高等学校教材

数字逻辑与计算机设计基础

刘 真 蔡懿慈 毕才术

高等教育出版社
HIGHER EDUCATION PRESS

高等学校教材

数字逻辑与计算机设计基础

刘　真　蔡懿慈　毕才术

高等教育出版社

内容提要

本书以一个计算机硬件系统的设计为主线，先介绍数字电路设计的基本知识，然后使用这些知识和计算机结构的基本知识设计一个简单的计算机，最后利用已有的芯片设计完成一个较完整的计算机硬件系统。全书共9章，分别为：数制及编码、组合逻辑电路、时序逻辑电路、计算机设计引论、总线、存储器、简易计算机设计、基于微处理器的计算机设计、接口与通信等，并用附录的形式介绍了8086微机原理及接口实验系统和8086指令系统的内容。本书包含了数字逻辑、计算机原理和微机原理等三门课程的必备知识，并有机地结合在一起，适合计算机软件专业、非计算机专业的本科生和各类专科生作为教材或自学教材使用。

图书在版编目(CIP)数据

数字逻辑与计算机设计基础/刘真，蔡懿慈，毕才术
—北京：高等教育出版社，2003.6
ISBN 7-04-013314-8

Ⅰ.数… Ⅱ.①刘…②蔡…③毕… Ⅲ①数字逻辑-高等学校-教材②电子计算机-设计-高等学校-教材 Ⅳ.TP302

中国版本图书馆CIP数据核字(2003)第055812号

出版发行	高等教育出版社	购书热线	010-64054588
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	010-82028899		http://www.hep.com.cn
经　　销	新华书店北京发行所		
印　　刷	北京未来科学技术研究所有限责任公司印刷厂		
开　　本	787×960　1/16	版　　次	2003年6月第1版
印　　张	13.75	印　　次	2003年6月第1次印刷
字　　数	240 000	定　　价	20.00元

本书如有缺页、倒页、脱页等质量问题，请到所购图书销售部门联系调换。

本书编委会

前　言

硬件设计课程由于设备、师资、学时等诸多因素所限，在许多大学计算机系的课程中处于相对次要的地位。同时，由于软件产业的发展创造了更多的就业机会，而软件开发对于硬件知识的依赖较少，因此学生们对软件的兴趣正在增加，而忽视了硬件设计的实践环节。但是，作为专业的计算机工作者和高级软件开发人员，如果不能很好地理解计算机的结构和工作原理，对于许多问题，包括软件设计方面的问题，就不可能找到最优的系统解决方案。

在硬件课程教学实践中，由于学生们很少有机会接触具体的工程问题和实际设备，同时由于各课程过分强调技术的细节，而忽视了相互之间的关系，学生们往往“只见树木不见森林”。本书编写的出发点正是为了弥补这些不足。

本书以一个计算机硬件系统的设计为主线，先介绍数字电路设计的基本知识，然后使用这些知识和计算机结构的基本知识设计一个简单的计算机，最后利用已有的芯片设计完成一个较完整的计算机硬件系统。为了使这条主线清晰，我们尽量略去了不必要的知识和内容，必备知识也尽可能用通俗、简单的方式叙述，而新兴的可编程逻辑器件技术和硬件描述语言由于篇幅所限没有介绍，对于想深入了解相关知识的读者，请参考有关的书籍。

原教育部副部长吕福源同志（现任国家商务部部长）提出了编写本书的设想，教育部高等教育司张尧学司长亲自主持了本书的编写组织工作，并在百忙之中仔细审阅了全部书稿，提出了许多指导性的修改意见，使本书得以进一步完善。北京邮电大学白中英先生也仔细审阅了全稿，并提出了许多宝贵意见，在此表示衷心的感谢。

本书针对软件学院学生的特点和要求，同时考虑到其他相关专业的教学需求组织编写。本书内容涉及数字逻辑、计算机原理、微机原理等三方面的知识，编者力图将它们有机地结合在一起，以期达到融汇贯通、深入浅出的目的。但限于编者水平，虽尽力而为，几经修改，仍与当初的设想和期望差距很大。鉴于时间所限，只好仓促付梓。不足之处，请读者批评指正。

