高等学校“十二五”重点规划教材
信 息 与 自 动 化 系 列

数字电子技术基础

（第2版）

主 编 张佳薇

1010101010001110101010

1100000 111100

HEUP哈尔滨工程大学出版社
Harbin Engineering University Press

数字电子技术基础

（第2版）

主　编　张佳薇

副主编　薛云峰　管雪梅

哈尔滨工程大学出版社

内容简介

本书在内容安排上，坚持"确保基础、精选内容、加强概念、侧重实践"的基本思想，结合新的课程体系和教学内容改革的要求编写而成。该教材的编写以集成电路为主，适当保留门电路和触发器方面的基本内容，把数字电路分析和设计的重点放到中规模集成电路上。重视集成电路的外特性，给出常用集成芯片的逻辑图和管脚分布，强调通过外特性来学习集成电路，提出通过功能框图来分析和设计较大规模数字电路，并引入数字系统概念。增加和充实了 EWB 电子仿真软件的讲解，使学生了解分析设计数字电路的新方法。

本书共分为 9 章，内容包括数字逻辑基础、门电路、组合逻辑电路、触发器、时序逻辑电路、半导体存储器、D/A 和 A/D 转换器、脉冲波形的产生与变换、EWB 软件教程。每章还配有一定数量的练习题，可供读者练习和思考。

本书内容取材合理，语言清楚简练，可作为高等院校的电子、通信、自动控制、电气、计算机等专业教材，还可以作为自学考试和从事电子技术的工程技术人员的学习参考书。

图书在版编目(CIP)数据

数字电子技术基础/张佳薇主编. —2 版. —哈尔滨:哈尔滨工程大学出版社,2011.8
ISBN 978 -7 -5661 -0209 -6

Ⅰ.①数… Ⅱ.①张… Ⅲ.①数字电路 - 电子技术 - 教材 Ⅳ.①TN79

中国版本图书馆 CIP 数据核字(2011)第 163200 号

出版发行 哈尔滨工程大学出版社
社　　址 哈尔滨市南岗区东大直街 124 号
邮政编码 150001
发行电话 0451 - 82519328
传　　真 0451 - 82519699
经　　销 新华书店
印　　刷 肇东市一兴印刷有限公司
开　　本 787mm×1 092mm　1/16
印　　张 17
字　　数 412 千字
版　　次 2009 年 2 月第 1 版　2011 年 8 月第 2 版
印　　次 2011 年 8 月第 1 次印刷
定　　价 35.00 元
http://press. hrbeu. edu cn
E-mail:heupress@ hrbeu. edu. cn

前　　言

本书从电子技术基础课程的定位出发，力图体现现代教育所要求的先进性、科学性和适用性。本着"确保基础、精选内容、加强概念、侧重实践"的基本思想，结合新的课程体系和教学内容改革的要求编写而成。

为了既满足教学内容的需要，又具有一定的实用性。该教材的编写以集成电路为主，适当保留门电路和触发器方面的基本内容，把数字电路分析和设计的重点移到中规模集成电路上。重视集成电路的外特性，给出常用集成芯片的逻辑图和管脚分布，强调通过外特性来学习集成电路，提出通过功能框图来分析和设计较大规模数字电路，并引入数字系统的概念。增加和充实了 EWB 电子仿真软件的讲解，使学生了解分析设计数字电路的新方法。

本书共分为九章。第 1 章数字逻辑基础，主要介绍数制和码制、逻辑代数、逻辑函数及化简；第 2 章门电路，主要介绍分立元件门电路、TTL 集成门电路和 CMOS 门电路；第 3 章组合逻辑电路，主要介绍组合逻辑电路的分析和设计方法、若干常用组合逻辑电路、组合逻辑电路中的竞争－冒险现象；第 4 章触发器，主要介绍触发器的电路结构与工作特点、触感器的逻辑功能及其描述方法；第 5 章时序逻辑电路，主要介绍同步、异步时序逻辑电路的分析和设计方法，以及几种常用的时序逻辑电路；第 6 章半导体存储器，主要介绍只读存储器和随机存储器的原理与构成；第 7 章数模转换器与模数转换器，主要介绍各种 D/A 和 A/D 转换器的原理和结构；第 8 章脉冲波形的产生与变换，主要介绍了施密特触发器、单稳态触发器和多谐振荡器的基本原理以及 555 定时器的应用；第 9 章 EWB 软件教程，介绍了电子设计自动化技术。

