全国高等职业技术院校电子类专业教材

QUANGUO GAODENG ZHIYE JISHU YUANXIAO DIANZILEI ZHUANYE JIAOCAI

数字集成电路应用基础

中国劳动社会保障出版社

QUANGUO GAODENG ZHIYE JISHU YUANXIAO DIANZILEI ZHUANYE JIAOCAI

全国高等职业技术院校电子类专业教材

数字集成电路应用基础

劳动和社会保障部教材办公室组织编写

中国劳动社会保障出版社

图书在版编目(CIP)数据

数字集成电路应用基础/汤湘林主编. —北京：中国劳动社会保障出版社，2005
全国高等职业技术院校电子类专业教材
ISBN 7-5045-4859-6

Ⅰ.数… Ⅱ.汤… Ⅲ.数字集成电路 Ⅳ.TN431.2

中国版本图书馆 CIP 数据核字(2005)第 014308 号

中国劳动社会保障出版社出版发行
(北京市惠新东街 1 号 邮政编码：100029)
出 版 人：张梦欣
*
北京外文印刷厂印刷装订 新华书店经销
787 毫米×1092 毫米 16 开本 12.75 印张 315 千字
2005 年 7 月第 1 版 2005 年 7 月第 1 次印刷
印数:4000 册
定价：23.00 元
读者服务部电话：010-64929211
发行部电话：010-64911190
出版社网址：http://www.class.com.cn

前言

为贯彻落实《国务院关于大力推进职业教育改革与发展的决定》，推进高等职业技术教育更好地适应经济结构调整、科技进步和劳动力市场的需要，推动高等职业技术院校实施职业资格证书制度，加快高技能人才的培养，劳动和社会保障部教材办公室在充分调研和论证的基础上，组织编写了高等职业技术院校系列教材。从2004年起，陆续推出数控类、电工类、模具设计与制造、电子商务、电子类、烹饪类等专业教材，并将根据需要不断开发新的教材，逐步建立起覆盖高等职业技术院校主要专业的教材体系。

在高等职业技术院校系列教材的编写过程中，我们始终坚持了以下几个原则：一是坚持高技能人才的培养方向，从职业（岗位）分析入手，强调教材的实用性；二是紧密结合高职技术院校、技师学院、高级技校的教学实际情况，同时，坚持以国家职业资格标准为依据，力求使教材内容覆盖职业技能鉴定的各项要求；三是突出教材的时代感，力求较多地引进新知识、新技术、新工艺、新方法等方面的内容，较全面地反映行业的技术发展趋势；四是打破传统的教材编写模式，树立以学生为主体的教学理念，力求教材编写有所创新，使教材易教易学，为师生所乐用。

电子类专业主要教材包括《模拟集成电路应用基础》《数字集成电路应用基础》《电子测量与仪器》《单片机原理与应用》《电子电路故障诊断及维修技术》《电子CAD》《电视机原理与技能训练》《常用通信终端设备原理与技能训练》《摄录像机原理与技能训练》，可供高职技术院校、技师学院、高级技校电子类专业使用。教材的编写参照了相关的国家职业标准、技术标准。

在上述教材编写过程中，我们得到有关省市劳动和社会保障部门、教育部门，以及高等职业技术院校、技师学院、高级技校的大力支持，在此表示衷心的感谢。同时，我们恳切希望广大读者对教材提出宝贵的意见和建议，以便修订时加以完善。

劳动和社会保障部教材办公室

简介

本教材为全国高等职业技术院校电子类专业教材，供各类高职院校、技师学院、高级技校相关专业使用。主要内容有：数字电路基础知识、组合逻辑电路、触发器与时序逻辑电路、脉冲波形的产生和整形电路、数/模和模/数转换电路、半导体存储器与可编程器件、实用数字电路制作等，并紧密结合教学内容配有实训练习。对于已具有数字电路基础知识的学生，可从第二章开始学习。本教材适用于一体化教学，也可用于职业培训。