编　者

2003 年 6 月

郑重声明

策划编辑　刘建元
责任编辑　倪文慧
封面设计　王凌波
责任印制　杨　明

目　录

第一章　数制及编码

1.1　进位计数制与数制转换

数制是人们表示数值大小的各种方法的统称。迄今为止，人类都是按照进位方式来实现计数的，这种数制称为进位计数制。大家熟悉的十进制，就是一种典型的进位计数制。在数字系统中，广泛采用的是二进制、八进制和十六进制。

一种进位计数包含着两个基本的因素：基数和位权。基数是计数制中所用到的数码的个数，如十进制中使用的0、1、2、…、9十个字符，所以十进制数的基数为10。一般地说，基数为 R 的计数制中，包含的是0、1、…、$R-1$ 等数码。位权则表示在一个进位计数制表示的数中，处在不同数位的数码代表着不同的数值，某一个数位的数值是由这一位数码的值乘上处在该位的一个固定常数。不同数位上的固定常数称为位权值，简称位权。例如，十进制数个位的位权值是 10^0，十位的位权值是 10^1，百位的位权值是 10^2，等等。一个 R 进制数 N，可以有两种表示方式：

(1) 并列表示方式或位置计数法

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

其中，n 为整数部分的数位，m 为小数部分的数位，R 表示基数，K_i 为不同数位的数值：

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

(2) 多项式表示法或以权展开式

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

或者写成和式：

$$(N)_R = \sum_{i=-m}^{n-1} K_iR^i$$

其中，R 代表进位制的基数，m、n 为正整数，n 代表整数部分的位数，m 代表小数部分的位数，K_i 代表 R 进位制中 R 个数字符号中的任何一个：

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

1.1.1 二进制

1. 二进制数的表示

基数 $R = 2$ 的数制为二进制。二进制数的数值表示只有“1”和“0”，进位规律是“逢二进一”，任意一个二进制数 N 的多项式表示为

$$(N)_2 = K_{n-1}2^{n-1} + K_{n-2}2^{n-2} + \cdots + K_1 2^1 + K_0 2^0 + K_{-1}2^{-1} + \cdots + K_{-m}2^{-m}$$

$$= \sum_{i=-m}^{n-1} K_i 2^i$$

其中，K_i 为 0 或 1，2 为位权。

例如，二进制数 1111.001 可以展开为：

$$1111.001 = 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$$

2. 二进制数的运算

二进制数的算术运算规律有加法规律和乘法规律。

加法规律为

$$0 + 0 = 0 \qquad 0 + 1 = 1 + 0 = 1 \qquad 1 + 1 = 10$$

乘法规律为

$$0 \times 0 = 0 \qquad 0 \times 1 = 1 \times 0 = 0 \qquad 1 \times 1 = 1$$

1.1.2 八进制

基数 $R = 8$ 的数制为八进制。八进制数的数位符号有 8 个，即 0、1、…、7，进位规律是“逢八进一”，任意的一个八进制数 N 的多项展开式为

$$(N)_8 = K_{n-1}8^{n-1} + K_{n-2}8^{n-2} + \cdots + K_1 8^1 + K_0 8^0 + K_{-1}8^{-1} + \cdots + K_{-m}8^{-m}$$

$$= \sum_{i=-m}^{n-1} K_i 8^i$$

其中，K_i 表示为 0 ~ 7 中的任意一个。

1.1.3 十六进制

基数 $R = 16$ 的数制为十六进制。十六进制数的表示符号有 16 个，分别为 0 ~ 9 以及用 A、B、C、D、E 和 F 表示的 10 ~ 15。进位规律为“逢十六进一”，任意的一个十六进制数 N 的多项表示式为

$$(N)_{16} = K_{n-1}16^{n-1} + K_{n-2}16^{n-2} + \cdots + K_1 16^1 + K_0 16^0 + K_{-1}16^{-1} + \cdots + K_{-m}16^{-m}$$

$$= \sum_{i=-m}^{n-1} K_i 16^i$$

其中，K_i 表示为 0 ~ 9 以及 A、B、C、D、E 和 F 中的任意一个。

1.1.4 二进制与八进制、十六进制之间的转换

1. 二进制数转换为八进制数

二进制数转换为八进制数时，整数部分从低位向高位每 3 位分为一组，最高一组不够3位时，用0补足；小数部分从高位向低位每3位一组，最后不足3位的，在低位补 0，然后分别把每 3 位的二进制数用相应的八进制数表示。

