模块一　数字逻辑基础

随着数字电子技术的快速发展，数字通信系统、高清晰数字电视、数字视听设备、数控机床等越来越多的数字化产品进入我们工作和生活的各个领域，让我们的生产、生活以及思维方式悄悄地发生着变革。那么，数字电子技术究竟是一门怎样的技术呢？为什么它能在近几十年取得如此瞩目的变化？现在就让我们一起探索其中的奥秘吧！

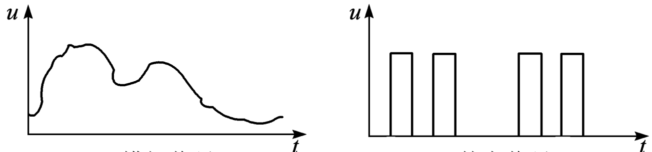
学习目标

了解数字集成电路的命名方法；
了解逻辑函数的意义，能进行逻辑函数的化简；
了解组合逻辑电路的设计流程，能进行简单应用电路的设计；
掌握数字集成电路的识别方法；
掌握仿真测试集成电路逻辑功能的方法，进而掌握实际数字集成电路的测试方法；
掌握如何选用数字集成电路芯片，能按照逻辑电路图搭建实际电路；
能使用仿真软件进行应用电路的设计。

学习单元一　数字逻辑基础概述

一、模拟信号和数字信号

自然界中存在各种各样的物理量，从变化规律来看，大致可以分为模拟量和数字量两大类。模拟量具有时间上连续变化、值域内任意取值的特点，如温度、速度、压力、交流电压等就是典型的模拟量；数字量具有时间上离散变化、数值也离散取值的特点，如机床上记录零件个数的计数信号就是典型的数字量。在电子设备中，无论是数字量还是模拟量都是以电信号的形式出现的。用于表示模拟量的电信号称为模拟信号，如图 1-1(a) 所示，模拟信号不易于进行存储、处理和传输。用于表示数字量的电信号称为数字信号，如图 1-1(b) 所示。



(a) 模拟信号　　(b) 数字信号

图 1-1　模拟信号和数字信号

数字信号在时间上和数值上均是离散的，常用数字 0 和 1 表示，这里的 0 和 1 是一种符号，称为逻辑 0 和逻辑 1，用来表示客观世界中相互关联又相互对立的两种状态，如高低、真假、开关等，因而称之为二值数字逻辑，简称数字逻辑。数字逻辑在电路上可以很方便地通过电子器件的开关特性来实现，也就是用高、低电平分别表示逻辑 1 和逻辑 0 两种状态。表 1-1 所示为电压与逻辑电平的对照关系。

表 1-1　电压与逻辑电平的对照关系

电　压	二值逻辑	逻辑电平
+5 V	1	H(高电平)
0 V	0	L(低电平)

表 1-1 中用“1”表示高电平，用“0”表示低电平，这是一种正逻辑；反之，则称为负逻辑。除特别说明外，本书都是采用正逻辑表示方法。

图 1-2 所示为信号 11010100 的数字波形，即数字信号用逻辑电平对时间的图形表示，一般都画成理想波形表示高低电平所经历的时间，其中 1 和 0 每位数据占用的最小时间为位时间，我们常说的比特率就是每秒钟所传输的数据位数，也称为数据率。

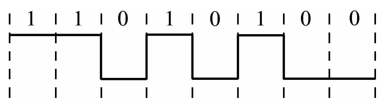


图 1-2　数字信号的传输波形

数字信号是一种脉冲信号，理想的脉冲波形的突变部分是瞬时的，不占用时间，如图 1-2 所示的方波信号。但在实际波形中，脉冲电压从零值跃变到最大值时或从最大值跃变到零值时都需要经过一定的时间。如图 1-3 所示，从脉冲幅值的 10% 到 90% 所经历的时间称为脉冲波形的上升时间(t_r)，从脉冲幅值的 90% 下降到 10% 所经历的时间称为脉冲波形的下降时间(t_f)。把脉冲幅值的 50% 的两个时间点之间的部分称为脉冲宽度(t_W)，它表示脉冲持续的时间，它占整个周期的百分比就称为占空比(q)，$q(\%) = \frac{t_W}{T} \times 100\%$。占空比是一个常用参数，显然，图 1-3 所示的方波信号的占空比为 50%。

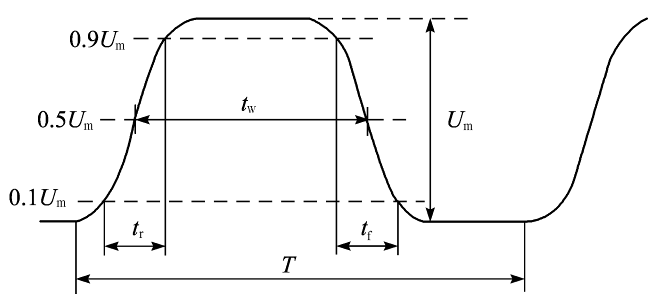


图 1-3　实际的脉冲波形

二、数字电路

工作于数字信号下的电路称为数字电路，它可以实现数字信号的变换、处理和传输。由于数字信号的离散特性，所以数字电路中的二极管、三极管以及由它们组成的集成电路主要工作在开关状态，因此我们更多关注的是它们工作的饱和区和截止区，而放大区只是其过渡状态。

如果把数字电路的基本单元逻辑门电路集成在一块半导体芯片上，就构成了数字集成电路，这也是当前应用的主流形式，它从 20 世纪 60 年代的小规模集成电路，发展到现在的中规模、大规模、超大规模、甚大规模集成电路，集成度不断提高，在工业自动化、通信系统等领域得到广泛应用。表 1-2 为从集成度角度对数字集成电路进行的分类。

表 1-2　从集成度角度对数字集成电路进行的分类

分　类	集成度(门的个数)	典型集成电路
小规模集成电路	最多 12 个	逻辑门、触发器
中规模集成电路	12 ～ 99	计数器、加法器
大规模集成电路	100 ～ 9 999	小型存储器、门阵列
超大规模集成电路	10 000 ～ 99 999	大型存储器、微处理器
甚大规模集成电路	10^6 以上	可编程逻辑器件、多功能专用集成电路

另外，如果按照半导体材料、结构和生产工艺还可以把数字集成电路分为 TTL 型和 CMOS 型器件，特别是 CMOS 工艺的发展，使得 CMOS 型集成电路具有更低的功耗、更高的集成度和工作速度，而且抗干扰能力强，目前逐渐在应用中占据了主导地位。

三、数字电路的特点

数字电路在信号的存储、处理和传输上比模拟电路具有更大的优势：

(1) 数字技术能够完成许多复杂的信号处理工作。数字电路主要对用 0 和 1 表示的数字信号进行运算和处理，只要能可靠地区分 0 和 1 这两种状态就可以正常工作，易于完成复杂信号的处理工作。

(2) 数字电路不仅能够完成算术运算，而且能够完成逻辑运算，具有逻辑推理和逻辑判断的能力，因此其也被称为数字逻辑电路或逻辑电路，这在控制系统中非常重要。

(3) 由数字电路组成的数字系统，抗干扰能力强，可靠性高，精确性和稳定性好，便于使用、维护和进行故障诊断，容易完成实时处理任务。

(4) 高速度，低功耗，可编程。现代化的生产工艺使得数字器件的工作速度越来越快，而功耗却可以越来越低，超大规模集成芯片的功耗甚至可以达到毫瓦级。另外，可编程器件的使用可以让用户根据自己的需要来定制芯片，提高了电路设计的灵活性，并大大缩短了研发周期。

但数字电路也有自身的局限性，自然界中大多数物理量都是模拟量，数字技术不能直接处理模拟信号，也不能直接使用处理后的数字信号，必须经过模 / 数和数 / 模转换器把模拟信号和数字信号进行相互转换，所以实际的电子系统通常都是模拟电路和数字电路的结合体，在发展数字电子技术的同时也要重视模拟电子技术的发展。

学习单元二　数制及二进制代码

一、进位计数制

人们在日常生活中经常使用十进制数来计数，即把 0 ～ 9 十个数码中的一个或几个按照一定的规律排列起来表达物体数量的多少。像这种多位数码的特定构成方式及从低位到高位的进位规则就称为进位计数制，简称数制。除了常用的十进制数，在计算机这样的数字系统中，广泛采用的还有二进制数、十六进制数等表示方式。

1. 十进制

十进制是用 0，1，2，…，9 十个不同的数码按一定的规律排成序列计数。数码的个数称作基数，十进制就是以 10 为基数的计数体制。当数码处于数字序列的不同位置时，它所表示的数值也不同。

