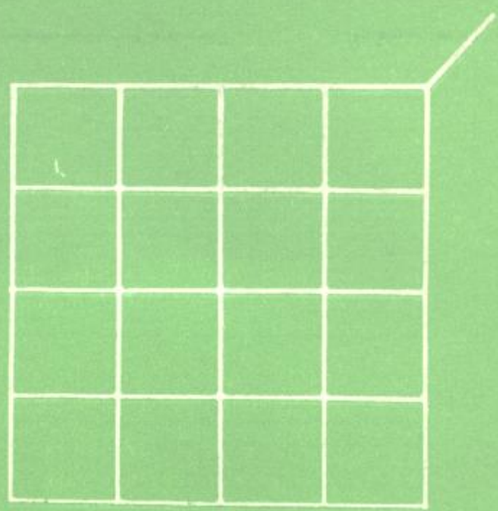
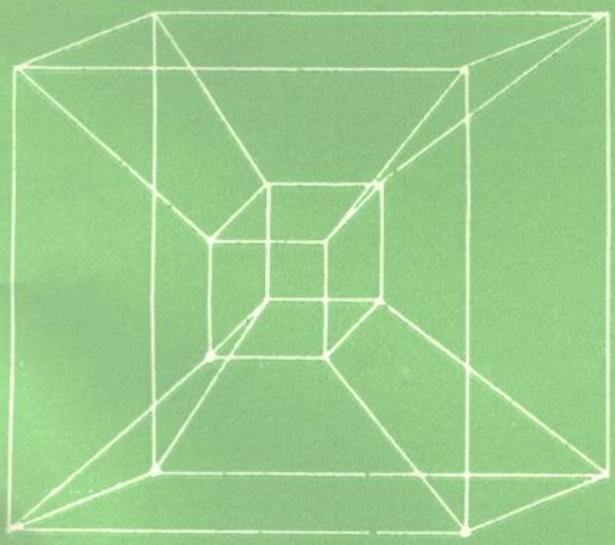
周南良 主编



数字逻辑



国防科技大学出版社

数 字 逻 辑

周南良 主编

国防科技大学出版社

[湘] 新登字 009 号

内 容 简 介

本书系统地介绍了数字逻辑的基本理论以及数字逻辑网络的分析和设计的基本方法。

全书共八章。第一至三章介绍数字逻辑设计的基础知识，包括数制与编码、布尔代数基础和布尔函数化简；第四至六章介绍数字逻辑网络分析和设计的基本方法，包括组合网络、同步时序网络和异步时序网络；第七、八两章介绍采用中、大规模集成电路的逻辑设计，以及介绍计算机辅助逻辑设计的基本概念和方法。

本书取材较新，基本概念叙述清楚。书中给出了大量例题，每章后还附有适量习题，书后附有习题答案，以便于自学。

本书可供大专院校计算机专业及有关专业作为教材，也可供科技人员参考。

数字逻辑

周南良　主编

责任编辑　王金荣

*

国防科技大学出版社出版发行

新华书店北京科技发行所经销

国防科技大学印刷厂印装

*

开本　787×1092　1/16　印张：17.5　字数：398 千字

1992 年 9 月第 1 版第 1 次印刷　印数：8000 册

ISBN　7-81024-190-7

TP・35　　定价：9.75 元

前　言

本书是为计算机及应用专业、计算机软件专业而编写的“数字逻辑”课程教材。

“数字逻辑”是计算机专业必修的专业基础课程。它主要讲述如何应用数字电路来进行数字系统的逻辑设计的基本理论和方法。对于从事计算机研制和应用的广大科技工作者来说，熟练地掌握数字逻辑设计的理论和方法是十分必要的。

数字逻辑设计的物质基础是电子元、器件。随着微电子技术的飞速发展，本课程的内容也在不断更新变化。本书是在我教研室使用多年的教材《数字逻辑》（张绍贤、朱锦桂编）的基础上，参考了国内兄弟院校的教材和国外有关文献，重新编写而成的。根据教学的需要，对原来经典的内容作了精简和修改。考虑到教材应反映当前新技术，增加了中、大规模集成电路有关内容（如门阵列技术等）；特别增加了计算机辅助逻辑设计这部分内容，取材较新，这是数字逻辑设计的发展趋向。

