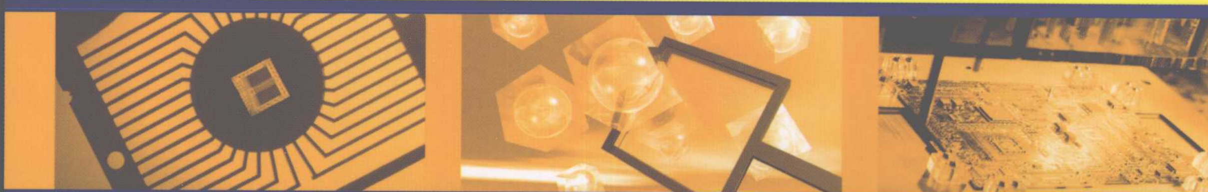
高职高专“十一五”电子信息类专业规划教材

数字电子技术及应用

SHUZI DIANZI JISHU JI YINGYONG



刘淑英 主编

赠送电子教案

机械工业出版社
CHINA MACHINE PRESS

高职高专“十一五”电子信息类专业规划教材

数字电子技术及应用

主　编　刘淑英

副主编　刘恩华　谢子青

参　编　马宁丽　胡建明　白　静

主　审　殷建国

机　械　工　业　出　版　社

本书的主要内容有：数字电路基础、逻辑门电路、组合逻辑电路、集成触发器、时序逻辑电路、脉冲波形的产生与整形电路、数/模和模/数转换器、半导体存储器和可编程序逻辑器件等。配合教学的实验内容穿插在相应的理论教学过程中，最后配有的综合实训使理论与实践紧密结合。本书始终贯彻“讲、学、练”相结合的原则，从能力培养的角度出发，培养学生分析问题和解决问题的能力。此外，每章都配有本章小结和习题，书末还配有部分习题的参考答案。

本书内容简明，通俗易懂，由浅入深，重点突出，理论联系实际。本书可作为高职高专院校应用电子技术、电气自动化技术和机电一体化技术等专业的教材或参考书，也可供从事电子技术工作的工程技术人员参考。

为方便教学，本书配有免费电子教案，凡选用本书作为授课教材的学校，均可来电索取，咨询电话：010-88379564。

图书在版编目(CIP)数据

数字电子技术及应用/刘淑英主编. —北京：机械工业出版社，2007.5

高职高专“十一五”电子信息类专业规划教材

ISBN 978-7-111-21488-5

Ⅰ.数... Ⅱ.刘... Ⅲ.数字电路—电子技术—高等学校：技术学校—教材 Ⅳ.TN79

中国版本图书馆 CIP 数据核字(2007)第 069284 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：于　宁　责任编辑：曲世海　版式设计：冉晓华

责任校对：陈延翔　封面设计：王伟光　责任印制：李　妍

北京铭成印刷有限公司印刷

2007 年 7 月第 1 版第 1 次印刷

184mm×260mm · 12.75 印张 · 315 千字

0 001—4 000 册

标准书号：ISBN 978-7-111-21488-5

定价：20.00 元

前　言

当今，电子技术迅猛发展，中大规模集成电路的发展和应用尤为突出，如可编程序逻辑器件，其出现的时间虽然不长，但已在各个领域得到了广泛的应用。所有这些给数字电子技术课程的教学提出了更高的要求，需要有新的教学内容、教学方法和教学手段与之相适应，为此我们编写了本书。

本书的主要内容有：数字电路基础、逻辑门电路、组合逻辑电路、集成触发器、时序逻辑电路、脉冲波形的产生与整形电路、数/模和模/数转换器、半导体存储器和可编程序逻辑器件及综合实训等。本书编写的指导思想是：保证基础知识，理论联系实际，注重能力培养。在内容安排上，突出基本理论、基本概念和基本分析方法；理论以够用为度，回避了集成电路内部繁琐的分析和推导；以器件的外特性及应用为主，注重实际应用和能力的培养；将相应的实验内容穿插在相应的教学过程中，理论与实践紧密结合。本书的最后一章为综合实训，通过对实际电路的设计、安装及调试，提高学生分析问题和解决问题的能力。本书概念清晰准确，通俗易懂，由浅入深，循序渐进，方便自学。每章后面都附有本章小结及习题，书后配有部分习题的参考答案，并配有与本书配套的电子教案，有利于组织教学和学生自学。