例如，十进制数 108.2 可写成

$$(108.2)_D = 1 \times 10^2 + 0 \times 10^1 + 8 \times 10^0 + 2 \times 10^{-1}$$

一般十进制数用下标“D”或“10”来表示。左边是最高位，右边是最低位，各位的数值就是这一位的数码乘上处于这位的固定常数，如最高位的数值就是这一位的数码 1 乘上处于这一位的固定常数 10^2，这里的固定常数称为“权”。10^2、10^1、10^0 分别为百、十、个位的权值，小数点右边的权值以 10 的负幂表示，相邻两位的权值正好相差基数的 10 位倍，位即遵循逢十进一的进位规则。因此，可以这样表示任意一个十进制数：

$$(N)_D = \sum_{i=-\infty}^{\infty} K_i \times 10^i$$

式中，K_i 为基数 10 的第 i 次幂的系数，它可以是 0 ～ 9 中的任意一个数字。

综上所述，十进制数的基本特点是：

(1) 采用 0，1，2，…，9 这十个不同的数码来计数，基数为 10。

(2) 计数规律是“逢十进一”或“借一当十”。

由于十进制数需要表示十个数码，用数字电路实现很复杂且不经济，因此数字电路中一般不直接采用十进制。

2. 二进制

二进制数与十进制数的排序规律相似，区别仅在于基数不同。仿照十进制的描述，可知二进制数的基本特点是：

(1) 采用 0 和 1 两个数码来计数，基数为 2。

(2) 计数规律是“逢二进一”，即 $1 + 1 = 10$（读作“壹零”）。

任意一个二进制数可表示为

$$(N)_B = \sum_{i=-\infty}^{\infty} K_i \times 2^i$$

式中，二进制数用下标“B”或“2”来表示，K_i 为 0 或 1。根据此式可以方便地把二进制数转换为十进制数。

例 1.2.1　$(1011)_B = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = (11)_D$

$(1001.1)_B = 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} = (9.5)_D$

二进制的运算规则有：

加法　$0+0=0$　$0+1=1$　$1+0=1$　$1+1=10$

乘法　$0 \times 0=0$　$0 \times 1=0$　$1 \times 0=0$　$1 \times 1=1$

二进制比较简单，只有 0 和 1 两个数码，在数字电路中能通过三极管的饱和与截止、电平的高与低等方便地表示两种状态，只要规定其中一种状态为“1”，另一种状态为“0”，就可以用来表示二进制数，而且二进制的运算简单，所以二进制在数字电路中被广泛应用。

3. 十六进制

用二进制表示数时位数很多，不便于书写和记忆，为了便于描述二进制数，通常采用易于转换的十六进制数。

十六进制数的基本特点是：

(1) 采用 0 ~ 9 和 A(10)、B(11)、C(12)、D(13)、E(14)、F(15) 共 16 个数码，基数为 16。

(2) 计数规律是“逢十六进一”。

任意一个十六进制数可表示为

$$(N)_H = \sum_{i=-\infty}^{\infty} K_i \times 16^i$$

式中，十六进制数用下标“H”或“16”来表示，K_i 为 0 ~ 9 和 A ~ F 中的任一数字。

二、不同数制之间的转换

出于习惯，人们通常采用十进制数计数，但数字系统内部运算都按二进制来进行，因此必须知道这几种数制间的相互转换关系。

1. 其他进制数转换成十进制数

根据二进制数、十六进制数的位权展开式展开相加，可以很方便地将一个数转换成十进制数。

例 1.2.2

$$\begin{aligned}(1011001.001)_B &= 2^6 \times 1 + 2^5 \times 0 + 2^4 \times 1 + 2^3 \times 1 + 2^2 \times 0 + \\ &\quad 2^1 \times 0 + 2^0 \times 1 + 2^{-1} \times 0 + 2^{-2} \times 0 + 2^{-3} \times 1 \\ &= 64 + 0 + 16 + 8 + 0 + 0 + 1 + 0 + 0 + 0.125 \\ &= (89.125)_D\end{aligned}$$

$$(4EA)_H = 4 \times 16^2 + 14 \times 16^1 + 10 \times 16^0 = (1258)_D$$

2. 十进制数转换成其他进制数

将十进制数转换成其他进制数时要对整数部分和小数部分分开转换。整数部分采用连除基数取余，再将余数逆序排列得到转换数据的整数部分；小数部分则采用连乘基数取整，再将整数顺序排列得到转换数据的小数部分。下面以十进制数转换为二进制数为例来说明转换的过程。

例 1.2.3　将十进制数 25.625 转换成二进制数。

解　整数部分的转换过程如下：

			余数	
2	25	……	1	最低位
2	12	……	0	
2	6	……	0	
2	3	……	1	
2	1	……	1	最高位
	0			

小数部分的转换过程如下：

		整数	
$0.625 \times 2 = 1.25$	……	1	最高位
$0.25 \times 2 = 0.5$	……	0	
$0.5 \times 2 = 1$	……	1	最低位

所以$(25.625)_D = (11001.101)_B$

3. 二进制数与十六进制数的相互转换

因为 $2^4 = 16$，所以 4 位二进制数共有 16 种组合状态，可以分别用来表示十六进制的 16 个数码。这样，二进制数转换为十六进制数时，每 4 位二进制数对应转换成 1 位十六进制数。整数部分从小数点往左每 4 位一组，最高位组若不够四位则补 0；小数部分从小数点往右每 4 位一组，最后一组不够四位也补 0。将十六进制数转换成二进制数，只要把每 1 位十六进制数转换成对应的 4 位二进制数即可。

例 1.2.4　$(0110\quad 1010\quad 1111.0010)_B = (6AF.2)_H$

$(A7E)_H = (1010\quad 0111\quad 1110)_B$

三、二进制代码

在数字系统中，常用一定位数的二进制数来表示一些符号信息，这就是二进制代码。1 位二进制代码可以表示 2 个信号，2 位二进制代码可以表示 4 个信号，依次类推，n 位二进制代码可以表示 2^n 个不同的信号。在信息符号与二进制代码之间建立这种一一对应的关系就是编码。

若要求编码的信息有 N 项，则所需的二进制代码的位数 n 应满足 $2^n \geqslant N$。

1. 二 - 十进制码

二 - 十进制码（简称 BCD 码）就是指用一组 4 位二进制码表示一位十进制数的编码方式。这种代码具有二进制数的形式，又具有十进制数的特点，可以作为人与计算机联系时的一种中间表示。4 位二进制码最多可以有 16 种不同的组合方式，可以从中取任意 10 种组合来表示 0 ～ 9 这十个数码。当采用不同的编码方案时，可以得到不同形式的 BCD 码，如表 1-3 所示。

表 1-3　几种常用的 BCD 码

十进制数	8421 码	2421 码	5421 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	1011	1000	1000	0111
6	0110	1100	1001	1001	0101
7	0111	1101	1010	1010	0100
8	1000	1110	1011	1011	1100
9	1001	1111	1100	1100	1101

最基本、最常用的是 8421BCD 码，选用 0000～1001 这 10 种组合来代表十进制的 0～9。各位二进制数的权值分别为 2^3、2^2、2^1、2^0（8、4、2、1），故称为 8421BCD 码。由于它保存了二进制位权的特点，所以将二进制码各自乘以其权值后相加，即得所代表的十进制数。因而它与十进制数之间的转换是一种直接按位转换，即一组 4 位二进制数码代表 1 位十进制数。

例 1.2.5　$(15)_D = (0001\quad 0101)_{8421BCD}$

$(0011\quad 0110\quad 1001\quad 0000)_{8421BCD} = (3690)_D$

2. 可靠性编码

可靠性编码可以减少代码在形成和传输过程中由于各位变化速度不同而产生错误的概率。格雷（Gray）码又称循环码，就是一种可靠性编码。从表 1-3 中的排列情况可以看出其特点就是任何相邻的两组代码中仅有一位不同，所以在传输过程中不容易出错。

3. ASCII 码

计算机系统中有数字、字符和各种专用符号。美国标准信息交换码（American Standard Code for Information Interchange）简称 ASCII 码，就是一种常用的用二进制代码表示符号的编码方式。代码由 7 位二进制码组成，共有 $2^7=128$ 种状态，可以用来表示 128 个字符，这些字符包括数字、英文字母、控制符及其他一些符号和标记，具体请参见附录 B。ASCII 码常用在计算机的输入、输出设备上。

学习单元三　逻辑代数基础

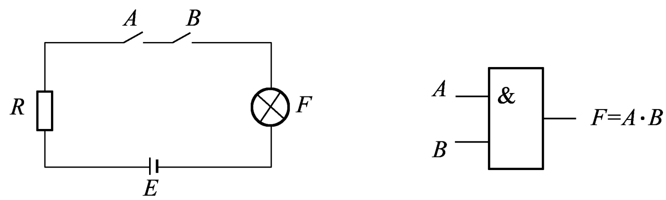
一、基本逻辑运算

逻辑代数又称布尔代数，由英国数学家乔治 · 布尔在 1847 年首先提出，它是研究逻辑函数（因变量）与逻辑变量（自变量）之间规律性的一门应用数学，是分析和设计逻辑电路的数学工具。和普通代数一样，逻辑代数也用字母 A、B、C 等来表示变量，但不同的是这些变量的取值范围只有 0 和 1 两种对立的逻辑状态，因而又称为二值逻辑变量。

