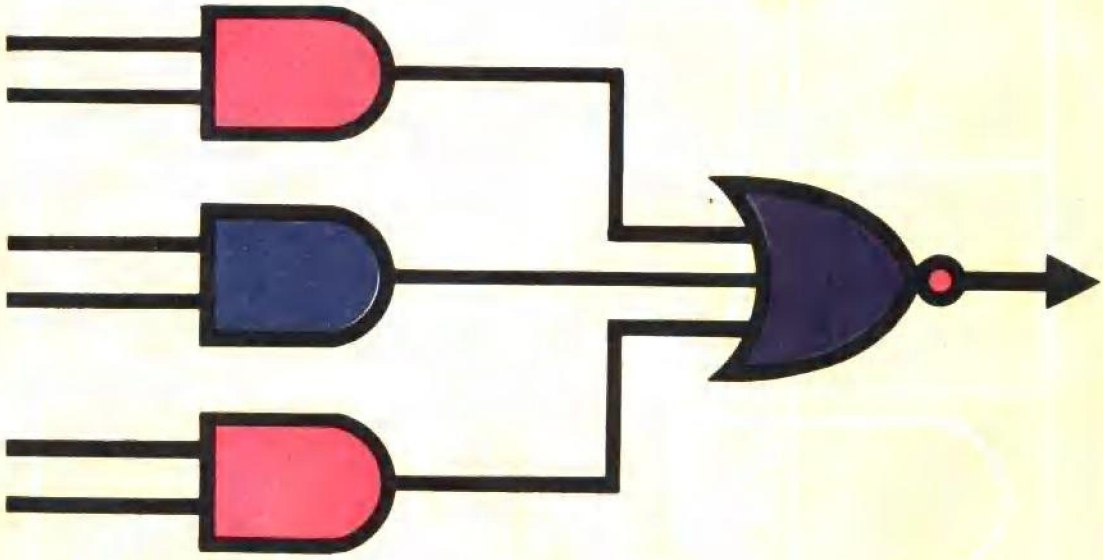
数字电路与逻辑设计

SHUZI DIANLU YU LUOJI SHEJI



李建勋 著

科学出版社

15.1011.64/54

0424952

数字电路与逻辑设计

李建勋 著

罗银芳 刘启业 译

李树贻 校

YP21/12

北京邮电学院 图书馆藏书

21113000812657

内 容 简 介

本书结合当前最新的数字电路技术，较全面系统地介绍了逻辑设计的基本理论、实现方法以及一些常用的逻辑线路.

全书共九章. 前六章讲述了布尔代数基础、组合电路的设计及化简、使用各种规模集成电路的逻辑设计、时序电路的分析和综合、竞争冒险的克服方法以及设计数字系统的流程图法等内容. 随后两章介绍了数字电路的故障检测和故障定位问题. 最后一章介绍了用微处理机进行逻辑设计的方法. 附录简述了半导体存贮器的原理、发展概况以及在实验室进行的一些实验.

本书可供计算机体系逻辑设计人员、硬件设计人员以及大专院校有关专业的高年级学生学习参考之用.

Samuel C. Lee

DIGITAL CIRCUITS AND LOGIC DESIGN

Prentice-Hall, 1976

数字电路与逻辑设计

李建勋 著

罗银芳　刘启业 译

李树贻 校

责任编辑 黄岁新

科学出版社出版

北京朝阳门内大街137号

中国科学院印刷厂印刷

新华书店北京发行所发行　各地新华书店经售

*

1981年4月第一版　开本：787×1092　1/16

1983年10月第二次印刷　印张：27 3/4

印数：10,201—16,400　字数：646,000

统一书号：15031·333

本社书号：2083·15—8

定价：4.25元

序

在过去五年中，数字电路和逻辑设计取得了巨大的进展. 越来越复杂的集成电路已经大量出现，而且价格也便宜了. 许多经典的逻辑设计规则，比如削减门和触发器数目的规则等，现在只适用于线路元件（指中规模集成电路 MSI 及大规模集成电路 LSI）的设计或者仅适用于那些不适合集成化的器件的设计. 当用 MSI 或 LSI 作为元件设计数字电路时，再想用经典的规则来削减系统的价格已经不行了.

目前在教科书中或教学中所讲的逻辑设计规则与用现有的集成电路进行的逻辑设计规则之间存在一定的差距. 因此，大学生在上完数字电路和逻辑设计课后并不能完成实际的、可靠的、便宜的电路设计，甚至设计最简单的电路也做不到.

本书就是要消除这一差距，消除数字电路和逻辑设计在课堂中和实际应用中的差距. 我们想在课堂上就使学生能结合目前工业产品的实际来了解电路设计的理论和实践. 我相信对一个大学生来说，在其全部学习规划中包含对工程实践的了解是非常重要的. 本书既包括了基础理论，也包括了实际设计，并且还举了一些近代逻辑设计的例子. 学完这本教程后，学生能熟悉逻辑设计的理论和实践，有利于参加实际工作.

在编写本书时注意到了以下几个问题：

1. 学生和指导教师都喜欢精练的、便于理解的“数学语言”. 本书中尽量采用简单易懂的符号，尽量避免不必要的定义.

2. 对于那些难于证明或解释的定理或方法，本书首先举例，然后才正式讲解或给出严格的证明. 这种方法作者看来有两种好处. 首先，它有助于读者了解某种方法或某个定理产生的背景. 其次，它能使读者先熟悉我们用到的数学符号和公式，然后逐步了解严格的形式证明.

