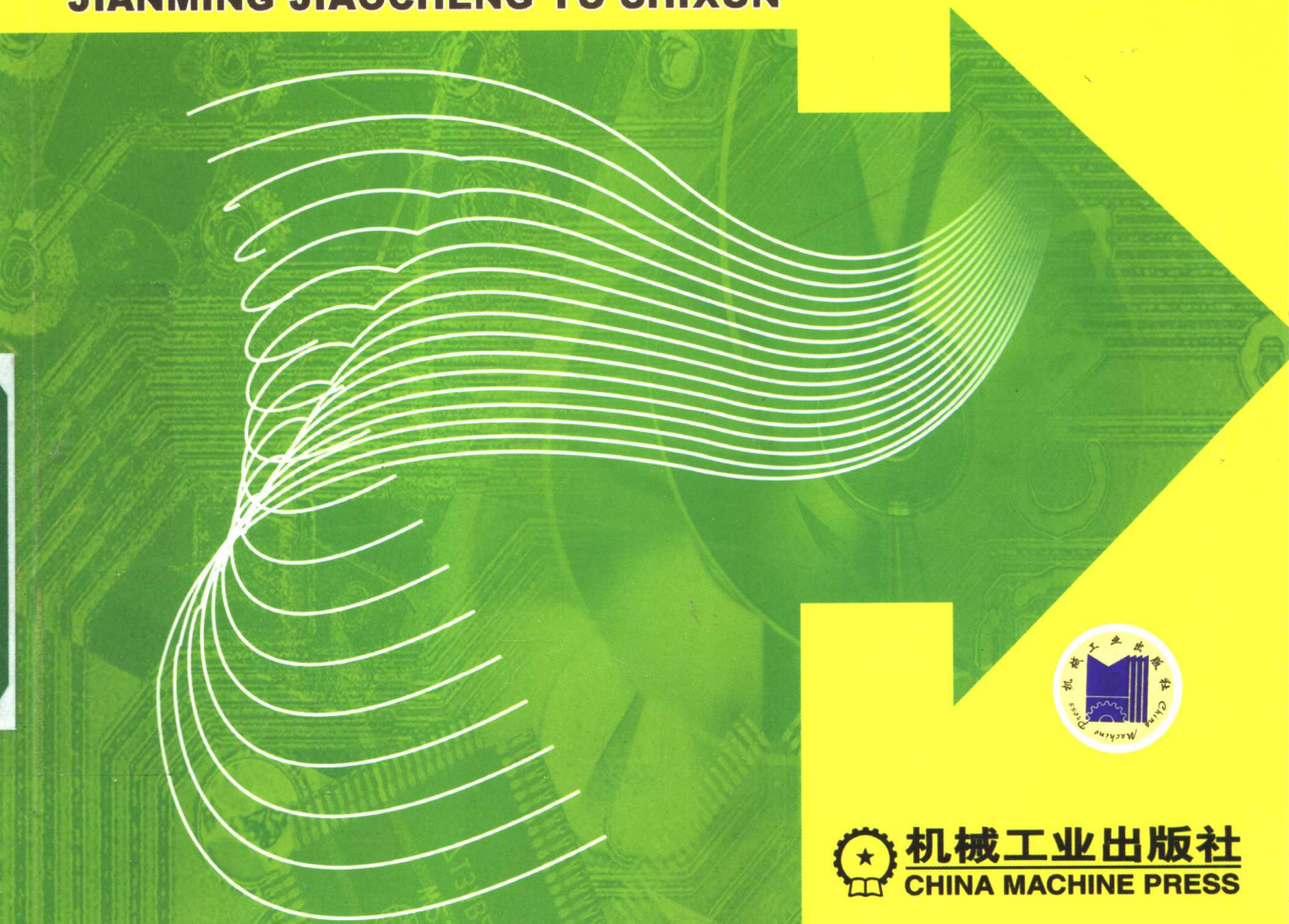
数字电子技术简明教程与实训

贾海瀛◎主编

SHUZI DIANZI JISHU
JIANMING JIAOCHENG YU SHIXUN



机械工业出版社
CHINA MACHINE PRESS

本书注重应用实践和基本技能的训练，内容主要包括数字逻辑基础、集成逻辑门、组合逻辑门电路、集成触发器、时序逻辑电路、脉冲单元电路、数/模和模/数转换器、存储器和可编程逻辑器件、课程设计与制作，并配有练习题、实验与技能操作训练。本书加强了FPGA、CPLD、MAX+plusⅡ软件、VHDL的介绍与应用。