基本的逻辑运算有三种：与、或、非。

1. 与运算

如图 1-4(a) 所示的串联开关电路是一个简单的与逻辑电路，开关 A、B 与灯 F 的状态关系如表 1-4(a) 所示。显然，只有当开关 A 和 B 都接通时，灯 F 才能亮，它们之间满足这样一种关系："只有当决定一件事情的条件全部具备之后，这件事情才会发生。"这种关系称为与逻辑。如果设图中的开关接通为 1，断开为 0，灯亮为 1，灯灭为 0，则由状态表可列出其真值表，如表 1-4(b) 所示。



(a) 电路图　　(b) 逻辑符号

图 1-4　与逻辑运算

表 1-4　与逻辑状态关系表和真值表

(a)

开关		灯 F
A	B	
断	断	灭
断	通	灭
通	断	灭
通	通	亮

(b)

A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

与运算的逻辑表达式为：$F = A \cdot B = AB$。

其中"•"表示与逻辑，通常可以省略。

与运算的运算规则有：$0 \cdot 0 = 0 \quad 0 \cdot 1 = 0 \quad 1 \cdot 0 = 0 \quad 1 \cdot 1 = 1$。

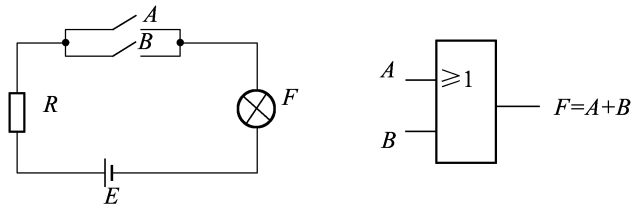
由此推出 $A \cdot 0 = 0 \quad A \cdot 1 = A \quad A \cdot A = A$。

输入、输出端能实现与运算的逻辑电路称为与门。其逻辑符号如图 1-4(b) 所示，图中的"&"表示与逻辑。

2. 或运算

图 1-5(a) 所示的并联开关电路为一个简单的或逻辑电路，开关 A、B 与灯 F 的状态关系如表 1-5(a) 所示。显然，当开关 A 和 B 任意一个接通，灯 F 都能亮，它们之间满足这样一种关系："决定一件事情的几个条件中，只要有一个条件得到满足，这件事情就会发生。"这种逻辑关系称为或逻辑，其真值表如表 1-5(b) 所示。

或运算的逻辑表达式为 $F = A + B$。



(a) 电路图　　　(b) 逻辑符号

图 1-5　或逻辑运算

符号“+”表示 A、B 做或运算，也称逻辑加。

或运算的运算规则有：$0+0=0\quad 0+1=1\quad 1+0=1\quad 1+1=1$。

由此推出 $A+0=A\quad A+1=1\quad A+A=A$。

或运算的逻辑符号如图 1-5(b) 所示。

表 1-5　或逻辑状态关系表和真值表

(a)

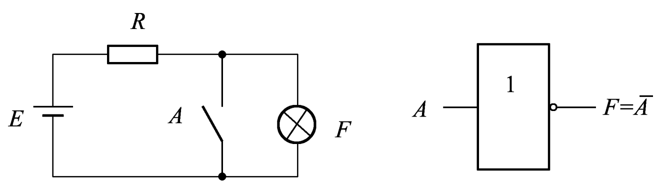
开　关		灯 F
A	B	
断	断	灭
断	通	亮
通	断	亮
通	通	亮

(b)

A	B	F
0	0	0
0	1	1
1	0	1
1	1	1

3. 非运算

图 1-6(a) 所示是非逻辑电路，开关 A 与灯 F 的状态关系如表 1-6(a) 所示。很明显，开关 A 接通时灯不亮，A 断开时灯亮。它们之间满足“某事的发生以另一件事不发生为条件”，这种逻辑关系称为非逻辑。其真值表如表 1-6(b) 所示。从表中可以看出，A 与 F 总是处于相反的逻辑状态，所以非运算的逻辑表达式为 $F=\overline{A}$。



(a) 电路图　　　(b) 逻辑符号

图 1-6　非逻辑运算

表 1-6　非逻辑状态关系表和真值表

(a)

开关 A	灯 F
断	亮
通	灭

(b)

A	F
0	1
1	0

非运算的运算规则有：$\overline{0}=1\quad \overline{1}=0$。

由此推出：$A+\overline{A}=1\quad A\cdot\overline{A}=0\quad \overline{\overline{A}}=A$。

非运算的逻辑符号如图 1-6(b) 所示。

4. 复合逻辑运算

与、或、非三种基本逻辑运算按不同的方式组合，还可以构成与非、或非、与或非、异或、同或等复合逻辑运算，并构成相应的复合门电路。

1）与非运算

将与和非运算组合在一起可以构成与非运算，或称与非逻辑。与非运算的逻辑符号和真值表分别如图 1-7 和表 1-7 所示。逻辑表达式为 $F=\overline{A\cdot B}$，输入、输出端能实现与非运算的电路称为与非门。

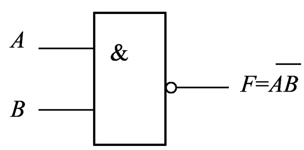


图 1-7　与非逻辑符号

表 1-7　与非逻辑真值表

A	B	F
0	0	1
0	1	1
1	0	1
1	1	0

2）或非运算

将或和非运算组合在一起构成或非运算，逻辑符号和真值表分别如图 1-8 和表 1-8 所示。

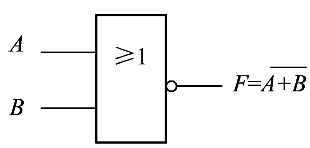


图 1-8　或非逻辑符号

表 1-8　或非逻辑真值表

A	B	F
0	0	1
0	1	0
1	0	0
1	1	0

逻辑表达式为 $F = \overline{A + B}$,能实现或非运算的电路称为或非门。

3）与或非运算

将与、或、非三种运算组合在一起可以构成与或非运算,逻辑符号如图 1-9 所示。

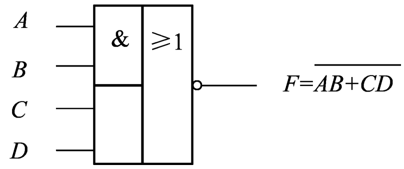


图 1-9　与或非逻辑符号

逻辑表达式为 $F = \overline{AB + CD}$。

4）异或运算

异或逻辑关系是:输入不同时,输出为 1;输入相同时,输出为 0。异或逻辑符号和真值表分别如图 1-10 和表 1-9 所示。由真值表可知输出 $F = 1$ 的条件是 $A \neq B$,而当 $A = B$ 时有$F = 0$。

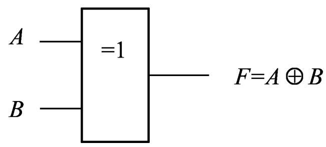


图 1-10　异或逻辑符号

表 1-9　异或逻辑真值表

A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

异或逻辑表达式为 $F = A\overline{B} + \overline{A}B = A \oplus B$。

异或逻辑功能可以通过异或门来实现。

5）同或运算

同或逻辑关系是:输入相同时,输出为 1;输入不同时,输出为 0。同或与异或逻辑正好相反,其逻辑符号和真值表分别如图 1-11 和表 1-10 所示。由真值表可知 $A = B$ 时 $F = 1$,$A \neq B$ 时 $F = 0$。

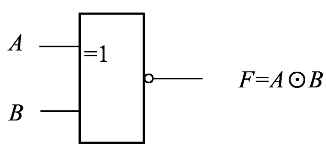


图 1-11　同或逻辑符号

表 1-10　同或逻辑真值表

A	B	F
0	0	1
0	1	0
1	0	0
1	1	1

同或逻辑表达式为 $F = AB + \bar{A}\,\bar{B} = A \odot B = \overline{A \oplus B}$。

同或逻辑功能可以通过同或门来实现。

二、基本定律、常用公式和基本规则

1. 基本定律和常用公式

根据逻辑与、或、非三种基本运算规则，可推导出逻辑运算的一些基本定律和常用公式，如表 1-11 所示。

表 1-11　基本定律和常用公式

基本定律	表达式	
0-1 律	$A + 0 = A$	$A \cdot 0 = 0$
	$A + 1 = 1$	$A \cdot 1 = A$
	$A + A = A$	$A \cdot A = A$
	$A + \bar{A} = 1$	$A \cdot \bar{A} = 0$
		$\bar{\bar{A}} = A$
结合律	$(A + B) + C = A + (B + C)$	$(AB)C = A(BC)$
交换律	$A + B = B + A$	$AB = BA$
分配律	$A(B + C) = AB + AC$	$A + BC = (A + B)(A + C)$
反演律	$\overline{A + B} = \bar{A} \cdot \bar{B}$	$\overline{AB} = \bar{A} + \bar{B}$
吸收律	$A + AB = A$	$A(A + B) = A$
	$AB + A\bar{B} = A$	$(A + B)(A + \bar{B}) = A$
	$A + \bar{A}B = A + B$	$A(\bar{A} + B) = AB$
多余项定律	$AB + \bar{A}C + BC = AB + \bar{A}C$	$(A + B)(\bar{A} + C)(B + C) = (A + B)(\bar{A} + C)$