3. 一个图形比一千句话更能说明问题. 因此，全书大量采用了表格、图形和流程图.

阅读本书的目录，就可以对本书的指导思想和内容有一个大致的了解. 书中所讲的材料都有许多例子说明. 每一小节后面都附有相当数量的习题. 全部材料是按以下顺序安排的：

第一章讲布尔代数基础，然后介绍开关代数基础及基本组合电路设计，其中包括完全定义和不完全定义开关函数的化简、计算机辅助开关函数化简、多输出开关函数化简等. 还讲述了设计组合逻辑电路的基本步骤.

第二章介绍各种常用数制和编码系统及其相互转换. 举例说明了一些常用组合电路的设计，如代码转换器、带符号及不带符号的二进制、十进制加法器、减法器等等. 需要指出的是，即使在目前采用“模块组装”电路的情况下，第一章所讲的化简方法仍然有用，至少在设计这一章中提到的电路元件时有用.

关于逻辑电路到电子电路的映射或电压符号命名问题将在第三章讲解，这一章的目的在于介绍现有的小规模集成电路 SSI 及中规模集成电路 MSI 和半导体存贮器以及以这些器件为基础的组合电路设计. 本章也讨论了只读存贮器（ROM）和随机访问存贮器

(RAM). 还介绍了用 ROM 阵列进行组合逻辑设计的方法. 在附录A中给出了各类半导体存贮器的工艺过程.

第四章是第三章的继续. 其中介绍了 SSI, MSI 及 LSI 时序集成电路及其在目前时序逻辑设计中的应用. 这些电路包括触发器、计数器、移位器等. 本章还介绍了 MOS/LSI 可编逻辑阵列 (PLA).

第五章讲解时序电路的通用数学模型——时序机. 介绍在时序机理论基础上建立的同步、异步时序电路的分析与综合过程. 介绍了时序电路中的竞争和冒险及其克服方法. 第六章给出了一种设计数字电路及数字系统的实用方法——流程图法. 这种方法已在工程技术人员中广泛应用.

由于集成电路的内部检测点少,而每个集成电路上的元件很多, IC 数字器件及系统的检测(从引出线检测)成了目前数字电路设计、生产、故障定位的难题. 第七章和第八章就讲这些问题. 第七章介绍了几种用于推导组合电路故障检测和故障定位的测试试验方法. 介绍了故障检测和故障定位所用到的固定目录测试试验法和自适应目录测试试验法,并对这两种方法作了比较. 这一章还介绍了如何用这些方法来测试集成电路.

在第八章,对于时序电路的故障检测给出了两种办法. 第一种为线路测试法,它要求实验员对电路及其可能发生的故障很了解. 第二种为转换-检查法,它不要求了解被测线路,但要知道转换表. 第二种方法也可以用于识别机器和机器状态. 本章还介绍了用第一种方法对同步时序电路中的单个及多个故障进行定位的方法.

第九章介绍了如何利用微处理机进行逻辑设计. 这是一种最新的逻辑设计方法. 这一章从介绍计算机的基本功能开始,然后给出了一般计算机(大型的、小型的)和微处理机的区别. 最后以常见的 4 位 Intel 4040 微处理机为例说明了用微处理机进行设计的过程,以及程序可编只读存贮器 (P/ROM) 的设计过程.

在每一章的末尾都附有参考文献介绍及参考书目. 书目肯定是不全的,作者也不想给全. 因为把全部参考书不经选择地读一遍对教师和学生来说都是一种浪费. 但是,由于疏忽,本书也可能漏掉一些重要的原始材料或是很好的参考文献.

本书是根据作者五年来为电子工程及计算机科学系高年级及进修班一年级学生讲课手稿整理出来的.

李建勋

此为试读,需要完整PDF请访问: www.ertongbook.com

目　录

第一章　开关代数和开关函数

本章将详细讨论二元布尔代数（B_2），即开关代数．布尔代数是开关代数的数学基础，因此先扼要地介绍布尔代数．由二元布尔代数 B_2 定义的布尔函数称为开关函数，本章将讨论它的一般定理及其二进制变量的显逻辑等效表示法，并给出它的两种基本的二级与或及或与实现，其中假设第一级与门及第一级或门的输入变量可以是原码，也可以是反码．为了化简二级与或及或与实现，将介绍三种有关开关函数的化简法——卡诺（Karnaugh）法、奎恩-麦克拉斯基（Quine-McCluskey）法和增项消项（iterative consensus）法．卡诺法是一种图介法，适合于手算．后两种是列表法，适合于机器运算．若某一开关函数满足下列条件：（1）其二级与或（或与）实现中所用的与（或）门数量最少；（2）其二级与或（或与）实现中任一个与（或）门再不能用输入头更少的与（或）门来代替，则这一开关函数为最简开关函数，或者说这一开关函数的积-和（和-积）表示式为最简表示式．上述两个条件若以函数的积（和）项来表示，则为：（1）函数具有最少的积（和）项；（2）函数的任一个积（和）项再不能用变量更少的积（和）项来代替．

在多开关函数中，化简（把一组开关函数同时化简）的标准，除了上述用于简单开关函数化简的两项标准外，另外还得使被化简的函数与其它被化简的函数之间有尽可能多的相同项．这样就可以使得执行这一多开关函数的基本二级与或（或与）实现中，器件应用最佳．

最后，举例介绍设计组合逻辑电路的方法．

1.1　布尔代数

布尔代数首先由乔治·布尔（George Boole）* 提出，其作用随着数字计算机的问世而突出出来．现在，布尔代数在数字电路和数字计算机的设计中已被广泛应用，并成为开关理论和逻辑设计的数学基础．

