第 1 章　绪　　论

§1.1　基本要求

（1）模拟信号、模拟电路与数字信号、数字电路的区别。
（2）常用数制及其相互转换。
（3）码制与常用编码。

§1.2　重点难点

重点：（1）数字信号的概念，矩形脉冲波形的主要参数。

（2）计数制中权的概念，十进制数与二进制数、八进制数、十六进制数的相互转换。

（3）编码的概念与常用编码。

难点：数字信号的不连续概念。

§1.3　内容提要

一、模拟信号、模拟电路和数字信号、数字电路的区别

模拟信号：在时间上和数值上均连续的信号称为模拟信号。最简单的模拟信号是正弦波信号，它是时间的连续函数，同时它的瞬时值在 $-U_m \sim +U_m$ 之间是数值连续的，即可以取到 $-U_m \sim +U_m$ 之间的连续数值。

模拟电路：能够处理（包括产生、放大、传输、运算等）模拟信号的电子电路称为模拟电子电路。

数字信号：数字信号是用二进制数表示连续信号在某一时刻瞬时值大小的数字量，它是时间上和数值上均不连续的信号。时间不连续的含义，是它只对应连续时间信号在有限个取样点处的瞬时值大小，因此在时间上是不连续的；数值不连续的含义是表示某个样本值的二进制数必须是量化单位 Δ 的整数倍数值（非整数部分被舍去），因此在数值上是不连续的。

数字信号波形：由于二进制数是由**0**和**1**两个数码组成，在数字电路中，若采用正逻辑定义，低电平用**0**表示，高电平用**1**表示，则数字信号波形就是由一系列对应于时钟信号周期的高电平、低电平组成的矩形脉冲波形。如图1.3.1所示，Q是对应于CP的二进制数字信号**01011010**的波形。

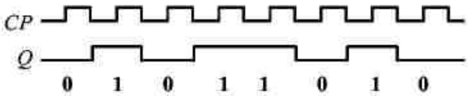


图1.3.1 数字信号波形

数字信号波形的特点：

（1）基本形状为矩形脉冲；

（2）矩形脉冲的幅度不变；

（3）低、高电平的宽度由具体的二进制数确定。

数字电路：能够处理（包括产生，传输，运算等）数字信号的电子电路称为数字电子电路。

二、矩形脉冲信号的主要参数

由于晶体管开关的非理想性，数字信号在实际传输过程中都不是理想的矩形脉冲，为了描述实际矩形脉冲信号的特性，定义了实际矩形脉冲波形的一些特性参数，如图1.3.2所示。它们主要有：

（1）上升沿时间t_r；

（2）下降沿时间t_f；

（3）有效宽度t_w；

（4）脉冲的幅度；

（5）周期T和占空比q

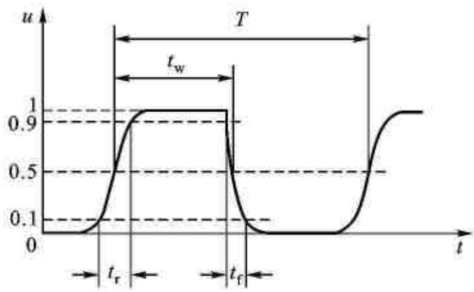


图1.3.2 矩形脉冲波形的主要参数

$$占空比\ q = t_w/T$$

影响数字电路工作速度的主要参数是t_r和t_f，t_r和t_f越小，电路工作速度就越快。

三、数制及其相互转换

1. 计数制的基数和权

基数：计数制中计数符号（又称为数码）的个数称为基数，这里用K表示。

基数 K 决定了计数制的进位规则，如十进制数有 0~9 十个数码，故 $K=10$，进位规则为逢十进一；二进制数只有 **0** 和 **1** 两个数码，故 $K=2$，进位规则为逢二进一；十六进制数有 0~9、A、B、C、D、E、F 十六个数码，故 $K=16$，进位规则为逢十六进一。

权：表征不同数位上数码的值，称为位权，简称权。权是基数的整数幂。如十进制数中，整数部分个位的权是 10^0，十位的权是 10^1，百位的权是 10^2，等等；小数部分第一位小数的权是 10^{-1}，第二位小数的权是 10^{-2}，等等。一般地说，对于任一个计数制的数 $N_K=a_{n-1}a_{n-2}\cdots a_1a_0.a_{-1}a_{-2}\cdots a_{-m}$，它们的权从高位到低位依次是 K^{n-1}，K^{n-2}，…，K^1，K^0，K^{-1}，K^{-2}，…，K^{-m}。

2. 数的按权展开式

根据权的定义，对于任意 K 进制数 $N_K=a_{n-1}a_{n-2}\cdots a_1a_0.a_{-1}a_{-2}\cdots a_{-m}$，可以写出以下展开式

$$\begin{aligned}N_K &= a_{n-1}K^{n-1}+a_{n-2}K^{n-2}+\cdots+a_1K^1+a_0K^0\\ &\quad +a_{-1}K^{-1}+a_{-2}K^{-2}+\cdots+a_{-m}K^{-m}\\ &=\sum_{i=-m}^{n-1}a_iK^i\end{aligned}$$