本书特别适合于应用电子技术、信息工程、自动控制、仪器仪表、机电一体化、通信、音响工程及电子电气类各专业作为高职高专教材使用，也可供中等专业学校师生、工程技术人员及自学者参考。

图书在版编目（CIP）数据

数字电子技术简明教程与实训/贾海瀛主编. —北京：机械工业出版社，2008.1

ISBN 978-7-111-23115-8

Ⅰ.数… Ⅱ.贾… Ⅲ.数字电路—电子技术—教材 Ⅳ.TN79

中国版本图书馆CIP数据核字（2007）第195579号

机械工业出版社（北京市百万庄大街22号 邮政编码100037）

策划编辑：靳 平 责任编辑：靳 平 版式设计：霍永明

责任校对：王 欣 封面设计：马精明 责任印制：李 妍

保定市中画美凯印刷有限公司印刷

2008年2月第1版第1次印刷

184mm×260mm·12印张·292千字

0001-4000册

标准书号：ISBN 978-7-111-23115-8

定价：26.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

销售服务热线电话：（010）68326294

购书热线电话：（010）88379639 88379641 88379643

编辑热线电话：（010）88379768

封面无防伪标均为盗版

前　言

本书是为满足当前数字电子技术发展的迫切需要而编写的。全书分为9章，内容涉及理论知识、实际应用、习题、实验与技能操作，并有1章课程设计与制作。该书可用于所有开设数字电子技术的专业。各专业可依照本专业的学时数情况，全部或部分选用本书所列内容。

在本书编写过程中，我们特别注意了以下几个问题：

1）所选内容力求符合高职高专现行数字电子课程理论和实践教学的需要。内容全面覆盖电类专业的教学要求，非电类专业可选用其中部分内容。

2）努力反映现代数字电子技术的新技术、新成果。教材中增加了可编程逻辑器件的内容，同时较适当地加重了集成器件及其应用方面的内容，简化原理介绍，使本书尽可能跟上数字电子技术领域的新发展。

3）考虑到各个学校现有的实验设备不尽相同，本教材中对各个实验内容要求及所用仪器等部分进行了“通用化”处理，以求摆脱只能在一种设备上进行实验的限制，从而体现了“实验原理通用、仪器设备通用”的原则。实训及课程设计部分也采用了各学校都能具备的常用仪器。因此，本教材的适用面较广。

本书第1、2章由李莉编写；第3、7、8、9章由贾海瀛编写；第4、5、6章由孟庆杰编写；附录和部分实验由张悦旺编写。全书由贾海瀛拟定了编写大纲、统稿、定稿并担任主编，孟庆杰、李莉担任副主编。此外，马燕、李英、郝智也参与了本书的编写工作。

在本书编写过程中，承蒙天津职业大学李雅轩教授、付植桐教授、罗月红老师、其他电子技术任课教师的大力支持和帮助，在此一并表示衷心的感谢。

由于时间仓促，加之作者水平有限，书中难免有错误和不妥之处，恳请读者批评指正。

编　者

目　录

第 1 章　数字逻辑基础

1.1　概述

1. 模拟量和数字量

自然界中的物理量千变万化，但按照变化规律可以将物理量分成模拟量和数字量两大类。在时间和数值上都是连续的量称为模拟量，如声音、气温、压力等。在时间和数值上都是离散的量称为数字量，如生产线上对零件计数、学校的学生人数统计等。

2. 数字信号和数字电路

表示数字量的信号称为数字信号，如果电子电路中的信号为数字信号，则此电路称为数字电路。

在数字电路中数字信号只有 0 和 1 两种数值，0 和 1 没有大小之分，只用来表示两种对立的状态，如表示电压的高低、脉冲的有无、开关的通断等。

数字电路具有误差小、抗干扰性强、精度高、数据易保存等优点。

1.2　数制

数制是计数进位制的简称。日常生活中最常用的是十进制计数法，而数字电路和计算机中主要采用二进制和十六进制计数，另外还有八进制计数。

1. 十进制数

在十进制中，有 0～9 十个数字，计数时“逢十进一，借一当十”。任意一个十进制数 N 可展开为

$$\begin{aligned}(N)_{10}=&K_{n-1}\times10^{n-1}+K_{n-2}\times10^{n-2}+\cdots+K_1\times10^1+K_0\times10^0+\\&K_{-1}\times10^{-1}+K_{-2}\times10^{-2}+\cdots+K_{-m}\times10^{-m}\\=&\sum_{i=-m}^{n-1}K_i\times10^i\end{aligned}\tag{1-1}$$