定义　1.1.1

布尔代数是一代数 $(B; \cdot, +, '; 0, 1)$，它由一布尔变量集 B（B 至少包含 0，1 两个元素）和三个操作（与 $\cdot$，或 $+$，非 $'$）组成，且对 B 中的任何元素 x、y 和 z，它们的 $x \cdot y$（x 和 y 的积）、$x + y$（x 和 y 的和）和 x'（x 的非）都仍在 B 内．

布尔代数有如下公理：

A1　幂等律：$x \cdot x = x$　$x + x = x$

A2　交换律：$x \cdot y = y \cdot x$　$x + y = y + x$

A3　结合律：$x \cdot (y \cdot z) = (x \cdot y) \cdot z$　$x + (y + z) = (x + y) + z$

---

* 乔·布尔，思维规律的研究，开园出版社，芝加哥，1854/1940．

A4　吸收律：$x\cdot(x+y)=x$　$x+(x\cdot y)=x$

A5　分配律：$x\cdot(y+z)=(x\cdot y)+(x\cdot z)$　$x+(y\cdot z)=(x+y)\cdot(x+z)$

A6　对于 0（无效，最小）元素和 1（有效，最大）元素，有如下一些公理：

对于每个 $x\in B$，存在一个唯一元素（1 元素）$1\in B$，使

$$x\cdot 1=1\cdot x=x$$

对于每个 $x\in B$，存在一个唯一元素（0 元素）$0\in B$，使

$$x+0=0+x=x$$

A7　对于每个 $x\in B$，存在一个叫 x 非的唯一元素 $x'\in B$，使得

$$x\cdot x'=0$$

$$x+x'=1$$

应当注意，$x\cdot y$ 和 $x+y$ 不是普通的代数运算，元素 0 和 1 没有普通代数中的 0 和 1 的意义.

下面举两个布尔代数的简单例子.

例 1.1.1

我们先看二元布尔代数 $B_2=(\{0,1\};\cdot,+,';0,1)$. 其三个操作 $\cdot$，$+$，和 $'$ 的定义如下：

$\cdot$	0	1
0	0	0
1	0	1

$+$	0	1
0	0	1
1	1	1

$'$	
0	1
1	0

下面再看单元素集 S 构成的幂集 $P(S)$. 显然，$P(S)$ 只包括空 $\varnothing$ 和 S 两个元素. 与二元布尔代数相类似，对于 $\varnothing$ 和 S 也可以列出它们的交集、和集及补集三个表：

$\cap$	$\varnothing$	S
$\varnothing$	$\varnothing$	$\varnothing$
S	$\varnothing$	S

$\cup$	$\varnothing$	S
$\varnothing$	$\varnothing$	S
S	S	S

$\sim$	
$\varnothing$	S
S	$\varnothing$

从上面两个集的操作表说明，上述两个布尔代数彼此同构（isomorphic）*.

例 1.1.2

考虑四元布尔代数 $B_4=(\{0,a,b,1\};\cdot,+,';0,1)$，其三个操作的定义如下：

$\cdot$	0	a	b	1
0	0	0	0	0
a	0	a	0	a
b	0	0	b	b
1	0	a	b	1

$+$	0	a	b	1
0	0	a	b	1
a	a	a	1	1
b	b	1	b	1
1	1	1	1	1

$'$	
0	1
a	b
b	a
1	0

现在进一步研究幂集 $P(I)$，其中 $I=\{a,b\}$，I 的三个真子集是 $\{a\}$，$\{b\}$ 和空，并分别用 S_a、S_b 和 $\varnothing$ 表示. $P(I)$ 是用下页一些表表示的三个操作的布尔代数：

---

* 如果两个代数存在一一对应和一个向另一个映射的关系，那末，这两个代数同构.

∩	∅	S_a	S_b	1
∅	∅	∅	∅	∅
S_a	∅	S_a	∅	S_a
S_b	∅	∅	S_b	S_b
1	∅	S_a	S_b	1

∪	∅	S_a	S_b	1
∅	∅	S_a	S_b	1
S_a	S_a	S_a	1	1
S_b	S_b	1	S_b	1
1	1	1	1	1

~	
∅	1
S_a	S_b
S_b	S_a
1	∅

由此可见，由 $I=\{a, b\}$ 构成的 $P(I)$ 与 B_4 同构. 可以证明（习题 1），每个布尔代数与一个具有∩(与)、∪(或)和 ~ (非)分别作为与、或和非三个集操作的幂集 $P(A)$ 同构. 因为 $A=\{a_1, a_2, \cdots, a_n\}$ 构成的幂集 $P(A)$ 有 2^n 个元素（习题 2），因此，一个布尔代数的元素数也是 2 的幂. 换言之，对于一个给定的布尔代数 B，存在一个正整数 n，且 B 的元素数为 2^n.

在研究其他的代数时，我们关心的是这些代数所定义的函数特性，在研究布尔代数时也是这样. 布尔函数是布尔代数向其自身的一个映射. 下面我们用所谓"递归"法，首先定义两个称为常函数和射影函数的简单函数，随后逐步给出其他的布尔函数的定义.

定义 1.1.2

布尔代数 B 的一个元素称为 B 上的常量.

定义 1.1.3

表示 B 中任何一个元素的符号称为 B 上的(布尔)变量.

下面定义布尔代数的布尔函数.

