第4章　组合逻辑电路

数字系统中的逻辑电路按其是否具有记忆功能分为组合逻辑电路和时序逻辑电路两大类型。

组合逻辑电路是指电路在任何时刻产生的稳定输出值仅仅取决于该时刻各输入值的组合，而与过去的输入值无关。组合逻辑电路的一般结构如图 4.1 所示。

图 4.1　组合逻辑电路的一般结构

图中，$X_1, X_2, \cdots, X_n$ 是电路的 n 个输入信号，$F_1, F_2, \cdots, F_m$ 是电路的 m 个输出信号。输出信号是输入信号的函数，表示为

$$F_i = f_i(X_1, X_2, \cdots, X_n) \qquad i = 1, 2, \cdots, m$$

从电路结构看，组合逻辑电路具有两个特点：

① 由逻辑门电路组成，不包含任何记忆元件；

② 信号是单向传输的，不存在任何反馈回路。

组合逻辑电路不但能独立完成各种复杂的逻辑功能，而且是时序逻辑电路的组成部分，它在数字系统中的应用十分广泛。

本章主要讨论组合逻辑电路分析和设计的基本方法，在此基础上介绍组合逻辑电路设计中几个常见的实际问题及其处理方法，并对组合逻辑电路中的竞争险象问题作一般讨论。

4.1　组合逻辑电路分析

所谓逻辑电路分析，是指对一个给定的逻辑电路，找出其输出与输入之间的逻辑关系。通过分析，不仅可以了解给定逻辑电路的功能，同时还能评价其设计方案的优劣，以便吸取优秀的设计思想、改进和完善不合理方案以及更换逻辑电路的某些组件等。由此可见，逻辑电路分析是研究数字系统的一种基本技能。

4.1.1　分析方法概述

组合逻辑电路分析的一般步骤如下。

1. 根据逻辑电路图写出输出函数表达式

为了确保写出的逻辑表达式正确无误，一般是在认清电路中所有逻辑器件和相互连线的

基础上，从输入端开始往输出端逐级推导，直至得到所有与输入变量相关的输出函数表达式为止。

2. 化简输出函数表达式

根据给定逻辑电路写出的输出函数表达式不一定是最简表达式，为了简单、清晰地反映输入/输出之间的逻辑关系，应对逻辑表达式进行化简。此外，描述一个电路功能的逻辑表达式是否达到最简，是评价该电路经济技术指标是否良好的依据。

3. 列出输出函数真值表

根据输出函数最简表达式，列出输出函数真值表。真值表详尽地给出了输入/输出取值关系，它通过逻辑值直观地描述了电路的逻辑功能。

4. 功能评述

根据真值表和化简后的函数表达式，概括出对电路逻辑功能的文字描述，并对原电路的设计方案进行评价，必要时提出改进意见和改进方案。

以上分析步骤是就一般情况而言的，实际应用中可根据问题的复杂程度和具体要求对上述步骤进行适当取舍。下面举例说明组合逻辑电路的分析过程。

4.1.2 分析举例

例 4.1 分析图 4.2(a)所示组合逻辑电路。

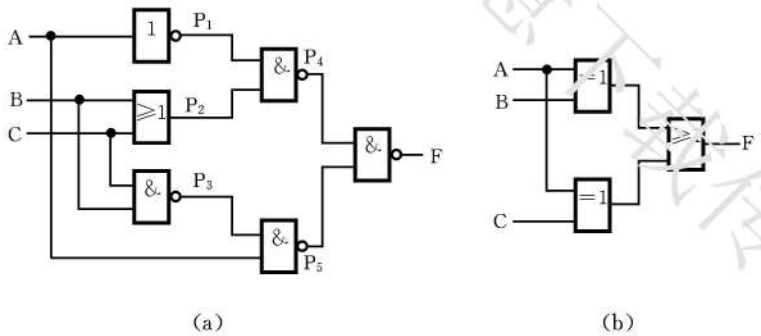


图 4.2 逻辑电路

解 (1) 根据逻辑电路图写出输出函数表达式

根据电路中各逻辑门的功能，从输入端开始逐级写出函数表达式如下：

$$P_1=\overline{A} \qquad P_2=B+C$$

$$P_3=\overline{BC} \qquad P_4=\overline{P_1 \cdot P_2}=\overline{\overline{A}(B+C)}$$

$$P_5=\overline{A \cdot P_3}=\overline{A\,\overline{BC}} \qquad F=\overline{P_4 \cdot P_5}=\overline{\overline{\overline{A}(B+C)} \cdot \overline{A\,\overline{BC}}}$$

(2) 化简输出函数表达式

用代数法对输出函数 F 的表达式化简如下：

$$F=\overline{\overline{\overline{A(B+C)}\cdot A\cdot\overline{BC}}}$$
$$=\overline{A}(B+C)+A\overline{BC}$$
$$=\overline{A}B+\overline{A}C+A\overline{B}+A\overline{C}$$
$$=A\oplus B+A\oplus C$$

(3) 根据化简后的函数表达式列出真值表

该函数的真值表如表 4.1 所示。

表 4.1 真值表

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

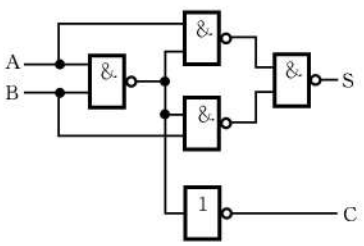
(4) 功能评述

由真值表可知，该电路仅当 A、B、C 取值同为 0 或同为 1 时输出 F 的值为 0，其他情况下输出 F 均为 1。换句话说，当输入取值一致时输出为 0，不一致时输出为 1，可见，该电路具有检查输入信号是否一致的逻辑功能，一旦输出为 1，则表明输入不一致。因此，通常称该电路为“不一致电路”。