式中，K_i 为第 i 位的系数，可以取 0～9 中的任何一个；10 为基数；10^i 为第 i 位的位权；n 为 N 的整数部分位数，m 为 N 的小数部分位数。例如，十进制数 256.18 可表示成

$$(256.18)_{10}=2\times10^2+5\times10^1+6\times10^0+1\times10^{-1}+8\times10^{-2}$$

下角标 10 表示十进制数，有时用 D（Decimal）表示。

2. 二进制数

二进制数只有 0 和 1 两个数，计数时“逢二进一，借一当二”。任意一个二进制数 N 可展开为

$$\begin{aligned}(N)_2=&K_{n-1}\times2^{n-1}+K_{n-2}\times2^{n-2}+\cdots+K_1\times2^1+K_0\times2^0+\\&K_{-1}\times2^{-1}+K_{-2}\times2^{-2}+\cdots+K_{-m}\times2^{-m}\end{aligned}$$

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

式中，K_i为第 i 位的系数，可以取 0 或 1 中的任何一个；2 为基数；2^i为第 i 位的位权；n 为 N 的整数部分位数，m 为 N 的小数部分位数。例如二进制数 101.01 可表示成

$$(101.01)_2 = 1\times2^2+0\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2}$$

下角标 2 表示二进制数，有时用 B（Binary）表示。

3. 十六进制数和八进制数

二进制数位数多，不便于书写和记忆，因而常把二进制数转换成十六进制数或八进制数。

十六进制数有 0～9、A、B、C、D、E、F 16 个数，计数时“逢十六进一，借一当十六”。任意一个十六进制数 N 可展开为

$$\begin{aligned}(N)_{16} &= K_{n-1}\times16^{n-1}+K_{n-2}\times16^{n-2}+\cdots+K_1\times16^1+K_0\times16^0+\\ &\quad K_{-1}\times16^{-1}+K_{-2}\times16^{-2}+\cdots+K_{-m}\times16^{-m}\\ &= \sum_{i=-m}^{n-1} K_i\times16^i\end{aligned} \tag{1-3}$$

式中，K_i为第 i 位的系数；16 为基数；16^i为第 i 位的位权；n 为 N 的整数部分位数，m 为 N 的小数部分位数。例如十六进制数 3C.2A 可表示成

$$(3C.2A)_{16} = 3\times16^1+C\times16^0+2\times16^{-1}+A\times16^{-2}$$

下角标 16 表示十六进制数，有时用 H（Hexadecimal）表示。

八进制数由 0～7 这八个数组成，计数时“逢八进一，借一当八”。任意一个八进制数 N 可展开为

$$\begin{aligned}(N)_8 &= K_{n-1}\times8^{n-1}+K_{n-2}\times8^{n-2}+\cdots+K_1\times8^1+K_0\times8^0+\\ &\quad K_{-1}\times8^{-1}+K_{-2}\times8^{-2}+\cdots+K_{-m}\times8^{-m}\\ &= \sum_{i=-m}^{n-1} K_i\times8^i\end{aligned} \tag{1-4}$$

式中，K_i为第 i 位的系数，可取 0～7 中的任何一个；8 为基数；8^i为第 i 位的位权；n 为 N 的整数部分位数，m 为 N 的小数部分位数。例如八进制数 17.52 可表示成

$$(17.52)_8 = 1\times8^1+7\times8^0+5\times8^{-1}+2\times8^{-2}$$

下角标 8 表示八进制数，有时用 O（Octal）表示。

4. 数制转换

（1）将 R 进制数转换成十进制数

这里的 R 进制可以是二进制、十六进制或八进制。将 R 进制数转换成十进制数时，只需要将 R 进制数按位权展开再相加即可得到十进制数。

例 1-1 将二进制数 $(1011.101)_2$转换成十进制数。

解： $(1011.101)_2 = 1\times2^3+0\times2^2+1\times2^1+1\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}=(11.625)_{10}$

例 1-2 将十六进制数 $(2A.18)_{16}$转换成十进制数。

解： $(2A.18)_{16}=2\times16^1+A\times16^0+1\times16^{-1}+8\times16^{-2}=(42.09375)_{10}$

例 1-3 将八进制数 $(205.45)_8$转换成十进制数。

解： $(205.45)_8=2\times8^2+0\times8^1+5\times8^0+4\times8^{-1}+5\times8^{-2}=(133.578125)_{10}$

