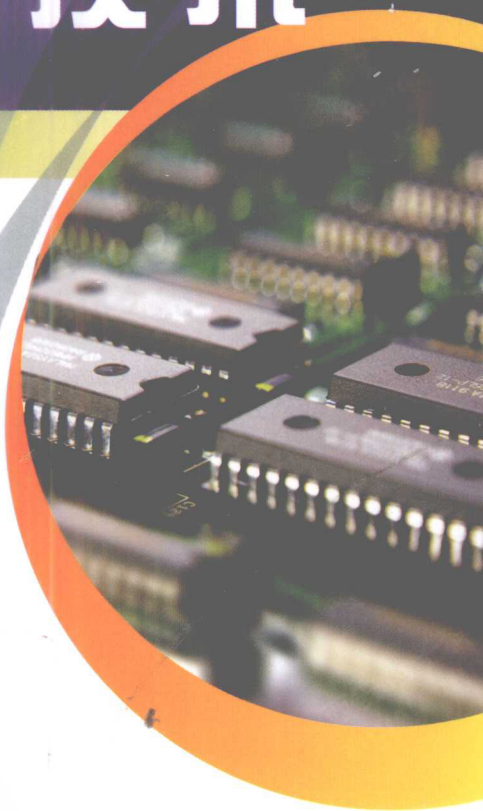
21 世纪 CENTURY

高等学校信息工程类专业规划教材

数字电子技术



主　编　董　敏
副主编　李旭红　王守华

西安电子科技大学出版社
http://www.xduph.com

面向21世纪高等学校信息工程类专业规划教材

数字电子技术

主　编　董　敏
副主编　李旭红　王守华

西安电子科技大学出版社

内 容 简 介

全书共分 8 章。第 1 章和第 2 章作为数字逻辑的理论基础，讨论了数制、码制和逻辑代数基础。第 3 章至第 5 章在小规模集成电路分析和设计基础上，讨论了组合逻辑和时序逻辑电路中的基本概念、分析方法及设计方法。第 6 章讨论了脉冲波形的产生与变换电路的结构、工作原理及参数计算。第 7 章讨论了数/模与模/数转换电路的结构、主要技术指标。第 8 章讨论了半导体存储器和可编程逻辑器件的结构特点及应用。

本书可作为电子工程、计算机、机电等相关专业的本科生教材，也可供电子技术领域的工程技术人员学习参考。

★ 本书配有电子教案，需要者可登录出版社网站，免费下载。

图书在版编目(CIP)数据

数字电子技术/董敏主编.

—西安：西安电子科技大学出版社，2010.8

面向 21 世纪高等学校信息工程类专业规划教材

ISBN 978 - 7 - 5606 - 2437 - 2

Ⅰ. ① 数… Ⅱ. ① 董… Ⅲ. ① 数字电路—电子技术 Ⅳ. ① TN79

中国版本图书馆 CIP 数据核字(2010)第 099832 号

策　　划　戚文艳

责任编辑　戚文艳

出版发行　西安电子科技大学出版社(西安市太白南路 2 号)

电　　话　(029)88242885　88201467　　邮　　编　710071

网　　址　www.xduph.com　　电子邮箱　xdupfxb001@163.com

经　　销　新华书店

印刷单位　西安文化彩印厂

版　　次　2010 年 8 月第 1 版　2010 年 8 月第 1 次印刷

开　　本　787 毫米×1092 毫米　1/16　印张 19.5

字　　数　461 千字

印　　数　1～3000 册

定　　价　28.00 元

ISBN 978 - 7 - 5606 - 2437 - 2/TN • 0566

XDUP　2729001－1

前　言

本书是根据教育部《关于进一步加强高等学校本科教学工作的若干意见》和《教育部、财政部关于实施高等学校本科教学质量与教学改革工作的意见》精神编写的。全书包括数字电路基础、逻辑代数基础、组合逻辑电路、触发器、时序逻辑电路、脉冲波形的产生与变换、数/模与模/数转换、半导体存储器和可编程逻辑器件等 8 章内容。