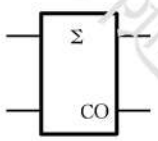
在某些可靠性要求非常高的系统中，往往是几套设备同时工作，一旦运行结果不一致，便由“不一致电路”发出报警信号，通知操作人员排除故障，以确保系统的可靠性。

其次，由分析可知，该电路的设计方案并不是最简的。根据化简后的输出函数表达式可采用异或门和或门画出实现给定功能的逻辑电路图，如图 4.2(b)所示。显然，它比原电路简单、清晰。

例 4.2 分析图 4.3(a)所示逻辑电路。



(a)



(b)

图 4.3 逻辑电路及其逻辑符号

表 4.2 真值表

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

解 根据给出的逻辑电路图可写出输出函数表达式

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

用代数化简法对输出函数化简：

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

根据简化后的表达式可列出真值表，如表 4.2 所示。

由真值表可以看出，若将 A、B 分别作为一位二进制数，则 S 是 A、B 相加的“和”，而 C 是相加产生的“进位”。该电路通常称作“半加器”，它能实现两个一位二进制数加法运算。半加器已被加工成小规模集成电路器件，其逻辑符号如图 4.3(b)所示。

例 4.3 分析图 4.4 所示组合逻辑电路。已知电路输入 ABCD 为 8421 码，说明该电路功能。

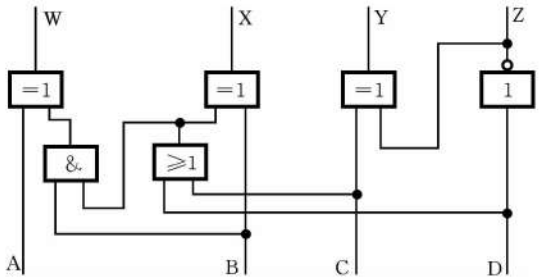


图 4.4 逻辑电路

解 根据图 4.4 所示逻辑电路，可写出输出函数表达式如下：

$$W=A\oplus B\cdot(C+D)$$

$$X=B\oplus(C+D)$$

$$Y=C\oplus\overline{D}$$

$$Z=\overline{D}$$

由于电路输入 ABCD 为 8421 码，所以 ABCD 只允许取值 0000～1001。根据所得输出函数表达式可列出真值表，如表 4.3 所示。

表 4.3 真值表

ABCD	WXYZ	ABCD	WXYZ
0000	0011	0101	1000
0001	0100	0110	1001
0010	0101	0111	1010
0011	0110	1000	1011
0100	0111	1001	1100

由真值表可知，电路输出 WXYZ 是一位十进制数的余 3 码，即该电路是一个将 8421 码转换成余 3 码的代码转换电路。

以上例子说明了组合逻辑电路分析的一般方法。从讨论过程可以看出，通过对电路进行分析，不仅可以找出电路输入、输出之间的关系，确定电路的逻辑功能，同时还能对某些设计不合理的电路进行改进和完善。

4.2 组合逻辑电路设计

根据问题要求完成的逻辑功能，求出在特定条件下实现该功能的逻辑电路，这一过程称为逻辑设计，又叫做逻辑综合。显然，逻辑设计是逻辑分析的逆过程。

4.2.1 设计方法概述

由于实际应用中提出的各种设计要求一般是用文字形式描述的，所以逻辑设计的首要任务是将设计要求转化为逻辑问题，即将文字描述的设计要求抽象为一种逻辑关系。就组合逻

辑电路而言，就是抽象出描述问题的逻辑表达式。

组合逻辑电路的设计过程大致如下。

1. 建立给定问题的逻辑描述

对设计要求进行逻辑描述是完成组合电路设计的第一步，也是最重要的一步，它是确保设计方案正确的前提。这一步的关键是：正确理解设计要求，弄清楚与给定问题相关的变量及函数，即电路的输入和输出；建立函数与变量之间的逻辑关系，最终得到描述给定问题的逻辑表达式。求逻辑表达式的常用方法有真值表法和分析法两种，后面将结合实例进行介绍。

2. 求出逻辑函数的最简表达式

基于小规模集成电路的组合逻辑电路设计是以最简方案为目标的，即要求逻辑电路中包含的逻辑门最少且连线最少。因此，要对逻辑表达式进行化简，求出描述设计问题的最简表达式。

3. 选择逻辑门类型并进行逻辑函数变换

根据简化后的逻辑表达式及问题的具体要求，选择合适的逻辑门，并将逻辑表达式变换成与所选逻辑门对应的形式。

4. 画出逻辑电路图

根据变换后的表达式画出逻辑电路图。

以上步骤是就一般情况而言的，根据实际问题的难易程度和设计者的熟练程度，有时可跳过其中的某些步骤。在设计过程中可视具体情况灵活掌握。

4.2.2　设计举例

例 4.4　设计一个 3 变量“多数表决电路”。

解　(1) 建立给定问题的逻辑描述

“多数表决电路”的逻辑功能就是按照少数服从多数的原则执行表决，确定某项决议是否通过。假设用 A、B、C 分别代表参加表决的 3 个逻辑变量，用函数 F 表示表决结果。并约定，逻辑变量取值为 0 表示反对，逻辑变量取值为 1 表示赞成；逻辑函数 F 取值为 0 表示决议被否决，逻辑函数 F 取值为 1 表示决议通过。那么，按照少数服从多数的原则可知，函数和变量的关系是：当 3 个变量 A、B、C 中有 2 个或 2 个以上取值为 1 时，函数 F 的值为 1，其他情况下函数 F 的值为 0。因此，可列出该逻辑函数的真值表，如表 4.4 所示。由真值表可写出函数 F 的最小项表达式为