（2）将十进制数转换成 R 进制数

将十进制数转换成 R 进制数，需要将整数部分和小数部分分别进行转换，然后再合并。

整数部分的转换步骤如下：①将十进制整数除以 R，余数作为 R 进制数的最低位；②把前一步的商再除以 R，余数作为次低位；③重复步骤（2），直至商为 0，最后的余数作为最高位。

例 1-4　将十进制数 $(23)_{10}$ 转换成二进制数。

解： 将 23 除以 2 取余数，即

	商	余数
2⌊23		
2⌊11	↵	1 最低位
2⌊5		1
2⌊2		1
2⌊1		0
0		1 最高位

所以　$(23)_{10}=(10111)_2$

例 1-5　将十进制数 $(100)_{10}$ 转换成八进制数。

解： 将 100 除以 8 取余数，即

	商	余数
8⌊100		
8⌊12	↵	4 最低位
8⌊1		4
0		1 最高位

所以　$(100)_{10}=(144)_8$

例 1-6　把十进制数 3988 转换成十六进制数。

解： 将 3988 除以 16 取余数，即

	商	余数
16⌊3988		
16⌊249	↵	4 最低位
16⌊15		9
0		F 最高位

所以　$(3988)_{10}=(F94)_{16}$

小数部分的转换步骤如下：①小数部分乘以 R，乘积的整数作为 R 进制的最高位；②乘积的小数部分继续乘以 R，乘积的整数作为 R 进制的次高位；③重复步骤②，直至乘积的小数部分为 0 或达到一定的精度为止，最后的整数作为 R 进制的最低位。

例 1-7　将十进制数 $(0.125)_2$ 转换成二进制数。

解： 将 0.125 乘以 2 取整数。

$$
\begin{array}{r}
0.125 \\
\times\quad 2 \\
\hline
[0].25 \\
\times\quad 2 \\
\hline
[0].5 \\
\times\quad 2 \\
\hline
[1].0
\end{array}
$$

所以　　$(0.125)_{10}=(0.001)_2$

例 1-8　将十进制数 $(0.3125)_2$转换成八进制数和十六进制数。

解：将 0.3125 分别乘以 8 和 16 取整数。

$$\begin{array}{r} 0.3125 \\ \times\quad 8 \\ \hline [2].5 \\ \times\quad 8 \\ \hline [4].0 \end{array} \qquad \begin{array}{r} 0.3125 \\ \times\quad 16 \\ \hline [5].0 \end{array}$$

所以　　$(0.3125)_{10}=(0.24)_8=(0.5)_{16}$

（3）二进制数与十六进制数的相互转换

二进制数转换为十六进制数时，整数和小数部分应分别转换。以小数点为基准，向左、右每 4 位为一组，整数部分的最高位组若不够 4 位，则在最高位前补 0；小数部分的最后一组若不够 4 位，则在最低位后补 0。然后再将每组 4 位二进制数转换成 1 位十六进制数。

例 1-9　将二进制数 $(1011010.011)_2$转换成十六进制数。

解：

$$\begin{array}{cccc} (0101 & 1010. & 0110)_2 \\ \downarrow & \downarrow & \downarrow \\ 5 & A & 6 \end{array}$$

所以　　$(1011010.011)_2=(5A.6)_{16}$

十六进制数转换为二进制数的过程相反，把每个十六进制数转换为 4 位二进制数。

例 1-10　将十六进制数 $(3C8.A9)_{16}$转换成二进制数。

解：

$$\begin{array}{ccccccc} (& 3 & C & 8 & . & A & 9 &)_{16} \\ & \downarrow & \downarrow & \downarrow & & \downarrow & \downarrow & \\ & 0011 & 1100 & 1000 & & 1010 & 1001 & \end{array}$$

所以　　$(3C8.A9)_{16}=(1111001000.10101001)_2$

（4）二进制数与八进制数的相互转换

二进制数转换为八进制数时，整数和小数部分要分别转换。以小数点为基准，向左、右每 3 位为一组，整数部分的最高位组若不够 3 位，则在最高位前补 0；小数部分的最后一组若不够 3 位，则在最低位后补 0。然后再将每组 3 位二进制数转换成 1 位八进制数。

例 1-11　将二进制数 $(1011010.01)_2$转换成八进制数。

解：

$$\begin{array}{cccccc} (001 & 011 & 010. & 010)_2 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 1 & 3 & 2 & 2 \end{array}$$