本教材由汤湘林主编，梁卫文副主编，傅乐鸿、胡晓苏、赵玉林参加编写；邵展图、程慧娟审稿。

目录

第一章　数字电路基础知识

本章介绍集成数字电路这门课所需要的基础理论知识，主要内容有：以二进制为基础的常用数制及码制；逻辑代数的基本公式和常用公式；逻辑函数的表示方法及化简；逻辑门电路。

§1—1　数制与码制

一、数制

数的概念大家并不陌生，人们经常用数来表示多少或大小，而且更多的是采用十进制数。而在数字电路中，经常采用的是二进制数、八进制数或十六进制数等。

1. 十进制

十进制数有10个数码：0、1、2、3、4、5、6、7、8、9。任何一个十进制数都是由这10个数码表示的。其计数规律为：逢十进一。通常将计数数码的个数称为基数，因此，十进制数的基数为10。任意一个十进制数 N，都可以表示为：

$$(N)_{10}=a_{n-1}\times10^{n-1}+a_{n-2}\times10^{n-2}+\cdots+a_1\times10^1+a_0\times10^0+a_{-1}\times10^{-1}+\cdots+a_{-m}\times10^{-m}$$

$$=\sum_{i=-m}^{n-1}a_i\times10^i \qquad (1—1—1)$$

式中，a_i 表示第 i 位的系数，即为0、1、2、3、4、5、6、7、8、9中的任一个数码，n 为整数部分的位数，m 为小数部分的位数。

例如，一个十进制数398.86，可以展开为：

$(398.86)_{10}=3\times10^2+9\times10^1+8\times10^0+8\times10^{-1}+6\times10^{-2}$

2. 二进制

二进制数只有两个数码：0和1。任何一个二进制数都可以由这两个数码表示。其计数规律为：逢二进一，基数为2。任意一个二进制数 N，都可以表示为：

$$(N)_2=a_{n-1}\times2^{n-1}+a_{n-2}\times2^{n-2}+\cdots+a_1\times2^1+a_0\times2^0+a_{-1}\times2^{-1}+\cdots+a_{-m}\times2^{-m}$$

$$=\sum_{i=-m}^{n-1}a_i\times2^i \qquad (1—1—2)$$

式中，a_i 只可取0和1。

例如，一个二进制数11011.01，可以展开为：

$(11011.01)_2=1\times2^4+1\times2^3+0\times2^2+1\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2}$

3. 八进制

八进制数有 8 个数码：0、1、2、3、4、5、6、7。任何一个八进制数都可以由这 8 个数码表示。其计数规律为：逢八进一，基数为 8。任意一个八进制数 N，都可以表示为：

$$(N)_8 = a_{n-1} \times 8^{n-1} + a_{n-2} \times 8^{n-2} + \cdots + a_1 \times 8^1 + a_0 \times 8^0 + a_{-1} \times 8^{-1} + \cdots + a_{-m} \times 8^{-m}$$

$$= \sum_{i=-m}^{n-1} a_i \times 8^i \qquad (1—1—3)$$

式中，a_i 可以取 0、1、2、3、4、5、6 和 7 中的任一个数码。

例如，一个八进制数 246.32，可以展开为：

$(246.32)_8 = 2\times8^2 + 4\times8^1 + 6\times8^0 + 3\times8^{-1} + 2\times8^{-2}$

4. 十六进制

十六进制数有 16 个数码：0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F。任何一个十六进制数都可以由这 16 个数码表示。其计数规律为：逢十六进一，基数为 16。任意一个十六进制数 N，都可以表示为：

$$(N)_{16} = a_{n-1} \times 16^{n-1} + a_{n-2} \times 16^{n-2} + \cdots + a_1 \times 16^1 + a_0 \times 16^0 + a_{-1} \times 16^{-1} + \cdots + a_{-m} \times 16^{-m}$$