$$F(A,B,C)=\sum m(3,5,6,7)$$

(2) 求出逻辑函数的最简表达式

作出函数 F 的卡诺图如图 4.5(a)所示，用卡诺图化简后得到函数的最简与-或表达式为

$$F(A,B,C)=AB+AC+BC$$

表 4.4 真值表

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

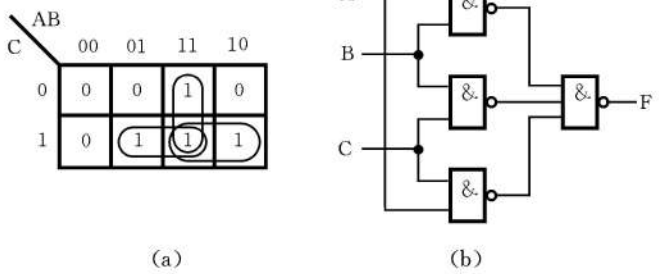


图 4.5 卡诺图及逻辑电路

(3) 选择逻辑门类型并进行逻辑函数变换

假定采用与非门组成实现给定功能的电路，则应将上述表达式变换成“与非-与非”表达式

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

(4) 画出逻辑电路图

由函数的与非-与非表达式，可画出实现给定功能的逻辑电路，如图 4.5(b)所示。

本例在建立给定问题的逻辑描述时，采用的是真值表法，即通过建立真值表得到相应的逻辑表达式。真值表的优点是规整、清晰，缺点是不方便，尤其当变量较多时十分麻烦。因此，针对具体问题通常采用的另一种方法是分析法，即通过对设计要求的分析、理解，直接写出逻辑表达式。

例 4.5 设计一个比较两个 3 位二进制数是否相等的数值比较器。

解 (1) 建立给定问题的逻辑描述

设两个 3 位二进制数分别为 $A=a_3a_2a_1$，$B=b_3b_2b_1$，比较结果用 F 表示。当 A=B 时，F 为 1；否则 F 为 0。显然，这是一个有 6 个输入变量、1 个输出函数的组合逻辑电路。

由于二进制数 A 和 B 相等，必须同时满足 $a_3=b_3$、$a_2=b_2$、$a_1=b_1$，而二进制中 $a_i=b_i$ 只有 a_i 和 b_i 同时为 0 或者同时为 1 两种可能，因此，该问题可用逻辑表达式描述如下：

$$F=(\bar{a}_3\bar{b}_3+a_3b_3)(\bar{a}_2\bar{b}_2+a_2b_2)(\bar{a}_1\bar{b}_1+a_1b_1)$$

(2) 求出逻辑函数最简表达式

将上述逻辑函数表达式展开成与-或表达式可以发现，该函数不能简化，表达式中包含 8 个 6 变量与项。

(3) 选择逻辑门类型并进行逻辑函数变换

假定采用异或门和或非门实现给定功能，可将逻辑表达式作如下变换：

$$\begin{aligned}F&=(\bar{a}_3\bar{b}_3+a_3b_3)(\bar{a}_2\bar{b}_2+a_2b_2)(\bar{a}_1\bar{b}_1+a_1b_1)\\&=\overline{(a_3\oplus b_3)}\,\overline{(a_2\oplus b_2)}\,\overline{(a_1\oplus b_1)}\\&=\overline{(a_3\oplus b_3)+(a_2\oplus b_2)+(a_1\oplus b_1)}\end{aligned}$$

(4) 画出逻辑电路图

根据变换后的表达式可画出逻辑电路，如图 4.6 所示。

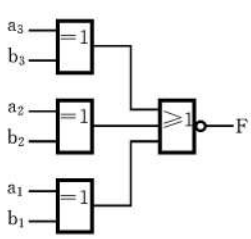


图 4.6 逻辑电路

例 4.6 设计一个乘法器，用于产生两个 2 位二进制数相乘的积。

解 因为一个 2 位二进制数表示的十进制数最大为 3，两个 2 位二进制数相乘的积最大为 9，所以相乘的积可用一个 4 位二进制数表示。由此可见，该电路有 4 个输入变量，4 个输出函数。设两个二进制

数分别为 A_1A_0 和 B_1B_0，相乘的积为 $M_3M_2M_1M_0$，下面用两种不同的方法抽象出电路的输出函数表达式，并选用合适的逻辑门完成相应电路设计。

方法Ⅰ　采用真值表法

(1) 借助真值表写出输出函数表达式

根据设计要求和对问题的假设可列出真值表，如表 4.5 所示。

表 4.5　真值表

A_1	A_0	B_1	B_0	M_3	M_2	M_1	M_0	A_1	A_0	B_1	B_0	M_3	M_2	M_1	M_0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	1	0	0	1	0	0	1	0
0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0
0	0	1	1	0	0	0	0	1	0	1	1	0	1	1	0
0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0
0	1	0	1	0	0	0	1	1	1	0	1	0	0	1	1
0	1	1	0	0	0	1	0	1	1	1	0	0	1	1	0
0	1	1	1	0	0	1	1	1	1	1	1	1	0	0	1