所以　　$(1011010.01)_2=(132.2)_8$

八进制数转换为二进制数的过程相反，把每个八进制数转换为 3 位二进制数。

例 1-12　将八进制数 $(53.24)_8$转换成二进制数。

解：

$$\begin{array}{ccccccc} (& 5 & 3 & . & 2 & 4 &)_8 \\ & \downarrow & \downarrow & & \downarrow & \downarrow & \\ & 101 & 011 & & 010 & 100 & \end{array}$$

所以　　$(53.24)_8 = (101011.0101)_2$

（5）十六进制数与八进制数的相互转换

根据十六进制数和八进制数与二进制数的关系，可将十六进制数（八进制数）先转换成二进制数，再转换成八进制数（十六进制数）。

例 1-13　将十六进制数 $(2A5F.6C)_{16}$ 转换成八进制数。

解：先将十六进制数转换成二进制数

$$(\ 2\quad A\quad 5\quad F\ .\ 6\quad C\ )_{16}$$
$$\downarrow\quad\downarrow\quad\downarrow\quad\downarrow\quad\downarrow\quad\downarrow$$
$$0010\quad 1010\quad 0101\quad 1111\quad 0110\quad 1100$$

即　　$(2A5F.6C)_{16} = (10101001011111.01101100)_2$

然后再将二进制数转换成八进制数

$$(010\ 101\ 001\ 011\ 111.\ 011\ 011)_2$$
$$\downarrow\quad\downarrow\quad\downarrow\quad\downarrow\quad\downarrow\quad\downarrow\quad\downarrow$$
$$2\quad 5\quad 1\quad 3\quad 7\quad 3\quad 3$$

所以　　$(2A5F.6C)_{16} = (25137.33)_8$

例 1-14　将八进制数 $(26.135)_8$ 转换成十六进制数。

解：先将八进制数转换成二进制数

$$(\ 2\quad 6\ .\ 1\quad 3\quad 5\ )_8$$
$$\downarrow\quad\downarrow\quad\downarrow\quad\downarrow\quad\downarrow$$
$$010\quad 110\quad 001\quad 011\quad 101$$

即　　$(26.135)_8 = (10110.001011101)_2$

然后再将二进制数转换成十六进制数

$$(0001\ 0110.\ 0010\ 1110\ 1000)_2$$
$$\downarrow\quad\downarrow\quad\downarrow\quad\downarrow\quad\downarrow$$
$$1\quad 6\quad 2\quad E\quad 8$$

所以　　$(26.135)_8 = (16.2E8)_{16}$

1.3　码制

不同的数码不仅可以表示数量的大小，还能表示不同的含义。利用数码来作为某一特定信息的代号叫作代码，如 901 路公交车、18 号信箱等。为便于记忆和处理，在编制代码时总要遵循一定的规则，这些规则叫做码制。

1.3.1　原码、反码和补码

通常情况下，我们记录一个带符号数时，要在数的前面加上“+”或“−”，表示正数或负数，如+5V 电压。但是在数字系统中，正负号用数码来表示。正负数的表示方法为：把一个数的最高位作为符号位，用 0 表示正数，用 1 表示负数。在数字电路中，带符号数通常有原码、反码和补码三种表示方法。

