高等学校教材

数字逻辑与数字系统

马义忠　常蓬彬　马　浚

高等教育出版社

高等学校教材

数字逻辑与数字系统

马义忠　常蓬彬　马　浚

高等教育出版社

内容提要

本书根据数字逻辑器件的发展历程，系统地阐述了数字逻辑系统的基本理论、分析方法和设计原理。突出基本原理及应用，使数字逻辑系统的设计从传统的单纯硬件设计方法变为计算机软、硬件协同设计。

全书共分13章，由逻辑代数、组合逻辑电路、时序逻辑电路、集成逻辑构件、可编程逻辑器件、数字系统设计方法及VHDL语言描述数字系统等7部分组成，每章均附有适量习题。

本书是根据计算机专业教学计划及相关信息类专业教学大纲编写的。全书概念清晰，取材丰富，体系新颖，可读性强。本书可作为高等院校计算机科学与技术、电子信息、通信类等专业的教材，也可作为成人教育的教材和相关专业科技人员的参考书。

图书在版编目（CIP）数据

数字逻辑与数字系统/ 马义忠，常蓬彬，马浚.
北京：高等教育出版社，2005.1
ISBN 7-04-016006-4

Ⅰ.数… Ⅱ.①马…②常…③马… Ⅲ.①数字逻辑—高等学校—教材②数字系统—高等学校—教材
Ⅳ.①TP302.2②TP271

中国版本图书馆CIP数据核字（2005）第000798号

策划编辑 倪文慧 **责任编辑** 倪文慧 **市场策划** 陈 振
封面设计 于文燕 **责任印制** 杨 明

出版发行	高等教育出版社	**购书热线**	010-58581118
社 址	北京市西城区德外大街4号	**免费咨询**	800-810-0598
邮政编码	100011	**网 址**	http://www.hep.edu.cn
总 机	010-58581000		http://www.hep.com.cn
		网上订购	http://www.landraco.com
经 销	北京蓝色畅想图书发行有限公司		http://www.landraco.com.cn
印 刷	北京市联华印刷厂		
开 本	787×1092 1/16	**版 次**	2005年1月第1版
印 张	19	**印 次**	2005年1月第1次印刷
字 数	410 000	**定 价**	25.00元

物料号： 16006－00

前　言

处于新世纪中的中国高等教育，面临世界范围的综合国力竞争，实质上是科学技术的竞争和民族素质的竞争。而对于中国高等教育的专业结构、课程体系、教学内容和教学方法，要从时代发展、技术进步的总体考虑，必须进行系统的调整与改革，跟踪世界先进水平。

现代科学技术的发展突飞猛进，电子技术更是一日千里，新理论、新发现从提出到实际应用，其周期大大缩短。就数字逻辑器件的功能和使用方法来说，经历了许多变化，从20世纪60年代小规模集成电路(SSI)的出现，70年代以后中规模集成电路(MSI)、大规模集成电路以及可编程逻辑器件的出现，80年代通用阵列逻辑(GAL)的研制成功，直到90年代在现系统编程(ISP)用户片的出现，在这样的发展历程中，由于数字逻辑器件更新换代的速度快，一方面使数字系统的设计方法发生了根本性变化，另一方面也向传统的“数字逻辑电路”课程的教学体系、教学内容、人才培养模式和任课教师提出了挑战。教材内容陈旧的局面再也不能继续下去了。

目前，大规模和超大规模的可编程逻辑器件得到了越来越广泛的应用。它们的设计采用的是计算机辅助设计技术，使电子系统的研制时间大大缩短，特别是在现系统可编程逻辑器件，可以在不改变硬件设置的情况下，在现场对系统进行组态，并可实现电子系统的遥控升级。

为了适应数字系统设计技术的发展，培养新世纪的科技人才，本书对数字逻辑方面的内容做了较大调整，除讲述必要的数字逻辑设计原理基础知识以外，对小规模电路的内容做了精简，加强了中大规模组件方面的内容，使读者能更熟练地掌握具体的使用技术。