本书可作为高职高专院校应用电子技术、电气自动化技术和机电一体化技术等专业的数字电子技术课程用书，也可供从事电子技术工作的工程技术人员参考。

本书由大连职业技术学院刘淑英任主编，负责全书的组织、统稿工作，并编写了第1章、第2章和第6章；江苏信息职业技术学院刘恩华任副主编，并编写了第3章；浙江工业职业技术学院谢子青任副主编，并编写了第5章；西安航空职业技术学院马宁丽编写了第4章及附录；重庆三峡职业学院胡建明编写了第8章和第9章；烟台职业学院白静编写了第7章；与本书配套的电子教案由烟台职业学院白静制作。

本书由大连职业技术学院殷建国主审。本书在编写过程中得到了编者所在单位领导、老师及企业技术人员的大力支持，在此一并表示感谢。

由于软件本身的原因，本书中的某些逻辑符号采用的是非国标符号，具体对应关系请参照附录B。

为方便教学，本书配有免费电子教案，凡选用本书作为教材的学校，均可来电索取，咨询电话：010-88379564。

由于编者水平有限，书中难免存在一些错误和不妥之处，敬请读者批评指正。

编　者

目　　录

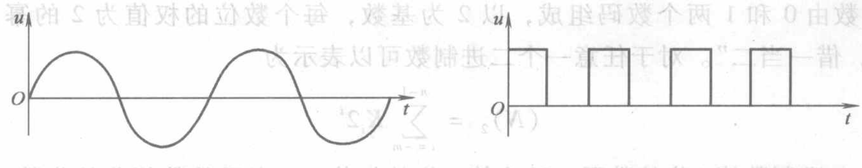
第 1 章　数字电路基础

内容提要： 本章主要介绍数字电路的特点，常用的数制及其相互转换方法，三种基本逻辑关系，逻辑函数的表示方法及其相互转换，逻辑代数的基本定律和规则，逻辑函数的两种化简方法——代数法和卡诺图法。

1.1 概述

1.1.1　数字信号和数字电路

在电子电路中可将所处理的信号分成两大类：一类是模拟信号，另一类是数字信号。所谓模拟信号是指在时间上和数值上都连续变化的信号，如温度、压力和速度等物理量通过传感器变成的电信号都是模拟信号；所谓数字信号是指在时间上和数值上都间断的、不连续变化的信号，如记录生产零件个数的记录信号、灯光闪烁等信号都属于数字信号。典型的模拟信号是正弦波，典型的数字信号是方波，如图 1-1 所示。用来传输和处理模拟信号的电路为模拟电路，传输和处理数字信号的电路为数字电路。



a）模拟信号　　　b）数字信号

图 1-1　模拟信号和数字信号

1.1.2　数字电路的特点

① 数字电路中的信号为数字信号，所以当电路处于稳定状态时，电路中的半导体器件（如二极管、晶体管）均工作在开关状态，即工作在饱和导通状态或截止状态。

② 数字信号只有两种状态，即有信号或无信号，而这两种状态可用 0 或 1 两个数码表示，因此用来表示或存储信号的电路比较简单，构成的单元电路也较简单，对元器件的精度要求不高，电路容易制造，集成度较高，成本较低。

③ 数字电路除了具有一定的“逻辑思维”能力外，还具有记忆功能，能够长期存储一定数量的信号，同时还可以采用标准的逻辑器件和可编程序逻辑器件来构成各种各样的数字系统，使用方便，通用性强，便于大批量生产，并可进行加密处理，使大量的有用信息资源得以长期保存。而且数字电路多采用集成电路，焊点少，连线少，工作可靠性高。

