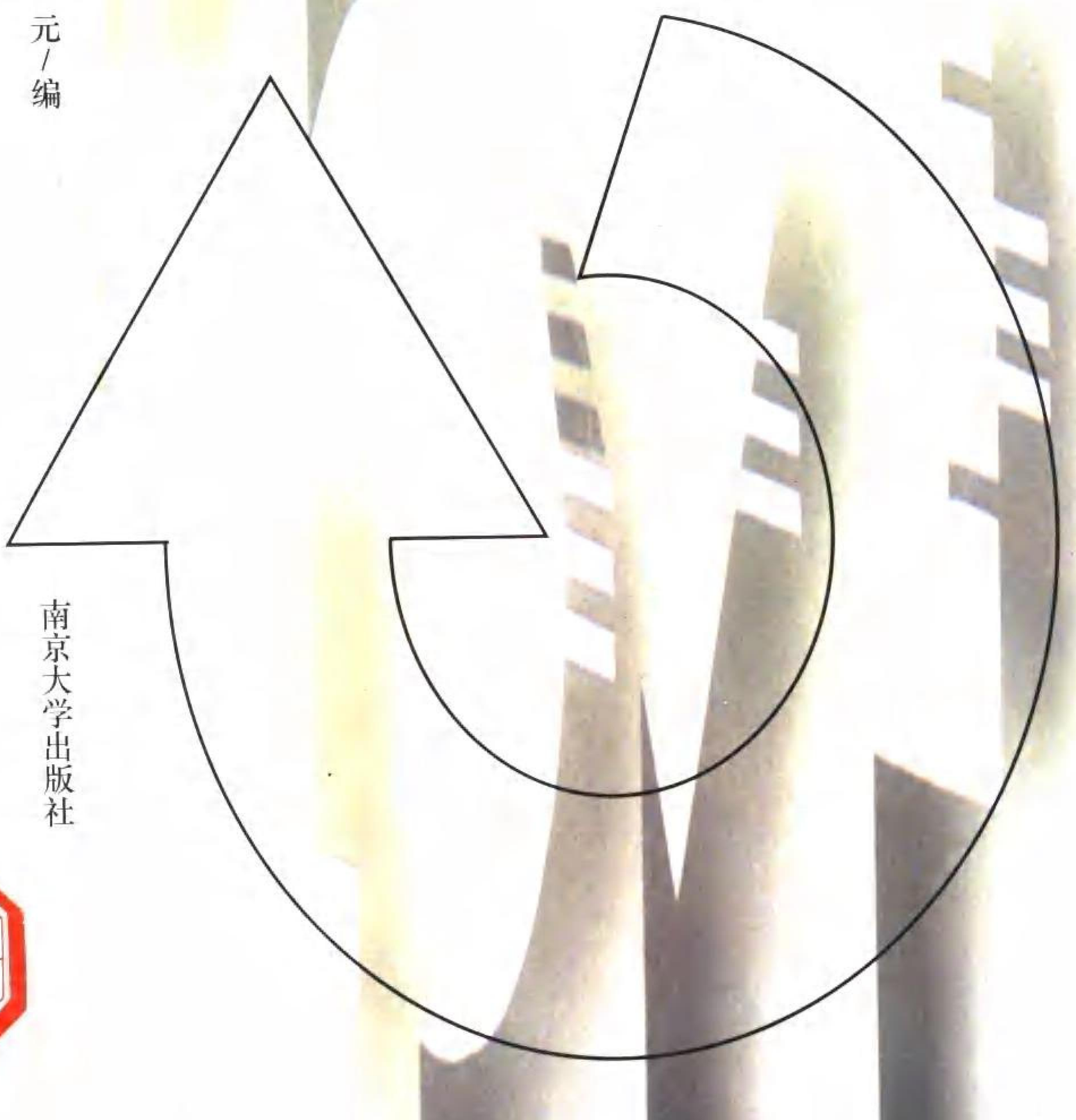
数字电路与逻辑设计

李　元/编



南京大学出版社

数字电路
与
逻辑设计

李　元　编

南京大学出版社
1997・南京

前　言

作为一门电子类的基础课——“数字电路与逻辑设计”已越来越受到人们的重视，近年来，随着微电子技术的发展，电子技术、尤其是数字技术得到了飞速的发展，这主要表现在新型器件层出不穷。例如 70 年代以来，以可编程逻辑器件为代表的新一代数字逻辑器件种类已经有可编程只读存储器(PROM)、可编程逻辑阵列(PLA)、可编程阵列逻辑(PAL)、通用逻辑阵列(GAL)、现场可编程门阵列(FPGA)等，这类新器件的开发与利用为数字电路与逻辑设计提供了崭新的舞台。然而，这些器件的开发与利用往往又需得到数字 CAD 工具软件的支撑。所以，如何在传统的数字电路与逻辑设计的“硬件”教学中适当引进一点“软件”的知识是有必要的。如何适当引进，如何使“数字电路与逻辑设计”这门基础课真正能成为“专用集成电路 CAD”、“微处理器原理”等后续课程的基础，这就是本书编者所考虑的基本思路。为此，在教材内容上增强了“可编程逻辑器件及其应用”中利用软件进行逻辑电路的设计；在第十章中，详细介绍了数字系统的分析法和设计方法。

全书内容有三部分共十章。第一部分为基础篇，介绍数字逻辑电路分析与设计的基础知识，内容涉及数制与码制、逻辑代数基础、开关元件与集成逻辑门、组合逻辑电路分析与设计、触发器、时序逻辑电路分析与设计。第二部分介绍可编程逻辑器件及应用，内容涉及 PROM、PLA、PAL、GAL 和 FPGA 的结构与工作原理，以及在逻辑设计中的应用。第三部分主要介绍脉冲波形的产生和整形、模数转换与数模转换。每章都有一定数量的实例和习题，以帮助读者提高分析能力和理解能力。

本书能得以出版，不仅得到了教研室沈振宇教授、张震武、金之仑等老师的支持和帮助，而且得到了电子科学与工程系的领导、学校教务处、校教材出版基金委员会的大力支持和帮助，在此深表谢意。

由于编者水平有限，书中难免有谬误欠妥之处，恳请读者批评、指正，作者将不胜感激。