在本书的编写过程中，得到了兰州大学教务处领导的大力支持，并得到兰州大学信息工程学院王玉曾教授、田亚菲教授的指导，在此一并表示感谢。

由于编者水平有限，敬请读者对本书的缺点及不足之处批评指正。

编者

2004年12月于兰州大学

目　录

第一章　数制与编码

本章主要介绍进位计数制的表示方法，以二进制为重点，讨论各种进制数的相互转换，带符号数的代码表示法，码制和字符的编码方法，等等。

1.1　进位计数制

1.1.1　十进制数的表示

用数字量表示物理量的大小时，仅用一位数码往往不够用。因此，用一组统一的符号和规则来表示数，常用进位计数的方法组成多位数码使用。我们把一组多位数码中每一位的构成方法以及从低位到高位的进位规则称为数制。

从原则上说，一个数可以用任何一种进位计数制来表示和运算，但不同的数制其运算方法及难易程度各不相同。选择什么样的进位计数制来表示数，对数字系统的性能影响很大。

在日常生活中，人们通常采用十进制数来计数，每位数可用十个数码之一来表示，即0、1、2、3、4、5、6、7、8、9。十进制的基数为10，基数表示进位制所具有的数字符号个数，十进制具有的数字符号个数为10。

十进制数的计算规律是由低位向高位进位时“逢十进一”。也就是说，每位累计不能超过9，计满10就应向高位进1。

当人们看到一个十进制数，如632.45时，就会立刻想到：这个数的最左位（即第一位）为百位（6代表600），第二位为十位（3代表30），第三位为个位（2代表2），小数点右面第一位为十分位（4代表4/10），第二位为百分位（5代表5/100）。这里百、十、个、十分之一和百分之一都是10的次幂。在一个进位制表示的数中，不同数位上的固定常数称之为“权”。十进制数632.45按权展开的形式如下

$$632.45 = 6 \times 10^2 + 3 \times 10^1 + 2 \times 10^0 + 4 \times 10^{-1} + 5 \times 10^{-2}$$

等式左边的表示方法称为位置计数表示法，等式右边则是其按权展开表示法。

一般说来，对于任意一个十进制数N，可用位置计数表示如下

$$(N)_{10} = (k_{n-1}k_{n-2}\cdots k_1k_0.k_{-1}k_{-2}\cdots k_{-m})_{10}$$

也可用按权展开表示法表示如下

$$(N)_{10} = k_{n-1} \times 10^{n-1} + k_{n-2} \times 10^{n-2} + \cdots + k_1 \times 10^1 + k_0 \times 10^0$$

$$+ k_{-1} \times 10^{-1} + k_{-2} \times 10^{-2} + \cdots + k_{-m} \times 10^{-m}$$
$$= \sum_{i=-m}^{n-1} k_i \times 10^i$$

式中：k_i 表示各个数字符号，为0～9这10个数码中的任意一个；n 为整数部分的位数；m 为小数部分的位数。

通常，对十进制数的表示，可以在数字的右下角标注 10 或 D。

1.1.2　二进制数的表示

数字系统中使用的进位制并不仅限于十进制，当进位基数为 2 时，称为二进制。在二进制中，只有 0 和 1 两个数码。二进制的计数规则是由低位向高位“逢二进一”，即每位计满 2 就向高位进 1，例如，$(1101)_2$ 就是一个二进制数。不同数位的数码表示的值不同，各位的权值是以 2 为底的连续整数幂，从右向左递增。

对于任意一个二进制数 N，用位置计数法表示为

$$(N)_2 = (k_{n-1}k_{n-2}\cdots k_1k_0.k_{-1}k_{-2}\cdots k_{-m})_2$$

用按权展开法表示为

$$(N)_2 = k_{n-1} \times 2^{n-1} + k_{n-2} \times 2^{n-2} + \cdots + k_1 \times 2^1 + k_0 \times 2^0 + k_{-1} \times 2^{-1}$$
$$+ k_{-2} \times 2^{-2} + \cdots + k_{-m} \times 2^{-m}$$
$$= \sum_{i=-m}^{n-1} k_i \times 2^i$$