对于上式需要说明一点，权是基数的整数次幂，各位数码的权进行整数次幂运算后的值都是十进制数。如某一位二进制数的权是 2^3，则 $2^3=8$ 是十进制数。

3. 数制的转换

数制的转换是把数由一种计数制的等值转换成另一种计数制。

（1）任意 K 进制数 N_K 转换为十进制数

转换方法：把 N_K 展开式中的每位数码用等值的十进制数替换后，计算出的数值就是与 N_K 等值的十进制数。

［**例 1.3.1**］ 将二进制数 **10110** 转换为十进制数。

解：$(\mathbf{10110})_2=\mathbf{1}\times2^4+\mathbf{0}\times2^3+\mathbf{1}\times2^2+\mathbf{1}\times2^1+\mathbf{0}\times2^0$

$=16+0+4+2+0=(22)_{10}$

对于十进制数，其下标可以不写。

［**例 1.3.2**］ 将十六进制数 2FA 转换为十进制数。

解：$(2FA)_{16}=2\times16^2+15\times16^1+10\times16^0$

$=512+240+10=762$

（2）十进制数转换为任意 K 进制数 N_K

转换分为整数部分转换和小数部分转换。

整数部分转换方法：除基取余法。取到的最大余数是 $K-1$，最小余数是 0。具体做法是用基数 K 连除十进制数，每除一次得到一个余数，直到商为 0。然后把余数按先得到的余数为低位（LSB），后得到的余数为高位（MSB）的顺序

排列成一行，就得到 K 进制整数。

小数部分转换方法：乘基取整法。取到的最大整数是 $K-1$，最小整数是 0。具体做法是用基数 K 连乘十进制小数，每乘一次取得 1 位整数，直到小数为 0，然后按先取到的整数为高位小数（小数点后第 1 位），后取到的为低位小数的顺序排列成一行，就得到 K 进制小数。若遇到小数乘不尽的情况，则根据精度要求保留到某一位小数即可。

［**例 1.3.3**］ 将十进制数 139 转换为二进制数和十六进制数。

解：（1）转换成二进制数，用除基取余法连除基数 2。即

$2\underline{\lvert 139}$	余数	
$2\underline{\lvert 69}$ ···	**1**	← LSB
$2\underline{\lvert 34}$ ···	**1**	
$2\underline{\lvert 17}$ ···	**0**	
$2\underline{\lvert 8}$ ···	**1**	
$2\underline{\lvert 4}$ ···	**0**	
$2\underline{\lvert 2}$ ···	**0**	
$2\underline{\lvert 1}$ ···	**0**	
0 ···	**1**	← MSB

$$(139)_{10}=(\mathbf{10001011})_2$$

（2）转换为十六进制数。用除基取余法连除基数 16。即

$16\underline{\lvert 139}$	余数	
$16\underline{\lvert 8}$ ···	B（11）	LSB
0 ···	8	MSB

$$(139)_{10}=(8B)_{16}$$

［**例 1.3.4**］ 将十进制小数 0.78125 转换为二进制小数和十六进制小数。

解：（1）转换为二进制小数，用基数 2 连乘小数。即

乘 2		取整	
$0.78125\times2=1.5625$	···	**1**	← 小数点后第一位
$0.5625\times2=1.125$	···	**1**	
$0.125\times2=0.25$	···	**0**	
$0.25\times2=0.5$	···	**0**	
$0.5\times2=1.0$	···	**1**	← 小数点后最后一位

$$(0.78125)_{10}=(\mathbf{0.11001})_2$$

（2）转换为十六进制小数，用基数 16 连乘小数。即

乘 16		取整
$0.78125\times16=12.5$	···	C（12）

$$0.5\times16=8.0 \qquad \cdots \qquad 8$$

$$(0.78125)_{10}=(0.C8)_{16}$$

（3）二进制数和十六进制数、八进制数的相互转换

转换方法：数码等值代换法。具体做法是用4位二进制数与一位等值十六进制数直接进行替换，即实现二进制数与十六进制数之间的相互转换。如果用3位二进制数与1位等值的八进制数直接进行替换，即实现二进制数与八进制数之间的相互转换。

［例 1.3.5］ 将数$(\mathbf{1011010.11})_2$转换为十六进制数和八进制数。

解：（1）转换为十六进制数。首先把二进制数以小数点为界，分别向左、向右每4位分为一节，整数高位的缺位和小数低位的缺位补**0**。然后用等值的十六进制数码替换每一节二进制数即可

$$(\mathbf{01011010.1100})_2=\underbrace{\mathbf{0101}}_{5}\ \underbrace{\mathbf{1010}}_{10}.\underbrace{\mathbf{1100}}_{12}=(5A.C)_{16}$$

（2）转换为八进制数，与转换为十六进制数类似，转换如下

$$(\mathbf{1011010.11})_2=\underbrace{\mathbf{001}}_{1}\ \underbrace{\mathbf{011}}_{3}\ \underbrace{\mathbf{010}}_{2}.\underbrace{\mathbf{110}}_{6}=(132.6)_8$$

本例的逆向转换就是十六进制数、八进制数转换为二进制数。

4. 二进制数的加、减运算

二进制加运算按逢二进一规则运算，减法运算按借一当二规则运算。但在数字电路中要实现二进制数的加、减运算，为避免将电路设计得太复杂，往往采用补码的形式进行二进制数的加、减运算，这样可以利用加法运算电路实现减法运算，从而节省电路的硬件成本。