定义 1.1.4

设 $x_1, x_2, \cdots, x_n$ 是布尔代数 B 的变量. 如果 f 能按照下述规则建立，则 B 向其自身映射的结果，f 成为 n 变量的布尔函数，用 $f(x_1, \cdots, x_n)$ 表示.

1. 设 a 为 B 的一个常数. $f(x_1, \cdots, x_n)=a$、$f(x_1, \cdots, x_n)=x_i$ 均为布尔函数，前者称为常函数，后者称为射影函数.

2. 如果 $f(x_1, \cdots, x_n)$ 是布尔函数，则 $(f(x_1, \cdots, x_n))'$ 也是布尔函数.

3. 如果 $f_1(x_1, \cdots, x_n)$ 和 $f_2(x_1, \cdots, x_n)$ 都是布尔函数，则 $f_1(x_1, \cdots, x_n)+f_2(x_1, \cdots, x_n)$ 和 $f_1(x_1, \cdots, x_n)\cdot f_2(x_1, \cdots, x_n)$ 也是布尔函数.

4. 有限次地应用上述规则所构成的任何函数，而且只有这样的函数，才叫布尔函数.

综合上述，可以说布尔函数是由常函数及射影函数经有限次地应用 $\cdot$，$+$，$'$ 等操作所构成的任意函数. 对于单变量函数，其射影函数即为恒同函数 $f(x)=x$.

例 1.1.3

下列函数是定义在 B_4 上的三变量 x_1，x_2 和 x_3 的布尔函数：

$$f(x_1, x_2, x_3)=ax_1+(bx_2x_3')'(x_2+x_3') \tag{1.1.1}$$

$x\cdot y$ 和 $x+y$ 虽然不是普通的代数运算，但我们仍采用普通代数运算中的一些习惯：

1. 略去变量间的点. 例如 $x\cdot y\cdot z$ 可以写成 xyz.

2. 略去布尔积外面的括号. 例如 $(uvw)+(xyz)$ 可以写成 $uvw+xyz$.

每个布尔函数都有如下两种范式:

1. 积-和范式:

$$f(x_1, \cdots, x_n) = \sum f(e_1, \cdots, e_n) x_1^{e_1} x_2^{e_2} \cdots x_n^{e_n} \tag{1.1.2}$$

其中,若 $x_i = e_i$, 则 $x_i^{e_i} = 1$; 若 $x_i \neq e_i$, 则 $x_i^{e_i} = 0$.

2. 和-积范式:

$$f(x_1, \cdots, x_n) = \prod (f(e_1, \cdots, e_n) + x_1^{e_1'} + x_2^{e_2'} + \cdots + x_n^{e_n'}) \tag{1.1.3}$$

其中,若 $x_i = e_i$, 则 $x_i^{e_i'} = 0$; 若 $x_i \neq e_i$, 则 $x_i^{e_i'} = 1$.

对上述两个公式的证明,作为读者的练习(习题 3).

例 1.1.4

设定义在 B_4 上的函数 $f(x_1, x_2)$ 真值表如下:

x_1	x_2	$f(x_1x_2)$
0	0	1
0	a	a
a	b	b
a	1	1
b	0	1
b	a	a
1	b	1
1	1	1
其他		0

那末,这函数的积-和范式为:

$$\begin{aligned} f(x_1, x_2) = {} & 1 \cdot x_1^0 x_2^0 + a \cdot x_1^0 x_2^a + b \cdot x_1^a x_2^b + 1 \cdot x_1^a x_2^1 \\ & + 1 \cdot x_1^b x_2^0 + a \cdot x_1^b x_2^a + 1 \cdot x_1^1 x_2^b + 1 \cdot x_1^1 x_2^1 \end{aligned} \tag{1.1.4}$$

练 习 1.1

1. 求证每个布尔代数都与具有三个集操作 $\cap$、$\cup$ 和 $\sim$ 分别作为与、或和非操作的幂集 $P(A)$同构.
2. 求证由 $A = \{a_1, a_2, \cdots, a_n\}$ 构成的幂集 $P(A)$ 有 2^n 个元素.
3. 证明一个布尔函数的两种范式.

1.2 开 关 代 数

在布尔代数中,最有用的是二元布尔代数 B_2(见例 1.1.1),即开关代数. 它是分析和设计数字系统中的开关电路的数学基础. 它包括由 1 表示的大数和由 0 表示的小数两个界元素. 根据开关代数定义的布尔函数称为开关函数. 本节将讨论开关函数的一系列性质,以及对函数可以进行直接化简的开关函数显逻辑等效表达式.

开关代数的基本性质,根据它所含的变量数可以分为三类,见表 1.2.1.