1. 原码

原码表示方法简单，只需要将符号位用 0 表示正数，1 表示负数即可，后面的数不变。

例如：　　$X_1=+10101$，　$X_2=-10101$

则用原码表示为　　$(X_1)_{原}=010101$，$(X_2)_{原}=110101$

需要指出的是，零的原码有两种表示形式：

$$(+0)_{原}=0.00\cdots0$$

$$(-0)_{原}=1.00\cdots0$$

原码虽然表示方法简单，但在数字系统中运算时比较麻烦，尤其是减法运算，需要先用大数减去小数，最好还要判断符号位，这样就导致运算速度降低。

2. 反码

在反码的符号表示方法中，仍然用 0 表示正数，1 表示负数。正数的表示方法与原码表示方法相同，负数除符号位外的各位按位取反。

例如：　　$X_1=+10101$，　$X_2=-10101$

则用反码表示为　　$(X_1)_{反}=010101$，$(X_2)_{反}=101010$

零的反码有两种形式：

$$(+0)_{反}=0.00\cdots0$$

$$(-0)_{反}=1.11\cdots1$$

3. 补码

在补码的符号表示方法中，仍然用 0 表示正数，1 表示负数。正数的补码与原码、反码相同，负数的补码除符号位外的各位按位取反后再在最低位加 1。

例如：　　$X_1=+10101$，　$X_2=-10101$

则用补码表示为　　$(X_1)_{补}=010101$，$(X_2)_{补}=101011$

零的补码只有一种形式：

$$(0)_{补}=0.00\cdots0$$

补码可以将数字系统的减法运算转化为用加法实现。

例 1-15　若 $X_1=+10101$，$X_2=+01101$，计算 X_1-X_2 的值。

解：根据二进制数的运算规则可知

$$\begin{array}{r} 10101 \\ -\ 01101 \\ \hline 01000 \end{array}$$

则　　$X_1-X_2=+01000$

下面采用补码的计算方法

$$X_1-X_2=X_1+(-X_2)=(X_1)_{补}+(-X_2)_{补}=010101+110011$$

用竖式表示

$$\begin{array}{r} 010101 \\ +\ 110011 \\ \hline \boxed{1}\,001000 \end{array}$$

丢弃进位位，则

$$X_1-X_2=+01000$$

这种加法运算更加适合于计算机系统中的运算，提高系统的运算速度。

3 种表示方法如表 1-1 所示。

表 1-1　原码、反码和补码的对应关系

十进制	二进制			十进制	二进制		
	原码	反码	补码		原码	反码	补码
−7	1111	1000	1001	0	0000	0000	0000
−6	1110	1001	1010	1	0001	0001	0001
−5	1101	1010	1011	2	0010	0010	0010
−4	1100	1011	1100	3	0011	0011	0011
−3	1011	1100	1101	4	0100	0100	0100
−2	1010	1101	1110	5	0101	0101	0101
−1	1001	1110	1111	6	0110	0110	0110
−0	1000	1111	0000	7	0111	0111	0111

1.3.2　二-十进制代码

为了用二进制数表示十进制数，常采用 BCD（Binary-Code-Decimal）码的表示方法。BCD 码即用一组 4 位二进制码表示 1 位十进制数的编码方法，也称为二-十进制码。BCD 码包括有权码和无权码。

1. 常用 BCD 码

4 位二进制码最多可以组成 16 种不同的代码，可从中任取 10 种组合来表示十进制中 0～9 这 10 个数。表 1-2 列出了几种常用的 BCD 码。

表 1-2　常用的 BCD 码

十进制数 \ BCD 码	8421 码	5421 码	2421 码	5211 码	余 3 码
0	0000	0000	0000	0000	0011
1	0001	0001	0001	0001	0100
2	0010	0010	0010	0100	0101
3	0011	0011	0011	0101	0110
4	0100	0100	0100	0111	0111
5	0101	1000	1011	1000	1000
6	0110	1001	1100	1001	1001
7	0111	1010	1101	1100	1010
8	1000	1011	1110	1101	1011
9	1001	1100	1111	1111	1100

表 1-2 中，8421 码、5421 码、2421 码和 5211 码都是有权码，即将二进制数按位权相加就可得到相应的十进制数。如 8421 码是 BCD 码中最常用的一种，从左到右每一位分别表示 8、4、2、1。2421 码，从左到右每一位分别表示 2、4、2、1，例如：

$$[1001]_{8421BCD}=1\times8+0\times4+0\times2+1\times1=(9)_{10}$$

$$[1011]_{2421BCD}=1\times2+0\times4+1\times2+1\times1=(5)_{10}$$

余3码是一种无权码，由8421码加上 $(3)_{10}=(0011)_2$ 得到。将两个余3码相加时，如果两个十进制数之和为10，应产生进位，而两个对应余3码之和正好等于二进制数16，于是便从高位自动产生进位信号而不再需要修正。

从表1-2中还可以看出，0和9、1和8、2和7、3和6、4和5的余3码互为反码，这对于求取对10的补码非常方便。2421码也有这个特点。

2. 格雷码（Gray Code）

格雷码是一种无权码，也称为循环码。其特点是：两个相邻代码只有1位不同，而且首尾（0和15）两个代码也只有1位不同，构成“循环”。这样，可以避免计数过程中出现的瞬间模糊状态，因为当数码变化时，格雷码仅改变1位，从而大大减少错码的可能性。表1-3所示为十进制数与4位格雷码的对应关系，从表中可以看出格雷码的另外一个特点：除0000外，关于中线对称。

表1-3 格雷码

十进制数	格雷码	十进制数	格雷码
0	0000	8	1100
1	0001	9	1101
2	0011	10	1111
3	0010	11	1110
4	0110	12	1010
5	0111	13	1011
6	0101	14	1001
7	0100	15	1000