$$= \sum_{i=-m}^{n-1} a_i \times 16^i \qquad (1—1—4)$$

式中，a_i 可以取 0～9、A、B、C、D、E 和 F 中的任一个数码。

例如，一个十六进制数 206B，可以展开为：

$(206B)_{16} = 2\times16^3 + 0\times16^2 + 6\times16^1 + B\times16^0$

从表 1—1 中可以比较十进制、二进制、八进制和十六进制数码之间的对应关系。

表 1—1　　常用数制间的对应关系

十进制数	二进制数	八进制数	十六进制数
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

5. 数制的相互转换

(1) 十进制数和二进制数之间的转换。

1) 二进制数转换为十进制数。

例 1—1 将二进制数 1011.01 转换成十进制数。

解：根据式 (1—1—2) 可得：

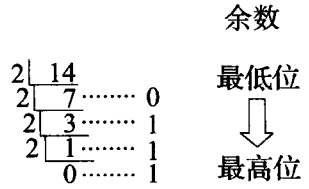
$(1011.01)_2=1\times2^3+0\times2^2+1\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2}=8+2+1+0.25=(11.25)_{10}$

2) 十进制数转换为二进制数　方法是整数部分按“除 2 取余倒记法”，即用二进制数的基数 2 去除十进制整数，第一次所得余数为二进制数的最低位，把得到的商再除以基数 2，所得余数为二进制数的次低位，依此类推，重复上面的过程，直至商为零时，所得到的余数为二进制数的最高位。

小数部分按“乘 2 取整顺记法”，即用二进制数的基数 2 去乘十进制小数，第一次所得结果的整数部分为二进制数的小数部分的最高位，把剩下的小数部分再乘以基数 2，所得结果的整数部分为二进制数的小数部分的次高位，依此类推，重复上面的过程，直至十进制数的小数部分为零或达到要求精度时为止。

例 1—2 将十进制数 14 转换成二进制数。

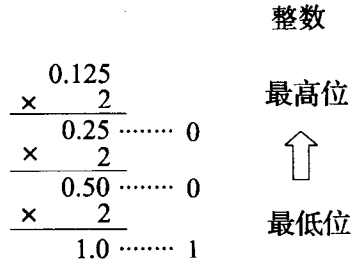
解：



所以，$(14)_{10}=(1110)_2$

例 1—3 将十进制数 0.125 转换成二进制数。

解：



所以，$(0.125)_{10}=(0.001)_2$

所以，与上例的整数部分合并，则 $(14.125)_{10}=(1110.001)_2$

(2) 二进制数、八进制数和十六进制数之间的转换　二进制数、八进制数、十六进制数的基数分别是 2、8、16，相互之间可以整除，因此它们之间的转换非常方便。

1) 二进制数与八进制数间的相互转换　由于 3 位二进制数符合“逢八进一”，正好能够表示一位八进制数，因此每 3 位二进制数分别用八进制数码表示就可以将二进制数转换成相应的八进制数。具体方法是：将二进制数整数部分自右至左每 3 位分为一组，最后不足 3 位时左面用零补足；小数部分则自左至右每 3 位分为一组，最后不足 3 位时在右面用零补足。

再将每 3 位二进制数对应的八进制数码写出即可。

例 1—4 将二进制数 $(11101011001.11001101)_2$ 转换成八进制数。

解：

011	101	011	001.	110	011	010
3	5	3	1.	6	3	2

所以，$(11101011001.11001101)_2 = (3531.632)_8$

若将八进制数转换成二进制数时，只要将每位八进制数转换成相应的 3 位二进制数码即可。

例 1—5 将八进制数 $(367.24)_8$ 转换成二进制数。

解：

3	6	7.	2	4
011	110	111.	010	100

所以，$(367.24)_8 = (11110111.0101)_8$

2）二进制数与十六进制数间的相互转换　同前述的二、八进制转换的道理一样，由于 4 位二进制数符合“逢十六进一”，正好能够表示一位十六进制数，因此每 4 位二进制数分别用十六进制数码表示就可以将二进制数转换成相应的十六进制数。具体方法是：将二进制数整数部分自右至左每 4 位分为一组，最后高位不足 4 位时左面用零补足；小数部分则自左至右每 4 位分为一组，最后低位不足 4 位时在右面用零补足。再将每 4 位二进制数对应的十六进制数码写出即可。