本书可作为本科生教材，同时也可为从事电子工程的工程师和参加各类电子制作竞赛的本科生提供有益的参考资料。

本书由东北林业大学、上海第二工业大学、上海工程技术大学三校联合编写。全书由张佳薇任主编，薛云峰、管雪梅任副主编。全书的第 1 章、第 2 章、第 3 章、第 5 章由张佳薇执笔，第 4 章、第 8 章由薛云峰、张佳薇执笔，第 6 章、第 9 章由管雪梅、王宇嘉执笔，第 7 章由薛云峰、管雪梅执笔。

作者在编写本教材的过程中，参阅了相关教材和专著，在此向各位原编著者致谢。

由于编者水平有限，书中难免存在一些缺点和错误，殷切希望广大读者批评指正。

编　者

2011 年 7 月

目　录

第1章　数字逻辑基础

1.1　绪　　论

1.1.1　数字技术的发展概况

电子技术的发展诞生了数字电路，而数字电路的进步又产生了计算机技术，现代计算技术的发展促进了科学技术的飞跃。现在，计算机已被广泛地应用于科学计算、数据处理、生产过程控制、军事科技、现代医疗技术和家用电器等领域中。计算机已经成为数字系统中最常见的、最有代表性的一种设备，是现代生活中不可缺少的工具。

数字电路系统是用数字来“处理”信息以实现计算和执行操作的电子网络。数字电路系统中所用的数字来自于一种特定的数制系统，该数制系统只有两个可能的值:0 或 1，这就是目前数字电路系统采用的二进制数制系统。每一位二进制数(0或1)称为 1 比特(bit)，简称为“二进制数字”。由于 1 位二进制数只有两个状态，而实际工作中可能出现的状态数很多，因此无法只用 1 位二进制数来完成所有的计算和操作，实际上是将若干位二进制数组合起来，如将 4 位二进制数划为 1 组，则这组二进制数将可以表示 16 种状态。现代的数字电路系统(如个人计算机)已有 8 位、16 位、32 位以至 64 位系统，位数越多，表示该系统处理的信息量越大。

数字系统是相当复杂的，它必须能完成如下任务：

- 将各种物理信息转换成数字系统可以理解的二进制“语言”；
- 仅用硬件电路来完成所要求的计算和操作；
- 将处理的结果以我们可以理解的物理形式返回给现实世界。

1.1.2　数字信号与数字电路

综观自然界中形形色色的物理量我们不难发现，尽管这些物理量性质各异，但就它们变化规律的特点而言，可以分为两大类。

有一类物理量的变化在时间上或数字上是连续的，我们把这一类物理量称为模拟量。用来表示模拟量的信号称为模拟信号，而处理模拟信号的电子电路叫做模拟电路。比如，当使用热电偶测量温度时，其输出的电压信号便是一个连续变化的模拟信号，在测量过程中任意时刻的电压取值都表示相应的温度。

另一类物理量的变化在数量上和时间上都是离散的。也就是说，它们的变化在时间上是不连续的，并且总是发生在一系列的离散瞬间。并且总是以某一个最小数量大小的整数倍的数值关系来增减变化。我们把这一类物理量称为数字量，用来表示数字量的信号叫做数字信号，而处理数字信号的电子电路叫做数字电路。通常，数字系统的信号只有两个离散量(0和1)，称为二进制信号。比如，开关的断开与闭合、晶体管的截止与导通、电路中的低电平与高电平等均为二进制信号。很明显，二进制信号容易被电路识别和处理，处理二进制