④ 数字电路研究的主要问题是电路的输入与输出之间的逻辑关系。

1.2 数制与码制

1.2.1 数制

数制是计数进位制的简称。在日常生活和生产中，人们习惯用十进制数，而在数字电路和计算机中，只能识别由“0”和“1”组成的数码，所以经常采用二进制数和十六进制数，有些地方采用八进制数。

1. 十进制数

十进制数有 0 ~ 9 十个数码，以 10 为基数。计数时，“逢十进一，借一当十”。数码在不同的位置代表的实际大小不同，例如，十进制数 616 可表示为

$$616 = 6 \times 10^2 + 1 \times 10^1 + 6 \times 10^0$$

式中，10^2、10^1、10^0 分别为百位、十位和个位的“位权”，简称为“权”，即相应位的数码所代表的实际数值，位数越高，权值越重。对于任意一个十进制数可以表示为

$$(N)_{10} = \sum_{i=-m}^{n-1} K_i 10^i \tag{1-1}$$

式中，K_i 为十进制数第 i 位的数码；n 表示整数部分的位数；m 表示小数部分的位数，n、m 都是正整数；10^i 为第 i 位的权值。例如，十进制数 209.04 可表示为

$$(209.04)_{10} = 2 \times 10^2 + 0 \times 10^1 + 9 \times 10^0 + 0 \times 10^{-1} + 4 \times 10^{-2}$$

2. 二进制数

二进制数由 0 和 1 两个数码组成，以 2 为基数，每个数位的权值为 2 的幂。计数时，“逢二进一，借一当二”。对于任意一个二进制数可以表示为

$$(N)_2 = \sum_{i=-m}^{n-1} K_i 2^i \tag{1-2}$$

式中，K_i 为二进制数第 i 位的数码；2^i 为第 i 位的权值；n 表示整数部分的位数；m 表示小数部分的位数，n、m 都是正整数。例如，二进制数 101.01 可以展开成

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

3. 十六进制数

十六进制有 0 ~ 9 和 A(10)、B(11)、C(12)、D(13)、E(14)、F(15)共十六个数码，以 16 为基数，每个数位的权值为 16 的幂。计数时，“逢十六进一，借一当十六”。对于任意一个十六进制数可以表示为

$$(N)_{16} = \sum_{i=-m}^{n-1} K_i 16^i \tag{1-3}$$

式中，K_i 为十六进制数第 i 位的数码；16^i 为第 i 位的权值；n 表示整数部分的位数；m 表示小数部分的位数，n、m 都是正整数。例如，十六进制数 D8.A 可以展开成

$$(D8.A)_{16} = 13 \times 16^1 + 8 \times 16^0 + 10 \times 16^{-1}$$

4. 八进制数

八进制数有 0 ~ 7 八个数码，以 8 为基数，每个数位的权值为 8 的幂。计数时，“逢八进一，借一当八”。任意一个八进制数按权展开的方法与二进制数、十进制数和十六进制数相

同，在此不再赘述。

1.2.2　二进制数与其他进制数的相互转换

1. 二进制数与十进制数的相互转换

（1）将二进制数转换为十进制数　按式(1-2)将二进制数按权展开后相加，即得与其等值的十进制数。

例 1-1　将$(101.01)_2$转换为十进制数。

解：$(101.01)_2 = 1\times2^2+0\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2}=(5.25)_{10}$

所以　$(101.01)_2=(5.25)_{10}$

(2) 将十进制数转换为二进制数　要将任意一个十进制数转换为二进制数，可将其整数部分和小数部分分别转换。整数部分采用“除 2 取余”法，即将给定的十进制数的整数部分依次被 2 除，所得余数自下而上排列起来；小数部分采用“乘 2 取整”法，即将给定的十进制数的小数部分依次被 2 乘，所得整数部分自上而下排列起来；最后将整数部分和小数部分组合到一起，即为对应的二进制数，小数点位置不变。