编者

内 容 提 要

本书不仅介绍了组成数字逻辑电路的基本单元电路及常用中大规模数字集成电路和新颖的可编程逻辑器件，而且还着重介绍了数字逻辑电路与系统的分析方法和设计方法。全书内容有三部分共十章。第一部分为基础篇，介绍数字逻辑及其电路的分析与设计基础知识，内容涉及数制与码制、逻辑代数基础、开关元件与集成逻辑门、组合逻辑电路分析与设计、触发器、时序逻辑电路分析与设计。第二部分介绍当今数字电路发展最活跃的新型器件——可编程逻辑器件及其应用，内容涉及PROM、PLA、PAL、GAL和FPGA的结构与工作原理，以及在逻辑设计中的应用。第三部分为模拟电路与数字电路的结合部分，主要介绍脉冲波形的产生与整形、模数转换与数模转换。最后，作为数字电路与逻辑设计的综合，介绍了数字系统的设计基础。本书内容充实，系统性强，力求做到理论与实际相结合。

本书可作为大专院校理、工科电子信息技术、计算机、无线电、声学、自动控制等专业学生的教材，也可作为相关专业工程技术人员参考书。

数字电路与逻辑设计
李 元 编
*
南京大学出版社出版
（南京大学校内 邮政编码：210093）
江苏省新华书店发行 江苏武进第三印刷厂印刷
*
开本 787×1092 1/16 印张 14.75 字数 365 千
1997 年 10 月第 1 版 1997 年 10 月第 1 次印刷
印数 1—2000
ISBN 7-305-03107-0/TN・7
定价：22.00 元
（南大版图书若有印装错误可向承印厂退换）

目　录

数制与编码 1

本章介绍数字电路常用的数制和编码。

1-1 进位计数制

数制是计数进位制的简称。在日常生活中，通常使用的是十进制。在十进制中，有 0,1,2,…,9 十个基数，计算时按“逢十进一”，“借一当十”的原则计算。例如 1997.6 这个数由 5 位数字组成，小数点左边有 4 位，右边有 1 位。这个数亦可写成 $1997.6=1\times10^3+9\times10^2+9\times10^1+7\times10^0+6\times10^{-1}$。