根据真值表可写出输出函数表达式如下：

$$M_3(A_1,A_0,B_1,B_0)=m_{15}$$

$$M_2(A_1,A_0,B_1,B_0)=m_{10}+m_{11}+m_{14}$$

$$M_1(A_1,A_0,B_1,B_0)=m_6+m_7+m_9+m_{11}+m_{13}+m_{14}$$

$$M_0(A_1,A_0,B_1,B_0)=m_5+m_7+m_{13}+m_{15}$$

(2) 求出逻辑函数最简表达式

利用卡诺图可求出输出函数的最简与-或表达式为

$$M_3=A_1A_0B_1B_0$$

$$M_2=A_1\overline{A}_0B_1+A_1B_1\overline{B}_0$$

$$M_1=A_1\overline{B}_1B_0+A_1\overline{A}_0B_0+\overline{A}_1A_0B_1+A_0B_1\overline{B}_0$$

$$M_0=A_0B_0$$

(3) 选择逻辑门类型并画出逻辑电路图

假定采用与门和或门实现给定功能，可画出实现给定功能的逻辑电路，如图 4.7 所示。

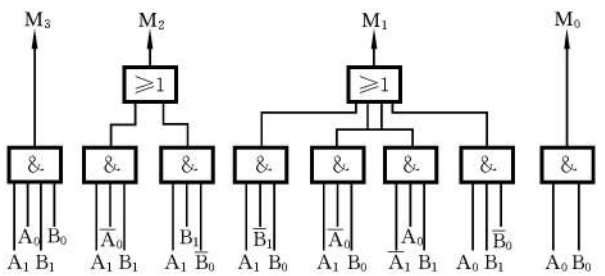


图 4.7　方法Ⅰ的逻辑电路

方法Ⅱ　采用分析法

(1) 通过分析写出输出函数表达式

按照二进制乘法运算法则和对问题的假设，可列出乘法计算过程如下：

			A_1	A_0
×(乘)			B_1	B_0
	C_2	C_1	$A_1\times B_0$	$A_0\times B_0$
+(加)		$A_1\times B_1$	$A_0\times B_1$	
	M_3	M_2	M_1	M_0

上述计算式中，积项 $A_i\times B_j$ $(i,j=0,1)$为两个 1 位二进制数相乘，可用两输入与门实现，即 $A_i\times B_j$的算术值等于 $A_i\cdot B_j$的逻辑值。因为 C_1为 A_1B_0和 A_0B_1相加产生的进位，它等于 A_1B_0和 A_0B_1相与，即 $C_1=A_1B_0A_0B_1$，C_2为 C_1和A_1B_1相加产生的进位，它等于 C_1和 A_1B_1相与，即 $C_2=C_1\cdot A_1B_1=A_1B_0A_0B_1\cdot A_1B_1=A_1B_0A_0B_1$；又因为两个 1 位二进制数相加的和等于对其进行异或运算的结果，所以可由分析得到该电路的输出函数表达式如下：

$$M_0=A_0B_0$$
$$M_1=A_1B_0\oplus A_0B_1$$
$$M_2=C_1\oplus A_1B_1=A_1B_0A_0B_1\oplus A_1B_1$$
$$M_3=C_2=A_1B_0A_0B_1$$

假定采用异或门和与门实现该乘法器的逻辑功能，可画出该乘法器的逻辑电路，如图 4.8 所示。

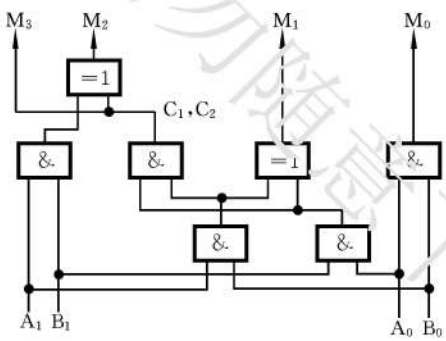


图 4.8　方法Ⅱ的逻辑电路

比较两种方法所得结果可以看出，图 4.7 所示电路中共用了 10 个逻辑门，而图 4.8 所示电路中只用了 7 个逻辑门，显然，后者比前者更简单。为了证明两者逻辑功能的一致性，读者可对方法Ⅱ中的表达式进行展开变换，即可得到方法Ⅰ中的最简与-或表达式。

4.2.3　设计中几个实际问题的处理

上面对组合逻辑电路设计的一般方法进行了讨论，并通过简单例子介绍了设计的全过程。然而，实际提出的设计要求是形形色色的，往往除了问题复杂之外，还存在某些特殊情况需要考虑。为了能在各种特殊情况下设计出最简的逻辑电路，必须针对具体问题作出具体的分析和处理。下面就几个常见问题进行讨论。

1. 包含无关条件的组合逻辑电路设计

前面讨论的设计问题中，对于电路输入变量的任何一种取值组合，都有确定的输出函数值

与之对应，换句话说，对于一个具有 n 个输入变量的组合逻辑电路，输出函数与 2^n 种输入取值组合均相关，假如有 m 种取值组合使函数的值为 1，则有 2^n-m 种取值组合使函数的值为 0，该输出函数可以用 m 个最小项之和表示。

但在某些实际问题中，常常由于输入变量之间存在的相互制约或问题的某种特殊限定等，使得输入变量的某些取值组合根本不会出现，或者虽然可能出现，但对在这些输入取值组合下函数的值是为 1 还是为 0 并不关心。通常把这类问题称为包含无关条件的逻辑问题；与这些输入取值组合对应的最小项称为**无关最小项**，简称为**无关项**或者**任意项**；描述这类问题的逻辑函数称为**包含无关条件的逻辑函数**。