表 1-1 中的公式可直接用真值表或其他基本定律来证明，用真值表时，需要列出等式左右两边函数的真值表，如果两边真值表相同，说明等式成立。

例 1.3.1　证明分配律 $A(B + C) = AB + AC$ 成立。

证明　将变量 A、B、C 的全部取值组合分别代入等式两边，结果如表 1-12 所示，即对于任意一组取值，等式 $A(B + C) = AB + AC$ 均成立。

表 1-12　例 1.3.1 真值表

A	B	C	$A(B+C)$	$AB+AC$
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

例 1.3.2　证明等式 $AB+\overline{A}C+BC=AB+\overline{A}C$ 成立。

证明　直接利用基本定律，有

$$\begin{aligned} AB+\overline{A}C+BC &= AB+\overline{A}C+(A+\overline{A})BC \\ &= AB+\overline{A}C+ABC+\overline{A}BC \\ &= AB(1+C)+\overline{A}C(1+B) \\ &= AB+\overline{A}C \end{aligned}$$

2. 基本规则

逻辑代数中还有三个重要规则，可以帮助我们利用已知基本定律方便地推导出更多的公式。

1）代入规则

在任何一个逻辑等式中，如果等式两边出现相同的变量，如变量 A，可以将所有含 A 的地方代之以同一个逻辑函数 F，等式仍然成立，这个规则就称为代入规则。

因为任何一个逻辑函数 F 只有两种可能的取值 0 和 1，所以代入规则正确，可以把代入规则看作是公理，利用它来扩大等式的应用范围。

例如，将摩根定律 $\overline{A+B}=\overline{A}\cdot\overline{B}$ 两边的 B 用函数 $B+C$ 代替，有

$$\overline{A+B+C}=\overline{A}\cdot\overline{B+C}=\overline{A}\cdot\overline{B}\cdot\overline{C}$$

从而说明摩根定律对于多变量的情况依然成立。

2）反演规则

对逻辑等式 F 取非（求其反函数）称为反演。可以通过反复使用摩根定律求得，也可以运用由摩根定律得到的反演规则一次写出。

在对逻辑等式 F 取非时，如果将等式 F 做了如下变换：

$$\text{将 } F \text{ 中所有的}\begin{cases} \text{运算符}\begin{cases} \cdot \rightarrow + \\ + \rightarrow \cdot \end{cases} \\ \text{变量}\begin{cases} \text{原变量} \rightarrow \text{反变量} \\ \text{反变量} \rightarrow \text{原变量} \end{cases} \\ \text{常量}\begin{cases} 0 \rightarrow 1 \\ 1 \rightarrow 0 \end{cases} \end{cases}$$

则得到的就是等式 F 的反函数 $\overline{F}$，这就是反演规则。

利用反演规则可以一次写出反函数表达式,使用起来非常方便,但要特别注意以下两个方面:

(1) 变换时要注意加括号来保持原式中的运算顺序。

(2) 不是在单个变量上面的非号应保持不变。

例 1.3.3 试求 $F = A \cdot \overline{B \cdot C} + (\overline{A} + \overline{B} \cdot \overline{C}) \cdot (A + C)$ 的反函数$\overline{F}$。

解 按照反演规则,注意保持运算顺序和反变量以外的非号不变,得

$$\overline{F} = (\overline{A} + \overline{\overline{B} + \overline{C}}) \cdot [A \cdot (B + C) + \overline{A} \cdot \overline{C}]$$

3) 对偶规则

$$\text{如果将逻辑表达式 } F \text{ 中所有的} \begin{cases} \text{运算符} \begin{cases} \cdot \rightarrow + \\ + \rightarrow \cdot \end{cases} \\ \text{常量} \begin{cases} 0 \rightarrow 1 \\ 1 \rightarrow 0 \end{cases} \end{cases}$$

就可得到一个新的表达式 F',F' 称为 F 的对偶式。

在变换时除了要保持原式中的运算顺序,还要注意和反演规则不同的是此处不应将原变量和反变量进行互换。

可以证明,如果两个逻辑式相等,那么它们的对偶式也一定相等,这就是对偶规则。

例如,$A + \overline{A}B = A + B$ 成立,则它的对偶式 $A(\overline{A} + B) = AB$ 也成立。

例 1.3.4 已知 $F = (\overline{A} + BC)\overline{\overline{C}D}$,求$\overline{F}$ 和 F'。

解 根据反演规则有$\overline{F} = A(\overline{B} + \overline{C}) + \overline{C + \overline{D}}$;

根据对偶规则有 $F' = \overline{A}(B + C) + \overline{\overline{C} + D}$。

学习单元四　逻辑函数及其化简

一、逻辑函数表达式

比较复杂一些的逻辑电路往往有多个逻辑变量,用来描述输入、输出逻辑变量之间因果关系的函数就称为逻辑函数。逻辑函数的描述方法有真值表、逻辑函数表达式、逻辑图、波形图和卡诺图等,下面先讨论逻辑函数建立的过程,一般实际问题往往要复杂得多,但基本过程都类似。

例 1.4.1 图 1-12 所示是一个控制楼梯照明的双联开关电路,A 装在楼上,B 装在楼下,楼下开灯可在楼上关灯,楼上开灯可在楼下关灯,即两个开关都同时处于上或下时电路才导通。试找出描述开关 A、B 与灯 F 之间逻辑关系的逻辑函数表达式。

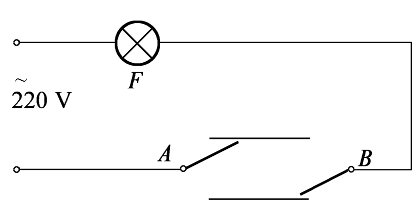


图 1-12　双联开关电路

解　要建立逻辑函数，一般可以根据实际问题先使用真值表来描述逻辑关系，再根据真值表找到逻辑函数表达式。

根据此问题取变量 A、B、F，设 $F=1$ 为灯亮，$F=0$ 为灯熄灭，开关向上扳为 1，开关向下扳为 0，则可把 A、B、F 之间的所有组合情况列出，得真值表如表 1-13 所示。

表 1-13　例 1.4.1 真值表

A	B	F
0	0	1
0	1	0
1	0	0
1	1	1

由真值表可知，只有在输入 $A=B=0$ 和 $A=B=1$ 两种组合时才能使灯亮(输出 $F=1$)，用原变量表示取 1，反变量表示取 0，可以写出电路的逻辑函数表达式为 $F=\overline{A}\,\overline{B}+AB$。

二、逻辑函数的代数化简法

逻辑函数可以有多种不同的表达方式，如果逻辑表达式比较简单，那么用来实现的逻辑电路所用元件也较少，这样不仅可以降低成本，而且可以提高电路的可靠性，因此我们经常需要通过化简的手段找出逻辑函数的最简形式，并且根据不同类型的表达式选择不同的化简标准。由于使用中最常见的是与或式，而且也容易同其他形式的表达式互相转换，所以我们主要介绍与或式的化简，即使得化简后的函数具有最少的项数且每一项中所含变量个数也最少。化简逻辑函数的常用方法主要有代数化简法和卡诺图化简法。

代数化简法又称为公式化简法，就是反复运用逻辑代数的基本定律和公式消去多余的乘积项或多余的因子。这种方法没有固定的步骤，需要熟练掌握公式和运用一些技巧。下面介绍一些常用的化简方法。

1. 并项法

利用公式 $A+\overline{A}=1$ 将两项合并为一项，同时消去一个变量。

例 1.4.2　$A\overline{B}\,\overline{C}+\overline{A}\,\overline{B}\,\overline{C}=(A+\overline{A})\overline{B}\,\overline{C}=\overline{B}\,\overline{C}$

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

2. 吸收法

利用公式 $A+AB=A$ 吸收多余项 AB。

例 1.4.3　$A+ABC+\overline{C}D=A+\overline{C}D$

$AB+ABC(\overline{D}+E)=AB$

3. 消去法

利用公式 $A+\overline{A}B=A+B$ 消去多余因子，或利用公式 $AB+\overline{A}C+BC=AB+\overline{A}C$ 消去多余项。

例 1.4.4　$AB+\overline{A}C+\overline{B}C=AB+(\overline{A}+\overline{B})C=AB+\overline{AB}C=AB+C$