推而广之，一个 n 位整数、m 位分数的十进制数 $(N)_{10}$ 可写成：

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

式中 $a_i\in(0,1,2,\cdots,9)$，n,m 均为正整数。10^i 是第 i 位的位值，称为“权”。

在数字电路中常用的是二进制。在二进制中只有两个基数 0,1。它的运算规则是“逢二进一”，“借一当二”。例如 $(N)_2=(101101.01)_2$，它可表示为：

$$\begin{aligned}(N)_2&=1\times2^5+0\times2^4+1\times2^3+1\times2^2+0\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2}\\&=(45.25)_{10}\end{aligned}$$

推而广之，一个 n 位整数、m 位分数的二进制数 $(N)_2$ 可记为：

$$\begin{aligned}(N)_2&=\sum_{i=-m}^{n-1}b_i\times2^i=b_{n-1}\times2^{n-1}+b_{n-2}\times2^{n-2}+\cdots+b_1\times2^1+b_0\times2^0+b_{-1}\times2^{-1}\\&\quad+b_{-2}\times2^{-2}+\cdots+b_{-m}\times2^{-m}\\&=(b_{n-1}b_{n-2}\cdots b_0b_{-1}b_{-2}\cdots b_{-m})_2\end{aligned}$$

在数字电路中，由于二进制表示一个较大的数值时，用的位数很多，读写时容易出错，因此还常常用到八进制和十六进制。在八进制中，基数 B=8，有 0,1,2,…,7 八个基数，运算规则是：“逢八进一，借一当八”。例如：$(15.46)_8=1\times8^1+5\times8^0+4\times8^{-1}+6\times8^{-2}$。在十六进制中，有十六个基数：0,1,…,9,A,B,C,D,E,F。运算规则是“逢十六进一，借一当十六”。例如：$(C36F)_{16}=C\times16^3+3\times16^2+6\times16^1+F\times16^0=(50031)_{10}$。

在实际运用中，八进制数是用二进制代码表示的，称为二进制编码的八进制数。因为 $2^3=8$，所以一位八进制数相当于三位二进制数。例 $(546)_8$ 其代码为 $(101100110)_2$。同样，十六进制

数也是用二进制代码表示的。因为 $16=2^4$,所以一位十六进制数相当于四位二进制数。例如:$(C36F)_{16}=(1100001101101111)_2$。

1-2　数制转换

二进制、八进制、十六进制及十进制是数字电路中常用的四种数制。本节举例说明它们之间转换的方法。

例 1.1　将二进制数 $(N)_2=(101110.01)_2$ 转换成十进制数 $(N)_{10}$。

解　$(N)_2=1\times2^5+0\times2^4+1\times2^3+1\times2^2+1\times2^1+0\times2^0+0\times2^{-1}+1\times2^{-2}$

$$=(46.25)_{10}$$

例 1.2　将十进制数 $(N)_{10}=(51.35)_{10}$ 转换成二进制数,要求分数部分转换位数为 6 位。

解　转换分成整数部分及分数部分。整数部分 $(51)_{10}$ 的转换过程为:将 $(51)_{10}$ 用 2 除之,得商 $(25)_{10}$,余数为 1,此即 b_0,依次类推,直至商为零为止:

$$\begin{array}{r|l l}
2 & 51 & \\ \hline
2 & 25 & b_0=1 \\ \hline
2 & 12 & b_1=1 \\ \hline
2 & 6 & b_2=0 \\ \hline
2 & 3 & b_3=0 \\ \hline
2 & 1 & b_4=1 \\ \hline
 & 0 & b_5=1
\end{array}$$

分数部分的转换过程如下:

$$\begin{array}{ll}
0.35\times2=0.70 & b_{-1}=0 \\
0.70\times2=1.40 & b_{-2}=1 \\
0.40\times2=0.80 & b_{-3}=0 \\
0.80\times2=1.60 & b_{-4}=1 \\
0.60\times2=1.20 & b_{-5}=1 \\
0.20\times2=0.40 & b_{-6}=0
\end{array}$$

首先将分数部分 $(0.35)_{10}$ 乘 2,得 $(0.70)_{10}$,由于 $(0.70)_{10}$ 的整数部分为 0,所以 $b_{-1}=0$。然后将 $(0.70)_{10}$ 再乘 2,得 $(1.40)_{10}$,其整数部分为 1,即 $b_{-2}=1$。依次类推,一直做到规定的位数,或者分数部分为零为止。