例 1-2　将$(44.375)_{10}$转换为二进制数。

解：整数部分 44 用“除 2 取余”法，小数部分 0.375 用“乘 2 取整”法。

		余数
2	44	
2	22	…… 0
2	11	…… 0
2	5	…… 1
2	2	…… 1
2	1	…… 0
	0	…… 1

	整数
0.375	
× 2	
0.750	…… 0
0.750	
× 2	
1.500	…… 1
0.500	
× 2	
1.000	…… 1
0.000	

最后结果为　$(44.375)_{10}=(101100.011)_2$

2. 二进制数与八进制数的相互转换

(1) 将二进制数转换为八进制数　将给定的二进制数每三位分一组，每组用一位八进制数表示。分组时应注意以小数点为界，向左对整数部分分组，向右对小数部分分组，不足三位的要加 0 补齐。

例 1-3　将$(1101001.1001)_2$转换为八进制数。

解：$(1101001.1001)_2$

$=(\ 001\quad 101\quad 001\ .\ 100\quad 100\ )_2$

$=(\ 1\quad 5\quad 1\ .\ 4\quad 4\ )_8$

所以$(1101001.1001)_2=(151.44)_8$

(2) 将八进制数转换为二进制数　将每位八进制数用三位二进制数表示，小数点位置不变。

例 1-4　将$(52.4)_8$转化为二进制数。

解：$(\ 5\quad 2\ .\ 4\ )_8$

$=(\ 101\quad 010\ .\ 100\ )_2$

所以$(52.4)_8 = (101010.100)_2$

3. 二进制数与十六进制数的相互转换

(1) 将二进制数转换为十六进制数　将给定的二进制数每四位分一组，每组用一位十六进制数表示。注意事项与二进制数转换为八进制数相同。

例 1-5 将$(111010100.011)_2$转化为十六进制数。

解： $(111010100.011)_2$

$= (0001\quad 1101\quad 0100\quad .\quad 0110)_2$

$= (1\quad D\quad 4\quad .\quad 6)_{16}$

所以$(111010100.011)_2 = (1D4.6)_{16}$

(2) 将十六进制数转换为二进制数　将每位十六进制数用四位二进制数表示，小数点位置不变。

例 1-6 将$(3A.5)_{16}$转化为二进制数。

解： $(3\quad A\quad .\quad 5)_{16}$

$= (0011\quad 1010\quad .\quad 0101)_2$

所以$(3A.5)_{16} = (00111010.0101)_2$

1.2.3　码制

数字系统中二进制数码不仅可以表示数值的大小，而且也可以表示特定的信息和符号。将若干位二进制数码按一定规律排列起来，用以表示特定信息的代码，称为二进制代码。用四位二进制数码表示一位十进制数，称为二—十进制码，简称 BCD 码，常用的 BCD 码分为有权码和无权码两类。

有权码用代码的权值命名，如 8421 码自左至右的权值为 8、4、2、1，它与普通的四位二进制数的权值相同，但在 8421 码中不允许出现 1010 ~ 1111 六种状态，只能用 0000 ~ 1001 十种状态，分别代表 0 ~ 9 十个数码。除 8421 码外，有权码还有 2421 码和 5421 码，其中 8421 码最为常用。

无权码每位无确定的权值，不能使用权展开式，但各有其特点和用途。例如格雷码(又叫循环码、反射码)，其相邻的两个编码只有一位码状态不同，在逻辑函数卡诺图化简中将会用到这一特点。表 1-1 列出了几种常用的 BCD 码。

表 1-1　常用的 BCD 码

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

1.3 逻辑代数

1.3.1 逻辑代数和逻辑变量

逻辑代数是由英国数学家乔治·布尔于 19 世纪提出来的，因此也称为布尔代数，它是一种描述事物逻辑关系的数学方法，是分析和设计数字电路的重要数学工具。在逻辑代数中逻辑变量同普通变量一样也是用英文字母表示，如 A、B、C、…、X、Y、Z 等，但与普通变量不同的是逻辑变量取值只有 0、1 两种可能，而且 0 和 1 并不表示具体的数值大小，只是表示两种相互对立的逻辑状态，如电灯的亮和灭，电动机的旋转与停止，电位的高和低等。