4. 配项法

利用公式 $A+\overline{A}=1$ 将某一项展开为两项以增加必要的乘积项，再利用前面的方法与其

他乘积项合并使项数减少，得到最简结果。

例 1.4.5 $AB+\overline{A}C+BC=AB+\overline{A}C+(A+\overline{A})BC=AB+\overline{A}C+ABC+\overline{A}BC$

$$=AB(1+C)+\overline{A}C(1+B)=AB+\overline{A}C$$

通常要综合运用上述方法来进行逻辑函数的化简。

例 1.4.6 化简函数 $F=AD+A\overline{D}+AB+\overline{A}C+BD+ACEF+\overline{B}E+DEF$。

解 (1) 用公式 $A+\overline{A}=1$ 合并 $AD+A\overline{D}=A$，得

$$F=A+AB+\overline{A}C+BD+ACEF+\overline{B}E+DEF$$

(2) 用公式 $A+AB=A$ 合并 $A+AB+ACEF=A$，得

$$F=A+\overline{A}C+BD+\overline{B}E+DEF$$

(3) 用公式 $A+\overline{A}B=A+B$ 合并 $A+\overline{A}C=A+C$，

用公式 $AB+\overline{A}C+BC=AB+\overline{A}C$ 合并 $BD+\overline{B}E+DEF=BD+\overline{B}E$，

最后得 $F=A+C+BD+\overline{B}E$。

三、逻辑函数的卡诺图化简法

使用代数化简法需要记忆大量公式，在化简过程中没有统一的模式或步骤可以遵循，并且很难判断化简结果是否最简。因此，在逻辑变量不是很多的情况下(一般不超过 6 个)，使用下面介绍的卡诺图化简法可以很方便地得到最简逻辑表达式。

1. 逻辑函数的最小项表达式

1) 逻辑函数的最小项及其性质

在 n 变量的逻辑函数中，如果某个乘积项含有逻辑问题的全部 n 个变量，每个变量都以它的原变量或反变量的形式出现且仅出现一次，这样的乘积项就称为 n 变量逻辑函数的最小项。

例如，对于三变量 A、B、C 来说，可以构成许多乘积项，但只有 8 个最小项：$\overline{A}\overline{B}\overline{C}$、$\overline{A}\overline{B}C$、$\overline{A}B\overline{C}$、$\overline{A}BC$、$A\overline{B}\overline{C}$、$A\overline{B}C$、$AB\overline{C}$、$ABC$，而 A、AB、$A\overline{A}B$、$\overline{A}\ \overline{B}B$ 等就不是最小项。因此，对于 n 个变量共有 2^n 个最小项。

为了叙述和书写方便，通常用“m_i”表示最小项来进行编号，即把最小项中的原变量用 1 表示，反变量用 0 表示，对应的组合取值转换成十进制数作为下标 i 的值。如 $\overline{A}B\overline{C}$ 对应的取值为 010，而 010 相当于十进制数 2，所以把 $\overline{A}B\overline{C}$ 记作 m_2，以此类推，可以得到三变量最小项编号如表 1-14 所示。

表 1-14　三变量最小项编号

变量 ABC 取值	最小项	编　号
000	$\overline{A}\,\overline{B}\,\overline{C}$	m_0
001	$\overline{A}\,\overline{B}C$	m_1
010	$\overline{A}B\overline{C}$	m_2
011	$\overline{A}BC$	m_3
100	$A\overline{B}\overline{C}$	m_4
101	$A\overline{B}C$	m_5
110	$AB\overline{C}$	m_6
111	ABC	m_7

由表 1-4 可以看出，最小项具有下列性质：

(1) 对于任意一个最小项，有且仅有一组变量取值使它等于 1，而取其他各组值时，此最小项的值均为 0。如对于 $\overline{A}B\overline{C}$，只有 ABC 取值为 010 时其值才为 1，而其他取值全为 0。

(2) 任意两个不同最小项的乘积恒为 0，根据性质(1)，当 ABC 取某个值时两个不同的最小项至少有一个为 0，因而乘积始终为 0。

(3) n 个变量的全部最小项的和恒为 1。

2) 最小项表达式

一个全以最小项组成的与或式逻辑函数就是最小项表达式。它可以写成如下形式：

$$
\begin{aligned}
F = F(A,B,C) &= \overline{A}B\,\overline{C} + \overline{A}\,\overline{B}\,\overline{C} + AB\,\overline{C} + A\,\overline{B}C \\
&= m_2 + m_0 + m_6 + m_5 = \sum m(0,2,5,6)
\end{aligned}
$$

任何一个逻辑函数都能展开成最小项表达式。

例 1.4.7　将 $F = AB + A\overline{B}C$ 化成最小项表达式。

解　AB 不是最小项，可以通过配项展开：

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

2. 用卡诺图表示逻辑函数

1) 逻辑变量的卡诺图

卡诺图是一种能直观表示最小项逻辑关系的方格图，是逻辑函数的图形表示。由于卡诺图中每一小方块都表示了一个最小项，并且最小项的逻辑相邻关系表现在几何位置上的相邻，所以通过卡诺图使得寻找、合并、化简最小项的工作变得更加直接、简便。

卡诺图的画法很多，这里仅介绍常用的一种。

如图 1-13 所示画出了二变量卡诺图的基本形式和简化形式。每个小方格代表一个最小项，并且任何两个逻辑相邻的最小项，其所对应的小方格在几何位置上也相邻，相邻的两格所表示的最小项都仅有一个因子不同。

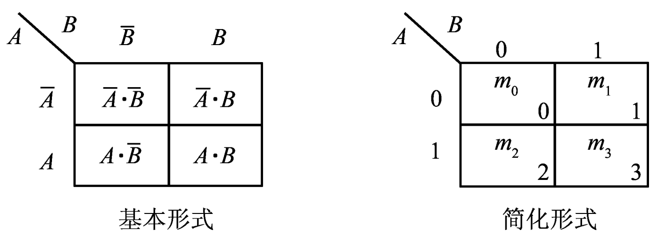


图 1-13　二变量卡诺图

如图 1-14 所示分别画出了三变量、四变量的卡诺图。

$A \backslash BC$	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

(a) 三变量卡诺图

$AB \backslash CD$	00	01	11	10
00	m_0	m_1	m_3	m_2
01	m_4	m_5	m_7	m_6
11	m_{12}	m_{13}	m_{15}	m_{14}
10	m_8	m_9	m_{11}	m_{10}

(b) 四变量卡诺图

图 1-14　多变量卡诺图

在多变量卡诺图中要注意相邻的特性，因为卡诺图可以卷起来看。也可以折叠起来看，在寻找相邻性上要注意上、下、左、右尤其是边和角的邻格。如四变量卡诺图中，当左、右卷起来看时，左边第一列与右边第四列相邻；当上、下卷起来看时，上边第一行还与下边第四行相邻，因此，四个角的小方格也具有相邻性。

2）用卡诺图表示逻辑函数

由于任何一个逻辑函数都可以展开成最小项表达式，所以只要根据变量数画出对应的卡诺图，然后按最小项编号把逻辑函数中包含的最小项在相应的方格中填 1，没有包含的项填 0（也可不填），就可以得到用卡诺图表示的逻辑函数。

例 1.4.8　画出 $F = AB + A\bar{B}C$ 的卡诺图。

解　先化成最小项表达式：

$$F = AB(C+\bar{C}) + A\bar{B}C = ABC + AB\bar{C} + A\bar{B}C$$

$$= m_7 + m_6 + m_5 = \sum m(5,6,7)$$

根据三变量的卡诺图形式，将所包含的最小项填进去，如图 1-15 所示。

$A \backslash BC$	00	01	11	10
0	0	0	0	0
1	0	1	1	1

图 1-15　$F = AB + A\bar{B}C$ 的卡诺图

实际上当我们熟悉卡诺图之后，可以不必每次都先求出最小项表达式，而只要根据函数与或式寻找变量的公共部位就可以得到卡诺图了。

如例 1.4.8 中的乘积项 AB 不是最小项，填图时可以寻找 A 和 B 的公共部位直接填入，所找的公共部位既要在图 1-16(a) 所示的阴影部分中，又要在图 1-16(b) 所示的阴影部分中，其公共部位恰好是 m_6、m_7 两格，如图 1-16(c) 所示，填好后与刚才的结果一致。

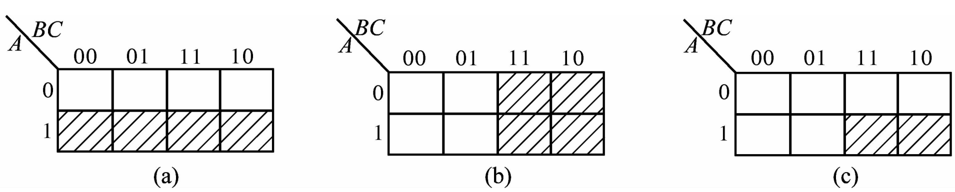


图 1-16　寻找变量的公共部位

3. 利用卡诺图化简逻辑函数

用卡诺图化简逻辑函数，实质上就是利用相邻性反复运用公式 $AB+A\overline{B}=A$ 合并最小项，消去相异的变量，得到最简与或式。具体的化简方法就是“画包围圈”。由于图中任意两个相邻项都仅有一个因子不同，所以两个相邻项合并可消去一个相异变量，4 个相邻项合并可消去两个相异变量，以此类推，2^n 个相邻项合并时，可消去 n 个相异变量。因此，画包围圈应遵循如下原则：