将整数部分和分数部分转换的结果相排列,便得到总的转换结果:

$$(51.35)_{10}=(110011.010110)_2$$

例 1.3　将 $(51.35)_{10}$ 转换成八进制数。

方法 1　首先将 $(51.35)_{10}$ 转换成二进制数 $(N)_2$,然后从小数点开始向左、向右每三位一组转换成八进制数。对于本例:

$$\begin{aligned}(51.35)_{10} &= (110011.010110)_2 \\ &= (63.26)_8\end{aligned}$$

方法 2　整数部分用不断除 8 的办法,分数部分用不断乘 8 的办法得到八进制数,对于本

例：

$$\begin{array}{r|l} 8 & 51 \\ \hline 8 & 6 \\ \hline & 0 \end{array}$$

余数 3

余数 6

$0.35 \times 8 = 2.80$　整数部分　2

$0.80 \times 8 = 6.40$　整数部分　6

两部分排列起来即为$(63.26)_8$。

例 1.4　将$(51.35)_{10}$转换成十六进制。

解　首先将$(51.35)_{10}$转换成二进制数$(N)_2$，然后从小数点向左、向右每四位一组转换成十六进制数。本例为：

$$(51.35)_{10} = (110011.010110)_2$$
$$= (33.58)_{16}$$

例 1.5　已知二进制数，转换成八进制和十六进制数。

① $(10110001101011.111100000110)_2 = (26153.7406)_8$

2 6 1 5 3　7 4 0 6

② $(10110001101011.111100000110)_2 = (2C6B.F06)_{16}$

2 C 6 B　F 0 6

例 1.6　已知八进制或十六进制数，转换成二进制数。

① $(673.124)_8 = (110111011.001010100)_2$

6 7 3　1 2 4

② $(306.D)_{16} = (001100000110.1101)_2$

3 0 6　D

1-3　原码、反码和补码

一、补数的基本概念

基数为 r，整数部分为 n 位的任一正数 N 的补数是：

$$\begin{array}{ll} r^n - N & N \neq 0 \\ 0 & N = 0 \end{array}$$

例 1.7　十进制数的补数。

$$(52520)_{10} = [10^5 - 52520]_{补} = (47480)_{补}$$
$$(0.3267)_{10} = [1 - 0.3267]_{补} = (0.6733)_{补}$$
$$(25.639)_{10} = [10^2 - 25.639]_{补} = (74.361)_{补}$$

例 1.8　二进制数的补数。

$$\begin{aligned}(101100)_2 &= [2^6 - 101100]_{补}\\ &= [1000000 - 101100]_{补}\\ &= [010100]_{补}\end{aligned}$$

$$(0.0110)_2 = (1 - 0.0110)_{补} = (0.1010)_{补}$$

二、机器数与原码、反码和补码

如何表示一个二进制数的正负?最简单的方法是在数的前面增设一个符号位，用符号位的"0"表示正数，"1"表示负数。例如：

$$+1101(\text{真值}) = (01101)(\text{机器数})$$
$$-1101(\text{真值}) = (11101)(\text{机器数})$$

符号位后面的数值称为机器数的真值。为了运算方便(把减法变为加法运算)，负数有三种表示方法——原码、反码和补码。

1. 原码

上述的机器数，即是原码表示法。原码的优点是简单、直观，而且用原码作乘法运算时比较方便，这时乘积的符号是两个乘数符号的异或。缺点是两个原码作加减法时，首先需要判断符号位是否相同，才能决定真值部分相加还是相减；其次需要判断两个真值的大小，才能确定谁是被减数，谁是减数。因此原码作加/减法运算时较为复杂。目前普遍采用补码进行加/减运算。

2. 反码

正数的反码与原码相同，最高位为符号位，用"0"表示正。负数的反码是将正数连同符号位按位取反。例如，

$$\underset{\text{符号位}}{\underset{\uparrow}{0}}\qquad \underbrace{1101}_{\text{真值}} \longrightarrow \underbrace{01101}_{\text{反码}}$$

$$\underset{\text{符号位}}{\underset{\uparrow}{1}}\qquad \underbrace{1101}_{\text{真值}} \longrightarrow \underbrace{00010}_{\text{反码}}$$