式中：k_i 表示各个数字符号，为 0 或 1；n 为整数部分的位数；m 为小数部分的位数。

通常，对二进制数的表示，可以在数字右下角标注 2 或 B。

在数字系统中，常用二进制来表示数字并进行运算，因为它具有以下特点。

（1）二进制数只有 0 或 1 两个数码，任何具有两个不同稳定状态的元件都可用来表示 1 位二制数。例如，晶体管的导通和截止、脉冲信号的“有”或“无”等。

（2）二进制数运算规则简单。其运算规则如表 1.1.1 所示。

表 1.1.1　二进制数的运算规则

加法规则	减法规则	乘法规则	除法规则
0+0=0；1+0=1	0−0=0；0−1=1	0×0=0；0×1=0	0÷1=0；1÷1=1
0+1=1；1+1=0	1−0=1；1−1=0	1×0=0；1×1=1	

下面举例说明二进制数的运算。

例 1.1.1　进行 1101+1011 运算。

解：

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

两个二进制数的加法运算和十进制数的加法运算相似，但采用“逢二进一”的法则，每位数累计到2时，本位就记为0，同时向相邻高位进1。

例1.1.2 进行11101－10011运算。

解：

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

二进制数的减法运算从低位起按位进行，在遇到0减1时，就要采用“借一当二”的法则向相邻高位借1，也就是从那一高位减去1。

例1.1.3 进行1101×1001运算。

解：

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

二进制数的乘法运算和十进制数的乘法运算相似，所不同的是对部分积进行累加时要按“逢二进一”的法则。

例1.1.4 进行10010001÷1011运算。

解

$$\begin{array}{r} 1\ 1\ 0\ 1\ \cdots\cdots\text{商} \\ 1\ 0\ 1\ 1\ \overline{)\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 1} \\ \quad 1\ 0\ 1\ 1 \\ \hline \quad 1\ 1\ 1\ 0 \\ \quad 1\ 0\ 1\ 1 \\ \hline \quad\quad 1\ 1\ 0\ 1 \\ \quad\quad 1\ 0\ 1\ 1 \\ \hline \quad\quad\quad 1\ 0\ \cdots\cdots\text{余数} \end{array}$$

二进制数的除法运算与十进制数的除法运算类似，但采用二进制数的运算规则。

(3) 二进制数只有两个状态，数字的传输和处理不容易出错，可靠性高。

(4) 二进制数的数码0和1,可与逻辑代数中逻辑变量的值"假"和"真"对应起来。也就是说,可用一个逻辑变量来表示一个二进制数码。这样,在逻辑运算中就可以使用逻辑代数这一数学工具。

1.1.3 其他进制数的表示

二进制数运算规则简单,便于电路实现,它是数字系统中广泛应用的一种数制。但用二进制表示一个数时,所用的位数比用十进制数表示的位数多,人们读写很不方便,容易出错,不便记忆。因此,人们常采用八进制数和十六进制数,这两种数制不但容易书写、阅读、便于记忆,而且具有二进制数的特点,十分容易将它们转换成二进制数。

八进制数的基数是8,采用的数码是0、1、2、3、4、5、6、7。计数规则是从低位向高位"逢八进一",对于相邻两位来说,高位的权值是低位权值的8倍。例如,数$(47.6)_8$就表示一个八进制数。由于八进制的数码和十进制前8个数码相同,为了便于区分,通常在八进制数字的右下角标注8或O。

十六进制数的基数为16,采用的数码是0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F。其中A、B、C、D、E、F分别代表十进制数字10、11、12、13、14、15。十六进制数的计算规则是从低位向高位"逢十六进一",对于相邻两位来说,高位的权值是低位权值的16倍。例如,$(54AF.8B)_{16}$就是一个十六进制数。通常,在十六进制数字的右下角标注16或H。