全书共八章，内容分成三部分。第一部分（第一、二、三章）介绍数字逻辑设计的基础知识，包括数制与编码、布尔代数基础与布尔函数化简；第二部分（第四、五、六章）介绍数字逻辑网络的分析与设计方法，包括组合网络、同步时序网络和异步时序网络；第三部分（第七、八章）介绍采用中、大规模集成电路的逻辑设计方法，以及采用计算机辅助逻辑设计的基本概念和方法。

本书在内容安排上，由易到难，层次清晰；文字叙述力求深入浅出，简明扼要；讨论设计方法力求概念清楚，步骤明确，便于读者自行设计所需逻辑电路。书中给出了大量例题，每章后附有适量习题，书后附录中给出部分习题答案，便于读者自学练习。

考虑到不同教学计划的要求，书中打有“*”号的章节，可根据情况作为选学或参考内容。

本书由周南良主编。第一至七章由周南良编写，第八章由周南良与焦彦平合编，焦彦平在 IBM PC 机上用 Turbo Pascal 4.0 调试了附录 A 中的源程序。

本书在编写出版过程中，曾得到国防科技大学计算机系兼研究所的大力支持和帮助。王风学教授审阅了本书的编写大纲和部分原稿，提出了许多指导性的意见。吴涛博士仔细审阅了全部书稿，杨超群副教授和刘依讲师审阅了部分章节，他们都提出了许多宝贵意见。谨向他们以及为本书的出版付出辛劳的所有同志表示诚挚的谢意。

由于笔者学识所限，书中不免有欠妥之处，恳请读者批评指正。

编　者

1992 年 4 月

目　　录

第一章　数制与编码

本章主要讨论数字系统中数的基本表示方法。首先讨论不同的进位计数制及其相互间如何转换；然后讨论二进制数在计算机中的表示方法，包括数的符号、数值以及小数点等如何表示。另外，本章还介绍计算机中常用的几种编码，包括十进制数的常用编码、字符编码以及可靠性编码等。

1.1　进位计数制

所谓进位计数制，就是按进位方式实现计数的一种规则，简称进位制。在日常生活中我们就是按这种进位制计数的，如十进制、十二进制、六十进制等等。

对于任何一个数，我们可以用不同的进位制来表示。我们先从熟悉的十进制开始，分析各种进位制的特点和表示方法。

十进制有十个数字符号，即 0、1、2、3、4、5、6、7、8、9. 将若干个这样的符号并列在一起可以表示一个十进制数，每位不超过“9”，由低位向高位进位是“逢十进一”。这是十进制的特点。

这里要引两个术语：一个叫“基数”，它表示某种进位制所具有的数字符号的个数，如十进制的基数为“十”；另一个叫“位权”或“权”，它表示某种进位制的数中不同位置上数字的单位数值，如十进制数 135.79，最左位为百位（1 代表 100），权为 10^2；第二位为十位（3 代表 30），权为 10^1；第三位为个位（5 代表 5），权为 10^0；小数点右边第一位为十分位（7 代表 1/10），权为 10^{-1}；第二位为百分位（9 代表 9/100），权为 10^{-2}.

基数和权是进位制的两个要素，根据基数和权的概念，我们可以将任何一个数表示成多项式的形式。例如：

$$135.79 = 1 \times 10^2 + 3 \times 10^1 + 5 \times 10^0 + 7 \times 10^{-1} + 9 \times 10^{-2}$$

对于一个一般的十进制数 N，它可表示成

$$(N)_{10} = (d_{n-1}d_{n-2}\cdots d_1 d_0 \cdot d_{-1}d_{-2}\cdots d_{-m})_{10} \tag{1.1}$$

或　$(N)_{10} = d_{n-1}(10)^{n-1} + d_{n-2}(10)^{n-2} + \cdots + d_1(10)^1 + d_0(10)^0 + d_{-1}(10)^{-1}$

$$+ d_{-2}(10)^{-2} + \cdots + d_{-m}(10)^{-m} = \sum_{i=-m}^{n-1} d_i(10)^i \tag{1.2}$$