信号的数字系统往往具有极高的可靠性和稳定性。

信号的离散有的是自然形成的,有的则是将连续过程有意加以量化后而得到的。例如,科学工作者在科学研究工作中常常要观察连续的过程,但由于过程持续时间太长,要做到连续观察和记录是非常困难的,如天气形势的记录是不可能连续记录下来的,否则信息量太大,系统很难处理。于是,气象台按一定时间间隔将数据记录下来,列成表格,这样就将连续的信息量化了,表格中的每一个数字都成为离散量。

数字电路与模拟电路相比主要有下列优点:

(1)由于数字电路是以二值数字逻辑为基础的,只有0和1两个基本数字,易于用电路来实现,比如可用二极管、三极管的导通与截止这两个对立的状态来表示数字信号的逻辑0和逻辑1。

(2)由数字电路组成的数字系统工作可靠,精度较高,抗干扰能力强。它可以通过整形很方便地去除叠加于传输信号上的噪声与干扰,还可利用差错控制技术对传输信号进行查错和纠错。

(3)数字电路不仅能完成数值运算,而且能进行逻辑判断和运算,这在控制系统中是不可缺少的。

(4)数字信息便于长期保存,比如可将数字信息存入磁盘、光盘等长期保存。

(5)数字集成电路产品系列多、通用性强、成本低。

由于具有一系列优点,数字电路在电子设备或电子系统中得到了越来越广泛的应用,计算机、计算器、电视机、音响系统、视频记录设备、光碟、长途电信及卫星系统等,无一不采用数字系统。

1.2 数制与码制

数制是进位计数制度的简称。我们把多位数码中每一位的构成方法以及从低位到相邻高位的进位规则称为数制。日常生活中有许多不同的数制,例如,最早采用、也是使用最广泛的十进制,“逢十进一”;而钟表计时采用六十进制,如 60 秒为 1 分钟,60 分钟为 1 小时;12 个月为 1 年,则采用的是十二进制;也有采用二进制的,如 2 只筷子为 1 双等。中国古代的八卦也是采用二进制信息来表示的。

数字系统处理的是离散元素,而这些离散元素通常以二进制数的形式出现。为此,本章将介绍数制和码制的概念、表示方法、性质及其相互转换,以便为进一步学习以后各章打下基础。

1.2.1 数制

1. 十进制(Decimal)

自古以来,人们常采用十进制数来计数,它的基数为 10,每位数可用下列十个数码之一来表示,即 0,1,2,3,4,5,6,7,8,9。

十进制数的计数规律是“逢十进一”。也就是说,每位数累计不能超过 10,计满 10 就应向高位进 1。

当我们面前出现一个十进制数,如 512.32 时,就会立刻联想到:这个数的最左边第一位

为百位(5代表500),第二位为十位(1代表10),第三位为个位(2代表2),小数点后而第一位为十分位(3代表3/10),第二位为百分位(2代表2/100)。这里百、十、个、十分之一和百分之一都是10的n次幂,它取决于系数所在的位置,这个位置称之为“权”。十进制数512.32从左至右各位的权分别是$10^2,10^1,10^0,10^{-1},10^{-2}$。因此,将512.32按权展开的形式为

$$512.32=5\times10^2+1\times10^1+2\times10^0+3\times10^{-1}+2\times10^{-2}$$

等式左边的表示方法称为位置计数法,等式右边则是其按权展开式。

一般说来,对于任意一个十进制数S,可用位置计数法表示如下

$$(S)_{10}=(a_{n-1}a_{n-2}\cdots a_1a_0a_{-1}a_{-2}\cdots a_{-m})_{10} \tag{1-1}$$

也可用按权展开式表示如下

$$\begin{aligned}(S)_{10}&=a_{n-1}\times10^{n-1}+a_{n-2}\times10^{n-2}+\cdots+a_1\times10^1+a_0\times10^0+a_{-1}\times10^{-1}+\\&\quad a_{-2}\times10^{-2}+\cdots+a_{-m}\times10^{-m}\\&=\sum_{i=-m}^{n-1}a_i\times10^i\end{aligned} \tag{1-2}$$

式中,a_i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(Decimal)。

2. 二进制(Binary)