例 1—6 将二进制数 $(1011101011010.1011001011)_2$ 转换成十六进制数。

解：

0001	0111	0101	1010.	1011	0010	1100
1	7	5	A.	B	2	C

所以，$(1011101011010.1011001011)_2 = (175A.B2C)_{16}$

若将十六进制数转换成二进制数，只要将每位十六进制数转换成相应的 4 位二进制数码即可。

例 1—7 将十六进制数 $(5F8C.E3)_{16}$ 转换成二进制数。

解：

5	F	8	C.	E	3
0101	1111	1000	1100.	1110	0011

所以，$(5F8C.E3)_{16} = (101111110001100.11100011)_2$

在计算机中机器数往往用二进制数来表示，而为了方便书写，在日常表述中一般用十六进制来进行数的表达，望读者一定要熟练掌握二进制数与十六进制数间的相互转换。

二、码制

数字系统只能识别 0 和 1，因此，各种数据都要转换为一定位数的二进制编码才能进行处理。而编码就是用一定位数的二进制数码来表示数字、字母和符号等信息的过程。

用以表示十进制数码、字母、符号等信息的一定位数的二进制数称为代码。

1. BCD 码

BCD码是用4位二进制数来表示一位十进制数，也称二一十进制代码。常用的BCD码与十进制数码对应关系见表1—2。

表1—2　　常用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
权	8421	无权码	无权码	2421	5421

（1）8421BCD码　8421BCD码是用4位二进制编码中的前10个码0000～1001来表示十进制数码。

例如：

$(13.8)_{10}=(00010011.1000)_{8421BCD}$

$(01100101.0010)_{8421BCD}=(65.2)_{10}$

（2）余3码　余3码也是用4位二进制编码表示1位十进制数。由表1－2可见，它是由8421码加3得到的。显然，余3码所代表的十进制数可以由下式求得：

$$(N)_{10}=8\times a_3+4\times a_2+2\times a_1+1\times a_0+3$$

式中，a_3、a_2、a_1和a_0为余3码各位的值（0或1）。

BCD码除了8421码和余3码外，常用的还有2421码、5421码、格雷码。

2. 可靠性编码

代码在形成或传输过程中免不了会发生错误。为了保证代码在形成和传输过程中不易出错，或在出错时易于发现和自动校正，就需要采用可靠性编码。常见的可靠性编码有以下两种。

（1）奇偶校验码　二进制代码在传输时可能会发生有的1错为0或有的0错为1的错误。奇偶校验码是具有检验这种差错能力的一种可靠性编码。它由若干个信息位加一位奇偶校验位组成，根据电路的需要采用奇校验或是偶校验。使代码中“1”的个数为奇数的，称为奇校验；使代码中“1”的个数为偶数的，称为偶校验。表1—3列出了以8421码为信息位的奇偶校验码。

奇偶校验码的特点是只能校验单个出错的情况，对于两位或两位以上出错的情况则无法校验出来。例如，传输的偶校验码为01001，如果发生一位错误变为11001，这时接收方在做奇偶校验时检测到“1”的个数不是偶数，则说明传输过程中出错。如果发生两位

错误变为11011，这时接收方在做奇偶校验时检测到“1”的个数仍为偶数，则无法判断传输过程中是否出错。由于在数据的一次传输过程中两次出错的机率比一次出错的机率要小得多，而且奇偶校验码比较容易实现，因此奇偶校验码仍然有实用性。此外，奇偶校验码虽然能校验出单个错误，但具体是哪一位出错，却不能判定，因此它不具备自动纠错的能力。

（2）格雷（Gray）码　格雷码又称循环码，其突出特点是任意两个相邻的码字之间仅有一位二进制数码不同，其他位相同。实际应用中采用这种码会大大降低误码率，从而提高数据传输的可靠性。常见的格雷码主要有典型格雷码、余3格雷码和步进格雷码等多种编码形式，见表1—4。