当采用最小项之和表达式描述一个包含无关条件的逻辑问题时，函数表达式中是否包含无关项以及对无关项是令其值为 1 还是为 0，并不影响函数的实际逻辑功能。因此，在化简这类逻辑函数时，利用这种随意性往往可以使逻辑函数得到更好的简化，从而使设计的电路达到更简。

例 4.7 设计一个组合逻辑电路，用于判别以余 3 码表示的 1 位十进制数是否为合数。

解 由题意可知，该电路输入为 1 位十进制数的余 3 码，输出为对其值进行判断的结果。设输入变量为 A、B、C、D，输出函数为 F，当 ABCD 表示的十进制数为合数(4,6,8,9)时，输出 F 为 1，否则 F 为 0。因为按照余 3 码的编码规则，ABCD 的取值组合不允许为 0000、0001、0010、1101、1110、1111，故该问题为包含无关条件的逻辑问题，与上述 6 种取值组合对应的最小项为无关项，即在这些取值组合下输出函数 F 的值可以随意指定为 1 或者为 0，通常记为“d”。据此可建立描述该问题的真值表，如表 4.6 所示。

表 4.6 真值表

A	B	C	D	F	A	B	C	D	F
0	0	0	0	d	1	0	0	0	0
0	0	0	1	d	1	0	0	1	1
0	0	1	0	d	1	0	1	0	0
0	0	1	1	0	1	0	1	1	1
0	1	0	0	0	1	1	0	0	1
0	1	0	1	0	1	1	0	1	d
0	1	1	0	0	1	1	1	0	d
0	1	1	1	1	1	1	1	1	d

根据真值表可写出 F 的逻辑表达式为

$$F(A,B,C,D)=\sum m(7,9,11,12)+\sum d(0,1,2,13,14,15)$$

用卡诺图化简函数 F 时，若不考虑无关项，如图 4.9(a)所示合并卡诺图上的 1 方格，则可得到化简后的逻辑表达式为

$$F(A,B,C,D)=A\overline{B}D+AB\overline{C}\,\overline{D}+\overline{A}BCD$$

如果化简时对无关条件加以利用，如图 4.9(b)所示，根据合并的需要将卡诺图中的无关项 d(13,14,15)当成 1 处理，而把 d(0,1,2)当成 0 处理，则可得到化简后的逻辑表达式为

$$F(A,B,C,D)=AB+AD+BCD$$

显然，后一个表达式比前一个表达式更简单。假定采用与非门构成实现给定逻辑功能的电路，可将 F 的最简表达式变换成与非-与非表达式：

$$F(A,B,C,D)=\overline{\overline{AB}+\overline{AD}+\overline{BCD}}=\overline{\overline{AB}\cdot\overline{AD}\cdot\overline{BCD}}$$

相应的逻辑电路如图 4.10 所示。

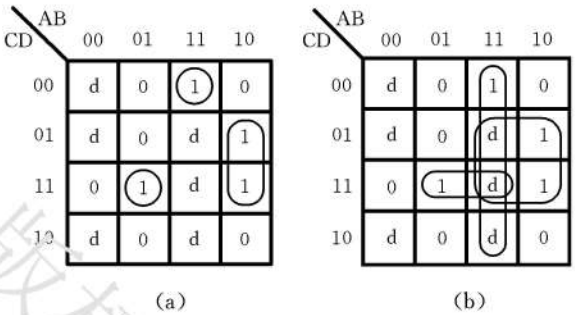


图 4.9 卡诺图

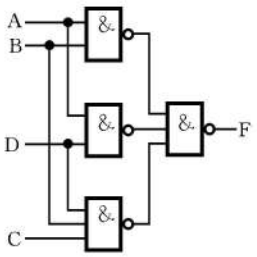


图 4.10 逻辑电路

由此可见，设计包含无关条件的组合逻辑电路时，恰当地利用无关项进行函数化简，通常可使设计出来的电路更简单。

2. 多输出函数的组合逻辑电路设计

实际问题中，大量存在着由同一组输入变量产生多个输出函数的问题，实现这类问题的组合逻辑电路称为多输出函数的组合逻辑电路。

设计多输出函数的组合逻辑电路时，如果只是孤立地求出各输出函数的最简表达式，然后画出相应逻辑电路图并将其拼在一起，通常不能保证逻辑电路整体最简。因为各输出函数之间往往存在相互联系，具有某些共同的部分，因此，应该将它们当作一个整体考虑，而不应该将其截然分开。使这类电路达到最简的关键在于函数化简时找出各输出函数的公用项，以便在逻辑电路中实现对逻辑门的共享，从而使电路整体结构最简。

例 4.8 假定某组合逻辑电路结构框图如图 4.11 所示，试用最少的与非门实现该电路功能。

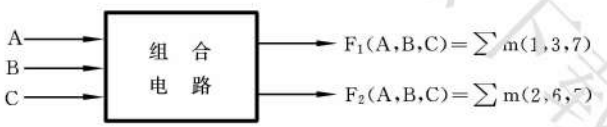


图 4.11 组合逻辑电路结构框图

解 为了求出用与非门构成的最简电路，首先应求出输出函数的最简与-或表达式，并变换成与非-与非表达式，然后画出相应的逻辑电路图。首先，若不考虑两个输出函数之间的共享问题，则可直接作出 F_1、F_2 的卡诺图，如图 4.12(a)、(b)所示。