数字系统中最常使用的进位制是二进制。在二进制数中,每一位只有**0**和**1**两个码,所以计数的基数为2。二进制数的计数规则是每位计满2就向高位进1,即“逢二进一”。例如,**1001**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S,用位置计数法表示为

$$(S)_2=(a_{n-1}a_{n-2}\cdots a_1a_0a_{-1}a_{-2}\cdots a_{-m})_2 \tag{1-3}$$

用按权展开式表示为

$$\begin{aligned}(S_2)&=a_{n-1}\times2^{n-1}+a_{n-2}\times2^{n-2}+\cdots+a_1\times2^1+a_0\times2^0+a_{-1}\times2^{-1}+\\&\quad a_2\times2^{-2}+\cdots+a_{-m}\times2^{-m}\\&=\sum_{i=-m}^{n-1}a_i\times2^i\end{aligned} \tag{1-4}$$

式中,a_i为数码**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(Binary)。

3. 十六进制(Hexadecimal)与八进制(Octal)

由于二进制数简单、容易实现,所以它是数字系统中,特别是计算机中广泛采用的一种数制。但如用二进制表示一个十进制数时,需用四位二进制数才能表示一位十进制数,所用的位数比用十进制数表示的位数多,读写很不方便,因此在实际工作中通常采用八进制或十六进制。

八进制数的基数是8,采用的数码是0~7。计数规则是“逢八进一”,相邻两位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,所以为了便于区分,通常在数字的右下角标注8或O(Octal)。

十六进制数的基数为16,分别用0~9,A(10),B(11),C(12),D(13),E(14),F(15)表示。十六进制的计数规则是“逢十六进一”,相邻两位高位的权值是低位权值的16倍。例

如,数$(54AFSB)_{16}$就是一个十六进制数。通常在数字的右下角标注16或H(Hexadecimal)。

与二进制数一样,任意一个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式的形式表示。

一般说来,对于任意的数S,都能表示成以r为基数的r进制数,数S的表示方法也有两种形式,即位置计数法和按权展开式。

位置计数法

$$(S)_r = (a_{n-1}a_{n-2}\cdots a_1a_0a_{-1}a_{-2}\cdots a_{-m})_r \tag{1-5}$$

用按权展开式表示为

$$\begin{aligned}(S)_r &= a_{n-1}\times r^{n-1} + a_{n-2}\times r^{n-2} + \cdots + a_1\times r^1 + a_0\times r^0 + a_{-1}\times r^{-1} + \\ &\quad a_{-2}\times r^{-2} + \cdots + a_{-m}\times r^{-m} \\ &= \sum_{i=-m}^{n-1} a_i\times r^i\end{aligned} \tag{1-6}$$

式中,a_i为数码$0\sim(r-1)$数码中的任意一个;r为该进位制的基数;n为整数部分的位数;m为小数部分的位数。

r进制数的计数规则是"逢r进一"。

不同数制的各种数码见表1-1,该表列出了当r为10,2,8,16时,各种进位计数制中开始的16个自然数。

表1-1 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**	0	0
1	**0001**	1	1
2	**0010**	2	2
3	**0011**	3	3
4	**0100**	4	4
5	**0101**	5	5
6	**0110**	6	6
7	**0111**	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*

4. 几种常用数制之间的转换

二进制数的缺点是书写时位数较长，不便记忆和阅读。因此，通常选用八进制数和十六进制数。这两种数制相对位数较短，便于书写和阅读，容易记忆，且非常容易将它们转换成二进制数。

在计算机和其他数字系统中，最常用的是二进制数。而人们日常习惯于使用十进制数，所以在数据处理过程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，经处理后，再将二进制数的计算结果转换成人们习惯的十进制数。这样就存在一个不同数制之间的相互转换问题。

(1)二进制转换成十进制

例 1.1　将二进制数**10011.101**转换成十进制数。

解　将每一位二进制数乘以位权，然后相加，可得

$$\begin{aligned}(\mathbf{10011.101})_B &= 1\times2^4+0\times2^3+0\times2^2+1\times2^1+1\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}\\ &=(19.625)_D\end{aligned}$$