用加法实现减法运算，就是把减数看作有符号数，然后与被减数相加（类似于代数中的求代数和运算），但是不能采用原码求和，而必须采用补码求和。若求得的和为正数，其和就是两数之差的原码；若求得的和为负数，其和为两数之差的补码，将结果再求补一次，就得到两数之差的原码。

（1）有符号二进制数的表示法

有符号二进制数是用符号位表示数的正负的。在二进制数的最高位前面加一位符号位，若符号位为**0**，则表示二进制数为正数；若符号位为**1**，则表示二进制数为负数。因此有符号数在书写时的最高位是符号位，其后的各位是数值位。

（2）二进制数的原码、反码、补码

对于数 $X>0$，如 $X=+\mathbf{1001}$，则有符号数表示为

$$[X]_{原}=[X]_{反}=[X]_{补}=\mathbf{01001}$$

最高位的 **0** 表示数值 **1001** 是正数。正数的原码、反码和补码是一样的。

对于数 $X<0$，如 $X=-\mathbf{1001}$，则有符号数表示为

$[X]_{原}=\mathbf{11001}$　　最高位的**1**表示数值**1001**是负数

$[X]_{反}=\mathbf{10110}$　　负数求反码是将数值位各位求反

$[X]_{补}=\mathbf{10111}$　　负数求补码是反码加**1**

$[[X]_{补}]_{补}=\mathbf{11001}=[X]_{原}$　　负数补码的补码是原码

采用补码运算时，若运算结果为正数，该结果就是原码；若运算结果为负数，该结果是补码，必须再求补一次，才能得到运算结果的原码。另外必须注意，2个 n 位二进制数相加的和不能大于 2^n-1，否则将发生溢出错误。要想避免溢出错误，只有加长二进制数的字长。字长 n 决定了电路允许的最大数值为 $|2^n-1|$。

（3）二进制数加、减运算举例

[例 1.3.6] 设 $X_1=+\mathbf{1001}$，$X_2=+\mathbf{0101}$，试用无符号数和有符号数分别求 X_1+X_2、X_1-X_2 和 X_2-X_1。

解：（1）无符号数运算

$$X_1+X_2=\mathbf{1001}+\mathbf{0101}=\mathbf{1110}$$

$$X_1-X_2=\mathbf{1001}-0101=\mathbf{0100}$$

$$X_2-X_1=-(\mathbf{1001}-\mathbf{0101})=-\mathbf{0100}$$

（2）有符号数运算

① 求 X_1+X_2

$$[X_1]_{补}=\mathbf{01001},\ [X_2]_{补}=\mathbf{00101}$$

$$[X_1]_{补}+[X_2]_{补}=\mathbf{01001}+\mathbf{00101}=\mathbf{01110}$$

结果大于0，说明该结果就是两数之和的原码。

② 求 $X_1-X_2=X_1+(-X_2)$

$$[X_1]_{补}=\mathbf{01001},\ [-X_2]_{补}=[-X_2]_{反}+\mathbf{1}=\mathbf{11011}$$

$$[X_1]_{补}+[-X_2]_{补}=\mathbf{01001}+\mathbf{11011}=\mathbf{00100}$$

$$\begin{array}{r} \mathbf{01001} \\ +\ \ \mathbf{11011} \\ \hline \boxed{\mathbf{1}}\ \ \mathbf{00100} \end{array}$$

$\uparrow$

自动舍去

运算结果为正数，是因为 $X_1>X_2$，所以 $X_1-X_2>0$，运算结果为两数之差的原码。

③ 求 $X_2-X_1=X_2+(-X_1)$

$$[X_2]_{补}=\mathbf{00101},\ [-X_1]_{补}=[-X_1]_{反}+1=\mathbf{10111}$$

$$[X_2]_{补}+[-X_1]_{补}=\mathbf{00101}+\mathbf{10111}=\mathbf{11100}$$

运算结果为负数，说明这一结果是 X_2-X_1 的补码，即

$$[X_2-X_1]_{补}=\mathbf{11100}$$

而 X_2-X_1 的原码是

$$[[X_2-X_1]_{补}]_{补}=\mathbf{10100}$$

因为 $X_2<X_1$，所以

$$X_2-X_1=-\mathbf{0100}$$

用有符号数表示就是 **10100**。

四、码制

码制：码制就是代码，是用 n 位数字符号表示 N 个客观信息量，这种表示又称为编码。如果用二进制数作为所要表示的信息编码，就得到二进制代码，简称二进制码。二进制码的位数 n 和要编码的信息量 N 之间必须满足关系式

$$2^n \geqslant N$$

1. 十进制数的二进制编码

十进制数的二进制编码就是用 n 位二进制数为 $0\sim(2^n-1)$ 个十进制数编码。常用的编码有两种。

(1) 自然二进制码：用 n 位二进制数为与之对应等值的 2^n 个 $(0\sim2^n-1)$ 十进制数进行编码，这样的二进制码称为自然二进制码。它是一种有权码，自然二进制码的加权系数和就是它所对应的十进制数。

(2) 格雷码：格雷码也是用 n 位二进制数为 2^n 个十进制数进行编码，但它是一种无权码。它的编码规则是：相邻两个十进制数的两个代码只有一位互反，其余各位相同，并且首数 0 和尾数 2^n-1 的两个代码也满足这一规律。表 1.3.1 列出了十进制数 0 ~ 15 的自然二进制码和格雷码。