1.3.2 三种基本逻辑运算和复合逻辑运算

在逻辑代数中最基本的逻辑关系有“与”、“或”和“非”三种，与之对应的也有三种最基本的逻辑运算：与运算、或运算和非运算。

1. 与逻辑及与运算

当决定某件事的所有条件全部具备时，这件事才发生，否则这件事就不发生，这种因果关系称为与逻辑关系。如图 1-2 所示电路为串联开关电路，A、B 是两个串联开关，L 是灯，只有当 A、B 开关都闭合时，电灯才亮，而若有一个开关打开，则灯就熄灭，这种灯的亮灭与开关通断之间的关系为与逻辑关系。

如果用 1 表示灯亮和开关闭合，用 0 表示灯灭和开关断开，那么可得如表 1-2 所示的与逻辑真值表。

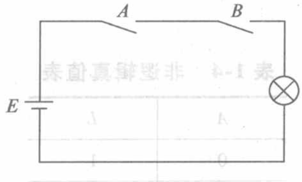


图 1-2　串联开关电路

表 1-2　与逻辑真值表

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

若用逻辑表达式来描述与逻辑，则可写成

$$L = A \cdot B \tag{1-4}$$

这里的“·”号表示与运算，通常可省略，简写为 $L = AB$。与运算的运算规则为：

$$0 \cdot 0 = 0,\ 0 \cdot 1 = 0,\ 1 \cdot 0 = 0,\ 1 \cdot 1 = 1$$

从上面的分析可以看出，与运算规则与普通代数中的乘法规则相似，所以与运算又称为逻辑乘。用以实现与运算的电路称为与门电路，简称与门。其逻辑符号如图 1-3 所示。

2. 或逻辑及或运算

当决定某件事的几个条件中，只要有一个或一个以上条件具备，这件事就会发生，否则就不发生，这种因果关系称为或逻辑关系。图 1-4 所示电路为并联开关电路，开关 A、B 只要有一个闭合，

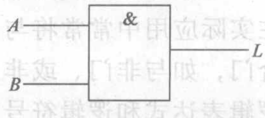


图 1-3　与门逻辑符号

电灯 L 就亮，只有当 A、B 都打开时灯才熄灭，这种灯的亮灭与开关通断之间的关系为或逻辑关系。若仍用 1 表示灯亮和开关闭合，用 0 表示灯灭和开关断开，则可得如表 1-3 所示的或逻辑真值表。

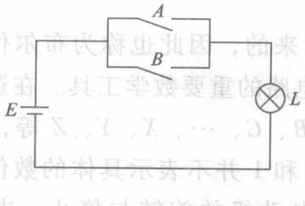


图 1-4　并联开关电路

表 1-3　或逻辑真值表

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

或逻辑的逻辑表达式为：

$$L = A + B \tag{1-5}$$

这里的“+”号表示或运算，或运算的运算规则为：

$$0+0=0,\ 0+1=1,\ 1+0=1,\ 1+1=1$$

由上述分析可以看出，或运算规则与普通代数的加法规则相似，所以或运算又称为逻辑加。要注意的是或运算与二进制的加法运算有所不同，尤其注意 $1+1=1$。用来实现或运算的电路称为或门电路，简称或门，其逻辑符号如图 1-5 所示。

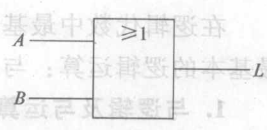


图 1-5　或门逻辑符号

3. 非逻辑及非运算

在某一事件中，若结果总是和条件呈相反状态，则这种逻辑关系称为非逻辑关系。图 1-6所示电路为开关与灯并联电路，当开关闭合时电灯熄灭，当开关断开时电灯亮。非逻辑真值表如表 1-4 所示。

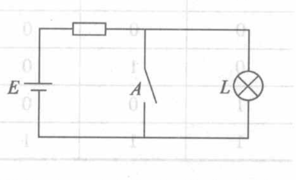


图 1-6　开关与灯并联电路

表 1-4　非逻辑真值表

A	L
0	1
1	0

非逻辑的逻辑表达式为

$$L = \overline{A} \tag{1-6}$$

$\overline{A}$ 读作“A 非”或“A 反”，非运算的运算规则为：$\overline{0}=1$，$\overline{1}=0$。非运算也称“反运算”。用来实现非运算的电路称为非门电路，简称非门，其逻辑符号如图 1-7 所示。