(2)十进制转换成二进制

可用“除 2 取余”法将十进制的整数部分转换成二进制。

例 1.2　将十进制数 23 转换成二进制数。

解　根据“除 2 取余”法的原理，按如下步骤转换

$$\begin{array}{r|l l l}
2 & 23 \cdots\cdots\cdots & \text{余}\mathbf{1} & b_0\\
2 & 11 \cdots\cdots\cdots & \text{余}\mathbf{1} & b_1\\
2 & 5 \cdots\cdots\cdots & \text{余}\mathbf{1} & b_2\\
2 & 2 \cdots\cdots\cdots & \text{余}\mathbf{0} & b_3\\
2 & 1 \cdots\cdots\cdots & \text{余}\mathbf{1} & b_4\\
 & 0 & &
\end{array}\quad \uparrow \text{读取次序}$$

则　$(23)_D=(\mathbf{10111})_B$

可用“乘 2 取整”的方法将任何十进制数的纯小数部分转换成二进制数。

例 1.3　将十进制数$(0.562)_D$转换成误差 ε 不大于 2^{-6} 的二进制数。

解　用“乘 2 取整”法，按如下步骤转换

$$\begin{array}{l c l}
 & \text{取整} & \\
0.562\times2=1.124 \cdots\cdots & \mathbf{1} & \cdots\cdots b_{-1}\\
0.124\times2=0.248 \cdots\cdots & \mathbf{0} & \cdots\cdots b_{-2}\\
0.248\times2=0.496 \cdots\cdots & \mathbf{0} & \cdots\cdots b_{-3}\\
0.496\times2=0.992 \cdots\cdots & \mathbf{0} & \cdots\cdots b_{-4}\\
0.992\times2=1.984 \cdots\cdots & \mathbf{1} & \cdots\cdots b_{-5}
\end{array}$$

由于最后的小数 0.984 >0.5，根据“四舍五入”的原则，b_{-6}应为 1。因此

$$(0.562)_D=(\mathbf{0.100011})_B$$

其误差 $\varepsilon<2^{-6}$。

(3)二进制转换成十六进制

由于十六进制基数为 16，而 $16=2^4$，因此，4 位二进制数就相当于 1 位十六进制数。因此，可用“4 位分组”法将二进制数转换为十六进制数。

例1.4 将二进制数**1001101.100111**转换成十六进制数

解 $(1001101.100111)_B=(0100\ 1101.1001\ 1100)_B=(4D.9C)_H$

同理,若将二进制数转换为八进制数,可将二进制数分为3位一组,再将每组的3位二进制数转换成一位八进制数即可。

(4)十六进制转换成二进制

由于每位十六进制数对应于4位二进制数,因此,十六进制数转换成二进制数,只要将每一位十六进制数变成4位二进制数,按位的高低依次排列即可。

例1.5 将十六进制数6E.3A5转换成二进制数。

解 $(6E.3A5)_H=(110\ 1110.0011\ 1010\ 0101)_B$

同理,若将八进制数转换为二进制数,只需将每一位八进制数变成3位二进制数,按位的高低依次排列即可。

(5)十六进制转换成十进制

可由"按权相加"法将十六进制数转换为十进制数。

例1.6 将十六进制数7A.58转换成十进制数。

解

$$\begin{aligned}(7A.58)_H&=7\times16^1+10\times16^0+5\times16^{-1}+8\times16^{-2}\\&=112+10+0.3125+0.03125=(122.34375)_D\end{aligned}$$

1.2.2 码制

不同的数码不仅可以表示数量的不同大小,而且还能用来表示不同的事物。在后一种情况下,这些数码已经没有表示数量大小的含义,只是表示不同事物的代码而已。由于数字系统是以二值数字逻辑为基础的,因此数字系统中的信息(包括数值、文字、控制命令等)都是用一定位数的二进制码表示的,这个二进制码称为代码。