表 1.3.1　十进制数的二进制编码

十进制数	自然二进制码	格　雷　码	十进制数	自然二进制码	格　雷　码
0	**0 0 0 0**	**0 0 0 0**	8	**1 0 0 0**	**1 1 0 0**
1	**0 0 0 1**	**0 0 0 1**	9	**1 0 0 1**	**1 1 0 1**
2	**0 0 1 0**	**0 0 1 1**	10	**1 0 1 0**	**1 1 1 1**
3	**0 0 1 1**	**0 0 1 0**	11	**1 0 1 1**	**1 1 1 0**
4	**0 1 0 0**	**0 1 1 0**	12	**1 1 0 0**	**1 0 1 0**
5	**0 1 0 1**	**0 1 1 1**	13	**1 1 0 1**	**1 0 1 1**
6	**0 1 1 0**	**0 1 0 1**	14	**1 1 1 0**	**1 0 0 1**
7	**0 1 1 1**	**0 1 0 0**	15	**1 1 1 1**	**1 0 0 0**

2. 十进制数码(0,1,2,…,9)的二进制编码——BCD码

BCD码就是用4位二进制数为十进制数的十个数码0~9进行编码。由于4位二进制数共有16种取值组合，因此有多种编码方式。常用的有8421BCD码，5421BCD码，2421BCD码，这些是有权码，无权码有余3码，格雷码(又称循环码)。表1.3.2列出了部分BCD码。

表1.3.2 常用BCD码编码表

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

用BCD码表示十进制数时，就是直接用BCD码代换十进制数中的每一位数码。如十进制数947，用8421BCD码表示为

$$947 = (\mathbf{100101000111})_{8421BCD}$$

§1.4 例题分析

[例1.4.1] 把二进制数**11010.101**转换为十进制数、八进制数和十六进制数。

解：(1) $(\mathbf{11010.101})_2 = 1\times2^4 + 1\times2^3 + 1\times2^1 + 1\times2^{-1} + 1\times2^{-3}$

$= 16 + 8 + 2 + 0.5 + 0.125 = (26.625)_{10}$

(2) $(\mathbf{11010.101})_2 = \underbrace{\mathbf{011}}_{3}\ \underbrace{\mathbf{010}}_{2}.\ \underbrace{\mathbf{101}}_{5} = (32.5)_8$

(3) $(\mathbf{11010.101})_2 = \underbrace{\mathbf{0001}}_{1}\ \underbrace{\mathbf{1010}}_{10}.\underbrace{\mathbf{1010}}_{10} = (1A.A)_{16}$

如果利用十六进制数求加权系数和获得十进制数就会简练些，如

$(\mathbf{11010.101})_2 = (32.5)_8$

$$(\mathbf{11010.101})_2=(1A.A)_{16}$$

$$\begin{aligned}(1A.A)_{16}&=1\times16^1+10\times16^0+10\times16^{-1}\\&=16+10+0.625=(26.625)_{10}\end{aligned}$$

［例 **1.4.2**］ 把十进制数 259.875 转换为二进制数。

解：用除 2 取余法和乘 2 取整法分别对整数和小数部分进行转换，可得

$$(259.875)_{10}=(\mathbf{100000011.111})_2$$

对整数部分进行转换时，要用除 2 取余法连除 9 次 2，才能得到结果，其演算很繁琐。可是把 259 写成 256 + 3，就简单了，因为 $(256)_{10}=2^8=(\mathbf{100000000})_2$，$(3)_{10}=(\mathbf{11})_2$，所以有

$$(259)_{10}=2^8+3=(\mathbf{100000000})_2+(\mathbf{11})_2=(\mathbf{100000011})_2$$

对于小数部分，也可类似写成

$$0.875=0.5+0.25+0.125=2^{-1}+2^{-2}+2^{-3}=(\mathbf{0.111})_2$$

因此有

$$(259.875)_{10}=(\mathbf{100000011.111})_2$$

这一结果告诉我们，记熟 $2^{10}=1024$，$2^9=512$，$2^8=256$，…以及 $2^{-1}=0.5$，$2^{-2}=0.25$，$2^{-3}=0.125$，…这些数，对于数制转换是十分有用的。

［例 **1.4.3**］ 把十进制小数 0.69 转换为二进制小数，二进制小数的精度为 2^{-4}。

解：用乘 2 取整法得

乘 2		取整
$0.69\times2=1.38$	…	**1**
$0.38\times2=0.76$	…	**0**
$0.76\times2=1.52$	…	**1**
$0.52\times2=1.04$	…	**1**

因此有

$$(0.69)_{10}\approx(\mathbf{0.1011})_2$$

［例 **1.4.4**］ 把数$(5FB)_{16}$转换为二进制数和十进制数。

解：(1) $(5FB)_{16}=\begin{array}{ccc}5 & F & B\\ \downarrow & \downarrow & \downarrow\\ \mathbf{0101} & \mathbf{1111} & \mathbf{1100}\end{array}=(\mathbf{010111111100})_2$

(2) $$\begin{aligned}(5FB)_{16}&=5\times16^2+15\times16^1+12\times16^0\\&=1280+240+12=(1532)_{10}\end{aligned}$$