本书具有以下特色：

1. 精炼内容，突出教材特点。

由于许多院校开设了“大规模集成电路设计”、“数字系统设计”、“硬件描述语言”等选修课，因此我们在本书中，对这些相关内容做了删减，使本书的篇幅得到了大幅度的压缩。由于课时的关系，教材中打有“*”的部分作为选学内容。

2. 注重学生能力的培养。

在每章小结中包括各章的重点内容、难点内容和需注意的问题，同时还给出了各章的例题精选及自我检测题，有助于学生抓住各章的重点、难点，提高学习效率和学习效果；在精选例题时，特别注重了题目的基础性、多样性、综合性和灵活性，并增加了一定的难度系数，以便提高学生分析问题及解决问题的能力。

3. 扩大教材的适用范围。

考虑到其他专业学生学习本课程的要求，本书补充了部分内容。如在时序电路的设计中，我们增加了利用隐含表进行状态化简的内容，以避免学生因某些知识的欠缺而导致学习上的困难。

本书的第 1 章由顾洁编写，第 2 章、第 8 章由王守华编写，第 4 章由李旭红编写，第 7 章由耿伟霞编写，第 3 章、第 5 章、第 6 章(除 6.5 节外)由董敏编写，第 6.5 节由吴文峰编写。全书由董敏统稿。

本书在编写过程中，还得到了程红丽、杨建翔、杨波、杨俊三、陈伟、薛颖轶及罗小莹等的支持，他们对本书的编写提供了许多宝贵的意见和有价值的资料，在此，对他们表示衷心的感谢。

限于编者的水平，书中的不足之处在所难免，敬请读者批评指正。

编　者

2010 年 5 月

目　录

第 1 章　数字电路基础

本章首先介绍有关数制和码制的一些基本概念和术语，然后给出数字电路中常用的数制和码制。此外，还具体介绍了不同数制之间的转换方法和二进制算术运算的原理和方法。

当我们观察自然界中各种物理量时不难发现，就其变化规律的特点而言，不外乎有两大类。一类是物理量的变化在时间上和数量上都是离散的。也就是说，它们的变化在时间上和数值上是不连续的，总是发生在一系列离散的瞬间。我们把这一类物理量称为数字量，把表示数字量的信号称为数字信号(参见图 1 (a))，并把工作在数字信号下的电路称为数字电路。例如：统计通过某一个桥梁的汽车数量，得到的就是一个数字量，最小数量单位的“1”代表一辆汽车，小于 1 的数值已经没有任何物理意义。数字信号在电路中常表现为突变的电压或电流。

另外一类是物理量的变化在时间上和数值上都是连续的信号。这一类物理量称为模拟量，把表示模拟量的信号称为模拟信号(参见图 1(b))，并把工作在模拟信号下的电子电路称为模拟电路。例如：热电偶工作时输出的电压或电流信号就是一种模拟信号，因为被测量的温度不可能发生突跳，所以测得的电压或电流无论在时间上还是数值上都是连续的。而且，这个信号在连续变化过程中的任何一个取值都有具体的物理意义，即表示一个相应的温度。

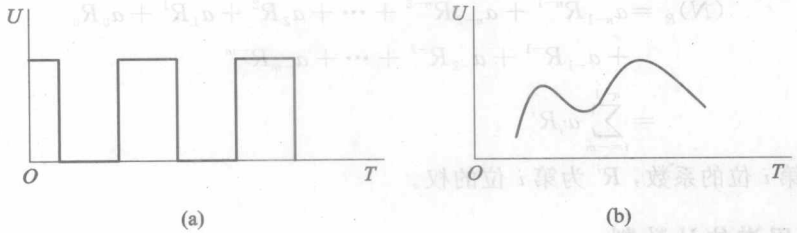


图 1　典型的数字信号与模拟信号

(a) 数字信号；(b) 模拟信号

随着计算机科学与技术的飞速发展，用数字电路进行信号处理的优势更加突出。为了充分发挥和利用数字电路在信号处理上的强大功能，可以先将模拟信号按比例转换成数字信号，然后送到数字电路进行处理，最后再将处理结果根据需要转换为模拟信号输出。