为了便于记忆和处理,在编制代码时总要遵循一定的规则,这些规则就叫做码制。

二进制编码方式有多种,二-十进制码,又称BCD码(Binary-Coded-Decimal),是其中一种常用的码。

BCD码——用二进制代码来表示十进制的0~9十个数。

要用二进制代码来表示十进制的0~9十个数,至少要用4位二进制数。4位二进制数有16种组合,可从这16种组合中选择10种组合分别来表示十进制的0~9十个数。选其中10种组合有多种方案,这就形成了不同的BCD码。具有一定规律的常用的BCD码见表1-2。

表1-2 常用BCD码

十进制数	8421码	2421码	5421码	余三码	余三循环码
0	0000	0000	0000	0011	0010
1	0001	0001	0001	0100	0110
2	0010	0010	0010	0101	0111
3	0011	0011	0011	0110	0101
4	0100	0100	0100	0111	0100
5	0101	1011	1000	1000	1100
6	0110	1100	1001	1001	1101
7	0111	1101	1010	1010	1111

表1-2(续)

十进制数	8421码	2421码	5421码	余三码	余三循环码
8	1 0 0 0	1 1 1 0	1 0 1 1	1 0 1 1	1 1 1 0
9	1 0 0 1	1 1 1 1	1 1 0 0	1 1 0 0	1 0 1 0
权	8 4 2 1 $b_3b_2b_1b_0$	2 4 2 1 $b_3b_2b_1b_0$	5 4 2 1 $b_3b_2b_1b_0$	无权	无权

注意,BCD码用4位二进制码表示的只是十进制数的一位。如果是多位十进制数,应先将每一位用BCD码表示,然后组合起来。

例1.7 将十进制数83分别用8421码、2421码和余3码表示。

解 由表1-2可得

$(83)_D=(1000\ 0011)_{8421}$

$(83)_D=(1110\ 0011)_{2421}$

$(83)_D=(1011\ 0110)_{余3}$

还有一种常用的四位无权码叫做格雷码(Gray),其编码如表1-3所示。这种码看似无规律,但它是按照"相邻性"编码的,即相邻两码之间只有一位数字不同。格雷码常用于模拟量的转换中,当模拟量发生微小变化而可能引起数字量发生变化时,格雷码仅改变1位,这样与其他码同时改变两位或多位的情况相比更为可靠,可减少出错的可能性。

表1-3 格雷码

十进制数	$G_3\ G_2\ G_1\ G_0$
0	0 0 0 0
1	0 0 0 1
2	0 0 1 1
3	0 0 1 0
4	0 1 1 0
5	0 1 1 1
6	0 1 0 1
7	0 1 0 0
8	1 1 0 0
9	1 1 0 1
10	1 1 1 1
11	1 1 1 0
12	1 0 1 0
13	1 0 1 1
14	1 0 0 1
15	1 0 0 0

1.2.3 算术运算和逻辑运算

在数字电路中,1位二进制数码1和0不仅可以表示数量的大小,而且也可以表示两种不同的逻辑状态。例如,我们既可以用1和0来表示电路的通和断、电灯的亮和暗这些直观的物理现象,也可以表示一件事情的是和非、好和坏、真和假、有和无等逻辑问题。我们称这些只有两种对立逻辑状态的逻辑关系为二值逻辑。

当用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,这种运算就称为算术运算,算术运算的结果是得到一个数量(算术)值。当两个二进制数码表示两个不同的逻辑状态时,它们之间可以按照规定的某种因果关系进行逻辑运算。逻辑运算的结果将得到一个逻辑值。逻辑运算和算术运算有根本的不同。下面我们先介绍二进制数的算术运算。

在数字电路中为什么选择使用二进制数制呢?这是因为用二进制数来表示数字和进行运算具有如下特点。

(1)二进制数只有0和1两个数码,任何具有两种不同稳定状态的元件都可用来表示1位二进制数,如晶体管的导通和截止、脉冲信号的有和无、灯的亮和灭等。

(2)二进制运算规则与其他数制的运算规则相比最简单,其运算规则具体如下:

加法规则 **0+0=0** **0+1=1**

$1+0=1$　　$1+1=0$　(同时向相邻高位进1)

减法规则　$0-0=0$　　$0-1=1$　(同时向相邻高位借1)