此为试读,需要完整PDF请访问: www.ertongbook.com

表 1.2.1　逻 辑 方 程 式

	$n=1$*
1.1. $x_1+x_1=x_1$	1.1′. $x_1\cdot x_1=x_1$
1.2. $x_1+x_1'=1$	1.2′. $x_1\cdot x_1'=0$
1.3. $1+x_1=x_1+1=1$	1.3′. $1\cdot x_1=x_1$
1.4. $0+x_1=x_1+0=x_1$	1.4′. $0\cdot x_1=0$
	$n=2$
2.1. $x_1+x_2=x_2+x_1$	2.1′. $x_1\cdot x_2=x_2\cdot x_1$
2.2. $x_1+x_1\cdot x_2=x_1$	2.2′. $x_1\cdot(x_1+x_2)=x_1$
2.3. $(x_1+x_2')\cdot x_2=x_1\cdot x_2$	2.3′. $x_1\cdot x_2'+x_2=x_1+x_2$
	$n=3$
3.1. $x_1+x_2+x_3=(x_1+x_2)+x_3$ $=x_1+(x_2+x_3)$	3.1′. $x_1\cdot x_2\cdot x_3=(x_1\cdot x_2)\cdot x_3$ $=x_1\cdot(x_2\cdot x_3)$
3.2. $x_1\cdot x_2+x_1\cdot x_3=x_1\cdot(x_2+x_3)$	3.2′. $(x_1+x_2)\cdot(x_1+x_3)=x_1+x_2\cdot x_3$
3.3. $(x_1+x_2)\cdot(x_1+x_3)(x_3+x_1')$ $=(x_1+x_2)\cdot(x_3+x_1')$	3.3′. $x_1\cdot x_2+x_2\cdot x_3+x_3x_1'=x_1\cdot x_2+x_3\cdot x_1'$
3.4. $(x_1+x_2)\cdot(x_1'+x_3)=x_1\cdot x_3+x_1'\cdot x_2$	

* n，二进制变量数.

下面介绍几个开关函数的一般定理，其中获·摩根（De Morgan）定理对开关函数十分有用.

定理 1.2.1（一般化的） 获·摩根定理

(a) $(x_1+x_2+\cdots+x_n)'=x_1'\cdot x_2'\cdot\cdots\cdot x_n'$

(b) $(x_1\cdot x_2\cdot\cdots\cdot x_n)'=x_1'+x_2'+\cdots+x_n'$

上述定理的证明，作为读者的练习(习题 1).

获·摩根定理说明互补函数之间的关系是不完善的. 为此，香农（Shannon）对此定理进行了一般化.

定理 1.2.2 香农定理

$$(f(x_1,x_2,\cdots,x_n,+,\cdot))'=f(x_1',x_2',\cdots,x_n,\cdot,+)$$

这就是说，任何函数之非，可以通过函数变量求反，·操作代替＋操作，＋操作代替·操作来取得.

证明：按获·摩根定理，任何 $(f(x_1,\cdots,x_n,+,\cdot))'$ 可以写成

$$\begin{aligned}(f(x_1,\cdots,x_n,+,\cdot))'&=(f_1(x_1,\cdots,x_n,+,\cdot)+f_2(x_1,\cdots,x_n,+,\cdot))'\\&=(f_1(x_1,\cdots,x_n,+,\cdot))'\cdot(f_2(x_1,\cdots,x_n,+,\cdot))'\end{aligned}$$

或者写成

$$\begin{aligned}(f(x_1,\cdots,x_n,+,\cdot))'&=(f_1(x_1,\cdots,x_n,+,\cdot)\cdot f_2(x_1,\cdots,x_n,+,\cdot))'\\&=(f_1(x_1,\cdots,x_n,+,\cdot))'+(f_2(x_1,\cdots,x_n,+,\cdot))'\end{aligned}$$

其中 f_1 和 f_2 是 f 的两部分函数. 对 f_1 和 f_2 重复上述过程，使 f 中的每个变量都用上获·摩根定理. 因为每对 f（或 f 的部分函数）应用一次获·摩根定理，就把部分函数（或子部分函数）求反且变换与或操作而取得函数 f（或部分函数）的非. 因此，当对 f 的每个

变量都进行获·摩根变换后，其结果必然是 $f(x_1', x_2', \cdots, x_n', \cdot, +)$·|*

应用香农定理时，开关函数变量之间的点及函数积外面的括号都不能去掉.

下面举例说明上述定理的应用.

例 1.2.1

若 $f(x_1, x_2, x_3) = (x_1 \cdot x_2)' \cdot x_3 + x_1 \cdot (x_2 + x_3')$，则这函数的非为

$$f'(x_1, x_2, x_3) = ((x_1 \cdot x_2) + x_3') \cdot (x_1' + (x_2' \cdot x_3))$$

下面介绍开关代数的另一个重要定理——展开定理.

定理 1.2.3 展开定理

(a) $f(x_1, x_2, \cdots, x_n) = x_1 \cdot f(1, x_2, \cdots, x_n) + x_1' \cdot f(0, x_2, \cdots, x_n)$

(b) $f(x_1, x_2, \cdots, x_n) = [x_1 + f(0, x_2, \cdots, x_n)][x_1' + f(1, x_2, \cdots, x_n)]$

这就是函数 f 对 x_1 的展开. 对于其他变量的展开式与此类似.

证明：若用 $x_1 = 1$, $x_1' = 0$ 代入 (a) 和 (b) 式，然后，$x_1 = 0$, $x_1' = 1$ 代入 (a) 和 (b) 式，则 (a), (b) 两式均成立.|

应用展开定理，可将 n 变量的任何开关函数展开.

例 1.2.2

函数 $f(x_1, x_2, x_3) = (x_1x_2)'x_3 + x_1(x_2 + x_3')$ 可以写成

$$f(x_1, x_2, x_3) = x_1 \cdot (x_2'x_3 + x_2 + x_3') + x_1' \cdot (x_3)$$

或

$$f(x_1, x_2, x_3) = [x_1 + x_3] \cdot [x_1' + (x_2'x_3 + x_2 + x_3')]$$

从展开定理又可得到如下定理.