1.1　数　制

数字信号通常都是以数码的形式给出的。不同的数码可以用来表示数量的不同大小。用数码表示数量大小时，仅用一位数码往往是不够的，因此经常需要用进位计数制的方法

组成多位数码。我们把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。数制是人们对数量计数的一种统计规则。在数字系统中广泛采用的是二进制、八进制和十六进制。

1.1.1　进位数制的基本概念

进位数制也叫位置计数制，其计数方法是把数划分为不同的数位，当某一数位累计到一定数量之后，该位又从零开始，同时向高位进位。在这种计数制中，同一个数码在不同的数位上所表示的数值是不同的。进位计数制可以用少量的数码表示较大的数，因而被广泛采用。

进位计数包含着两个基本因素：进位基数和数位的权值。

进位基数：在一个数位上，规定使用的数码符号的个数叫该进位计数制的进位基数或进位模数，记作 R。例如十进制，每个数位规定使用的数码符号为 0，1，2，…，9，共 10 个，故其进位基数 $R=10$。

数位的权值：某个数位上数码为 1 时所表征的数值，称为该数位的权值，简称“权”。各个数位的权值均可表示成 R_i 的形式，其中 R 是进位基数，i 是各数位的序号。数位按如下方法确定：整数部分，以小数点为起点，自右向左依次为 0，1，2，…，$n-1$；小数部分，以小数点为起点，自左向右依次为 -1，-2，…，$-m$。n 是整数部分的位数，m 是小数部分的位数。

某个数位上的数码 a_i 所表示的数值等于数码 a_i 与该位的权值 R^i 的乘积。所以，R 进制的数

$$(N)_R = a_{n-1}a_{n-2}\cdots a_2a_1a_0 \cdot a_{-1}a_{-2}\cdots a_{-m} \tag{1.1.1}$$

又可以写成如下多项式的形式：

$$\begin{aligned}(N)_R &= a_{n-1}R^{n-1} + a_{n-2}R^{n-2} + \cdots + a_2R^2 + a_1R^1 + a_0R_0 \\ &\quad + a_{-1}R^{-1} + a_{-2}R^{-2} + \cdots + a_{-m}R^{-m} \\ &= \sum_{i=-m}^{n-1} a_iR^i\end{aligned} \tag{1.1.2}$$

式中：a_i 为第 i 位的系数，R^i 为第 i 位的权。

1.1.2　常用进位计数制

1. 十进制

十进制是最经常使用的进位计数制。在十进制中，每一位规定使用的数码为 0，1，2，…，9，共 10 个数码，故其进位基数 R 为 10。超过 9 的数必须用多位数表示，其中低位与相邻高位之间的关系为“逢十进一”。

十进制数用下标“D”(Decimal)表示，也可省略。例如：

$$(368.258)_D = 3\times10^2 + 6\times10^1 + 8\times10^0 + 2\times10^{-1} + 5\times10^{-2} + 8\times10^{-3}$$

十进制数中各位的权值为 10^i，i 是各数位的序号。所以任何一个十进制数 D 均可展开为

$$D = \sum k_i \times 10^i \tag{1.1.3}$$

其中，k_i 是第 i 位的系数，它可以是 0～9 这 10 个数码中的任何一个。若整数部分的位数是 n，小数部分的位数为 m，则 D 可以包含从 $n-1$ 到 0 的所有正整数和从 -1 到 $-m$ 的所有负整数。

2. 二进制

目前在数字电路中应用最广泛的是二进制。在二进制中，每个数位规定使用的数码仅为 0 和 1，共 2 个数码，所以其进位基数 R 为 2。低位与相邻高位之间的关系为“逢二进一”。各位的权值为 2^i，i 是各数位的序号。根据式(1.1.2)，任何一个二进制数均可展开为

$$D = \sum k_i \times 2^i \tag{1.1.4}$$

二进制数用下标“B”(Binary) 表示。例如：

$$(1011.01)_B = 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0 + 0\times 2^{-1} + 1\times 2^{-2}$$