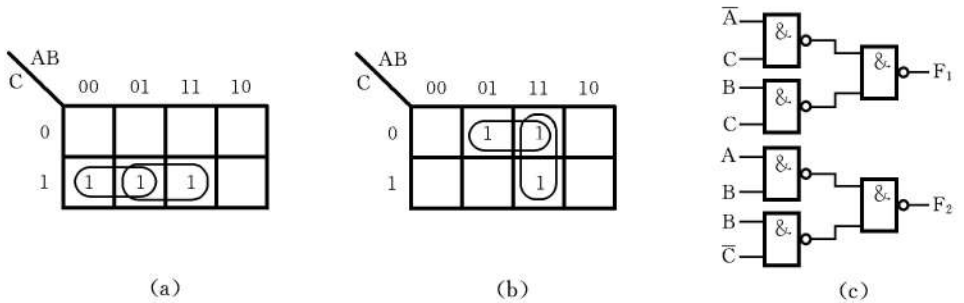


图 4.12 卡诺图及逻辑电路之一

化简后的输出函数表达式为

$$F_1=\overline{A}C+BC=\overline{\overline{\overline{A}C+BC}}=\overline{\overline{\overline{A}C}\cdot\overline{BC}}$$

$$F_2=AB+B\overline{C}=\overline{\overline{AB+B\overline{C}}}=\overline{\overline{AB}\cdot\overline{B\overline{C}}}$$

根据化简后的输出函数表达式可画出相应的逻辑电路，如图 4.12(c)所示。

如果考虑两个输出函数应充分“共享”的问题，则在对 F_1、F_2进行化简时，可按图 4.13(a)、(b)所示卡诺图进行处理，化简后的输出逻辑表达式为

$$F_1=\overline{A}C+ABC=\overline{\overline{\overline{A}C+ABC}}=\overline{\overline{\overline{A}C}\cdot\overline{ABC}}$$

$$F_2=ABC+B\overline{C}=\overline{\overline{ABC+B\overline{C}}}=\overline{\overline{ABC}\cdot\overline{B\overline{C}}}$$

根据修改后的表达式，可画出相应的逻辑电路，如图 4.13(c)所示。显然，通过找出两个函数的公共项，使其“共享”同一个逻辑门，从而使电路从整体上得到了进一步简化。

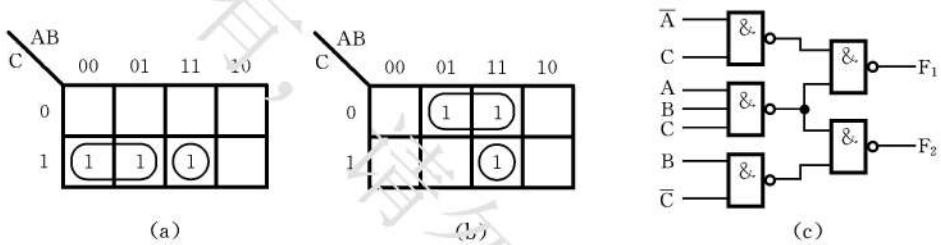


图 4.13　卡诺图及逻辑电路之二

例 4.9　设计一个全加器。

解　全加器是一个能对两个一位二进制数及来自低位的“进位”进行相加，产生本位“和”及向高位“进位”的逻辑电路。由此可知，该电路有 3 个输入变量，2 个输出函数。设被加数、加数及来自低位的“进位”分别用 A_i、B_i及 C_{i-1}表示，相加产生的“和”及“进位”用 S_i和 C_i表示。根据二进制加法运算法则可列出全加器的真值表，如表 4.7 所示。

由真值表可写出输出函数表达式：

$$S_i(A_i,B_i,C_{i-1})=\sum m(1,2,4,7)$$

$$C_i(A_i,B_i,C_{i-1})=\sum m(3,5,6,7)$$

假定采用卡诺图化简上述函数，则可作出相应卡诺图，如图 4.14 所示。

表 4.7　全加器真值表

A_i	B_i	C_{i-1}	S_i	C_i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

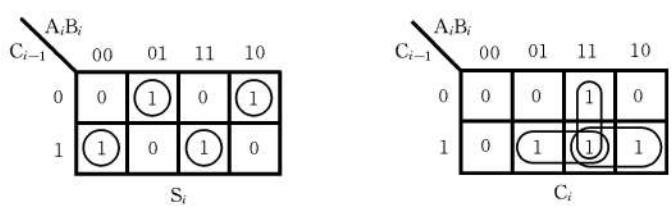


图 4.14　函数 S_i和 C_i的卡诺图

经化简后的输出函数表达式为

$$S_i=\overline{A}_i\overline{B}_iC_{i-1}+\overline{A}_iB_i\overline{C}_{i-1}+A_i\overline{B}_i\overline{C}_{i-1}+A_iB_iC_{i-1}$$

$$C_i=A_iB_i+A_iC_{i-1}+B_iC_{i-1}$$

其中，S_i的标准与-或式即最简与-或式。当采用异或门和与非门组成实现给定功能的电路时，可对表达式作如下变换：

$$\begin{aligned}S_i&=\overline{A}_i\overline{B}_iC_{i-1}+\overline{A}_iB_i\overline{C}_{i-1}+A_i\overline{B}_i\overline{C}_{i-1}+A_iB_iC_{i-1}\\&=\overline{A}_i(\overline{B}_iC_{i-1}+B_i\overline{C}_{i-1})+A_i(\overline{B}_i\overline{C}_{i-1}+B_iC_{i-1})\\&=\overline{A}_i(B_i\oplus C_{i-1})+A_i\overline{(B_i\oplus C_{i-1})}\\&=A_i\oplus B_i\oplus C_{i-1}\end{aligned}$$

$$C_i=A_iB_i+A_iC_{i-1}+B_iC_{i-1}=\overline{\overline{A_iB_i}\cdot\overline{A_iC_{i-1}}\cdot\overline{B_iC_{i-1}}}$$