表1—3　　8421奇偶校验码

十进制数	8421奇校验码	8421偶校验码
0	0000　1	0000　0
1	0001　0	0001　0
2	0010　0	0010　1
3	0011　1	0011　0
4	0100　0	0100　1
5	0101　1	0101　0
6	0110　1	0110　0
7	0111　0	0111　1
8	1000　0	1000　1
9	1001　1	1001　0

表1—4　　十进制数格雷码

十进制数	典型格雷码	余3格雷码	步进格雷码
0	0000	0010	00000
1	0001	0110	00001
2	0011	0111	00011
3	0010	0101	00111
4	0110	0100	01111
5	1110	1100	11111
6	1010	1101	11110
7	1011	1111	11100
8	1001	1110	11000
9	1000	1010	10000

3. 常用字符代码

对数字、字母和各种专用符号进行的编码统称为字符代码。常用的字符代码有ASCⅡ码（美国标准信息交换码）、ISO码（国际标准化组织码）和我国的国家标准码。

ASCⅡ码广泛应用于通信及计算机中，已被确认为国际标准代码。它是由7位二进制数组合而成的编码，它可以表示0～9十个数字、26个字母和各种常用符号及控制字符等。

ISO码主要应用于信息传送中，它是国际标准化组织编制的一组8位二进制编码。

我国的国家标准码和ASCⅡ码基本相同，只增加了第8位作为奇偶校验位，因此属于8位码。

§1—2　逻辑代数基础

一、逻辑代数的几个基本概念

1. 逻辑代数及基本运算

当自变量的取值（定义域）只有0和1（非0即1），函数的取值（值域）也只有0和1

(非 0 即 1) 两个数——这种代数就是逻辑代数。它由逻辑变量（只能取 0 或 1)、逻辑常量（“0”或“1”）以及与、或、非三种基本逻辑运算组成。

逻辑代数中的变量称为逻辑变量，用大写字母 A，B，C，…，X，Y，Z 表示。逻辑变量的取值只有两种，即逻辑 0 或逻辑 1，这里的“0”和“1”称为逻辑常量。

逻辑代数的基本运算有“与”“或”“非”三种。

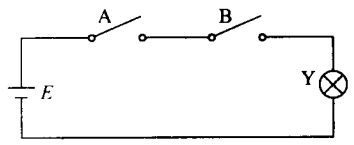
(1) 与运算（逻辑乘） 由图 1—1a 所示的开关 A、B 串联控制灯泡 Y 的电路可知，只有当开关 A、B 都闭合时，灯泡 Y 才会亮；否则灯泡 Y 就不亮。这种仅当决定事件（Y）发生的所有条件（开关 A、B 闭合）均满足时，事件（Y）才能发生（灯泡 Y 亮）的逻辑关系称为与逻辑，对应的逻辑运算称为与运算。

逻辑表达式为：

$$Y = A \cdot B$$

若用 1 表示开关闭合和灯泡亮，用 0 表示开关断开和灯泡不亮，可得逻辑真值表如图 1—1b 所示。

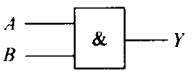
实现与逻辑运算的电路称为与门。与门的逻辑符号如图 1—1c 所示。



a)电路图

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

b)真值表



c)逻辑符号

图 1—1 与逻辑

与运算的运算法则定义为：

$$0 \cdot 0 = 0; \quad 0 \cdot 1 = 0; \quad 1 \cdot 0 = 0; \quad 1 \cdot 1 = 1$$

(2) 或运算（逻辑加） 由图 1—2a 所示的开关 A、B 并联控制灯泡 Y 的电路可知，只要开关 A 和 B 中有一个闭合或两个全闭合时，灯泡 Y 就会亮；只有当开关 A 和 B 全部断开，灯泡 Y 才不亮。这种当决定事件（Y）发生的各种条件（开关 A 或 B 闭合）中，只要有一个条件具备，事件（Y）就发生（灯泡 Y 亮）的逻辑关系称为或逻辑，对应的逻辑运算称为或运算。