式中，n 表示整数部分的位数；m 表示小数部分的位数；10 表示基数，$(10)^i$ 为第 i 位的权；d_i 表示各个数字符号，在十进制中有

$$d_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

通常，我们把式(1.1)称为并列表示法，把式(1.2)称为多项式表示法或按权展开式。

在数字系统中使用的进位制并不限于十进制。广义地，一个 R 进制的数 N，它可表示成

$$(N)_R=(r_{n-1}r_{n-2}\cdots r_1r_0\cdot r_{-1}r_{-2}\cdots r_{-m})_R$$
$$=r_{n-1}R^{n-1}+r_{n-2}R^{n-2}+\cdots+r_1R^1+r_0R^0+r_{-1}R^{-1}+r_{-2}R^{-2}+\cdots+r_{-m}R^{-m}$$
$$=\sum_{i=-m}^{n-1}r_iR^i$$

式中，n 表示整数的位数，m 表示小数的位数；R 为基数，在 R 进制中 R 应写成“10“；r_i 是 R 进制中各个数字符号，即有

$$r_i\in\{0,1,2,\cdots,R-1\}$$

数制是人类在实践中创造的，对于一个数，原则上讲我们可以用任何一种进位制来记数或进行算术运算。但是，不同的进位制的运算方法及难易程度各不相同。因此，选择什么样的进位制来表示数，对数字系统的性能影响很大。在数字系统中，常用二进制来表示数和进行运算。这是因为二进制只有 0 和 1 两个数字符号，容易用物理状态来表示；二进制运算规则简单，便于进行算术运算；此外，采用二进制来表示数可以节省设备，其运算逻辑电路的设计也比较方便。

二进制算术运算十分简单，规则如下：

加法规则　0+0=0，0+1=1+0=1，1+1=10

乘法规则　0×0=0，0×1=1×0=0，1×1=1

下面举几个二进制数四则运算的例子，从中领会它的运算规则。

例　两个二进制数相加，采用“逢二进一”的法则。

$$\begin{array}{r} 1\ 1\ 0\ 1 \\ +)\quad 1\ 0\ 0\ 1 \\ \hline 1\ 0\ 1\ 1\ 0 \end{array}$$

例　两个二进制数相减，采用“借一当二”的法则。

$$\begin{array}{r} 1\ 1\ 0\ 1 \\ -)\ 0\ 1\ 1\ 0 \\ \hline 0\ 1\ 1\ 1 \end{array}$$

例　两个二进制数相乘，其方法与十进制乘法运算相似，但采用二进制运算规则。

$$\begin{array}{r} 1\ 0\ 1\ 1 \\ \times)\quad 1\ 1\ 0\ 1 \\ \hline 1\ 0\ 1\ 1 \\ 0\ 0\ 0\ 0 \\ 1\ 0\ 1\ 1 \\ 1\ 0\ 1\ 1 \\ \hline 1\ 0\ 0\ 0\ 1\ 1\ 1\ 1 \end{array}$$

例　两个二进制数相除，其方法与十进制除法运算相似，但采用二进制运算规则。

```
          1 0 1 0  ……商
1 1 0 1)1 0 0 0 1 0 0 1
        1 1 0 1
        ---------
          1 0 0 0 0
            1 1 0 1
          ---------
              1 1 1 ……余数
```

虽然数字系统广泛采用二进制，但当二进制数的位数很多时，书写和阅读很不方便，容易出错。为此，人们通常采用二进制的缩写形式——八进制和十六进制。

八进制的基数 $R=8$，每位可取 8 个不同的数字符号（即 0，1，2，3，4，5，6，7），其进位规则是“逢八进一”。

由于一位八进制的 8 个数字符号正好相应于三位二进制数的八种不同组合，所以八进制与二进制之间有简单的对应关系：

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

这样，八进制与二进制之间数的转换就极为方便。

例 将二进制数 11010.1101 转换为八进制数。

$$\underbrace{011}_{3}\ \underbrace{010}_{2}\ .\ \underbrace{110}_{6}\ \underbrace{100}_{4}$$

所以 $(11010.1101)_2=(32.64)_8$

由二进制转换成八进制的方法是：以小数点为界，将二进制数的整数部分从低位开始，小数部分从高位开始，每三位分成一组、头尾不足三位的补 0；然后将每组的三位二进制数转换为一位八进制数。

例 将八进制数 357.6 转换为二进制数。

$$\begin{array}{ccccc} 3 & 5 & 7 & . & 6 \\ \downarrow & \downarrow & \downarrow & & \downarrow \\ 011 & 101 & 111 & . & 110 \end{array}$$