［例 **1.4.5**］ 把数$(267.25)_8$ 转换为十六进制数。

解：要先把八进制数转换成二进制数，再转换成十六进制数。

$$
\begin{array}{l}
(267.25)_8 = \begin{array}{ccccccc} 2 & 6 & 7 & . & 2 & 5 \\ \downarrow & \downarrow & \downarrow & & \downarrow & \downarrow \\ \mathbf{010} & \mathbf{110} & \mathbf{111} & & \mathbf{010} & \mathbf{101} \end{array} = (\mathbf{010110111.010101})_2
\end{array}
$$

$$(\mathbf{10110111.010101})_2 = \underbrace{\mathbf{1011}}_{11}\ \underbrace{\mathbf{0111}}_{7}.\ \underbrace{\mathbf{0101}}_{5}\ \underbrace{\mathbf{0100}}_{4} = (B7.54)_{16}$$

[例 1.4.6] 用 8421BCD 码表示数$(\mathbf{1011010})_2$。

解：BCD 码是十进制数码的二进制编码，因此必须把二进制数转换成十进制数，才能用 BCD 码表示。

$$(\mathbf{1011010})_2 = 2^6 + 2^4 + 2^3 + 2^1 = (90)_{10}$$

$$(90)_{10} = (\mathbf{10010000})_{8421BCD}$$

[例 1.4.7] 设 $X_1 = (7.25)_{10}$，$X_2 = (13.5)_{10}$，试用有符号数求 $X_1 + X_2$ 和 $X_1 - X_2$。

解：(1) 先转换 X_1、X_2 为有 5 位整数的二进制数(因 $X_1 + X_2 > 16$，故需用 5 位二进制整数记录两数之和)。

$$X_1 = (7.25)_{10} = (\mathbf{00111.01})_2$$

$$X_2 = (13.5)_{10} = (\mathbf{01101.10})_2$$

(2) 求 $X_1 + X_2$。

$$[X_1]_{补} = \mathbf{000111.01},\ [X_2]_{补} = \mathbf{001101.10}$$

$$[X_1]_{补} + [X_2]_{补} = \mathbf{000111.01} + \mathbf{001101.10} = \mathbf{010100.11}$$

$$(\mathbf{010100.11})_2 = 1 \times 2^4 + 1 \times 2^2 + 1 \times 2^{-1} + 1 \times 2^{-2} = (20.75)_{10}$$

(3) 求 $X_1 - X_2 = X_1 + (-X_2)$。

$$[X_1]_{补} = \mathbf{000111.01},\ [-X_2]_{补} = \mathbf{110010.10}$$

$$[X_1]_{补} + [-X_2]_{补} = \mathbf{000111.01} + \mathbf{110010.10} = \mathbf{111001.11}$$

$$[X_1 - X_2]_{原} = [\mathbf{111001.11}]_{补} = \mathbf{100110.01}$$

写成无符号数为

$$-(\mathbf{00110.01})_2 = (-6.25)_{10}$$

第 2 章　逻辑代数基础

§ 2.1　基本要求

（1）掌握基本逻辑运算和导出逻辑运算及其真值表，熟记它们的逻辑符号。

（2）掌握逻辑代数基本定律、变换规则及其应用。

（3）掌握逻辑函数的 5 种表示方法及其相互对应关系。

（4）熟悉逻辑函数表达式的 5 种基本形式及其相互变换。

（5）掌握逻辑函数的卡诺图化简法，熟悉逻辑函数的代数化简法。

§ 2.2　重点难点

重点：（1）常用逻辑运算，逻辑代数基本定律、规则及应用。

（2）逻辑函数的表示方法及相互对应关系。

（3）逻辑函数式的变换与化简。

难点：逻辑函数式的变换与代数化简法。

§ 2.3　内容提要

一、逻辑变量的二值性及其意义、逻辑函数及其表示法

1. 逻辑变量

具有两个对立状态的变量称为逻辑变量。如电路中的高电平和低电平；开关的闭合与断开；晶体管的饱和与截止等，都可视为逻辑变量。数字电路中的正逻辑系统，规定用 **1** 表示高电平，用 **0** 表示低电平，因此逻辑变量只有 **0** 或 **1** 两种取值，故逻辑变量又称为二值变量。逻辑变量取值 **0** 或 **1** 仅表示变量的两种对立状态，不具有数值大小的意义。

2. 逻辑函数

逻辑变量之间依一定的对应法则存在着确定的因果关系，这种因果关系称

为逻辑函数。描述逻辑函数的方法有 5 种，它们是真值表、逻辑函数表达式(简称逻辑式)、卡诺图、逻辑图和波形图。这 5 种描述方法之间是对应的，可以相互转换，转换关系如图 2.3.1 所示。

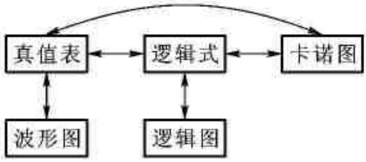


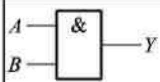
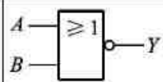
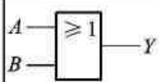
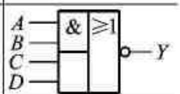
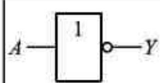
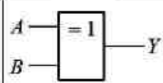
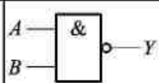
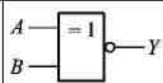
图 2.3.1 逻辑函数的表示及相互转换

二、逻辑运算、逻辑代数基本定律和重要规则

1. 基本逻辑运算和导出逻辑运算

基本逻辑运算有**与**、**或**、**非**运算，导出逻辑运算有**与非**、**或非**、**与或非**、**异或**和**同或**运算。每一种运算都有对应的逻辑符号，它们是构成数字逻辑系统的基本单元。表 2.3.1 列出了这些逻辑运算和相应的逻辑符号。