二进制数只需两个状态便可表示，机器实现容易，但二进制书写太长。

3. 八进制

八进制中，每一位上规定使用的数码为 0～7 八个不同的数码，故其进位基数为 8。低位与相邻高位之间的关系为“逢八进一”。各位的权值为 8^i，i 是各数位的序号。任何一个八进制数均可展开为

$$D = \sum k_i \times 8^i \tag{1.1.5}$$

八进制数用下标“O”(Octal)表示。例如：

$$(152.47)_O = 1\times 8^2 + 5\times 8^1 + 2\times 8^0 + 4\times 8^{-1} + 7\times 8^{-2}$$

4. 十六进制

在十六进制中，每一位上有 16 个不同的数码，分别用 0～9、A(10)、B(11)、C(12)、D(13)、E(14)、F(15)表示。其进位基数为 16。低位与相邻高位之间的关系为“逢十六进一”。各位的权值为 16^i，i 是各个数位的序号。任何一个十六进制数均可展开为

$$D = \sum k_i \times 16^i \tag{1.1.6}$$

十六进制数用下标“H”(Hexadecimal)表示，例如：

$$(BD2.3C)_H = B\times 16^2 + D\times 16^1 + 2\times 16^0 + 3\times 16^{-1} + C\times 16^{-2}$$
$$= 11\times 16^2 + 13\times 16^1 + 2\times 16^0 + 3\times 16^{-1} + 12\times 16^{-2}$$

目前在微机系统中普遍采用 8 位、16 位和 32 位二进制并行运算，而 8 位、16 位和 32 位的二进制数可以用 2 位、4 位和 8 位的十六进制数表示，因而用十六进制符号书写程序十分简便。

表 1.1.1 是十进制数与等值二进制数、八进制数、十六进制数的对照表。

表 1.1.1　不同进制数的对照表

十进制	二进制	八进制	十六进制
00	0000	00	0
01	0001	01	1
02	0010	02	2
03	0011	03	3
04	0100	04	4
05	0101	05	5

续表

十进制	二进制	八进制	十六进制
06	0110	06	6
07	0111	07	7
08	1000	10	8
09	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

在计算机应用系统中，二进制主要用于机器内部的数据处理，八进制和十六进制主要用于书写程序，十进制主要用于运算最终结果的输出。

1.2 数制转换

1.2.1 十进制数转换成其他进制数

一个任意的十进制数可以由整数部分和小数部分构成，若设整数部分为 M_1，小数部分为 M_2，则

$$(M)_{10} = (M_1)_{10} + (M_2)_{10}$$

将它转换为 R 进制，根据式(1.1.2)

$$\begin{aligned}(M)_{10} &= \sum_{i=-m}^{n-1} a_i R^i \\ &= a_{n-1}R^{n-1} + a_{n-2}R^{n-2} + \cdots + a_2R^2 + a_1R^1 + a_0R^0 \\ &\quad + a_{-1}R^{-1} + a_{-2}R^{-2} + \cdots + a_{-m}R^{-m} \end{aligned} \tag{1.2.1}$$

于是整数部分为：

$$(M_1)_{10} = a_{n-1}R^{n-1} + a_{n-2}R^{n-2} + \cdots + a_2R^2 + a_1R^1 + a_0R^0 \tag{1.2.2}$$

小数部分为：

$$(M_2)_{10} = a_{-1}R^{-1} + a_{-2}R^{-2} + \cdots + a_{-m}R^{-m} \tag{1.2.3}$$

下面讨论如何确定 a_i 的值。

1. 整数部分转换

整数部分转换，采用基数连除法。把十进制整数 M 转换成 R 进制数的步骤如下：

(1) 将 M 除以 R，记下所得的商和余数。

(2) 将上一步所得的商再除以 R，记下所得商和余数。

(3) 重复做第(2)步，直到商为 0。

(4) 将各个余数转换成 R 进制的数码，并按照和运算过程相反的顺序把各个余数排列起来，即为 R 进制的数。

我们将这种方法取名为除以 R 取余法，逆序排列。

例 1.2.1　$(11)_D=(?)_B$

解

$$
\begin{array}{r|l}
2 & 11 \quad \cdots\cdots \quad 1 \quad \text{最低位} \\
\hline
2 & 5 \quad \cdots\cdots \quad 1 \\
\hline
2 & 2 \quad \cdots\cdots \quad 1 \\
\hline
2 & 1 \quad \cdots\cdots \quad 0 \\
\hline
 & 0 \quad \cdots\cdots \quad 1 \quad \text{最高位}
\end{array}
$$