定理 1.2.4

(a) (1) $x_1 \cdot f(x_1, x_2, \cdots, x_n) = x_1 \cdot f(1, x_2, \cdots, x_n)$

(2) $x_1 + f(x_1, x_2, \cdots, x_n) = x_1 + f(0, x_2, \cdots, x_n)$

(b) (1) $x_1' \cdot f(x_1, x_2, \cdots, x_n) = x_1' \cdot f(0, x_2, \cdots, x_n)$

(2) $x_1' + f(x_1, x_2, \cdots, x_n) = x_1' + f(1, x_2, \cdots, x_n)$

应用表 1.2.1 中的公式和上述定理 (定理 1.2.1—1.2.4) 可容易地把 n 变量的 2^{2^n} 个开关函数化简为积-和范式. $n = 2$ 的所有开关函数的积-和范式列于表 1.2.2 中. 但当 n 大于 2 时，表格规模迅速增大; $n = 3$，表有 256($= 2^{2^3}$) 行; $n = 4$，表有 65,536($= 2^{2^4}$) 行.

下面对表 1.2.2 作一说明:

表 1.2.2 中第一列说明二变量开关函数的积-和式有 16 种; 第二列指出每种积-和范式有多少项; 第三列又分为四小列，每列对应范式中的一个项，内容可 0 可 1，若为 0，范式中不包括此项，若为 1，范式中包括此项; 第四列给出了由两变量构成的 16 种布尔函数

---

* | 表示证明毕.

表 1.2.2　范式二变量开关函数简化公式

函数数	积-和范式项数	$x_1'x_2'$	$x_1'x_2$	x_1x_2'	x_1x_2	积-和 范 式	简 化 函 数
1	0	0	0	0	0	0	0
2	1	0	0	0	1	x_1x_2	x_1x_2
3	1	0	0	1	0	x_1x_2'	x_1x_2'
4	1	0	1	0	0	$x_1'x_2$	$x_1'x_2$
5	1	1	0	0	0	$x_1'x_2'$	$x_1'x_2'$
6	2	0	0	1	1	$x_1x_2' + x_1x_2$	x_1
7	2	1	1	0	0	$x_1'x_2' + x_1'x_2$	x_1'
8	2	0	1	0	1	$x_1'x_2 + x_1x_2$	x_2
9	2	1	0	1	0	$x_1'x_2' + x_1x_2'$	x_2'
10	2	1	0	0	1	$x_1'x_2' + x_1x_2$	$x_1x_2 + x_1'x_2'$
11	2	0	1	1	0	$x_1'x_2 + x_1x_2'$	$x_1x_2' + x_1'x_2$
12	3	0	1	1	1	$x_1'x_2 + x_1x_2' + x_1x_2$	$x_1 + x_2$
13	3	1	0	1	1	$x_1'x_2' + x_1x_2' + x_1x_2$	$x_1 + x_2'$
14	3	1	1	0	1	$x_1'x_2' + x_1'x_2 + x_1x_2$	$x_1' + x_2$
15	3	1	1	1	0	$x_1'x_2' + x_1'x_2 + x_1x_2'$	$x_1' + x_2'$
16	4	1	1	1	1	$x_1'x_2' + x_1'x_2 + x_1x_2' + x_1x_2$	1

的积-和范式；第五列是第四列的函数的最简式，这两种表示形式虽不相同，但彼此同构，而且构成如图 1.2.1 所示的点阵。

具有三个或少于三个变量的函数，可以用表 1.2.1中的公式和定理 1.2.1—1.2.4 化简。

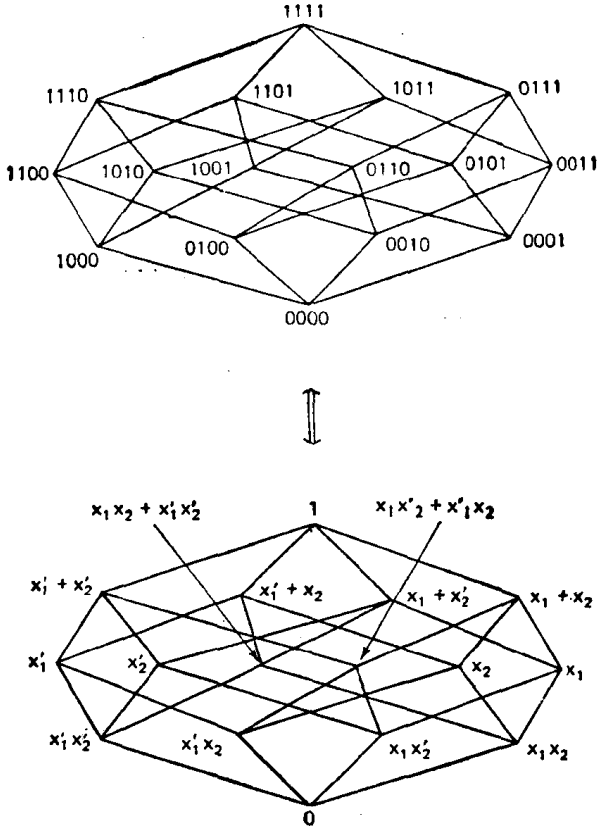


图 1.2.1　16 种简化函数的点阵表示

例如

$$\begin{aligned} f(x_1, x_2, x_3) &= x_1'x_2x_3' + x_1'x_2x_3 + x_1x_2'x_3' + x_1x_2x_3 \\ &= x_1'x_2(x_3' + x_3) + x_1x_2'x_3' \quad (\text{用表 1.2.1 的 1.1}) \\ &\quad + (x_1' + x_1)x_2x_3 \\ &= x_1'x_2 + x_1x_2'x_3' + x_2x_3 \quad (\text{用表 1.2.1 的 1.2}) \end{aligned} \tag{1.2.1}$$