与二进制数一样,八进制数和十六进制数均可用位置计数法的形式和按权展开法的形式表示。

一般说来,对于任意的数N,都能表示成以R为基数的R进制数。数N的位置记数法为

$$(N)_R = (k_{n-1}k_{n-2}\cdots k_1k_0.k_{-1}k_{-2}\cdots k_{-m})_R$$

而相应的按权展开表示法为

$$\begin{aligned}(N)_R &= k_{n-1}\times R^{n-1} + k_{n-2}\times R^{n-2} + \cdots + k_1\times R^1 + k_0\times R^0 \\ &\quad + k_{-1}\times R^{-1} + k_{-2}\times R^{-2} + \cdots k_{-m}\times R^{-m} \\ &= \sum_{i=-m}^{n-1} k_i R^i\end{aligned}$$

式中,k_i表示各个位的数字符号,为0~$(R-1)$数码中的任意一个;R为进位制的基数;n为整数部分的位数;m为小数部分的位数。

R进制的计数规则是从低位向高位"逢R进一"。

常用的不同数制的各种数码如表1.1.2所示,该表列出了当R为10、2、8、16时各种进位计数制中开头的16个自然数。

表 1.1.2 不同进位计数制的各种数码

十进制数($R=10$)	二进制数($R=2$)	八进制数($R=8$)	十六进制数($R=16$)
0	0000	00	0
1	0001	01	1
2	0010	02	2
3	0011	03	3
4	0100	04	4
5	0101	05	5
6	0110	06	6
7	0111	07	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

1.2 数制转换

1.2.1 二进制数与十进制数的转换

在计算机和其他数字系统中,普遍使用二进制数。采用二进制数的数字系统只能处理二进制数和用二进制代码形式表示的其他进制数。而人们习惯于使用十进制数,所以,在信息处理中,首先必须把十进制数转换成计算机能加工和处理的二进制数进行运算,然后再将二进制数的计算结果转换成人们习惯的十进制数。

二进制数转换成十进制数很方便,只要将二进制数写成按权展开式,并将式中各乘积项的积算出来,然后各项相加,即可得到与该二进制数相对应的十进制数。例如

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

十进制数转换成二进制数时,需将待转换的数分成整数部分和小数部分,并分别加以转换。一个十进制数可写成

$$(N)_{10}=(\text{整数部分})_{10}.(\text{小数部分})_{10}$$

转换时，首先将(整数部分)$_{10}$转换成(整数部分)$_2$，然后再将(小数部分)$_{10}$转换成(小数部分)$_2$。待整数部分和小数部分确定后，就可写成

$$(N)_2=(\text{整数部分})_2.(\text{小数部分})_2$$

1. 整数转换

十进制数的整数部分采用“除2取余”法进行转换，即把十进制数除以2，取出余数1或0作为相应二进制数的最低位，把得到的商再除以2，取余数1或0作为二进制数的次低位，依次类推，继续上述过程，直到商为0，所得余数为最高位。

例如，将十进制整数58转换为二进制整数，先把它写成如下形式：

$$\begin{aligned}(58)_{10}&=(k_{n-1}k_{n-2}\cdots k_1k_0)_2\\&=k_{n-1}\times2^{n-1}+k_{n-2}\times2^{n-2}+\cdots+k_1\times2^1+k_0\times2^0\\&=2(k_{n-1}\times2^{n-2}+k_{n-2}\times2^{n-3}+\cdots+k_1)+k_0\end{aligned}$$

只要求出等式中的各个系数$k_{n-1},k_{n-2},\cdots,k_1,k_0$，便得到二进制整数。将上式两边除以2，得

$$(29)_{10}=k_{n-1}\times2^{n-2}+k_{n-2}\times2^{n-3}+\cdots+k_1+\frac{k_0}{2}$$

两数相等，整数部分和小数部分必须对应相等，等式左边余数为0，则k_0为0。

因而得

$$(29)_{10}=2(k_{n-1}\times2^{n-3}+k_{n-2}\times2^{n-4}+\cdots+k_2)+k_1$$

将等式两边再除以2，得

$$(14+\frac{1}{2})_{10}=k_{n-1}\times2^{n-3}+k_{n-2}\times2^{n-4}+\cdots+k_2+\frac{k_1}{2}$$