1.4 逻辑代数的运算

逻辑是指事物的因果关系，或者说条件和结果的关系，这些因果关系可以用逻辑运算来表示，也就是用逻辑代数来描述。逻辑代数又称为布尔代数，是英国数学家乔治·布尔（George Boole）在1847年提出的，是用来描述客观事物逻辑关系的数学方法。后来香农（Shannon）将布尔代数应用于解决开关电路和数字逻辑电路的分析和设计中，所以也把布尔代数称为开关代数或逻辑代数。

逻辑代数中的变量称为逻辑变量，逻辑变量的取值只有0和1两种可能，称为逻辑0状态和逻辑1状态。这里0和1不表示数值大小，只代表两种不同的逻辑状态，如信号的有无、电平的高低或灯的亮灭等。

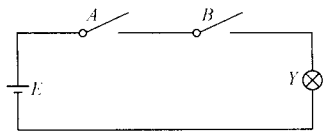
1.4.1 逻辑代数的基本逻辑运算

逻辑代数的基本运算有与、或、非三种。

1. 与逻辑

当决定某一事件的全部条件都具备时，该事件才会发生，这样的因果关系称为与逻辑关系，简称与逻辑。可以用图1-1所示电路加以说明，图1-1中只有当 A、B 两个开关同时闭

合时，指示灯 Y 才会亮，功能描述如表 1-4 所示。



电路图

图 1-1　与逻辑关系举例

表 1-4　与逻辑功能表

开关 A	开关 B	灯 Y
断开	断开	灭
断开	闭合	灭
闭合	断开	灭
闭合	闭合	亮

两输入变量与逻辑表达式为

$$Y=A\cdot B \tag{1-5}$$

式中，Y 是逻辑函数；A 和 B 是逻辑变量；运算符号“·”读作“与”（或读作“逻辑乘”），表示与逻辑。可以推广到多输入变量的一般形式为

$$Y=A\cdot B\cdot C\cdot D\cdots$$

在不致引起混淆的前提下，“·”常被省略，可以简写为

$$Y=ABCD\cdots$$

对图 1-1，用逻辑变量 A、B 代表电路中的两个开关，我们用 1 表示开关闭合，用 0 表示开关断开；用 Y 表示指示灯的状态，用 1 表示灯亮，用 0 表示灯不亮。则可以列出逻辑变量和逻辑函数之间的逻辑关系表格，称为真值表，如表 1-5 所示。

表 1-5　与逻辑真值表

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

实现与逻辑的电路称作与门，与逻辑和与门的逻辑图形符号如图 1-2a 所示。图中第一行为国家标准规定的图形符号，第二行为过去沿用的图形符号，第三行为国外书刊和资料上的常见图形符号。

2. 或逻辑

当决定某一事件的所有条件中，只要有一个条件具备，该事件就会发生，这样的因果关系叫做或逻辑关系，简称或逻辑。如图 1-3 所示，两个开关 A、B 中只要有一个接通，指示灯 Y 就会亮，功能描述如表 1-6 所示。

两输入变量或逻辑表达式为

$$Y=A+B \tag{1-6}$$

运算符号“+”读作“或”（或读作“逻辑加”）表示或逻辑。可以推广到多输入变量的一般形式为

$$Y=A+B+C+D+\cdots$$

或逻辑的真值表如表 1-7 所示。实现或逻辑的电路称作或门，或逻辑和或门的逻辑图形符号如图 1-2b 所示。

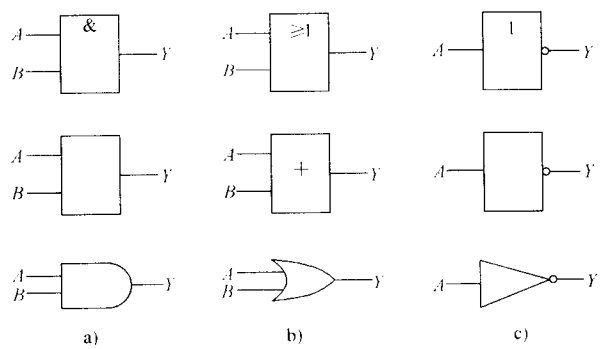


图 1-2　基本逻辑的逻辑图形符号

a）与逻辑图形符号　b）或逻辑图形符号　c）非逻辑图形符号

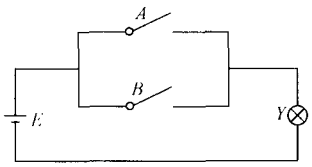


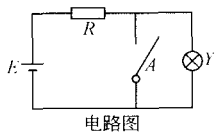
图 1-3　或逻辑关系举例

表 1-6　或逻辑功能表

开关 A	开关 B	灯 Y
断开	断开	灭
断开	闭合	亮
闭合	断开	亮
闭合	闭合	亮

3. 非逻辑

当某一条件具备了，事情不会发生；而此条件不具备时，事情反而发生。这种逻辑关系称为非逻辑关系，简称非逻辑。如图 1-4 所示，开关 A 接通时，指示灯 Y 不亮，开关 A 断开时，指示灯亮，功能描述如表 1-8 所示。