例如，$(11110.11)_2$ 可表示为

011	110	.	110
3	6	.	6

即 $(11110.11)_2 = (36.6)_8$

2. 八进制数转换为二进制数

八进制数转换为二进制数时，把每位八进制数用三位二进制数表示。

例如，$(413.62)_8$ 可表示为

4	1	3	.	6	2
100	001	011	.	110	010

即 $(413.62)_8 = (100001011.11001)_2$

3. 二进制数转换为十六进制数

二进制数转换为十六进制数时，整数部分由小数点向左，每四位一组，最高一组不足 4 位时，前面补 0；小数部分由小数点向右，每 4 位一组，最后不足 4 位的，在低位补 0，然后分别把每 4 位二进制数用相应的十六进制数表示。

例如，$(1110111.101)_2$ 可表示为

0111	0111	.	1010
7	7	.	A

即 $(1110111.101)_2 = (77.A)_{16}$

4. 十六进制数转换为二进制数

十六进制数转换为二进制数时，把每位十六进制数用相应的 4 位二进制数表示。

例如，$(41B.2)_{16}$ 可表示为

4	1	B	.	2
0100	0001	1011	.	0010

即 $(41B.2)_{16} = (10000011011.001)_2$

1.1.5　二进制数与十进制数之间的转换

1. 二进制数转换为十进制数

采用多项式替代法把二进制数按权展开，利用十进制运算法则求出其值，即可将二进制数转换为十进制数。

例如：

$$\begin{aligned}(101.101)_2 &= 1\times 2^2 + 1\times 2^0 + 1\times 2^{-1} + 1\times 2^{-3}\\ &= 4 + 1 + 0.5 + 0.125\\ &= (5.625)_{10}\end{aligned}$$

2. 十进制数转换为二进制数

采用基数除／乘法，分别将十进制数的整数部分和小数部分转换为二进制数，整数部分用基数除法，小数部分用基数乘法，然后用小数点将两部分连接起来。

例如，将十进制数$(125.843)_{10}$转换为二进制数。

对于整数部分，

$$(125)_{10} = K_{n-1}2^{n-1} + K_{n-2}2^{n-2} + \cdots + K_1 2^1 + K_0 2^0$$

显然，等式右边除K_0项外都有2的因子。因此，用2除$(125)_{10}$，所得余数即为K_0，即

$$2\overline{)125}$$
$$62\cdots\cdots\text{余数 } 1 = K_0$$

并得到等式：

$$(62)_{10} = K_{n-1}2^{n-2} + K_{n-2}2^{n-3} + \cdots + K_2 2^1 + K_1$$

同样，再用2除$(62)_{10}$，余数则为K_1，即

$$2\overline{)62}$$
$$31\cdots\cdots\text{余数 } 0 = K_1$$

再用这样的方法一直继续下去，直到商为0为止。

$$
\begin{array}{r|l}
2 & 125 \quad \text{余数为 1，所以 } K_0=1 \\
2 & 62 \quad \text{余数为 0，所以 } K_1=0 \\
2 & 31 \quad \text{余数为 1，所以 } K_2=1 \\
2 & 15 \quad \text{余数为 1，所以 } K_3=1 \\
2 & 7 \quad \text{余数为 1，所以 } K_4=1 \\
2 & 3 \quad \text{余数为 1，所以 } K_5=1 \\
2 & 1 \quad \text{余数为 1，所以 } K_6=1 \\
\multicolumn{2}{r}{0}
\end{array}
$$