即　$(11)_D=(1011)_B$

例 1.2.2　$(427)_D=(?)_O$

解

$$
\begin{array}{r|l}
8 & 427 \quad \text{余数} \\
\hline
8 & 53 \quad \cdots\cdots \quad 3 \quad \text{最低位} \\
\hline
8 & 6 \quad \cdots\cdots \quad 5 \\
\hline
 & 0 \quad \cdots\cdots \quad 6 \quad \text{最高位}
\end{array}
$$

即　$(427)_D=(653)_O$

例 1.2.3　$(427)_D=(?)_H$

解

$$
\begin{array}{r|l}
16 & 427 \quad \text{余数} \\
\hline
16 & 26 \quad \cdots\cdots \quad 11=B \quad \text{最低位} \\
\hline
16 & 1 \quad \cdots\cdots \quad 10=A \\
\hline
 & 0 \quad \cdots\cdots \quad 1=1 \quad \text{最高位}
\end{array}
$$

即　$(427)_D=(1AB)_H$

2. 小数部分转换

小数部分转换，采用基数连乘法。把十进制的纯小数 M 转换成 R 进制数的步骤如下：

(1) 将 M 乘以 R，记下整数部分。

(2) 将上一步乘积中的小数部分再乘以 R，记下整数部分。

(3) 重复做第(2)步，直到小数部分为 0 或者满足精度要求为止。

(4) 将各步求得的整数转换成 R 进制的数码，并按照和运算过程相同的顺序排列起来，即为所求的 R 进制数。

我们将这种方法取名为乘以 R 取整法，顺序排列。

例 1.2.4　$(0.85)_D=(?)_H$

解

$$
\begin{array}{l}
0.85\times16=13.6\cdots\cdots 13=D \quad \text{最高位} \\
0.6\times16=9.6\cdots\cdots \quad 9=9 \quad \downarrow \\
0.6\times16=9.6\cdots\cdots \quad 9=9 \quad \text{最低位}
\end{array}
$$

即　$(0.85)_D=(0.D99\cdots)_H$

例 1.2.5　$(0.35)_D=(?)_O$

解

$$0.35\times8=2.8\cdots\cdots\cdots\cdots2 \quad \text{最高位}$$
$$0.8\times8=6.4\cdots\cdots\cdots\cdots 6 \quad \downarrow$$
$$0.4\times8=3.2\cdots\cdots\cdots\cdots 3 \quad \downarrow$$
$$0.2\times8=1.6\cdots\cdots\cdots\cdots 1 \quad \text{最低位}$$

即　$(0.35)_D=(0.2631)_O$

1.2.2　非十进制数转换成十进制数

不同数制之间的转换方法有若干种。把非十进制数转换成十进制数采用按权展开相加法。具体步骤是：首先把非十进制数写成按权展开的多项式，然后按十进制数的计数规则求其和。

例 1.2.6　$(2A.8)_H=(?)_D$

解

$$(2A.8)_H=2\times16^1+A\times16^0+8\times16^{-1}$$
$$=32+10+1.5=(42.5)_D$$

例 1.2.7　$(165.2)_O=(?)_D$

解

$$(165.2)_O=1\times8^2+6\times8^1+5\times8^0+2\times8^{-1}$$
$$=64+48+5+0.25=(117.25)_D$$

例 1.2.8　$(10101.11)_B=(?)_D$

解

$$(10101.11)_B=1\times2^4+0\times2^3+1\times2^2+0\times2^1+1\times2^0+1\times2^{-1}+1\times2^{-2}$$
$$=16+0+4+0+1+0.5+0.25=(21.75)_D$$