电路图

图 1-4　非逻辑关系举例

表 1-7　或逻辑真值表

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

非逻辑表达式为

$$Y=\overline{A} \tag{1-7}$$

运算符号“－”读作“非”。

非逻辑的真值表如表 1-9 所示。实现非逻辑的电路称作非门，非逻辑和非门的逻辑图形符号如图 1-2c 所示。逻辑图形符号中用小圆圈“。”表示非运算，图形符号中的“1”表示缓冲。

与、或、非 3 种基本运算的运算优先级别依次为：非、与、或。

表 1-8　非逻辑功能表

开关 A	灯 Y
断开	亮
闭合	灭

表 1-9　非逻辑真值表

A	Y
0	1
1	0

1.4.2　逻辑代数的复合逻辑运算

在数字系统中，除应用与、或、非三种基本逻辑运算之外，还广泛应用与、或、非的不同组合，称为复合逻辑运算。最常见的复合逻辑运算有与非、或非、与或非、异或、同或等。

1. 与非逻辑

与和非的复合运算称为与非运算，是将输入变量先进行与运算，再进行非运算。

与非逻辑表达式为

$$Y=\overline{AB} \tag{1-8}$$

与非逻辑的真值表如表 1-10 所示。由真值表可见，只有所有输入逻辑变量同时为 1 时，输出才为 0。可以推广到多输入变量的一般形式为

$$Y=\overline{ABCD\cdots}$$

实现与非逻辑的电路称作与非门，与非门的逻辑图形符号如图 1-5a 所示。逻辑符号中用小圆圈“。”表示非运算。

2. 或非逻辑

或和非的复合运算称为或非运算，是将输入变量先进行或运算，再进行非运算。

或非逻辑表达式为

$$Y=\overline{A+B} \tag{1-9}$$

或非逻辑的真值表如表 1-11 所示。由真值表可见，只有所有输入逻辑变量同时为 0 时，输出才为 1。可以推广到多输入变量的一般形式为

$$Y=\overline{A+B+C+D+\cdots}$$

表 1-10　与非逻辑真值表

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

表 1-11　或非逻辑真值表

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

实现或非逻辑的电路称作或非门，或非门的逻辑图形符号如图 1-5b 所示。

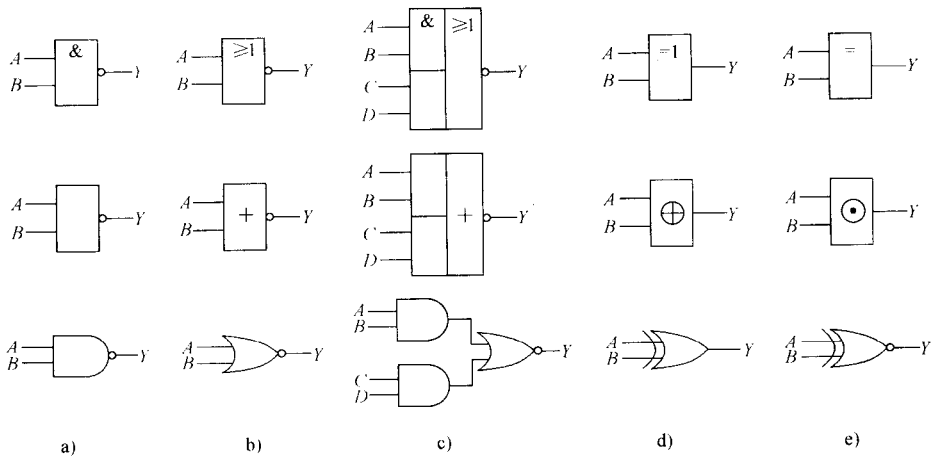


图 1-5　复合逻辑的逻辑图形符号

a）与非逻辑图形符号　b）或非逻辑图形符号　c）与或非逻辑图形符号

d）异或逻辑图形符号　e）同或逻辑图形符号

3. 与或非逻辑

与、或和非的复合运算称为与或非运算。

与或非逻辑表达式为