所以 $(357.6)_8=(11101111.11)_2$

十六进制的基数 $R=16$，每位可取 16 个不同的数字符号（即 0，1，2，3，4，5，6，7，8，9，A，B，C，D，E，F），其进位规则是“逢十六进一”。

同理，由于一位十六进制的 16 个数字符号正好相应于四位二进制数的十六种不同的组合，所以，十六进制与二进制之间有简单的对应关系：

十六进制	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
二进制	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

这样，十六进制与二进制之间数的转换也很方便。

例 将二进制数 1010110110.110111 转换为十六进制数。

$$\underbrace{0010}_{2}\ \underbrace{1011}_{B}\ \underbrace{0110}_{6}\ .\ \underbrace{1101}_{D}\ \underbrace{1100}_{C}$$

所以 $(1010110110.110111)_2=(2B6.DC)_{16}$

例 将十六进制数 $5D.\ 6E$ 转换为二进制数。

$$5\quad D\quad .\quad 6\quad E$$

$$\underset{0101}{\downarrow}\quad\underset{1101}{\downarrow}\quad .\quad\underset{0110}{\downarrow}\quad\underset{1110}{\downarrow}$$

所以　　$(5D.6E)_{16}=(1011101.0110111)_2$

由此可见，采用八进制和十六进制要比用二进制书写简短，易读易记，而且转换也方便。因此，计算机工作者普遍采用八进制或十六进制来书写和表达。

表 1.1 列出了当基数 R 为 10，2，8 和 16 时表示数值零到二十的不同进位制数。

表 1.1　不同基数的进位制数

$R=10$	$R=2$	$R=8$	$R=16$
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
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14

1.2　数制转换

在计算机和其它数字系统中普遍采用二进制，采用二进制的数字系统只能处理二进制数或用二进制编码形式表示的其它进位制数。由于人们习惯于使用十进制数，所以在用计算机进行信息处理时，首先必须把十进制数转换成二进制数才能被计算机所接受，然后进行运算，运算结果又必须从二进制转换成人们习惯的十进制数。

这一节我们研究不同进位制之间的相互转换的方法。

1.2.1　多项式替代法

我们先来看一个简单例子。

例　将二进制数 1101.101 转换成十进制数。

先把二进制数的并列表示法展开成多项式表示法，则有

$$(1101.101)_2=[1\times(10)^{11}+1\times(10)^{10}+0\times(10)^{1}+1\times(10)^{0}$$
$$+1\times(10)^{-1}+0\times(10)^{-10}+1\times(10)^{-11}]_2$$

再把等式右边的二进制数替代成十进制数，则得

$$(1101.101)_2=[1\times2^{3}+1\times2^{2}+0\times2^{1}+1\times2^{0}+1\times2^{-1}$$

$$+0\times2^{-2}+1\times2^{-3}]_{10}$$

在十进制中计算等式右边之值，得

$$(1101.101)_2=(8+4+1+0.5+0.125)_{10}=(13.625)_{10}$$

这一方法可以推广到任意两个 α、β 进制数之间的转换，其方法是：先将 α 进制的数在 α 进制中按权展开，然后替代成相应 β 进制中的数，最后在 β 进制中计算即可得 β 进制的数。

例 $(123.4_8')=(?)_{10}$

$$\begin{aligned}(123.4)_8&=[1\times(10)^2+2\times(10)^1+3\times(10)^0+4\times(10)^{-1}]_8 \quad \text{(展开)}\\&=(1\times8^2+2\times8^1+3\times8^0+4\times8^{-1})_{10} \quad \text{(替代)}\\&=(64+16+3+0.5)_{10} \quad \text{(在十进制中计算)}\\&=(83.5)_{10}\end{aligned}$$

例 $(201.2)_3=(?)_2$

$$\begin{aligned}(201.2)_3&=[2\times(10)^2+0\times(10)^1+1\times(10)^0+2\times(10)^{-1}]_3 \quad \text{(展开)}\\&=(10\times11^{10}+0\times11^1+1\times11^0+10\times11^{-1})_2 \quad \text{(替代)}\\&=(10010+1+0.101010\cdots)_2 \quad \text{(在二进制中计算)}\\&=(10011.101010\cdots)_2\end{aligned}$$

由以上两例可看出，多项式替代法由于要在 β 进制中进行计算，当它为十进制时，计算较方便，而当它为其它进制时，计算就很不方便。因此，这种方法用于 α 进制→十进制较方便。