1.2.3　二进制数与八进制数、十六进制数之间的转换

二进制数转换成八进制数(或十六进制数)时，其整数部分和小数部分可以同时进行转换。其方法是：以二进制数的小数点为起点，分别向左、向右，每三位(或四位)分一组。对于小数部分，最低位一组不足三位(或四位)时，必须在有效位右边补 0，使其足位。然后，把每一组二进制数转换成八进制(或十六进制)数，并保持原排序。对于整数部分，最高位一组不足位时，可在有效位的左边补 0，也可不补。

例 1.2.9　$(1011011111.10011)_B=(?)_O=(?)_H$

解

$$\overline{1011}\,\overline{011}\,\overline{111}.\overline{100}\,\overline{110}$$
$$1\quad3\quad3\quad7\quad.\quad4\quad6$$

所以　$(1011011111.10011)_B=(1337.46)_O$

$$\overleftarrow{1011}\,\overleftarrow{011}\,\overleftarrow{111}.\overrightarrow{100}\,\overrightarrow{11000}$$
$$2\quad D\quad F\quad.\quad9\quad8$$

即

$$(1011011111.10011)_B=(2DF.98)_H$$

例 1.2.10　$(36.24)_O=(?)_B$

解

$$(36.24)_O=(001\ 110.010\ 100)_B$$
$$\quad\ \ 3\quad 6\ .\ 2\quad 4$$

例 1.2.11　$(3DB.46)_H=(?)_B$

解　$(3DB.46)_H=(001111011011.01000110)_B$

$$3\quad D\quad B\ .\ 4\quad 6=(1111011011.0100011)_B$$

1.3　码　　制

不同的数码不仅可以用来表示数量的不同大小，而且可以用来表示不同的事物或事物的不同状态。在用于表示不同事物的情况下，这些数码已经不再具有表示数量大小的含义，它们只是不同事物的代号而已。例如食品外包装上的条形码。这些条形码仅仅代表不同的厂家生产的不同物品，没有数量大小的含义。

数字系统中用于表示数量或事物的数码称为代码。为了便于记忆和查找，在编制代码时要遵循一定的规则，这些规则就称为码制。每个人都可以根据自己的需要选定编码规则，编制出一组代码。当然，考虑到信息交换的需要，还必须制定出一些大家共同使用的通用代码。例如，目前国际上通用的美国信息交换标准代码(ASCII 码)就属于这一种。

用代码来表示给定的数字和信息符号的过程称为编码。信息符号可以是十进制数符 0，1，2，…，9；字符 A，B，…；运算符“+”，“−”，“=”等。

1.3.1　带符号数的代码表示

在数字系统中，通常使用二进制代码。在二进制代码中，直接表示正、负数的方法是：把一个数最高位作为符号位，用“0”表示“+”，用“1”表示“−”。符号位连同其后的数字位一起作为一个数，称为机器数，它表示的数值则称为这个机器数的真值。例如：

$$X_1=+0.1101;\ X_2=-0.1101$$

表示成机器数为：

$$X_1=0.1101;\ X_2=1.1101$$

在数字系统中，表示机器数的方法很多，目前常用的有原码、反码和补码。

1. 原码

原码表示法又称符号—数值表示法。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。例如：

$$X=-1101;\ (X)_{原}=11101$$

2. 反码

反码的符号表示与原码相同，正数反码的数值部分保持不变，而负数反码的数值是原码的数值按位求反。例如：

$$X_1=+1101$$

则

$$(X_1)_{反} = 01101 = 1101$$

$$X_2 = -1101$$

则

$$(X_2)_{反} = 10010$$

3. 补码

补码的符号表示与原码相同。正数补码的数值部分也与原码相同。负数的补码是这样得到的：将数值部分按位求反，再在最低位加 1。例如：

$$X_1 = +1101$$

则

$$(X_1)_{补} = (X_1)_{原} = (X_1)_{反} = 01101 = 1101$$

$$X_2 = -1101$$

则

$$(X_2)_{补} = 10011$$

1.3.2　带符号数的加、减运算

当两个二进制数码表示两个数量大小时，它们之间可以进行数值运算，这种运算称为算术运算。二进制算术运算和十进制算术运算的规则基本相同，唯一的区别在于二进制是“逢二进一”，而不是十进制数的“逢十进一”。