得$(125)_{10} = (1111101)_2$。

十进制小数转换为二进制小数的方法是：不断用2乘要转换的十进制小数，将每次所得的整数(0或1)，依次记为 K_{-1}、K_{-2}、…。若乘积的小数部分最后能为0，那么最后一次乘积的整数部分记作 K_{-m}，则 $0.K_{-1}K_{-2}\cdots K_{-m}$ 即为十进制小数的二进制表达式。因为十进制小数并不都是能用有限位的二进制小数精确表示，通常则是根据精度要求 m 位，作为十进制小数的二进制表示的近似表达式。

例如：将$(0.843)_{10}$转换为二进制数。

$$(0.843)_{10} = K_{-1}2^{-1} + K_{-2}2^{-2} + \cdots + K_{-m}2^{-m}$$

两边乘以2，得

$$(1.686)_{10} = K_{-1} + K_{-2}2^{-1} + \cdots + K_{-m}2^{-m+1}$$

所以 $K_{-1} = 1$；再对$(0.686)_{10} = K_{-2}2^{-1} + K_{-3}2^{-2} + \cdots + K_{-m}2^{-m+1}$

两边再乘以2，得

$$(1.372)_{10} = K_{-2} + K_{-3}2^{-1} + K_{-4}2^{-2} + \cdots + K_{-m}2^{-m+2}$$

所以 $K_{-2} = 1$；假如精度要求 $m = 2$，转换过程如下：

$$
\begin{array}{r}
0.843 \\
\times \quad 2 \\
\hline
1.686
\end{array}
\qquad \text{整数部分为 1，所以 } K_{-1} = 1;
$$

$$
\begin{array}{r}
0.686 \\
\times \quad 2 \\
\hline
1.372
\end{array}
\qquad \text{整数部分为 1，所以 } K_{-2} = 1;
$$

因此，$(0.843)_{10} = (0.11)_2$

$(125.843)_{10} = (1111101.11)_2$

1.2 编 码

虽然计算机是采用二进制数进行处理，但人们输入的不仅仅是二进制数，而是数字、字母甚至符号，这些数字、字母和符号也必须用二进制数来表示，各种不同的表示方法就称为编码。

1.2.1 十进制数的二进制编码

用二进制数码按照不同规律编码来表示十进制数，使其既具有二进制的形式，又具有十进制数的特点，便于传递、处理。

一位十进制数有 0 ~ 9 十个不同数码，至少需要 4 位二进制数编码。当采用 4 位二进制数进行编码时，共有 16 种代码。从 16 种代码中取 10 种代码来表示十进制的 10 个字符的编码方式很多。一般分为有权码和无权码。有权码是指 4 位二进制数中的每一位都对应有固定的权。无权码是指 4 位二进制数中的每一位无固定的权，而要遵循另外的规则。最常用的十进制数的二进制编码有以下几种。

1. 8421 码

8421 码为有权码。它是十进制代码中最常见的代码，也称二 — 十进制码，简称 BCD 码(Binary Code Decimal)。4 位二进制码从高位至低位每位的权分别为 2^3、2^2、2^1、2^0，即为 8、4、2、1。

2. 5421 码

5421 是一种有权码，其权自高位至低位，每位分别为 5、4、2、1。所以，十进制数 X 用 5421 码 $a_3a_2a_1a_0$ 表示为

$$X = a_3 \times 5 + a_2 \times 4 + a_1 \times 2 + a_0 \cdot 1$$

3. 2421 码

2421 码为另一种有权码，也是四位代码，每位权从高位到低位为 2、4、2、1，若一个 2421 编码的二进制数码为 $a_3a_2a_1a_0$ 时，它表示的十进制数值为

$$X = a_3 \times 2 + a_2 \times 4 + a_1 \times 2 + a_0 \times 1$$

4. 余三码

余三码是一种无权码，因为它是将 4 位 8421 码首位各多余的 3 组去掉而得到的，所以称为余三码，它可由 8421 码加 0011 得到。

以上四种十进制数的二进制编码格式如表 1.1 表示。其中，8421 码的 1010 ~

1111 是没有意义的;余三码表中,0 和 9、1 和 8、2 和 7、3 和 6、4 和 5 互为反码。因此,用余三码做十进制数的算术运算是比较方便的。

表 1.1 常用的 BCD 编码

十进制数	8421	5421	2421	余三码
0	0000	0000	0000	0011
1	0001	0001	0001	0100
2	0010	0010	0010	0101
3	0011	0011	0011	0110
4	0100	0100	0100	0111
5	0101	1000	0101	1000
6	0110	1001	0110	1001
7	0111	1010	0111	1010
8	1000	1011	1110	1011
9	1001	1100	1111	1100