逻辑表达式为：

$$Y = A + B$$

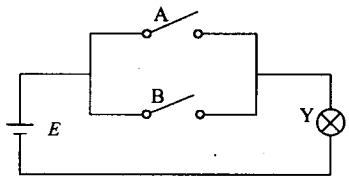
同样，若用 1 表示开关闭合和灯泡亮，用 0 表示开关断开和灯泡不亮，可得逻辑真值表如图 1—2b 所示。

实现或逻辑运算的电路称为或门。或门的逻辑符号如图 1—2c 所示。

或运算的运算法则定义为：

$$0+0=0; \quad 0+1=1; \quad 1+0=1; \quad 1+1=1$$

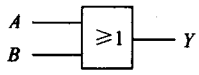
(3) 非运算（逻辑非） 由图 1—3a 所示的开关 A 控制灯泡 Y 的电路可知，当开关 A



a) 电路图

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

b) 真值表



c) 逻辑符号

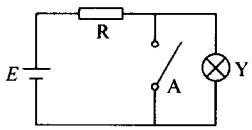
图 1—2　或逻辑

闭合时，灯泡 Y 不亮；而当开关 A 断开时，灯泡 Y 反而亮。这种当决定事件（Y）发生的条件（开关 A 闭合）满足时，事件不发生（灯泡 Y 不亮）；条件（开关 A 断开）不满足时，事件反而发生（灯泡 Y 亮）的逻辑关系称为非逻辑，对应的逻辑运算称为非运算。

逻辑表达式为：

$$Y=\overline{A}$$

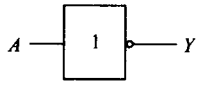
同样道理，若用 1 表示开关闭合和灯泡亮，若用 0 表示开关断开和灯泡不亮，可得逻辑真值表如图 1—3b 所示。



a) 电路图

A	Y
0	1
1	0

b) 真值表



c) 逻辑符号

图 1—3　非逻辑

实现非逻辑运算的电路称为非门。非门的逻辑符号如图 1—3c 所示。

非运算的运算法则定义为：

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

2. 逻辑代数的复合运算

实际的逻辑运算中除了上述三种基本的与、或、非运算外，还可以由这三种基本运算组成各种复合逻辑运算。常见的有与非、或非、与或非、异或等。

逻辑表达式分别为：

与非：$Y=\overline{A\cdot B}$

或非：$Y=\overline{A+B}$

与或非：$Y=\overline{AB+CD}$

异或：$Y=\overline{A}B+A\overline{B}=A\oplus B$

复合逻辑符号如图 1—4 所示。

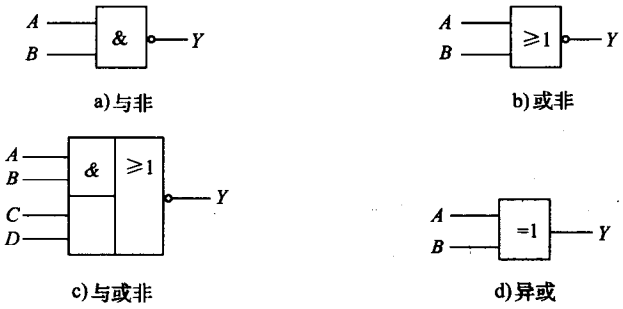


图 1—4　复合逻辑符号

二、逻辑代数的基本定律与规则

1. 基本定律与公式

(1) 0—1 律

$$\begin{cases} A+1=1 \\ A \cdot 0=0 \end{cases}$$

(2) 自等律

$$\begin{cases} A+0=A \\ A \cdot 1=A \end{cases}$$

(3) 重叠律（同一律）

$$\begin{cases} A+A=A \\ A \cdot A=A \end{cases}$$

(4) 互补律

$$\begin{cases} A+\overline{A}=1 \\ A \cdot \overline{A}=0 \end{cases}$$

(5) 还原律（非非律）

$$\overline{\overline{A}}=A$$

(6) 交换律

$$\begin{cases} A+B=B+A \\ A \cdot B=B \cdot A \end{cases}$$

(7) 结合律

$$\begin{cases} (A+B)+C=A+(B+C) \\ (A \cdot B) \cdot C=A \cdot (B \cdot C) \end{cases}$$