（1）必须包含函数所有的最小项，即为 1 的小方格必须全部含在包围圈中。

（2）包围圈只能圈 2^n 个方格，且圈越大越好，因为圈越大消去的相异变量越多，得到的结果越简单。

（3）不同的包围圈可以重复圈同一个区域，但每个圈中至少要包含一个尚未被圈过的 1。

（4）包围圈的圈数要尽可能的少，这也意味着最后乘积项的数目少，圈数越少，电路中所用的门数也越少。

画好包围圈，就可以进行最小项合并了。将包围圈中既含有原变量又含有反变量的那些变量消去，而保留变量取值不变的项，如图 1-17 所示合并最小项的几种情况。最后把所有乘积项相加就得到最简与或表达式了。

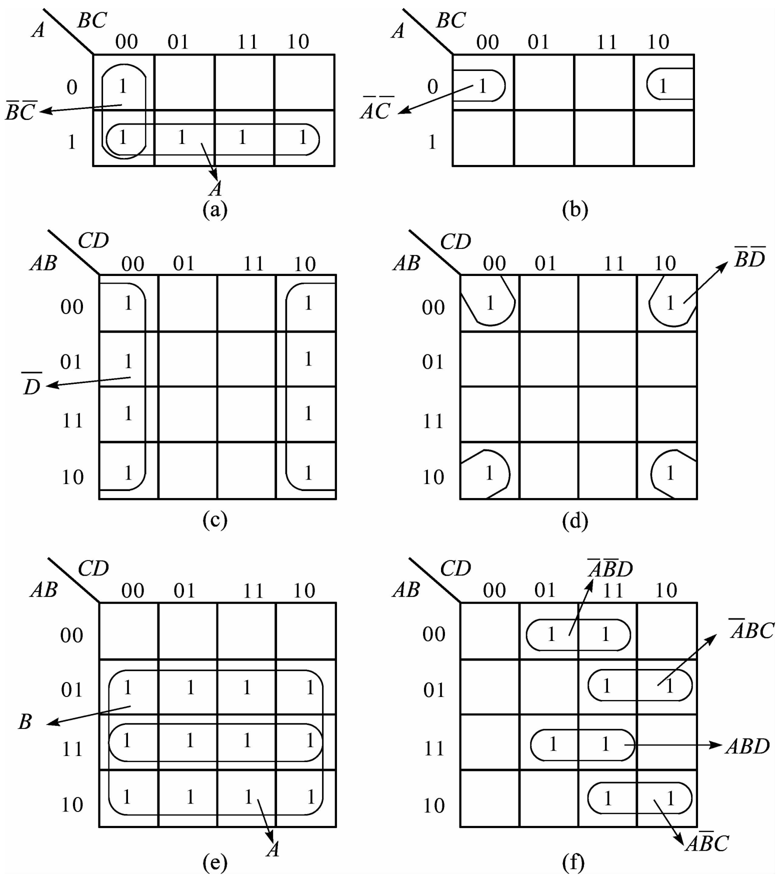


图 1-17　合并最小项

因此，归纳起来，用卡诺图化简逻辑函数可分成以下三个步骤进行：

（1）根据逻辑函数建立卡诺图，注意要包括所有的逻辑变量。

（2）将相邻含 1 的小方格画入包围圈，对应每个包围圈合并成一个新的乘积项。

(3) 将所有包围圈对应的乘积项相加即可得到最简与或式。

例 1.4.9 用卡诺图法化简逻辑函数 $F=\overline{A}B+A\overline{B}+BC+AB\overline{C}+\overline{A}\,\overline{B}\,\overline{C}$。

解 画出逻辑函数 F 的卡诺图,如图 1-18 所示,按照合并规律画出最小项的包围圈,得最简式为 $F=A+B+\overline{C}$。

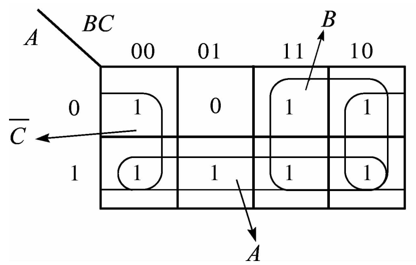


图 1-18 例 1.4.9 的卡诺图

例 1.4.10 用卡诺图法化简逻辑函数 $F(A,B,C,D)=\sum m(0,2,4,8,10,12)$。

解 由表达式画出卡诺图,如图 1-19 所示,画包围圈合并最小项,得最简与或表达式为 $F=\overline{C}\,\overline{D}+\overline{B}\,\overline{D}$。

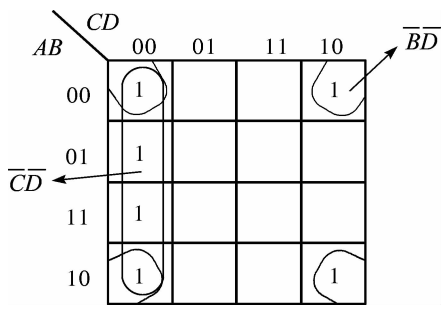


图 1-19 例 1.4.10 的卡诺图

4. 具有约束项的逻辑函数化简

约束条件反映了逻辑函数中各逻辑变量之间的制约关系。约束条件所含的最小项称为约束项,它表示输入变量某些取值组合不允许出现,或者不影响逻辑函数的输出,因此也被称为无关项、任意项,一般用 d_i 表示,i 仍为最小项序号,填入卡诺图时约束项用"×" 表示。

约束项可以视需要取值为 1 或 0,而不会影响其函数值,可以充分利用约束项使表达式大大简化。

例 1.4.11 某逻辑电路的输入 $ABCD$ 是十进制数 X 的 8421BCD 码,设计一个逻辑电路实现四舍五入的判断功能,即当 $X\geqslant 5$ 时,输出 $F=1$,否则输出 $F=0$,求 F 的最简与或表达式。

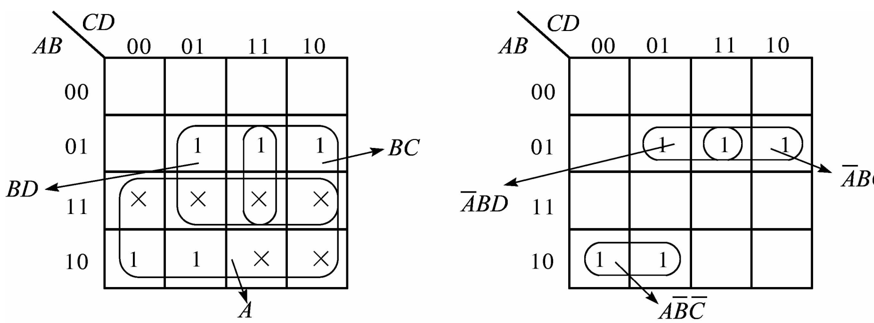
解 根据题意,列出真值表如表 1-15 所示,因为 0 ~ 9 这 10 个数码对应的 8421BCD 码是 0000 ~ 1001,而取值 1010 ~ 1111 是不允许出现的,也就是说,这 6 个最小项是约束项。由真值表可以写出含有约束项的逻辑函数表达式为 $F=\sum m(5,6,7,8,9)+\sum d(10,11,12,13,14,15)$。

表 1-15　例 1.4.11 真值表

X	$A\ B\ C\ D$	F	X	$A\ B\ C\ D$	F
0	0 0 0 0	0	8	1 0 0 0	1
1	0 0 0 1	0	9	1 0 0 1	1
2	0 0 1 0	0	10	1 0 1 0	×
3	0 0 1 1	0	11	1 0 1 1	×
4	0 1 0 0	0	12	1 1 0 0	×
5	0 1 0 1	1	13	1 1 0 1	×
6	0 1 1 0	1	14	1 1 1 0	×
7	0 1 1 1	1	15	1 1 1 1	×

画出卡诺图如图 1-20 所示，化简可得 $F=A+BD+BC$。

如果不考虑约束项，那么 $F=\overline{A}BD+\overline{A}BC+A\overline{B}\,\overline{C}$，显然，利用约束项后化简结果更简单。