比较等式两边，等式左边余数为1，则取k_1为1。

依次类推，可得系数k_2，k_3，…，k_{n-2}，k_{n-1}。

根据上面讨论的方法，可用下列形式很方便地将十进制整数转换成二进制整数。

2 | 58
2 | 29 ⟶ k_4=0 (余数0最低位)
2 | 14 ⟶ k_1=1 (余1)
2 | 7 ⟶ k_2=0 (余0)
2 | 3 ⟶ k_3=1 (余1)
2 | 1 ⟶ k_4=1 (余1)
0 ⟶ k_5=1 (余数1最高位)

因此，$(58)_{10}=(111010)_2$。

2. 纯小数转换

十进制数的小数部分采用“乘2取整”法进行转换，即先将十进制小数乘以2，取其整数1或0作为二进制小数的最高位；然后将乘积的小数部分再乘以2，并再取整数作为次高位。重复上述过程，直到小数部分为0或达到所要求的精度。

例如，将十进制小数0.625转换为二进制小数，需把它写成如下形式：

$$\begin{aligned}(0.625)_{10} &= (0.k_{-1}k_{-2}\cdots k_{-m})_2 \\ &= k_{-1}\times 2^{-1} + k_{-2}\times 2^{-2} + \cdots + k_{-m}\times 2^{-m} \\ &= \frac{k_{-1}}{2} + \frac{1}{2}(k_{-2} + \cdots + k_{-m}\times 2^{-m+1})\end{aligned}$$

只要求出各系数 k_{-1}，k_{-2}，…，k_{-m}，便可得到二进制小数。

将上式两边乘2，得

$$(1.25)_{10} = k_{-1} + (k_{-2}\times 2^{-1} + \cdots + k_{-m}\times 2^{-m+1})$$

根据两个数相等，其整数部分和小数部分必须分别相等的道理，k_{-1}等于左边的整数，则 k_{-1} 为1。

等式右边括号内的数仍为小数，因而

$$(0.25)_{10} = \frac{k_{-2}}{2} + \frac{1}{2}(k_{-3}\times 2^{-1} + \cdots + k_{-m}\times 2^{-m+2})$$

再将等式两边乘2，得

$$(0.5)_{10} = k_{-2} + (k_{-3}\times 2^{-1} + \cdots + k_{-m}\times 2^{-m+2})$$

比较等式两边的整数，又取 k_{-2} 为0。如此连续乘2，直到小数部分等于0，即可求得系数 k_{-1}，k_{-2}，…，k_{-m}。

根据上面讨论的方法，可用下列形式很方便地将十进制小数转换成二进制小数。

```
    0.625
×)      2
-----------
   [1].250      整数1(k_-1)最高小数位
×)      2
-----------
   [0].500      整数0(k_-2)次高位小数位
×)      2
-----------
   [1].000      整数1(k_-3)最低位小数位
```

最后得：$(0.625)_{10} = (0.101)_2$。

必须指出：运算时，式中的整数不参加连乘。

在十进制数的小数部分转换中，有时连续乘2不一定能使小数部分等于0，这说明该十进制小数不能用有限位二进制小数表示。这时，只要取足够多的位数，使其误差达到所要求

的精度就可以了。下面举例说明。

例 1.2.1　将十进制数 0.18 转换成二进制数，精确到小数点后 4 位。

解：

$$
\begin{array}{rl}
 & 0.18 \\
\times) & \quad 2 \\
\hline
 & \boxed{0}.36 \qquad \text{整数 } 0(k_{-1}) \\
\times) & \quad 2 \\
\hline
 & \boxed{0}.72 \qquad \text{整数 } 0(k_{-2}) \\
\times) & \quad 2 \\
\hline
 & \boxed{1}.44 \qquad \text{整数 } 1(k_{-3}) \\
\times) & \quad 2 \\
\hline
 & \boxed{0}.88 \qquad \text{整数 } 0(k_{-4}) \\
\times) & \quad 2 \\
\hline
 & \boxed{1}.76 \qquad \text{整数 } 1(k_{-5})
\end{array}
$$