对于三个以上变量的函数化简，此法就不方便了。本章在后面将介绍三种化简 n 变量开关函数的常用方法。

练　习　1.2

1. 证明定理 1.2.1。

2. 用表 1.2.1 中的公式和定理 1.2.1—1.2.4 化简下列函数：

(1) $f_1(x_1, x_2) = x_1x_2' + x_1'x_2$;

(2) $f_2(x_1, x_2) = x_1 + x_1'x_2$;

(3) $f_3(x_1, x_2, x_3) = x_1x_2x_3 + x_1'x_2'x_3 + x_1'x_2x_3' + x_1x_2'x_3' + x_1'x_2'x_3'$;

(4) $f_4(x_1, x_2, x_3) = x_1x_2 + x_2'x_3' + x_1x_2x_3 + x_1x_2'x_3'$;

(5) $f_5(x_1, x_2, x_3) = x_1x_2 + x_1'x_3 + x_2x_3$;

(6) $f_6(x_1, x_2, x_3) = x_1x_3 + x_1x_2x_3 + x_1'x_3 + x_1'x_2'x_3' + x_2x_3$.

1.3　完全定义开关函数和不完全定义开关函数

开关函数 $f(e_1, e_2, \cdots, e_n)$ 的值只能是 0 或 1，因此，式 (1.1.2) 和式 (1.1.3) 可以表示为：

积-和范式：

$$f(x_1, \cdots, x_n) = \sum x_1^{e_1}x_2^{e_2}\cdots x_n^{e_n} \tag{1.3.1}$$

和-积范式：

$$f(x_1, \cdots, x_n) = \prod(x_1^{e_1'} + x_2^{e_2'} + \cdots + x_n^{e_n'}) \tag{1.3.2}$$

式中的符号 $\sum$ 表示使得 $f(e_1, \cdots, e_n) = 1$ 的 $x_1, \cdots, x_n$ 值的所有组合；符号 $\prod$ 表示使得 $f(e_1, \cdots, e_n) = 0$ 的 $x_1, \cdots, x_n$ 值的所有组合。式中 $x_i^{e_i}(x_i^{e_i'})$，根据 e_i 是 1 值或 0 值，分别为 $x_i(x_i')$ 或 $x_i'(x_i)$。例如，对于式(1.2.1)开关函数，$f(x_1, x_2, x_3) = x_1'x_2 + x_1x_2'x_3' + x_2x_3$ 的两个范式分别为：

$$f(x_1, x_2, x_3) = x_1'x_2x_3' + x_1'x_2x_3 + x_1x_2'x_3' + x_1x_2x_3 \tag{1.3.3}$$

和

$$f(x_1, x_2, x_3) = (x_1 + x_2 + x_3)(x_1 + x_2 + x_3')(x_1' + x_2 + x_3')(x_1' + x_2' + x_3) \tag{1.3.4}$$

式 (1.3.1) 中的每个项称为最小项，式 (1.3.2) 中的每个项称为最大项。根据开关函数的范式定义[式 (1.3.1) 和 (1.3.2)]，积-和范式中的最小项可以从函数真值表中映射等于 1 的各行得到，和-积范式中的最大项可以从函数真值表中映射等于 0 的各行得到。表1.3.1 是一简单的例子。比如说表中的 010 行，对应于而且只能对应于唯一的一个最小项 $x_1'x_2x_3'$。这是因为当且仅当 $x_1 = 0$, $x_2 = 1$ 和 $x_3 = 0$ 时，才有 $x_1'x_2x_3' = 1$。对于其他三行 011，100 和 111，同它们相对应的只能是 $x_1'x_2x_3$, $x_1x_2'x_3'$ 和 $x_1x_2x_3$。由此可以得出以下规

则：

规则 1.3.1

一个开关函数的积-和范式最小项，可从其真值表映射等于 1 的各行得到．变量的 0 值表示该变量的非，变量的 1 值表示该变量本身．

同样，表中的 000 行对应且仅对应的最大项是 $x_1 + x_2 + x_3$，这是因为当且仅当 $x_1 = 0$，$x_2 = 0$ 和 $x_3 = 0$ 时，才有 $x_1 + x_2 + x_3 = 0$．其他 001，101 和 110 三行的最大项分别为 $x_1 + x_2 + x_3'$，$x_1' + x_2 + x_3'$ 和 $x_1' + x_2' + x_3$ 的道理与上面相同．由此，我们可以得出：

规则 1.3.2

一个开关函数的和-积范式最大项，可从其真值表映射等于 0 的各行得到．变量的 0 值表示该变量本身，变量的 1 值表示该变量的非．

表 1.3.1

行数 r	最小项和最大项	x_1 x_2 x_3	式 (1.2.1) 的 $f(x_1, x_2, x_3)$
0	$M_0 = x_1 + x_2 + x_3$	0 0 0	0
1	$M_1 = x_1 + x_2 + x_3'$	0 0 1	0
2	$m_2 = x_1'x_2x_3'$	0 1 0	1
3	$m_3 = x_1'x_2x_3$	0 1 1	1
4	$m_4 = x_1x_2'x_3'$	1 0 0	1
5	$M_5 = x_1' + x_2 + x_3'$	1 0 1	0
6	$M_6 = x_1' + x_2' + x_3$	1 1 0	0
7	$m_7 = x_1x_2x_3$	1 1 1	1