(a) 利用约束项化简　　(b) 不利用约束项化简

图 1-20　例 1.4.11 的卡诺图

技能实训　裁判表决器电路的设计与调试

一、项目制作目的

(1) 了解并掌握裁判表决器电路的设计和制作方法。

(2) 掌握用仿真软件进行实际电路设计和调试的方法。

二、项目要求

(1) 设计电路应能完全满足项目题目的要求。

(2) 绘出裁判表决器电路的逻辑图。

(3) 完成裁判表决器电路的仿真调试。

(4) 完成裁判表决器电路的模拟接线安装。

三、项目步骤

(1) 根据项目要求列出真值表。设 A 代表主裁判，B、C、D 分别代表副裁判，真值表

如图 1-21 所示。

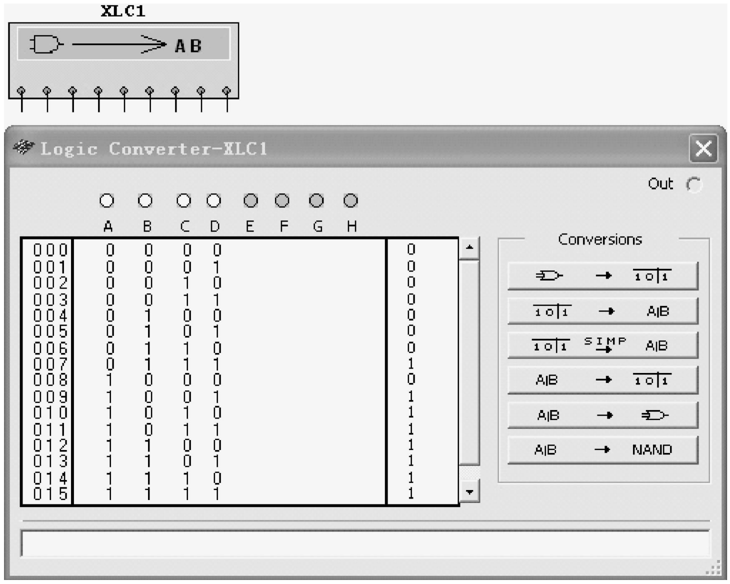


图 1-21　裁判表决器电路的真值表

(2) 由真值表到逻辑表达式和电路的逻辑图。逻辑表达式为 $F=BCD+AB+AC+AD$ 或 $F=\overline{\overline{BCD}\cdot\overline{AB}\cdot\overline{AC}\cdot\overline{AD}}$。由仿真软件生成的逻辑图如图 1-22 所示。

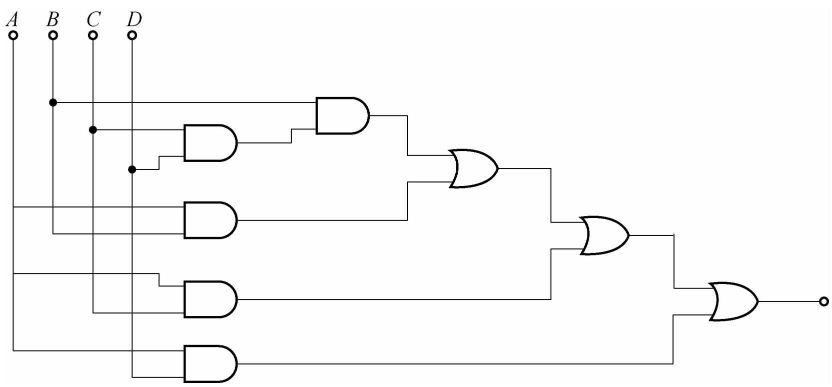


图 1-22　裁判表决器电路的逻辑图

需要说明的是，由于是软件自动生成的逻辑图，所以元件逻辑门的选取不尽合理，但该逻辑图的逻辑功能完全满足要求。具体选择什么型号的元件，由设计者根据工作需要进行选取。

(3) 选择合适的集成电路芯片，连接并调试。

(4) 发现调试中的问题并解决。

(5) 确定合适的裁判表决器设计电路，如图 1-23 所示。

图 1-23 所示电路此时的状态，仿真的是举重运动员成绩有效时的情形。

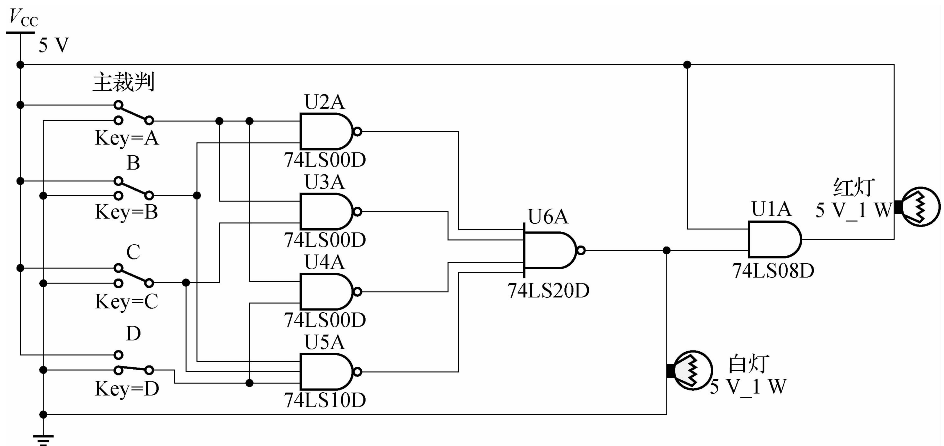


图 1-23　裁判表决器的仿真电路

（6）进行裁判表决器电路的模拟接线安装，打开控制工具栏中的调用面包板按钮，如图 1-24 所示。模拟接线后的效果，如图 1-25 所示。

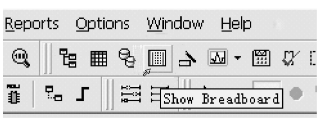


图 1-24　面包板的调用

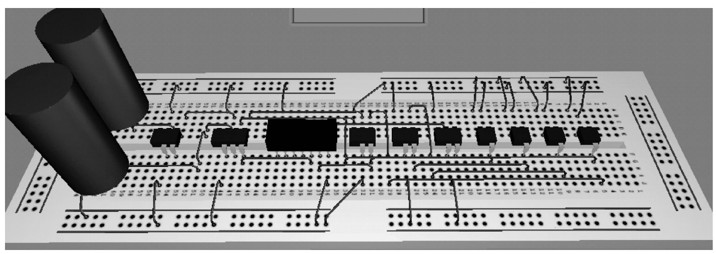


图 1-25　裁判表决器电路接线

四、注意事项

（1）模拟接线完全按照在实际接线时的操作要求，在面包板上进行接线。
（2）表决结果的显示可以用灯泡，也可以选用发光二极管。

五、项目考核

<table>
<tr><td>班　级</td><td></td><td>姓　名</td><td></td><td>组　号</td><td></td><td rowspan="2">扣分记录</td><td rowspan="2">得　分</td></tr>
<tr><td>项　目</td><td>配　分</td><td colspan="2">考核要求</td><td colspan="2">评分细则</td></tr>
<tr><td>正确使用逻辑转换仪</td><td>15 分</td><td colspan="2">能使用逻辑转换仪将实际问题转化为真值表、逻辑表达式的形式</td><td colspan="2">（1）不会使用逻辑转换仪，扣 10 分；
（2）未能正确将实际问题转化为真值表、逻辑表达式，扣 5 分</td><td></td><td></td></tr>
</table>

续表

<table>
<tr><td>班　　级</td><td></td><td>姓　　名</td><td></td><td>组　　号</td><td></td><td rowspan="2">扣分记录</td><td rowspan="2">得　　分</td></tr>
<tr><td>项　　目</td><td>配　　分</td><td colspan="2">考核要求</td><td colspan="2">评分细则</td></tr>
<tr><td>能正确产生逻辑图</td><td>25 分</td><td colspan="2">能使用逻辑转换仪产生逻辑图，并能进行修正</td><td colspan="2">(1) 不能使用逻辑转换仪产生逻辑图，扣 10 分；
(2) 逻辑图不合理，未能修正，每处扣 5 分；
(3) 逻辑图不正确，扣 10 分</td><td></td><td></td></tr>
<tr><td>能根据逻辑图建立裁判表决器的仿真电路</td><td>35 分</td><td colspan="2">能正确进行仿真，能满足本题目的要求</td><td colspan="2">(1) 连接方法不正确，每处扣 5 分；
(2) 不能正确进行仿真，扣 5 分；
(3) 结果不准确，每次扣 5 分；
(4) 电路出现问题，不能调试，每次扣 5 分；
(5) 仿真电路不能实现题目要求，扣 10 分</td><td></td><td></td></tr>
<tr><td>能正确地进行仿真接线</td><td>15 分</td><td colspan="2">能正确地在面包板上进行仿真接线</td><td colspan="2">(1) 不会使用仿真面包板接线，扣 10 分；
(2) 接线错误，每处扣 5 分</td><td></td><td></td></tr>
<tr><td>安全文明操作</td><td>10 分</td><td colspan="2">(1) 安全用电，无人为损坏仪器、元件和设备；
(2) 保持环境整洁，秩序井然，操作习惯良好；
(3) 小组成员协作和谐，态度正确；
(4) 不迟到、早退、旷课</td><td colspan="2">(1) 违反操作规程，每次扣 5 分；
(2) 工作场地不整洁，扣 5 分</td><td></td><td></td></tr>
<tr><td colspan="7">总　　分</td><td></td></tr>
</table>

模块小结

(1) 数字信号在时间上和数值上均是离散变化的，工作于数字信号下的电路就是数字电路。在数字电路中采用高、低电平来表示数字信号的 1 和 0 两种状态，适于完成复杂的信号处理工作，在信号的存储、处理和传输上占有很大的优势。但是，信号的放大、转换和功能的执行却离不开模拟电路，因此，数字电路和模拟电路的发展总是相辅相成、互相促进的。