反码的一个重要用途是产生补码。

3. 补码

正数的补码与原码相同，负数的补码是它的反码+1。例如，

$$\underset{\text{符号位}}{\underset{\uparrow}{0}}\qquad \underbrace{1101}_{\text{真值}} \longrightarrow \underbrace{01101}_{\text{补码}}$$

$$\underset{\text{符号位}}{\underset{\uparrow}{1}}\qquad \underbrace{1101}_{\text{真值}} \longrightarrow \underbrace{00011}_{\text{补码}}$$

利用补码的概念可以方便地进行两个数的加/减法。

例 1.9　01010100－01000100(带有符号位的两个数相减)

解　01010100 ——→补码为　01010100

-01000100 ——→补码为　10111100

补码相加：　100010000　（将进位 1 丢掉）

∴01010100－01000100＝00010000　（正数的补码即是原码）

例 1.10　利用补数(补码)的概念进行两个数相减。

(1)　$(72532-3250)_{10}$

解　∵3250 的补数是 96750

∴$72532-3250=[72532+96750]_{补}$

$=[1\ 69282]_{补}=69282$

式中 1 是进位，去掉，所得结果就是:69282。

(2)　$(3250-72532)_{10}$

解　∵72532 的补数是 27468

∴$(3250-72532)_{10}=(3250+27468)_{补}=(30718)_{补}$

所得结果没有进位，因此需要从补码还原至原码：－69282。

(3)　$(1010100)_2-(1000100)_2$

解　$(1010100)_2-(1000100)_2$

$=(1010100+0111100)_{补}$

$=(1\ 0010000)_{补}$　丢掉进位 1

$=(0010000)_2$

(4)　$(01000100)_2-(01010100)_2$

解　$(01000100+10101100)_{补}$

$=(11110000)_{补}$　没有进位，最高位符号位表示该数是负数

∴$(1110000)_{补}$的原码是－0010000

即 01000100－01010100＝－0010000

1-4　若干常用的编码

一、表示 0～9 十个数的二进制代码的方法很多，常见的有 8-4-2-1 码，余 3 码，5-2-1-1 码等。如表 1-1 所示。

表 1-1

十进制数	BCD 码 8-4-2-1	余三码	BCD 码 5-2-1-1	余三循环码	二元五进制 5043210 码	步进码	格雷码 (Gray 码)
0	0000	0011	0000	0010	0100001	00000	0000
1	0001	0100	0001	0110	0100010	10000	0001
2	0010	0101	0100	0111	0100100	11000	0011
3	0011	0110	0101	0101	0101000	11100	0010
4	0100	0111	0111	0100	0110000	11110	0110

5	0101	1000	1000	1100	1000001	11111	1110
6	0110	1001	1001	1101	1000010	01111	1010
7	0111	1010	1100	1111	1000100	00111	1011
8	1000	1011	1101	1110	1001000	00011	1001
9	1001	1100	1111	1010	1010000	00001	1000

例 1.11　利用 8421BCD 码和格雷码表示$(1997)_{10}$。

解　$(1997)_{10}=(0001100110010111)_{8421}=(0001100010001011)_G$

表中，格雷码是一种具有纠错能力的码，其特点是相邻码之间仅有一位不同。例如，在 8421 码中，7～8 的编码是 0111～1000，有四位变化，而格雷码的 7～8 是 1011～1001 只有一位变化，这种特点在许多场合是有用的。

二、字母数字码

字母数字码是用二进制代码来表示字母、数字、信息符号的一组代码。常用的有 8 位 ISO 码、ASCII 码、EBCDIC 码等。表 1-2 为美国信息交换标准码(ASCII 码)的编码表。这种编码常用于计算机的键盘编码和一些通讯设备的字母数字码。

表 1-2

$b_4b_3b_2b_1\backslash b_7b_6b_5$	000	001	010	011	100	101	110	111
0000	NUL	DEL	SP	0	@	P		p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	”	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	’	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	!
1101	CR	GS	—	=	M	]	m	}
1110	SO	RS	·	>	N	↑	n	～
1111	SI	US	/	?	O	↓	O	DEL