4. 常见的几种复合逻辑关系

与运算、或运算和非运算是逻辑代数中最基本的三种运算，在实际应用中常常将与门、或门和非门组合起来，形成常用的复合门，如与非门、或非门、与或非门、异或门以及同或门等，其逻辑表达式和逻辑符号如表 1-5 所示。

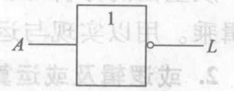
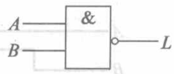
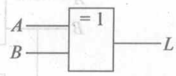
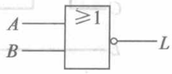
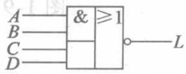
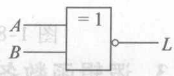


图 1-7　非门逻辑符号

表 1-5 常见的复合逻辑关系

复合逻辑关系名称	逻辑表达式	逻辑符号	复合逻辑关系名称	逻辑表达式	逻辑符号
与非	$L=\overline{A\cdot B}$		异或	$L=\overline{A}B+A\overline{B}$	
或非	$L=\overline{A+B}$				
与或非	$L=\overline{AB+CD}$		同或	$L=\overline{A}\,\overline{B}+AB$	

1.3.3 逻辑函数的表示方法及相互转换

1. 逻辑函数

在前面讨论的逻辑关系中，当输入逻辑变量的取值确定之后，输出逻辑变量的取值也就被相应地确定了，输出逻辑变量与输入逻辑变量之间存在一定的对应关系，我们将这种对应关系称为逻辑函数。

2. 逻辑函数的表示方法

逻辑函数的表示方法较多，常见的有逻辑函数表达式、真值表、逻辑图和波形图等。

（1）逻辑函数表达式　用与、或、非等逻辑运算表示逻辑函数中各个变量之间逻辑关系的代数式称为逻辑函数表达式或逻辑表达式。式(1-4)、式(1-5)、式(1-6)是最基本的逻辑函数表达式，在逻辑函数表达式中单个字母上无非号的为原变量，如 A、B、C 等，有非号的为反变量，如 $\overline{A}$、$\overline{B}$、$\overline{C}$ 等。

（2）真值表　真值表是以表格的形式反映输入逻辑变量的取值组合与函数值之间的对应关系。它的特点是直观、明了，特别是在把一个实际逻辑问题抽象为数学问题时，使用真值表最为方便。

真值表的列写方法：每一个变量均有 0、1 两种取值，n 个变量共有 2^n 种不同的取值，将这 2^n 种不同的取值按顺序(一般按二进制递增顺序)排列起来，同时在相应位置上填入函数的值，便可得到逻辑函数的真值表。