真值表中各行的变量值与最小项和最大项之间有一一对应关系，因此可以得到一个最小项和最大项的简单表示法．习惯上用 m 表示最小项，用 M 表示最大项．另外，十进制和二进制的相互转换是唯一的，所以真值表的行不用二进制 000，001，010 等表示，而用 0，1，2 等表示．这样，与之对应的最小项和最大项，就可以用带有下标 r 的 m 和 M 表示（见表 1.3.1 的第一列和第二列）．根据这些规定，式 (1.3.3) 和 (1.3.4) 可以表示为：

$$f(x_1, x_2, x_3) = m_2 + m_3 + m_4 + m_7 \tag{1.3.5}$$

和

$$f(x_1, x_2, x_3) = M_0 \cdot M_1 \cdot M_5 \cdot M_6 \tag{1.3.6}$$

把式 (1.3.5) 和 (1.3.6) 进一步简化，得到

$$f(x_1, x_2, x_3) = \sum(2, 3, 4, 7)$$

和

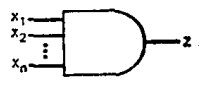
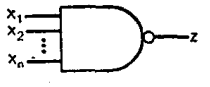
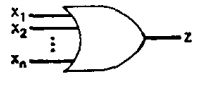
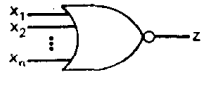
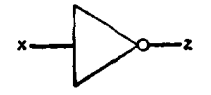
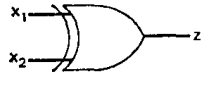
$$f(x_1, x_2, x_3) = \Pi(0, 1, 5, 6)$$

我们看到，函数值为 0 的 0，1，5 和 6 行不包括在和式 $\sum(2, 3, 4, 7)$ 中，函数值为 1 的 2，3，4 和 7 行不包括在积式 $\Pi(0, 1, 5, 6)$ 中．这就是说，如果这行是最小项（最大项），那它就不可能是最大项（最小项）．换言之，一个完全定义的开关函数的最小项的集和最大项的集是互斥的．因此，若知一个开关函数的积-和（和-积）范式，那末，这开关函数的和-

积(或-和)范式，就可以从最大项(最小项)的集中并应用规则 1.3.1 (1.3.2) 得到.

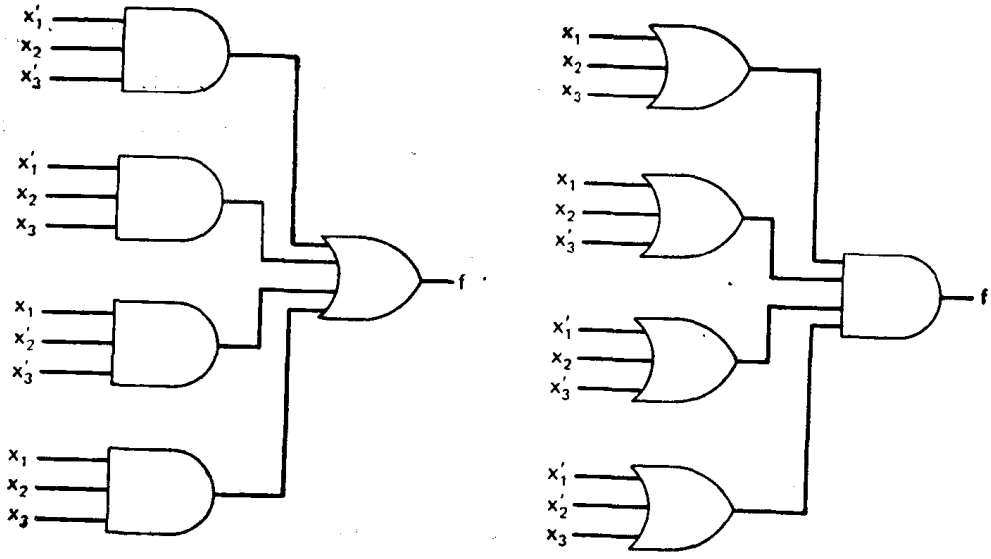
表1.3.2给出了与、或、非、与非、或非和异或等六种基本门符号. 这些符号是根据美国兵役局标准符号MIL-STD-806B画的. 与门、或门和非门分别实现"与"操作、"或"操作和"禁止"操作;与非门和或非门分别实现"与非"和"或非"操作;具有 x 和 y 两输入端的异或门实现 $xy' + x'y$ 操作. 与非门、或非门和异或门将在第三章讨论. 用与门、或门和非门作为基本单元，可以组成两种开关函数实现，即二级与或实现和二级或与实现，它们分别实现函数的积-和范式和和-积范式. 在上述实现中，输入变量可以为原码，也可以为反码. 例如，图 1.3.1 给出了公式 (1.3.3) 和 (1.3.4) 的二级实现. 这种二级电路具有对逻辑设计者很有用的两个特性:

表 1.3.2　美国兵役局标准符号
MIL-STD-806B

门	符号	门	符号
与		与非	
或		或非	
非		异或	

1. 因为电路只有两级门，所以电路响应时间近似等于单个门响应时间的两倍，这比多级门电路的响应时间短.

2. 在后面的章节中将会看到，二级电路不仅容易构成，而且故障检测和定位都很容易.



(a) 式 (1.3.3) 的二级与或实现　　(b) 式 (1.3.4) 的二级或与实现

图 1.3.1　式 (1.3 3) 和 (1.3.4) 函数的基本二级实现的例子

此为试读，需要完整PDF请访问：www.ertongbook.com