相应的逻辑电路如图 4.15(a)所示。该电路就单个函数而言均已达到最简，但从整体考虑则并非最简。当按多输出函数组合电路进行设计时，可对函数C_i作如下变换：

$$\begin{aligned}C_i&=\overline{A}_iB_iC_{i-1}+A_i\overline{B}_iC_{i-1}+A_iB_i\overline{C}_{i-1}+A_iB_iC_{i-1}\\&=(\overline{A}_iB_i+A_i\overline{B}_i)C_{i-1}+A_iB_i(\overline{C}_{i-1}+C_{i-1})\\&=(A_i\oplus B_i)C_{i-1}+A_iB_i\\&=\overline{\overline{(A_i\oplus B_i)C_{i-1}}\cdot\overline{A_iB_i}}\end{aligned}$$

经变换后，S_i和C_i的逻辑表达式中有公用项$A_i\oplus B_i$，因此，在组成电路时，可令其共享同一异或门，从而使整体得到进一步简化，其逻辑电路如图 4.15(b)所示。

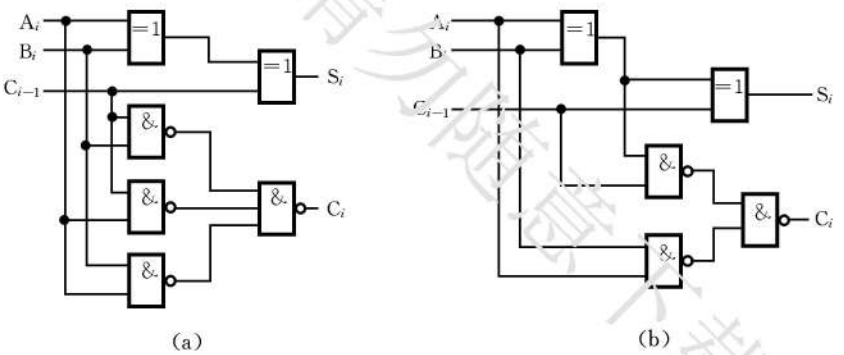


图 4.15　全加器逻辑电路

图 4.16 给出了用集成门电路实现全加器功能时的芯片引脚连接图。

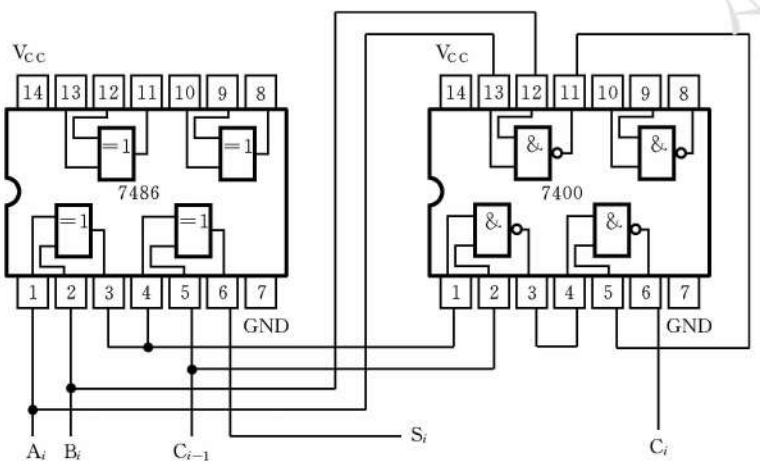


图 4.16　全加器的芯片引脚连接图

实现图 4.15(a)所示电路的功能需要 3 种芯片，可用一块异或门芯片 7486、一块 2 输入与非门芯片 7400 和一块 3 输入与非门芯片 7410。而实现图 4.15(b)所示电路的功能则只需要两种芯片，用一块异或门芯片 7486 和一块与非门芯片 7400 即可，图 4.16 给出了一个实现该电路功能的芯片引脚连接图。值得指出的是，由于器件提供的逻辑门数目多于电路所需逻辑门数目，所以在功能实现时对芯片内部逻辑门的选择方案不是唯一的，不同的选择方案对应不同的芯片引脚连接图。

3. 无反变量提供的组合逻辑电路设计

在对某些实际问题进行设计的过程中，常常为了减少各部件之间的连线，只给所设计的逻辑电路提供原变量，不提供反变量。设计这类电路时，直截了当的办法是当需要某个反变量时，就用一个非门将相应的原变量转换成反变量，但这样处理往往是不经济的。因此，通常采用适当的方法进行处理，以便在无反变量提供的前提下，使逻辑电路尽可能简单。

例 4.10 输入变量中无反变量时，用与非门实现逻辑函数

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

的功能。

解 因为给定函数已经是最简与-或表达式，故可直接画出逻辑电路，如图 4.17(a)所示，但这个电路并不是最简的。如果对函数 F 的表达式作如下整理：

$$\begin{aligned}F(A,B,C,D)&=\overline{A}B+B\overline{C}+A\overline{B}C+AC\overline{D}\\&=B(\overline{A}+\overline{C})+AC(\overline{B}+\overline{D})\\&=B\overline{AC}+AC\overline{B\overline{D}}\end{aligned}$$