例如，两个二进制数 1001 和 0101 的算术运算有

加法运算

```
  1001
+0101
------
  1110
```

减法运算

```
  1001
-0101
------
  0100
```

乘法运算

```
    1001
  ×0101
  ------
    1001
   0000
  1001
 0000
 -------
 0101101
```

除法运算

```
        1.11…
      ------
0101 ) 1001
       0101
       ----
       1000
       0101
       ----
        0110
        0101
        ----
        0010
```

从上面的例子中可以看出，二进制算术运算有两个特点，即二进制数的乘法运算可以通过若干次的“被乘数(或零)左移 1 位”和“被乘数(或零)与部分积相加”这两种操作完成；而二进制的除法运算能通过若干次的“除数右移 1 位”和从“被除数或余数中减去除数”这两种操作完成。

如果我们再将减法操作转化为某种形式的加法运算，那么加、减、乘、除运算就全部

可以用“移位”和“相加”两种操作实现了。利用上述特点能使运算电路大大简化。这也是数字电路中普遍采用二进制数的重要原因之一。

在做减法运算时，如果两个数是用原码表示的，则首先需要比较两个数的绝对值大小，然后以绝对值大的一个作为被减数，绝对值小的作为减数，求出差值，并以绝对值大的一个数的符号作为差值的符号。不难看出，这个操作过程比较复杂，而且需要使用数值比较电路和减法运算电路。如果能用两数的补码相加代替上述的减法运算，那么计算过程中就无需使用数值比较电路和减法运算电路，从而使运算器的电路结构大为简化。

下面说明补码的运算原理。

我们先讨论一个生活中常见的事例。例如，你在 5 点钟的时候发现自己的手表停在 10 点上。要想调整手表有两种方法：第一种拨法是把时针往回拨 5 格，10－5＝5；另一种拨法是往前拨 7 格，10＋7＝17，由于表盘的最大数只有 12，超过 12 以后的“进位”将自动消失，于是就只剩下减去 12 的余数了，即 17－12＝5，也将指针拨回到了 5 点。这个例子说明，10－5 的减法运算可以用 10＋7 的加法运算代替。因为 5 和 7 相加正好等于进位的模数 12，所以我们称 7 为－5 对模 12 的补数，也称为补码。

从这个例子中可以得出一个结论，就是在舍去进位的条件下，减去某个数可以用加上它的补码来替代。这个结论同样也适用于二进制数的运算。

例如：1011－0111＝0100 的减法运算，在舍去进位的条件下，可以用 1011＋1001＝0100 的加法运算代替。因为 4 位二进制数的进位基数是 16(10000)，所以 1001(9)恰好是 0111(7)对模 16 的补码。

基于上述原理，对于有效数字(不包括符号位)为 n 位的二进制数 N，它的补码表示方法为

$$(N)_{\text{COMP}}=\begin{cases}N & (N\text{ 为正数})\\ 2^n-N & (N\text{ 为负数})\end{cases}\tag{1.3.1}$$

即正数(当符号位为 0 时)的补码与原码相同，负数(当符号位为 1 时)的补码等于 2^n-N，符号位保持不变。

为了避免在求补码的过程中做减法运算，通常是先求出 N 的反码$(N)_{\text{INV}}$，然后在负数的反码上加 1 而得到补码。二进制数的反码$(N)_{\text{INV}}$是这样定义的：

$$(N)_{\text{INV}}=\begin{cases}N & (N\text{ 为正数})\\ (2^n-1)-N & (N\text{ 为负数})\end{cases}\tag{1.3.2}$$

由上式可知，当 N 为负数时，$(N)_{\text{INV}}+N=2^n-1$，而 2^n-1 是 n 位全为 1 的二进制数，所以只要将 N 中每一位的 1 改为 0、0 改为 1，就得到了$(N)_{\text{INV}}$。

例 1.3.1　写出带符号位二进制数 00011010(＋26)、10011010(－26)、00101101(＋45)和 10101101(－45)的补码和反码。

解　根据式(1.3.1)和式(1.3.2)得到

原码	反码	补码
00011010	00011010	00011010
10011010	11100101	11100110
00101101	00101101	00101101
10101101	11010010	11010011