习　题

1. 把下列二进制数换算成十进制数：

(1)1100101　　(2)0.1101　　(3)101011.101

2. 把下列十进制数换算成二进制数：

(1)1997　　(2)312.32　　(3)0.85

3. 完成下列各数的换算：

(1)$(4A6B)_{16}=(?)_2=(?)_{10}$

(2)$(56.84)_{10}=(?)_2$

(3)$(6375)_{10}=(?)_2=(?)_8=(?)_{16}$

逻辑代数基础

2

本章介绍逻辑代数的基本概念、基本公式、运算规律及逻辑函数的化简方法。

2-1 逻辑代数的三种基本逻辑关系

逻辑代数又称布尔代数，它是由英国数学家乔治·布尔(George Boole)于1849年首先提出来的一种研究逻辑关系的二进制数字系统。首先，什么叫"逻辑"？逻辑是logic的译音，它表示事物的因果关系。例如，所有开关都具有完全确定的两种状态，开和关，这两种状态可以用"0"和"1"两种数字表征。如我们定义开关接通为"1"，开关断开为"0"，电灯"亮"为"1"，电灯"灭"为"0"，那么由若干个开关和电灯就能组成基本的逻辑关系。图2-1(a)为两个开关A、B和电灯L及电源组成了一个串联电路。显然，只有当A与B都接通，灯L才亮。我们可以列出输入变量A、B取值的组合和输出L的一一对应关系，如表2-1所示。这个表叫做真值表。由表可知，只有当A＝1，B＝1，L才等于1。A、B中只要有一个为0，则L＝0。用数字语言表示：

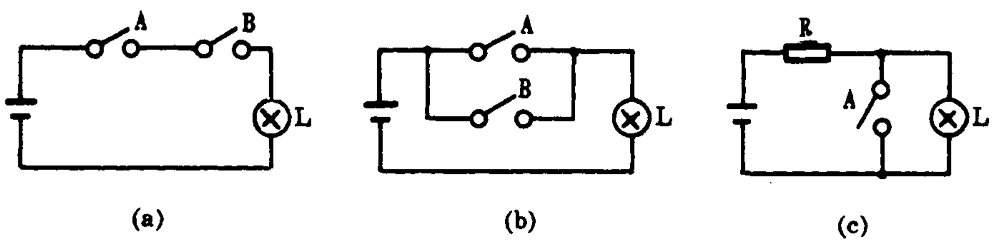


图2-1 基本逻辑开关电路

L＝A·B。这种逻辑关系称为"与"逻辑。式中符号"·"叫逻辑乘(或叫"与"运算)，不是普通代数中的乘号。为了书写方便，符号"·"可以不写。逻辑乘的表达式亦可推广到多变量的一般形式：

$$L = A \cdot B \cdot C \cdot D \cdots = ABCD \cdots$$

实现"与"逻辑的电路叫"与门"。

图2-1(b)所示为两个开关的并联电路。其真值表为表2-2。该电路表示只要有一个开关接通，灯就亮，即只要A＝1，或B＝1，则L＝1，其表达式为L＝A＋B。这种逻辑关系称为"或"逻辑(逻辑加)，其符号为"＋"。实现"或"逻辑的电路叫"或门"。

图2-1(c)表示逻辑"非"(又称逻辑"反")，其真值表为表2-3。其逻辑关系是当A接通，则灯L不亮，当A断开，则灯L亮。即A＝0，L＝1；A＝1，L＝0。我们用逻辑关系式表示：$L=\overline{A}$。

表 2-1　逻辑“与”的真值表

A	B	L
0	0	0
0	1	0
1	0	0
1	1	1

表 2-2　逻辑“或”的真值表

A	B	L
0	0	0
0	1	1
1	0	1
1	1	1

表 2-3

A	L
0	1
1	0

这种逻辑关系称为逻辑“非”。实现“非”逻辑的电路称为“非门”。上述三种逻辑关系的符号如图 2-2 所示。(a)为国内传统画法;(b)为国外传统画法;(c)为电气图用图形符号国家标准。本书采用国内传统画法。