表 2.3.1 基本逻辑运算及对应的逻辑符号

运算关系	逻　辑　式	逻 辑 符 号	运算关系	逻　辑　式	逻 辑 符 号
与运算	$Y=AB$		或非运算	$Y=\overline{A+B}$	
或运算	$Y=A+B$		与或非运算	$Y=\overline{AB+CD}$	
非运算	$Y=\overline{A}$		异或运算	$Y=A\overline{B}+\overline{A}B$ $=A\oplus B$	
与非运算	$Y=\overline{AB}$		同或运算	$Y=\overline{A}\,\overline{B}+AB$ $=A\odot B$	

实现基本运算的电路称为逻辑门，如**与**门、**或**门、**非**门、**与非**门、**或非**门、**与或非**门、**异或**门和**同或**门。关于**异或**运算和**同或**运算需特别指出以下几点：

（1）**异或**运算和**同或**运算互为反函数，即

$$\overline{A\oplus B}=\overline{A}\oplus B=A\oplus\overline{B}=A\odot B$$

$$\overline{A\odot B}=\overline{A}\odot B=A\odot\overline{B}=A\oplus B$$

（2）奇数个 **1** 连续**异或**运算的结果为 **1**；偶数个 **1** 连续**异或**运算的结果为 **0**。

（3）偶数个 **0** 连续**同或**运算的结果为 **1**；奇数个 **0** 连续**同或**运算的结果为 **0**。

（4）**0** 连续**异或**运算结果为 **0**，**1** 连续**同或**运算结果为 **1**。

逻辑运算的优先顺序是**非**—**与**—**或**，括号可改变运算顺序。

2. 逻辑代数的基本定律和公式

逻辑代数的基本定律表明了逻辑代数的运算法则。表 2.3.2 列出了逻辑代数的基本定律和对应的公式。

表 2.3.2　逻辑代数基本定律和公式

基本定律	基 本 公 式	基本定律	基 本 公 式
0 - 1 律	$\mathbf{0}\ A=\mathbf{0}$，$\mathbf{0}+A=A$ $\mathbf{1}\ A=A$，$\mathbf{1}+A=\mathbf{1}$ $\mathbf{0}\oplus A=A$，$\mathbf{0}\odot A=\overline{A}$ $\mathbf{1}\oplus A=\overline{A}$，$\mathbf{1}\odot A=A$	结合律	$A\ B\ C=A\ (B\ C)$， $A+B+C=A+(B+C)$， $A\odot B\odot C=A\odot(B\odot C)$， $A\oplus B\oplus C=A\oplus(B\oplus C)$
互补律	$A\ \overline{A}=\mathbf{0}$，$A+\overline{A}=\mathbf{1}$ $A\odot\overline{A}=\mathbf{0}$，$A\oplus\overline{A}=\mathbf{1}$	分配律	$A\ (B+C)=AB+AC$， $A+BC=(A+B)(A+C)$ $A\ (B\oplus C)=AB\oplus AC$ $A+(B\odot C)=(A+B)\odot(A+C)$
重叠律	$A\ A=A$，$A+A=A$		
还原律	$\overline{\overline{A}}=A$ $\overline{A}\oplus\overline{B}=A\oplus B$，$\overline{A}\odot\overline{B}=A\odot B$	吸收律	$A+AB=A$，$A+\overline{A}B=A+B$ $AB+\overline{A}C+BC=AB+\overline{A}C$
交换律	$A\ B=B\ A$，$A+B=B+A$ $A\odot B=B\odot A$，$A\oplus B=B\oplus A$	摩根定律	$\overline{A\ B}=\overline{A}+\overline{B}$，$\overline{A+B}=\overline{A}\ \overline{B}$

3. 三个重要规则

（1）代入规则：把一个逻辑等式两边的同一个变量用一个逻辑函数代换后，逻辑式仍然相等。

代入规则的应用：

① 拓展基本公式，如

$$A+\overline{A}B=A+B\xrightarrow{\overline{A}\to A}\overline{A}+AB=\overline{A}+B$$

② 证明逻辑等式，如

$$A\oplus B=A\overline{B}+\overline{A}B\xrightarrow{\overline{B}\to B}A\oplus\overline{B}=AB+\overline{A}\ \overline{B}=A\odot B$$

③ 逻辑定理的推广，如

$$\overline{AB}=\overline{A}+\overline{B}\xrightarrow{BC\to B}\overline{ABC}=\overline{A}+\overline{BC}=\overline{A}+\overline{B}+\overline{C}$$

（2）反演规则：把逻辑函数式中的**与**运算“ ”变为**或**运算“+”，**或**运算“+”变为**与**运算“ ”，原变量变为反变量，反变量变为原变量，**0** 变为 **1**，**1** 变为 **0**，并保持原式中的运算优先顺序不变，就得到一个新的逻辑函数式，它是原函数的反函数。反演规则主要用于求反函数。