1.2.2 基数乘/除法

基数乘/除法分为基数乘法和基数除法两种。对于整数的转换，采用基数除法；对于小数的转换，采用基数乘法。下面分别介绍这两种方法。

一）基数除法

引例 将十进制整数 25 转换为二进制数，即

$$(25)_{10}=(?)_2$$

我们来推导转换的方法。设转换结果为

$$\begin{aligned}(25)_{10}&=(k_{n-1}k_{n-2}\cdots k_1k_0)_2\\&=(k_{n-1}2^{n-1}+k_{n-2}2^{n-2}+\cdots+k_12^1+k_02^0)_{10}\end{aligned}\qquad(1.3)$$

在十进制中计算，将式(1.3)两边除以 2，则得

$$12+\frac{1}{2}=(k_{n-1}2^{n-2}+k_{n-2}2^{n-3}+\cdots+k_12^0)+\frac{k_0}{2}$$

两数相等，则它们整数部分和小数部分必定分别相等，故有

$$12=k_{n-1}2^{n-2}+k_{n-2}2^{n-3}+\cdots+k_12^0 \qquad(1.4)$$

$$\frac{1}{2}=\frac{k_0}{2}\qquad k_0=1$$

将式(1.4)两边同除以 2，可得

$$6=(k_{n-1}2^{n-3}+k_{n-2}2^{n-4}+\cdots+k_22^0)+\frac{k_1}{2}$$

故有

$$6 = k_{n-1}2^{n-3} + k_{n-2}2^{n-4} + \cdots + k_2 2^0$$

$$0 = \frac{k_1}{2} \qquad k_1 = 0$$

可见，所要求的二进制数$(k_{n-1}k_{n-2}\cdots k_1k_0)_2$的最低位$k_0$是十进制数25除以2所得余数；次低位$k_1$是所得商12再除以2所得的余数；依次类推，继续用2除，直到商为0为止。于是，各次所得的余数即为要求的二进制数$k_0 \sim k_{n-1}$之值。此法又称除2取余法。

可以将上述过程写成简单算式如下：

2	25	余数	低位
2	12	$1=k_0$	↑
2	6	$0=k_1$	|
2	3	$0=k_2$	|
2	1	$1=k_3$	|
	0	$1=k_4$	高位

所以，转换结果为$(25)_{10}=(11001)_2$

上述将十进制整数转换为二进制整数的方法可以推广到任何两个α、β进制数之间的转换。其方法是：先将α进制的整数在α进制中连续除以β，求得各次余数$(k_i)_\alpha$；然后将各余数替代成β进制中相应的数字符号$(k_i')_\beta$；最后按照并列表示法列出即得β进制的整数。应该指出，由于要在α进制中运算，而人们对十进制运算非常熟悉，所以基数除法用在十进制→β进制较方便。

例 $(785)_{10}=(?)_8$

8	785	余数	低位
8	98	1	↑
8	12	2	|
8	1	4	|
	0	1	高位

所以$(785)_{10}=(1421)_8$

例 $(687)_{10}=(?)_{16}$

16	687	余数	低位
16	42	15	↑
16	2	10	|
	0	2	高位

由于 $(15)_{10}=(F)_{16}$，$(10)_{10}=(A)_{16}$，$(2)_{10}=(2)_{16}$

所以 $(687)_{10}=(2AF)_{16}$

二）基数乘法

引例 将十进制小数0.6875转换为二进制数，即$(0.6875)_{10}=(?)_2$

设转换结果为

$$(0.6875)_{10} = (0.k_{-1}k_{-2}\cdots k_{-m})_2$$
$$= (k_{-1}2^{-1} + k_{-2}2^{-2} + \cdots + k_{-m}2^{-m})_{10} \qquad (1.5)$$

在十进制中计算，将式(1.5)两边乘以 2，则得

$$1.3750 = k_{-1} + (k_{-2}2^{-1} + k_{-3}2^{-2} + \cdots + k_{-m}2^{-m+1})$$

两数相等，则它们整数部分和小数部分必定分别相等，故有

$$k_{-1} = 1$$
$$0.3750 = k_{-2}2^{-1} + k_{-3}2^{-2} + \cdots + k_{-m}2^{-m+1} \qquad (1.6)$$