(2) 数制是人们对计数进位的简称，生活中人们常用十进制数；而在数字电路中，则采用二进制数，它只使用 0 和 1 两个数码，遵从“逢二进一”的计数规律，将它按权展开即可得到十进制数，十进制数也可以通过“连除基数取余、连乘基数取整”的方法转换成二进制数。由于二进制数位数很多，不便于书写和记忆，通常采用易于转换的十六进制数来描述二进制数。

(3) 逻辑代数又称布尔代数，是分析和设计逻辑电路的数学工具。逻辑变量是用来表示

逻辑关系的一种二值变量，取值范围只有 0 和 1 两种对立的逻辑状态。基本的逻辑运算主要有三种：与、或、非，逻辑运算符号如图 1-26 所示。

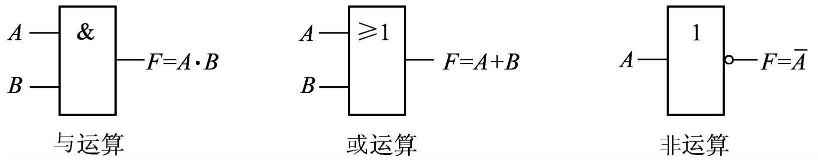


图 1-26　几种逻辑运算符号

（4）代入规则、反演规则和对偶规则是三个重要规则，可以帮助我们利用已知基本定律方便地推导出更多的公式。在运用时要注意写反函数和对偶式的区别。

$$\text{写反函数：}\begin{cases}\text{运算符}\begin{cases}\cdot \rightarrow +\\ + \rightarrow \cdot\end{cases}\\ \text{变量}\begin{cases}\text{原变量} \rightarrow \text{反变量}\\ \text{反变量} \rightarrow \text{原变量}\end{cases}\\ \text{常量}\begin{cases}0 \rightarrow 1\\ 1 \rightarrow 0\end{cases}\end{cases}\qquad \text{写对偶式：}\begin{cases}\text{运算符}\begin{cases}\cdot \rightarrow +\\ + \rightarrow \cdot\end{cases}\\ \text{常量}\begin{cases}0 \rightarrow 1\\ 1 \rightarrow 0\end{cases}\end{cases}$$

两者在变换时均要注意保持原式中的运算顺序，并且不是在单个变量上面的非号应保持不变。

（5）一个逻辑问题总可以用逻辑函数来描述。逻辑函数表达式越简单，逻辑电路就越简单，这有利于降低成本、提高电路的可靠性，因此逻辑函数的化简是本章的重点内容。常用的逻辑函数的化简主要有代数化简法和卡诺图化简法，代数化简法没有固定的方法和步骤，需要熟练运用各种定理和公式，有时还需要有一定的技巧和经验。常用公式化简方法如表 1-16 所示。

表 1-16　常用公式化简方法

方　法	所用公式	说　明
并项法	$A+\overline{A}=1$	将两项合并为一项，同时消去一个变量
吸收法	$A+AB=A$	吸收多余项 AB
消去法	$A+\overline{A}B=A+B$ $AB+\overline{A}C+BC=AB+\overline{A}C$	消去多余因子或多余项
配项法	$A+\overline{A}=1$	将某一项展开为两项以增加必要的乘积项

（6）卡诺图是一种能直观表示最小项逻辑关系的方格图，是逻辑函数的图形表示。每一小方块都表示了一个最小项，并且最小项的逻辑相邻关系表现在几何位置上的相邻。卡诺图化简法利用了几何相邻的特点，化简过程简单直观，有一定的步骤和方法可循，容易得到最简式。用卡诺图化简逻辑函数可分成以下三个步骤进行：

① 根据逻辑函数建立卡诺图，注意要包括所有的逻辑变量。

② 按照画包围圈的原则，将相邻含 1 的小方格画入包围圈，对应每个包围圈合并成一个新的乘积项。

③ 将所有包围圈对应的乘积项相加即可得到最简与或式。

根据约束项的特点合理使用约束项，可以使化简结果更简单。

习题 1

1.1 一数字信号的波形如图 1-27 所示，试写出该波形所表示的二进制数。

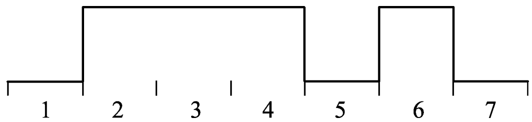


图 1-27 题 1.1 图

1.2 将下列十进制数转换为二进制数、十六进制数、8421BCD 码。

(1)26 (2)87 (3)255 (4)11.375

1.3 将下列二进制数转换为十进制数、十六进制数。

(1)1011 (2)1111111111 (3)11000101 (4)1010101.101

1.4 将下列十六进制数转换为十进制数、二进制数。

(1)3E (2)7D8 (3)3AF.E

1.5 已知 A、B 的波形如图 1-28 所示。设 $F=A\oplus B$，试画出 F 对应 A、B 的波形。

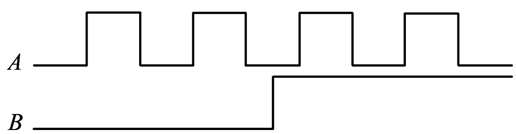


图 1-28 题 1.5 图

1.6 用真值表证明下列逻辑等式。

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

(2)$\overline{AB+\overline{A}C}=A\overline{B}+\overline{A}\,\overline{C}$。

1.7 用公式法证明下列逻辑等式。

(1)$\overline{\overline{A}B+A\overline{B}}=(A+\overline{B})(\overline{A}+B)$。

(2)$ABC+\overline{A}\,\overline{B}\,\overline{C}=\overline{A\overline{B}+B\overline{C}+C\overline{A}}$。

1.8 写出下列函数的对偶式 F'。

(1)$F=\overline{A}\,\overline{B}+CD$。

(2)$F=\overline{A+B+\overline{C}+\overline{D+E}}$。

(3)$F=AB+\overline{B\overline{C}+\overline{CD}}$。

(4)$F=A\overline{BC}+(\overline{A}+\overline{B}\,\overline{C})\cdot(A+C)$。

1.9 写出题 1.8 中函数的反函数 $\overline{F}$。

1.10 列出下列问题的真值表，并写出逻辑表达式。

(1) 设三变量 A、B、C，当变量组合值中出现奇数个 1 时，输出(F_1) 为 1，否则为 0。

(2) 设三变量 A、B、C，当输入端信号不一致时，输出(F_2) 为 1，否则为 0。

(3) 列出三变量多数表决器的真值表(输出用 F_3 表示)。

1.11　用代数法化简下列各式。

(1)$F=\overline{A}B\,\overline{C}+A\,\overline{C}+\overline{B}\,\overline{C}$。

(2)$F=ABC+\overline{A}+\overline{B}+\overline{C}$。

(3)$F=A(\overline{A}+B)+B(B+C)+B$。

(4)$F=(\overline{\overline{A}B}+C)ABD+BD$。

(5)$F=AC+\overline{B}\,\overline{C}+\overline{\overline{A}+B}$。

(6)$F=\overline{A}\,\overline{B}\,\overline{C}+\overline{A}B\,\overline{C}+\overline{A}BC+ABC$。

(7)$F=\overline{\overline{AB+\overline{A}\,\overline{B}}\;\overline{BC+\overline{B}\,\overline{C}}}$。

(8)$F=A\,\overline{C}\,\overline{D}+BC+\overline{B}D+A\,\overline{B}+\overline{A}C+\overline{B}\,\overline{C}$。

1.12　逻辑函数项 $A\,\overline{B}C$ 的逻辑相邻项有哪些?

1.13　画出下列各逻辑函数的卡诺图。

(1)$F_1(A,B,C)=\sum m(0,5,6,7)$。

(2)$F_2(A,B,C,D)=ABC+BCD+\overline{A}\,\overline{B}CD$。

1.14　写出图 1-29 中各卡诺图的逻辑函数式。

A \ BC	00	01	11	10
0	0	0	1	0
1	0	1	1	1

(a)

AB \ CD	00	01	11	10
00	1	0	0	1
01	1	1	1	1
11	1	0	0	1
10	1	0	0	1

(b)

图 1-29　题 1.14 图

1.15　用卡诺图化简下列逻辑函数。

(1)$F(A,B,C)=\overline{A}\,\overline{B}C+\overline{A}\,\overline{B}+C+ABC$。

(2)$F(A,B,C)=\overline{B}+ABC+\overline{A}\cdot\overline{C}+\overline{AB}$。

(3)$F(A,B,C)=\sum m(0,2,4,6)$。

(4)$F(A,B,C,D)=\sum m(0,2,5,6,7,8,10,13,14,15)$。

1.16　化简下列逻辑函数。

(1)$F(A,B,C)=\sum m(1,4)+\sum d(3,5,6,7)$。

(2)$F(A,B,C,D)=\sum m(2,3,5,7,8,9)+\sum d(10,11,12,13,14,15)$。