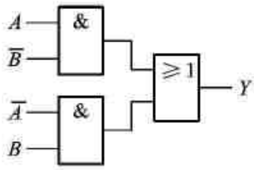
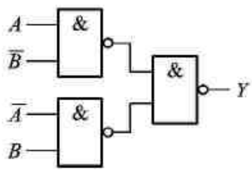
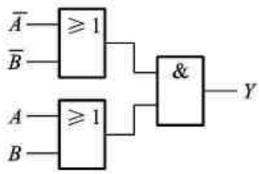
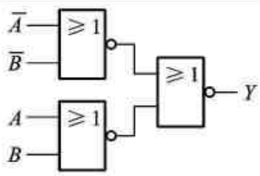
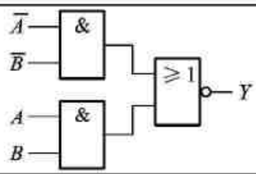
（3）对偶规则：把逻辑函数式中的**与**运算“ ”变为**或**运算“+”，**或**运算“+”变为**与**运算“ ”，**0** 变为 **1**，**1** 变为 **0**，变量不变，且保持原式中的运算

优先顺序不变，就得到一个新的逻辑函数式，它是原函数的对偶函数。对偶规则的应用，提供了一种证明逻辑等式的方法，如能证明对偶式成立，则原式也一定成立。此外还可利用对偶变换扩充基本逻辑公式。

三、逻辑函数式的变换

逻辑函数式可以有多种形式，常用的基本逻辑式有**与或**式、**与非**式、**或与**式、**或非**式和**与或非**式。不同的逻辑式，需要不同的门电路来实现，因此，变换逻辑式的意义就在于使逻辑设计适用于不同的逻辑门电路。下面以**异或**逻辑 $Y=A\oplus B$ 为例，给出它的 5 种基本逻辑式和对应的逻辑图，如表 2.3.3 所示。

表 2.3.3 异或逻辑 $Y=A\oplus B$ 的 5 种基本逻辑式

名　称	逻辑式变换	逻 辑 图
与或式	$Y=A\bar{B}+\bar{A}B$	
与非式	$Y=A\bar{B}+\bar{A}B$ $=\overline{\overline{A\bar{B}+\bar{A}B}}$ $=\overline{\overline{A\bar{B}}\ \overline{\bar{A}B}}$	
或与式	$Y=A\bar{B}+\bar{A}B$ $=(A\bar{B}+\bar{A})(A\bar{B}+B)$ $=(\bar{A}+\bar{B})(A+B)$	
或非式	$Y=(\bar{A}+\bar{B})(A+B)$ $=\overline{\overline{(\bar{A}+\bar{B})(A+B)}}$ $=\overline{\overline{\bar{A}+\bar{B}}+\overline{A+B}}$	
与或非式	$Y=\overline{\overline{\bar{A}+\bar{B}}+\overline{A+B}}$ $=\overline{AB+\bar{A}\ \bar{B}}$	

四、逻辑函数的代数化简法

代数化简法是运用逻辑代数的基本定律、基本公式和变换规则，对逻辑函数式进行变换，以达到减少乘积项或消去变量，获得最简逻辑式的目的。

1. 最简**与或**逻辑式的标准

不同形式的逻辑式对应有不同的最简形式，但一般来说，都可以从最简**与或**式经等效变换来获得。最简**与或**式的标准是：

（1）逻辑式中的乘积项最少；

（2）每个乘积项中的变量数最少。

2. 逻辑式的代数化简法

（1）并项法：利用 $AB+A\overline{B}=A$，把两项合并为一项，并消去一个变量；

（2）吸收法：利用 $A+AB=A$ 和 $AB+\overline{A}C+BC=AB+\overline{A}C$，消去多余项；

（3）消去法：利用 $A+\overline{A}B=A+B$ 和 $\overline{A}+AB=\overline{A}+B$，消去多余变量；

（4）配项法：利用 $A=A(B+\overline{B})$ 和 $A=A+B\overline{B}$，把一项变成二项，或利用 $AB+\overline{A}C=AB+\overline{A}C+BC$，添加冗余项，再和其他项合并，达到化简的目的。

代数法化简逻辑函数式时，首先应用逻辑代数基本定律对逻辑式进行适当变换，把各项进行适当组合，使之成为方法（1）~（3）的形式，从而实现化简。代数化简法具有适用范围广，不受任何条件限制的特点，但没有一定的步骤可循，要求有熟练的运算技巧和经验，并且化简结果是否达到最简，难以判断。

五、逻辑函数卡诺图化简法

1. 逻辑函数式的两种标准式

（1）最小项和标准**与或**式

① 最小项：对于 n 个变量的逻辑函数**与或**表达式，如果乘积项中包含全部变量，且每个变量在该乘积项中以原变量或反变量的形式只出现一次，则该乘积项称为最小项，记为 m_i。n 个变量的全部最小项共有 2^n 个。把最小项中的原变量取值为 **1**，反变量取值为 **0** 得到的一组二进制数所对应的十进制数，就是该最小项下标 i 的值。如最小项 $A\overline{B}C$，当 $A\overline{B}C$ 取值为 **101** 时，对应十进制数 5，因此记为 m_5。

② 最小项的性质：

对应任意一组变量取值，只有一个最小项的值为 **1**，其余各最小项的值均为 **0**；

同一组变量取值，任意两个最小项的乘积恒为 **0**；

不同的最小项，使其值为 **1** 的变量取值不同；

对于任一组变量取值，全体最小项的和为 **1**。

③ 相邻最小项：只有一个互为反变量的两个最小项，称为相邻最小项，简称相邻项。如 ABC 和 $AB\overline{C}$ 是相邻项。显然，两个相邻项可以相加合并为一项，同时消去一个变量（消去互反变量）。