将式(1.6)两边再分别乘以 2，可得

$$0.75 = k_{-2} + (k_{-3}2^{-1} + \cdots + k_{-m}2^{-m+2})$$

故有

$$k_{-2} = 0$$
$$0.75 = k_{-3}2^{-1} + k_{-4}2^{-2} + \cdots + k_{-m}2^{-m+2}$$

可见，所要求的二进制数$(0.k_{-1}k_{-2}\cdots k_{-m})_2$的最高位$k_{-1}$是十进制数 0.6875 乘 2 所得的整数部分；其小数部分再乘以 2 所得的整数部分即为k_{-2}之值；依次类推，继续用 2 乘，每次所得的乘积整数部分即为要求的二进制数k_3～k_{-m}之值。此法又称乘 2 取整法。

可以将上述过程写成简单算式如下：

0.6875		
× 2	整数	
1.3750	……1＝k_{-1}	高位
0.3750		
× 2		
0.7500	……0＝k_{-2}	
0.7500		
× 2		
1.5000	……1＝k_{-3}	
0.5000		
× 2		
1.0000	……1＝k_{-4}	低位

所以，转换结果为$(0.6875)_{10}=(0.1011)_2$

上述将十进制小数转换为二进制小数的方法可以推广到任何两个α、β进制小数之间的转换，其方法是：先将α进制的小数在α进制中连续乘以β，求得各次乘积的整数部分$(k_i)_\alpha$；然后将各整数替代成β进制中相应的数字符号$(k_i')_\beta$；最后按照并列表示法列出即得β进制的小数。转换时小数的位数由乘积等于 0 或由所需精度确定。如何根据精度要求确定小数的位数，我们将在本节的最后介绍。同样应该指出，由于要在α进制中乘以β，所以基数乘法和基数除法一样，用在十进制→β进制较为方便。

利用基数乘/除法可以将一个十进制混合小数很方便地转换为任何β进制的数。只要将整数部分和纯小数部分按上述规则分别进行转换，然后将所得的数组合起来即可。

例　$(78.12)_{10}=(?)_5$

整数部分

5	78	余数	低位 ↑
5	15	……3	
5	3	……0	
	0	……3	高位

小数部分

0.12		
× 5	整数	高位 ↓
0.60	……0	
0.60		
× 5		
3.00	……3	低位

所以 $(78.12)_{10}=(303.30)_5$

例 $(45.3)_{10}=(?)_{16}$

整数部分

16	45	余数	低位 ↑
16	2	……13	
	0	……2	高位

小数部分

0.3	整数	
× 16		
4.8	……4	高位 ↓
0.8		
× 16		
12.8	……12	
0.8		
× 16		
12.8	……12	低位
⋮		

将十进制数替换成十六进制数字符号，有

$$(13)_{10}=(D)_{16} \qquad (4)_{10}=(4)_{16}$$

$$(2)_{10}=(2)_{16} \qquad (12)_{10}=(C)_{16}$$

所以 $(45.3)_{10}=(2D.4CC\cdots)_{16}$

1.2.3 混合法

前面介绍的两种方法尽管原则上都适用于任意 α、β 进位制数之间的转换，但考虑到人们对十进制运算十分熟悉，所以多项式替代法用于 α 进制→十进制的转换较方便，而基数乘除法用于十进制→β 进制数的转换较方便。这样，对于任意两种进位制数的转换，即 α 进制→β 进制，一种比较方便的方法是利用十进制作桥梁，先把 α 进制数转换为十进制数；然后再将十进制数转换为 β 进制数。这种方法是多项式替代法和基数乘除法的混合使用，我们称之为混合法。其示意图如图 1.1 所示。

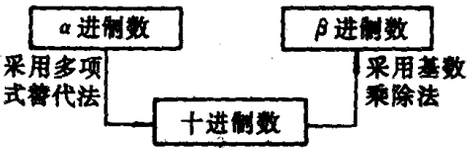


图 1.1 混合法示意图

例 $(121.02)_4=(?)_3$

先用多项式替代法将四进制数转换为十进制数，有

$$(121.02)_4=(1\times4^2+2\times4^1+1\times4^0+2\times4^{-2})_{10}$$

$$=(16+8+1+0.125)_{10}=(25.125)_{10}$$

再用基数乘除法将它转换为三进制：