(8) 分配律

$$\begin{cases} A+B \cdot C=(A+B) \cdot (A+C) \\ A \cdot (B+C)=AB+AC \end{cases}$$

(9) 吸收律

$$\begin{cases} A+AB=A \\ A \cdot (A+B)=A \end{cases} \qquad \begin{cases} A+\overline{A}B=A+B \\ A \cdot (\overline{A}+B)=AB \end{cases}$$

(10) 反演律

$$\begin{cases}\overline{A+B}=\overline{A}\cdot\overline{B}\\ \overline{A\cdot B}=\overline{A}+\overline{B}\end{cases}$$

2. 基本规则

(1) 代入规则　任何一个含有变量 A 的等式，如果将所有出现 A 的位置都用同一个逻辑函数代替，则等式仍然成立。这个规则称为代入规则。

例如，已知等式 $\overline{AB}=\overline{A}+\overline{B}$，用函数 $Y=AC$ 代替等式中的 A，根据代入规则，等式仍然成立，即有：$\overline{(AC)\ B}=\overline{AC}+\overline{B}=\overline{A}+\overline{B}+\overline{C}$。

(2) 反演规则　对于任何一个逻辑表达式 Y，如果将表达式中的所有“·”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，那么所得到的表达式就是函数 Y 的反函数 $\overline{Y}$（或称补函数）。这个规则称为反演规则。例如：

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

$$Y=A+\overline{B+\overline{C}+\overline{D+\overline{E}}}\Rightarrow\overline{Y}=\overline{A}\cdot\overline{\overline{B}\cdot C\cdot\overline{\overline{D}\cdot E}}$$

(3) 对偶规则　对于任何一个逻辑表达式 Y，如果将表达式中的所有“·”换成“+”，“+”换成“·”，“0”换成“1”，“1”换成“0”，而变量保持不变，则可得到一个新的函数表达式 Y'，Y' 称为函数 Y 的对偶函数。这个规则称为对偶规则。例如：

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

$$Y=A+\overline{B+\overline{C}+\overline{D+\overline{E}}}\Rightarrow Y'=A\cdot\overline{B\cdot\overline{C}\cdot\overline{D\cdot\overline{E}}}$$

对偶规则的意义在于：如果两个函数相等，则它们的对偶函数也相等。利用对偶规则，可以使要证明及要记忆的公式数目减少一半。例如：

$$AB+A\overline{B}=A\Rightarrow(A+B)(A+\overline{B})=A$$

$$A(B+C)=AB+AC\Rightarrow A+BC=(A+B)(A+C)$$

注意：在运用反演规则和对偶规则时，必须按照逻辑运算的优先顺序进行：先算括号，接着与运算，然后或运算，最后非运算，否则容易出错。

三、逻辑函数的表示方法

1. 逻辑函数的定义

如果对应于输入逻辑变量 A、B、C、…的每一组确定值，输出逻辑变量 Y 就有唯一确定的值，则称 Y 是 A、B、C、…的逻辑函数。记为

$$Y=f(A,B,C,\cdots)$$

式中，A、B、C、…称为输入逻辑变量；Y 称为输出逻辑变量，是 A、B、C、…的函数；f 表示函数关系。

注意：与普通代数不同的是，在逻辑代数中，不管是变量还是函数，其取值都只能是 0 或 1，而且这里的 0 和 1 只表示两种不同的逻辑状态，没有具体数量的含义。

2. 逻辑函数相等的概念

设有两个逻辑函数

$$Y_1=f(A,B,C,\cdots),\ Y_2=g(A,B,C,\cdots)$$

它们的变量都是 A、B、C、…，如果对应于变量 A、B、C、…的任何一组变量取值，Y_1 和 Y_2 的值都相同，则称 Y_1 和 Y_2 是相等的，记为 $Y_1=Y_2$。