④ 标准**与或**式：每个**与**项都是最小项的**与或**式称为标准**与或**式。又称为最小项表达式。标准**与或**式的简记形式为

$$\begin{aligned} Y &= \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC \\ &= m_3 + m_5 + m_6 + m_7 \\ &= \sum m(3,5,6,7) \end{aligned}$$

逻辑函数的标准**与或**式和逻辑函数真值表是一一对应的，根据真值表写标准**与或**式的方法是：把函数值为 **1** 的各变量取值对应的最小项相加，即为逻辑函数的标准**与或**式。一般的**与或**式可以通过乘$(X+\overline{X})$，再应用分配律 $A(B+C)=AB+AC$ 转换成标准**与或**式。这里的 X 代表**与**项中缺少的变量。

（2）最大项和标准**或与**式

① 最大项：对于 n 个变量的逻辑函数**或与**式，如果**或**项中包含全部变量，且每个变量以原变量或反变量的形式只出现一次，则该**或**项称为最大项，记为 M_i。n 个变量的全部最大项共有 2^n 个。把最大项中的原变量取值为 **0**，反变量取值为 **1** 得到的一组二进制数所对应的十进制数，就是该最大项下标 i 的值。如最大项$(\overline{A}+B+\overline{C})$记为 M_5。

② 最大项的性质。表 2.3.4 所示为三变量 A、B、C 的最大项真值表，由此表可归纳出最大项的性质：

表 2.3.4　三变量最大项真值表

$A\ B\ C$	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M_7
0 0 0	0	1	1	1	1	1	1	1
0 0 1	1	0	1	1	1	1	1	1
0 1 0	1	1	0	1	1	1	1	1
0 1 1	1	1	1	0	1	1	1	1
1 0 0	1	1	1	1	0	1	1	1
1 0 1	1	1	1	1	1	0	1	1
1 1 0	1	1	1	1	1	1	0	1
1 1 1	1	1	1	1	1	1	1	0

对于任意一组变量取值，只有一个最大项的值为 **0**，其余各最大项的值均为 **1**；

对于任意一组变量取值，任意两个最大项之和（或运算）恒为 **1**；

对于变量的任一组取值，全部最大项的积为 **0**。

③ 相邻最大项：只有一个互反变量的两个最大项称为相邻最大项，也称为相邻项。两个相邻最大项可以相乘合并为一项，同时消去一个变量（消去了互反变量），如相邻最大项($\overline{A}+B+C$)和($A+B+C$)相乘有

$$(\overline{A}+B+C)(A+B+C)=\overline{A}(B+C)+A(B+C)+(B+C)$$
$$=(B+C)\quad \text{消去了互反变量 } A$$

④ 标准**或与**式：每个**或**项都是最大项的**或与**式，称为标准**或与**式，又称最大项表达式。标准**或与**式也有简记形式，如

$$Y(A,B,C)=(A+B+C)(A+\overline{B}+C)(\overline{A}+\overline{B}+\overline{C})$$
$$=M_0\ M_2\ M_7=\prod M(0,2,7)$$

逻辑函数的标准**或与**式和逻辑函数真值表是一一对应的。根据真值表写标准**或与**式的方法是：把函数值为 **0** 的各变量取值对应的最大项相乘，即为逻辑函数的标准**或与**式。一般的**或与**式，可以通过加零($X\ \overline{X}$)，再应用分配律 $A+BC=(A+B)(A+C)$转换为标准**或与**式。这里的 X 代表**或**项中缺少的变量。

［**例 2.3.1**］ 试将逻辑函数 $Y=(A+\overline{B})(\overline{B}+C)$转换为标准**或与**式。

解：
$$Y=(A+\overline{B})(\overline{B}+C)$$
$$=(A+\overline{B}+C\ \overline{C})(\overline{B}+C+A\ \overline{A})$$
$$=(A+\overline{B}+C)(A+\overline{B}+\overline{C})(A+\overline{B}+C)(\overline{A}+\overline{B}+C)$$
$$=M_2\ M_3\ M_2\ M_6$$
$$=\prod M(2,3,6)$$

(3) 两种标准表达式的转换

① 同一逻辑函数，可以有两种不同的标准逻辑表达式。

② 变量相同，变量排列顺序也相同时，下标相同的最小项和最大项是互为反函数，即

$$\overline{m_i}=M_i;\ \overline{M_i}=m_i$$

③ 同一逻辑函数的标准**与或**式和标准**或与**式的转换。

设
$$Y(A,B,C)=\sum m(1,2,4,7)$$
则
$$\overline{Y}(A,B,C)=\sum m(0,3,5,6)$$
$$=m_0+m_3+m_5+m_6$$

对$\overline{Y}(A,B,C)$求反有

$$Y(A,B,C)=\overline{m_0+m_3+m_5+m_6}=\overline{m_0}\ \overline{m_3}\ \overline{m_5}\ \overline{m_6}$$
$$=M_0\ M_3\ M_5\ M_6=\prod M(0,\ 3,\ 5,\ 6)$$

所以
$$Y(A,B,C)=\sum m(1,2,4,7)=\prod M(0,3,5,6)$$

由上述分析可归纳出如下关系：如果把 n 个变量组成的 2^n 个最小(大)项的下标看作一个集合，则标准**与或**式中的最小项下标的集合与标准**或与**式中最