例 1-7　某逻辑函数有 A、B 两个变量，当 A、B 取值相同时输出为 1，不相同时输出为 0，列出真值表。

解：因为函数有两个变量，所以共有四种不同的取值，根据题意确定每组变量取值对应的函数值，得表 1-6。

表 1-6 例 1-7 真值表

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

（3）逻辑图　将逻辑函数表达式中的与、或、非等运算关系用相应的逻辑符号表示出来，就形成逻辑函数的逻辑图。

例 1-8　画出 $L=AB+BC$ 的逻辑图。

解：逻辑函数 L 有三个输入变量，其中 AB、BC 为与运算，由与门实现，各与运算的输出再作或运算，由或门实现，其逻辑图如图 1-8 所示。

（4）波形图　波形图是反映输入变量和输出变量变化规律的图形，图 1-9 是在给定 A、B、C 波形之后画出的函数 $L=AB+BC$ 的波形。

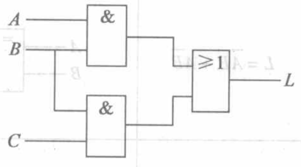


图 1-8　例 1-8 逻辑图

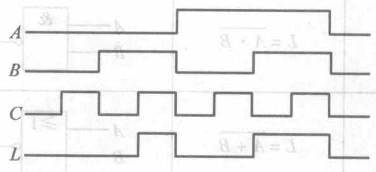


图 1-9　例 1-8 波形图

3. 逻辑函数各种表示方法间的相互转换

一个逻辑函数有几种不同的表示方法，各种表示方法之间可以相互转换，其转换方法通过下面几个例题来说明。

例 1-9　已知逻辑函数表达式 $L=B+\overline{A}C$，求真值表。

解：因为函数 L 有三个变量，故有 $2^3=8$ 种不同的组合状态。先列出 8 种组合状态，再将每种组合状态对应的输入变量代入表达式中进行逻辑运算，求出函数值，最后得出真值表，如表 1-7 所示。

例 1-10　已知逻辑函数真值表如表 1-8 所示，求逻辑函数表达式。

表 1-7　例 1-9 真值表

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

表 1-8　例 1-10 真值表

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

解：根据真值表求表达式时，只需将函数值 $L=1$ 对应的输入变量以逻辑乘的形式表示（1 用原变量表示，0 用反变量表示），再将所有 $L=1$ 对应的乘积项进行逻辑加运算，即为逻辑函数表达式，所以有

$$L=\overline{A}\,\overline{B}C+\overline{A}B\overline{C}+A\overline{B}\,\overline{C}+ABC$$

例 1-11　已知函数 L 的逻辑图如图 1-10 所示，写出函数 L 的逻辑表达式。

解：根据给定的逻辑图写表达式时，可逐级写出输出端的函数表达式。

$$L_1=A\overline{B}C，L_2=A\overline{B}\,\overline{C}，L_3=\overline{A}\,\overline{B}C$$

最后得到函数 L 的表达式为　$L=A\overline{B}C+A\overline{B}\,\overline{C}+\overline{A}\,\overline{B}C$

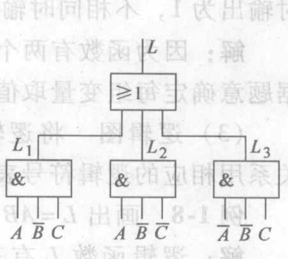


图 1-10　例 1-11 逻辑图

1.4 逻辑代数的基本定律和规则

1.4.1 基本定律

1. 常量之间的关系

$$0 \cdot 0 = 0 \qquad 1 + 1 = 1$$

$$0 \cdot 1 = 0 \qquad 0 + 1 = 1 \qquad 1 + 0 = 1$$

$$1 \cdot 1 = 1 \qquad 0 + 0 = 0$$

$$\overline{0} = 1 \qquad \overline{1} = 0$$

2. 常量和变量之间的关系

$$A + 0 = A \qquad A \cdot 1 = A$$

$$A + 1 = 1 \qquad A \cdot 0 = 0$$

$$A + \overline{A} = 1 \qquad A \cdot \overline{A} = 0$$

3. 变量之间的关系

(1) 交换律 $A + B = B + A$ $\quad A \cdot B = B \cdot A$

(2) 结合律 $(A + B) + C = A + (B + C)$ $\quad (A \cdot B) \cdot C = A \cdot (B \cdot C)$

(3) 分配律 $A \cdot (B + C) = AB + AC$ $\quad A + B \cdot C = (A + B) \cdot (A + C)$

(4) 重叠律 $A + A = A$ $\quad A \cdot A = A$

(5) 反演律 $\overline{A + B} = \overline{A} \cdot \overline{B}$ $\quad \overline{AB} = \overline{A} + \overline{B}$

(6) 还原律 $\overline{\overline{A}} = A$

以上定律的正确性可以用真值表证明，若等式两边的真值表相同，则等式成立。具体证明过程读者可以自己做。

4. 几个常用公式

$$AB + A\overline{B} = B \qquad A + AB = A$$

$$A + \overline{A}B = A + B \qquad AB + \overline{A}C + BC = AB + \overline{A}C$$