1.2.2 带符号的二进制数的编码

在数字系统中,正、负数的表示方法是:把一个数的最高位作为符号位,并用"0"表示"+";用"1"表示"-",连同符号位在一起作为一个数,称之为机器数,它的原来的数值形式则称为这个机器数的真值。

例如: $X_1 = +0.1011$; $X_2 = -0.111$

表示成机器数为 $X_1 = 0.1011$; $X_2 = 1.111$

常用的表示机器数的方法有原码、反码和补码。

1. 原码

正数的符号位用"0"表示;负数的符号位用"1"表示;数值部分保持不变。

(1) 小数原码的定义

若二进制数 X 为小数,$X = \pm 0.X_{-1}X_{-2}\cdots X_{-m}$,则 X 的原码表示为

$$[X]_{原} = \begin{cases} X & \text{当 } 0 \leqslant X < 1 \\ 1 - X & \text{当 } -1 < X \leqslant 0 \end{cases}$$

例如:$X_1 = +0.11011$ 则 $[X]_{原} = 0.11011$

$X_2 = -0.1101$ 则 $[X]_{原} = 1 - (-0.1101) = 1.1101$

(2) 整数原码的定义

若二进制数 X 为整数,$X = \pm X_{n-1}X_{n-2}\cdots X_0$,则的原码定义为

$$[X]_{原} = \begin{cases} X & 当 0 \leqslant X < 2^n \\ 2^n - X & 当 -2^n < X \leqslant 0 \end{cases}$$

例如：$X = -11011$

$$\begin{aligned} [X]_{原} &= 2^n - X \\ &= 10000 - (-11011) \\ &= 10000 + 11011 \\ &= 111011 \end{aligned}$$

(3) 零的原码有两种表示形式

$$[+0]_{原} = 0.00\cdots0$$

$$[-0]_{原} = 1.00\cdots0$$

2. 反码

正数的符号位用“0”表示，负数的符号位用“0”表示；反码数值部分的形成和它的符号位有关。正数反码的数值和原码的数值相同，而负数反码的数值是原码的数值按位求反。

(1) 整数反码的定义

若二进制数 X 为整数，$X = \pm X_{n-1}X_{n-2}\cdots X_0$，则 X 的反码定义为

$$[X]_{反} = \begin{cases} X & 当 0 \leqslant X < 2^n \\ (2^{n+1} - 1) + X & 当 -2^n < X \leqslant 0 \end{cases}$$

例如：$X_1 = +11011$，则$(X_1)_{反} = 011011 = 11011$

$X_2 = -1101$，则$(X_2)_{反} = (2^5 - 1) + X$

$$\begin{aligned} &= (100000 - 000001) + (-1101) \\ &= 11111 - 1101 \\ &= 10010 \end{aligned}$$

(2) 小数反码的定义

若二进制数 X 为小数，$X = \pm 0.X_{-1}X_{-2}\cdots X_{-m}$，则 X 的反码定义为

$$[X]_{反} = \begin{cases} X & 当 0 \leqslant X < 1 \\ 2 - 2^{-n} + X & 当 -1 < X \leqslant 0 \end{cases}$$

例如：$X_1 = +0.11011$，则$[X_1]_{反} = 0.11011$

$X_2 = -0.1101$，则$[X_2]_{反} = 2 - 2^{-4} + X$

$$\begin{aligned} &= 10.0000 - 0.0001 - 0.1101 \\ &= 1.0010 \end{aligned}$$

(3) 零的反码表示有两种形式