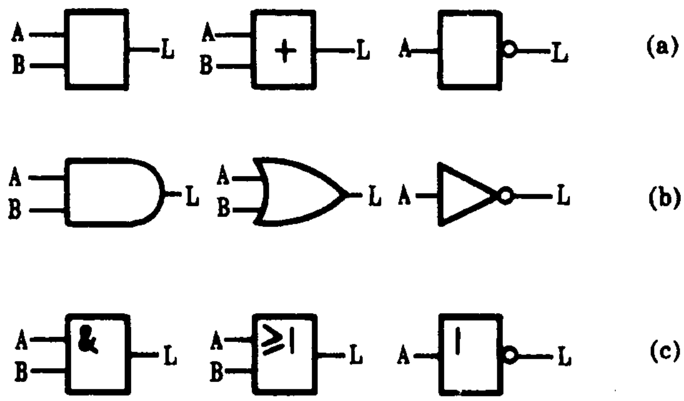


图 2-2　三种逻辑关系的符号

2-2　正逻辑和负逻辑

上一节讨论的三种逻辑关系的定义是开关接通为“1”,开关断开为“0”,灯亮为“1”,灭为“0”。若定义开关接通为“0”,开关断开为“1”,灯亮为“0”,灯灭为“1”,则上述的图 2-1(a)所示的“与”逻辑就成为“或”逻辑,见表 2-4 所示。而图 2-1(b)所示的“或”逻辑就成为“与”逻辑,见表 2-5 所示。我们把 2-1 节定义的逻辑称为正逻辑,而把这里定义的逻辑称为负逻辑。因此,正逻辑的与门就是负逻辑的或门,正逻辑的或门就是负逻辑的与门。在本书中若没有具体的说明,我们皆用正逻辑,即逻辑“1”表示高电平,逻辑“0”表示低电平。

表 2-4

A	B	L
0	0	0
0	1	1
1	0	1
1	1	1

表 2-5

A	B	L
0	0	0
0	1	0
1	0	0
1	1	1

2-3 逻辑代数的基本公式和基本定理

逻辑代数的基本公式如表 2-6 所示。

表 2-6

序号	定律名称	公式	
1	0-1 律	$A \cdot 0=0$	$A+1=1$
2	自等律	$A \cdot 1=A$	$A+0=A$
3	重叠律	$A \cdot A=A$	$A+A=A$
4	互补律	$A \cdot \overline{A}=0$	$A+\overline{A}=1$
5	交换律	$A \cdot B=B \cdot A$	$A+B=B+A$
6	结合律	$A \cdot (BC)=(AB) \cdot C$	$A+(B+C)=(A+B)+C$
7	分配律	$A \cdot (B+C)=AB+AC$	$A+(B \cdot C)=(A+B) \cdot (A+C)$
8	还原律	$\overline{\overline{A}}=A$	
9	反演律	$\overline{AB}=\overline{A}+\overline{B}$	$\overline{A+B}=\overline{A} \cdot \overline{B}$
10	吸收律	$A+AB=A$	$A \cdot (A+B)=A$
		$A+\overline{A}B=A+B$	$A \cdot (\overline{A}+B)=AB$
		$AB+\overline{A}C+BC=AB+\overline{A}C$	$(A+B)(\overline{A}+C)(B+C)=(A+B)(\overline{A}+C)$

基本公式证明的方法常用的是真值表法，例如证明分配率：$A+BC=(A+B) \cdot (A+C)$，真值表如表 2-7 所示。

表 2-7

A	B	C	BC	A+B	A+C	A+BC	(A+B)(A+C)
0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0
0	1	0	0	1	0	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

从表 2-7 看出，原变量 A、B、C 的各种组合的结果 $A+BC$ 都是和 $(A+B) \cdot (A+C)$ 相同，因此分配律得以证明。

逻辑代数的基本定理有三个：代入定理、反演定理和对偶定理。

1. 代入定理：在任何一个逻辑等式中，若将其某一个变量以另一个逻辑函数式代入，则此等式仍成立。例如，$\overline{A \cdot B}=\overline{A}+\overline{B}$，若将该等式中变量 A 以 $(C \cdot D)$ 代入，则可得到：

$$\overline{(C \cdot D) \cdot B}=\overline{C \cdot D}+\overline{B}=\overline{C}+\overline{D}+\overline{B}$$

2. 反演定理

对于任意一个逻辑式 L，如果把其中所有的"·"换成"+"，"+"换成"·"，0 换成 1，1 换