由上面最后一个式子可得出推论：$AB + \overline{A}C + BCD = AB + \overline{A}C$

1.4.2 三个重要规则

1. 代入规则

在任何一个逻辑等式中，如果将等式两边的某一变量都用一个函数代替，那么等式依然成立，这个规则称为代入规则。利用代入规则，可以将上述反演律推广到三个变量。如已知等式$\overline{AB} = \overline{A} + \overline{B}$，若用 $Z = CD$ 代替等式中的 B，则：

$$\overline{AZ} = \overline{A} + \overline{Z}$$

$$\overline{ACD} = \overline{A} + \overline{CD} = \overline{A} + \overline{C} + \overline{D}$$

同理可将变量个数推广到 n 个。

2. 反演规则

若求一个逻辑函数 L 的反函数，则只需将函数中所有的“·”换成“+”，“+”换成“·”；“0”换成“1”，“1”换成“0”；原变量换成反变量，反变量换成原变量，则所得到

的逻辑函数式就是逻辑函数 L 的反函数 $\overline{L}$。利用反演规则可以很方便地求出一个逻辑函数的反函数。但在运用反演规则时必须注意运算符号的先后顺序，即先括号、后与、最后或，另外单个变量的反变量应变成原变量，其余的多层非号应保留。

例 1-12 求 $L_1 = A\overline{B} + C\,\overline{D}E$ 和 $L_2 = A + \overline{B + \overline{C} + \overline{\overline{D + \overline{E}}}}$的反函数。

解：

$$\overline{L_1} = (\overline{A} + B)(\overline{C} + D + \overline{E})$$

$$\overline{L_2} = \overline{A} \cdot \overline{\overline{B} \cdot C \cdot \overline{\overline{D}} \cdot E}$$

3. 对偶规则

对于任意一个逻辑函数 L，将函数中的“·”换成“+”，“+”换成“·”；“0”换成“1”，“1”换成“0”，将所得到新的逻辑函数式记作 L'，L'是 L 的对偶式。

对于两个逻辑函数，如果原函数相等，那么其对偶式、反函数也相等。

例 1-13 求 $L_1 = A\overline{B} + C\overline{D}E$ 和 $L_2 = \overline{\overline{A} \cdot \overline{B \cdot \overline{C}}}$的对偶式。

解：

$$L_1' = (A + \overline{B})(C + \overline{D} + E)$$

$$L_2' = \overline{\overline{A} + \overline{B + \overline{C}}}$$

1.5 逻辑函数的代数化简法

1.5.1 逻辑函数常用的几种表达式

在逻辑关系不变的前提下，逻辑函数的表达式不是惟一的，常用的逻辑函数表达式有如下几种：

$L = A\overline{B} + BC$　　与-或表达式

$L = (A + B)(\overline{B} + C)$　　或-与表达式

$L = \overline{\overline{A\overline{B}} \cdot \overline{BC}}$　　与非-与非表达式

$L = \overline{\overline{A + B} + \overline{\overline{B} + C}}$　　或非-或非表达式

$L = \overline{\overline{A}\,\overline{B} + B\overline{C}}$　　与-或-非表达式

在逻辑关系确定之后，真值表是惟一的，由真值表得到的表达式为与-或表达式，而且与-或表达式也容易转换成其他形式，因此下面将重点讨论与-或表达式的化简方法。

1.5.2 化简的意义及最简的概念

对于一个与-或表达式，其形式也不是惟一的，有繁简之分，如：

$$\begin{aligned} L &= AB + \overline{B}C \\ &= AB(C + \overline{C}) + (A + \overline{A})\overline{B}C \\ &= ABC + AB\overline{C} + A\overline{B}C + \overline{A}\,\overline{B}C \end{aligned}$$

上面三个逻辑函数表达式的逻辑关系相同，但表达式不同，而表达式不同，实现的逻辑电路也不同。显然表达式越简单，对应的逻辑电路也越简单，使用的电路元器件越少且经济可靠，所以在设计逻辑电路之前，必须对逻辑函数进行化简，以求得“最简”的逻辑表达式，最后得到最简的逻辑电路。所谓“最简”就是在保证逻辑关系不变的前提下乘积项的