十进制数 0.18 连续四次乘 2 后，其小数部分等于 0.88，仍不为 0。由于要求精确到小数点后 4 位，因此将 0.88 再乘 2，小数点后第 5 位为 1，得

$$(0.18)_{10} \approx (0.00101)_2$$

如果一个十进制数既有整数部分又有小数部分，转换时，整数部分采用“除 2 取余”法，小数部分采用“乘 2 取整”法，然后再把转换的结果合并起来即可。

1.2.2　二进制数与八进制数、十六进制数的转换

八进制数的基数 $R=8$，3 位二进制数恰好是 8 个状态，即 $8=2^3$；十六进制数的基数为 $R=16$，4 位二进制数恰好是 16 个状态，即 $16=2^4$。二进制数、八进制数和十六进制数之间具有 2 的整指数倍关系，因而可直接进行转换。

将二进制数转换为八进制或十六进制的方法是：从小数点开始，分别向左、右按 3 位一组转换为八进制，或按 4 位一组转换为十六进制，最后不满 3 位或 4 位的则需补 0。将每组以对应的等值八进制数或十六进制数代替，举例如下：

二进制数：010　101　111 · 000　101　101　100

八进制数：2　5　7 · 0　5　5　4

二进制数：1010　1111 · 0001　0110　1100

十六进制数：A　F · 1　6　C

最后得:$(257.0554)_8 = (10101111.0001011011)_2 = (AF.16C)_{16}$

若将八进制数或十六进制数转换成二进制数时,可按上述方法的逆过程进行。

1.3 带符号数的代码表示

1.3.1 真值与机器数

通常,我们都用符号“+”表示正,用符号“-”表示负。“+”和“-”无非是表示两种对立的状态标志,如同计算机中可采用的“0”和“1”一样。因此,在计算机中表示正、负号的最简单的方法是约定用0表示“+”,用1表示“-”。

一个带符号的数由两部分组成:一部分表示数的符号,另一部分表示数的数值。对于一个 n 位二进制数,若数的第一位为符号位,则剩下的 $n-1$ 位就表示数的数值部分。一般用正号“+”和负号“-”来表示带符号的二进制数,叫做符号数的**真值**。数的真值形式是一种原始形式,不能直接用于计算机中。当把符号数值化后,就可在计算机中使用它。数的符号是一个具有正、负两种值的离散信息,它可以用一位二进制数来表示。习惯上以0表示正数,而以1表示负数。计算机中使用的符号数叫做**机器数**。

例如,二进制正数 +0.1011 在机器中的表示如图1.3.1(a)所示,二进制负数 -0.1011 在机器中的表示如图1.3.1(b)所示。

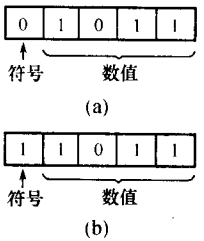

图1.3.1 二进制数在机器中的表示

由二进制数的加、减、乘、除4种运算可知,乘法运算实际上是做移位加法运算,而除法运算则是做移位减法运算。这就是说,在机器中只需要做加、减两种运算。但做减法运算时,必须先比较两个数绝对值的大小,将绝对值大的数减绝对值小的数,最后在相减结果的前面加上正确的符号。虽然逻辑电路可以实现减法运算,但所需的电路复杂,运算时间较长。为了能使减法运算变成加法运算,人们提出了三种机器数的表示形式,称为原码、反码和补码。

1.3.2 原码

原码又称为“符号-数值表示”。在以原码形式表示的正数和负数中,第1位表示符号位,对于正数,符号位记作0,对于负数,符号位记作1,其余各位表示数值部分。

例如两个带符号的二进制数分别为 N_1 和 N_2,其真值形式为

$$N_1 = +100110 \qquad N_2 = -010101$$