$1-0=1$　　$1-1=0$

乘法规则　$0\times0=0$　　$0\times1=0$

$1\times0=0$　　$1\times1=1$

除法规则　$0\div1=0$　　$1\div1=1$

下面举几个二进制数运算的例子。

例1.8　对1001+1011进行加法运算。

解

$$\begin{array}{r} 1001 \\ +\ 1011 \\ \hline 10100 \end{array}$$

由此可见,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,即任何位数累计到2时,本位就记为0,且向相邻高位进1。

例1.9　对10100-1110进行减法运算。

解

$$\begin{array}{r} 10100 \\ -\ 1110 \\ \hline 110 \end{array}$$

在二进制数减法中采用了"借一当二"的法则,减法运算从低位起按位进行,在遇到0减1时,就要向相邻高位借1,也就是从相邻高位减去1。

例1.10　对1011×1001进行乘法运算。

解

$$\begin{array}{r} 1011 \\ \times\ 1001 \\ \hline 1011 \\ 0000 \\ 0000 \\ +\ 1011 \\ \hline 1100011 \end{array}$$

从二进制数的乘法运算过程可以看出,二进制数的乘法运算和十进制数的乘法运算相似,只不过对乘积部分进行累加时要按照"逢二进一"的原则来运算。

例1.11　对10100100÷1001进行除法运算。

解

$$\begin{array}{rl} 10010 & \text{——商} \\ 1001\overline{)10100100} & \\ \underline{1001} & \\ 1010 & \\ \underline{1001} & \\ 10 & \text{——余数} \end{array}$$

从二进制数的除法运算过程可以看出,二进制数的除法运算同十进制数的除法运算相类似,但采用二进制数的除法运算规则。

在数字电路和数字电子计算机中,二进制数的正、负号也是用0和1表示的。在定点运算的情况下,以最高位作为符号位,正数为0,负数为1。以下各位为0和1表示数值。用这种方

式表示的数码称为原码。例如

$$(\boxed{0}\,1011101)_2=(+93)_{10}$$

符号位

$$(\boxed{1}\,1011101)_2=(-93)_{10}$$

符号位

为了简化运算电路,在数字电路中两数相减的运算是用它们的补码相加来完成的。二进制数的补码是这样定义的:

(1)最高位为符号位,正数为0,负数为1;

(2)正数的补码和它的原码相同;

(3)负数的补码可以通过将原码的数值位逐位求反,然后在最低位上加1得到。

例 计算$(1001)_2-(0101)_2$

解 根据二进制的运算规则可知

$$\begin{array}{r} 1001 \\ -\ 0101 \\ \hline 0100 \end{array}$$

在采用补码运算时,首先求出$(+1001)_2$和$(-0101)_2$的补码,它们是

$$[+1001]_{补}=\boxed{0}\,1001$$

符号位

$$[-0101]_{补}=\boxed{1}\,1011$$

符号位

然后将两个补码相加并舍去进位

$$\begin{array}{r} 01001 \\ +\ 11011 \\ \hline 舍去\leftarrow 100100 \end{array}$$

则得到与前面相同的结果。这样就把减法运算转化成了加法运算。

1.3 逻辑代数及其基本逻辑运算

1849年,英国数学家乔治·布尔(George Boole)首先提出了描述客观事物逻辑关系的数学方法——布尔代数。后来,由于布尔代数被广泛地应用于解决开关电路和数字逻辑电路的分析与设计上,所以也把布尔代数叫做开关代数或逻辑代数。数字电路实现的是逻辑关系,逻辑关系是指某事物的条件(或原因)与结果之间的关系。

1.3.1 基本逻辑运算

逻辑代数中只有三种基本运算:与、或、非。

1. 与运算

与运算——只有当决定一件事情的条件全部具备之后,这件事情才会发生。我们把这种因果关系称为与逻辑。

(1)可以用列表的方式表示上述逻辑关系,称为真值表。

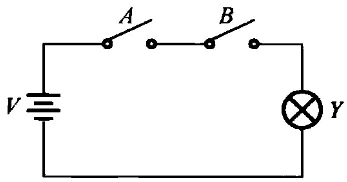
(2)如果用二值逻辑0和1来表示,并设1表示开关闭合或灯亮;0表示开关不闭合或灯不亮,则得到如图 1 -1(c)所示的表格,称为逻辑真值表。

(3)若用逻辑表达式来描述,则可写为

$$Y=A\cdot B$$

与运算的规则为“有0得0,全1得1”。

(4)在数字电路中能实现与运算的电路称为与门电路,其逻辑符号如图 1 -1(d)所示。



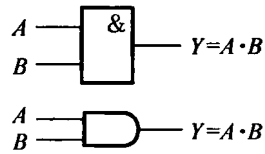
(a)

A	B	灯Y
不闭合	不闭合	不亮
不闭合	闭合	不亮
闭合	不闭合	不亮
闭合	闭合	亮

(b)

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

(c)



(d)

图 1 -1 与逻辑运算

(a)电路图;(b)真值表;(c)逻辑真值表;(d)逻辑符号

与运算可以推广到多变量:$Y=A\cdot B\cdot C\cdots$

2. 或运算

或运算——决定一件事情的几个条件中,只要有一个或一个以上条件具备,这件事情就会发生。我们把这种因果关系称为**或逻辑**。

或运算的真值表如图 1 -2(b)所示,逻辑真值表如图 1 -2(c)所示。若用逻辑表达式来描述,则可写为

$$Y=A+B$$

或运算的规则为“有1得1,全0得0”。

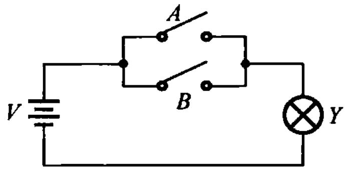
在数字电路中能实现或运算的电路称为或门电路,其逻辑符号如图 1 -2(d)所示。或运算也可以推广到多变量:$Y=A+B+C\cdots$

3. 非运算

非运算——某事情发生与否,仅取决于一个条件,而且是对该条件的否定。即条件具备时事情不发生;条件不具备时事情才发生。

例如图 1 -3(a)所示的电路,当开关 A 闭合时,灯不亮;而当 A 不闭合时,灯亮。其真值表如图 1 -3(b)所示,逻辑真值表如图 1 -3(c)所示。若用逻辑表达式来描述,则可写为

$$Y=\overline{A}$$



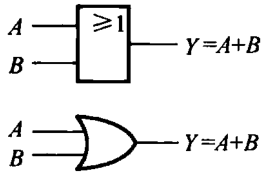
(a)

开关A	开关B	灯Y
不闭合	不闭合	不亮
不闭合	闭合	亮
闭合	不闭合	亮
闭合	闭合	亮

(b)

A	B	$Y=A+B$
0	0	0
0	1	1
1	0	1
1	1	1

(c)



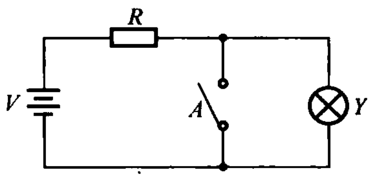
(d)

图 1－2　或逻辑运算

(a)电路图;(b)真值表;(c)逻辑真值表 ;(d)逻辑符号

非运算的规则为:$\overline{0}=1$;$\overline{1}=0$。

在数字电路中实现非运算的电路称为非门电路,其逻辑符号如图 1－3(d)所示。



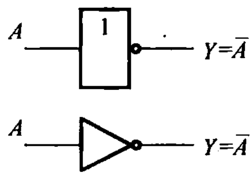
(a)

开关A	灯Y
不闭合	亮
闭合	不亮

(b)

A	$Y=\overline{A}$
0	1
1	0

(c)



(d)

图 1－3　非逻辑运算

(a)电路图;(b)真值表;(c)逻辑真值表;(d)逻辑符号

1.3.2　其他常用复合逻辑运算

任何复杂的逻辑运算都可以由这三种基本逻辑运算组合而成。在实际应用中为了减少逻辑门的数目,使数字电路的设计更方便,还常常使用其他几种常用复合逻辑运算,如与非、或非、与或非、同或、异或等。