根据整理后的表达式可画出对应的逻辑电路，如图 4.17(b)所示。显然，图4.17(b)比图 4.17(a)更合理。

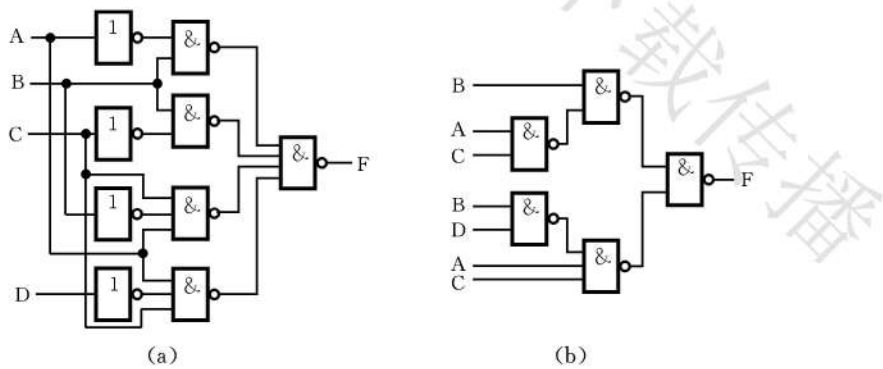


图 4.17 逻辑电路

例 4.11 在输入无反变量提供时，用最少的逻辑门实现逻辑函数 $F(A,B,C)=\overline{A}B+B\overline{C}+A\overline{B}C$ 的功能。

解 给定逻辑函数表达式已为最简与或表达式，直接使用与非门和反相器实现该函数功能的逻辑电路如图 4.18(a)所示，电路中共用了 3 个反相器和 4 个与非门。

图 4.18(a)所示电路是不是最简的呢？不是！为了得到更简单的电路，可对给定逻辑函数表达式作如下变换：

$$\begin{aligned}F(A,B,C)&=\overline{A}B+B\overline{C}+A\overline{B}C\\&=(\overline{A}+\overline{C})B+A\overline{B}C\\&=\overline{AC}B+AC\overline{B}\\&=AC\oplus B\end{aligned}$$

根据变换后的逻辑表达式可画出实现原函数功能的更简逻辑电路，如图 4.18(b)所示。

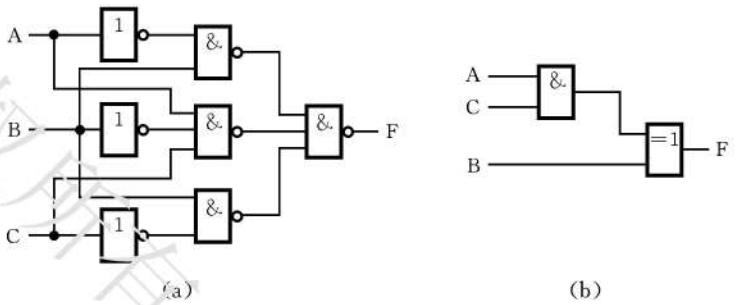


图 4.18　逻辑电路

由上述两个例子可以看出，在输入无反变量提供的场合，即使逻辑函数表达式已为最简，但直接实现时所得到的电路也不一定最简，通常对逻辑表达式作适当变换，可以减少电路中非门的数量，更好地简化电路结构。然而，当描述某种设计要求的表达式被确定下来后，并不是都可以作类似变换的。在实际问题的设计中，电路的复杂度往往和设计过程中某些步骤的处理方案直接相关。

例 4.12　设计一个组合逻辑电路，用来判断献血者与受血者血型是否相容。血型相容规则如表 4.8 所示，表中用“√”表示两者血型相容。

表 4.8　血型相容规则表

受血 / 献血	A	B	AB	O
A	√		√	
B		√	√	
AB			√	
O	√	√	√	√

表 4.9　血型编码(1)

血　型	献 W	献 X	受 Y	受 Z
A	0	0	0	0
B	0	1	0	1
AB	1	0	1	0
O	1	1	1	1

解　根据题意，电路输入变量为献血者血型和受血者血型。血型共 4 种，可用两个变量的 4 种编码进行区分。设变量 WX 表示献血者血型，YZ 表示受血者血型，血型编码如表 4.9 所示。电路输出用 F 表示，当输血者与受血者血型相容时，F 为 1，否则 F 为 0。可根据血型相容规则直接写出输出函数 F 的表达式：

$$\begin{aligned}F&=\overline{W}\,\overline{X}(\overline{Y}\,\overline{Z}+Y\overline{Z})+\overline{W}X(\overline{Y}Z+Y\overline{Z})+W\overline{X}Y\overline{Z}+WX(\overline{Y}\,\overline{Z}+\overline{Y}Z+Y\overline{Z}+YZ)\\&=\overline{W}\,\overline{X}\,\overline{Z}+\overline{W}X\overline{Y}Z+\overline{W}XY\overline{Z}+W\overline{X}Y\overline{Z}+WX\\&=\overline{W}\,\overline{Z}(\overline{X}+XY)+(\overline{W}\,\overline{Y}Z+W)X+W(\overline{X}Y\overline{Z}+X)\\&=\overline{W}\,\overline{Z}(\overline{X}+Y)+(W+\overline{Y}Z)X+W(X+Y\overline{Z})\\&=\overline{W}\,\overline{X}\,\overline{Z}+\overline{W}Y\overline{Z}+WX+X\overline{Y}Z+WX+WY\overline{Z}\\&=\overline{W}\,\overline{X}\,\overline{Z}+Y\overline{Z}+WX+X\overline{Y}Z\end{aligned}$$

由化简后的表达式可知，在无反变量提供的情况下，若通过直接加非门产生反变量，则组

成实现给定功能的电路时需 9 个逻辑门，其中 4 个非门用来产生 4 个输入变量的反变量。用与非门组成实现给定功能的逻辑电路如图 4.19(a)所示。

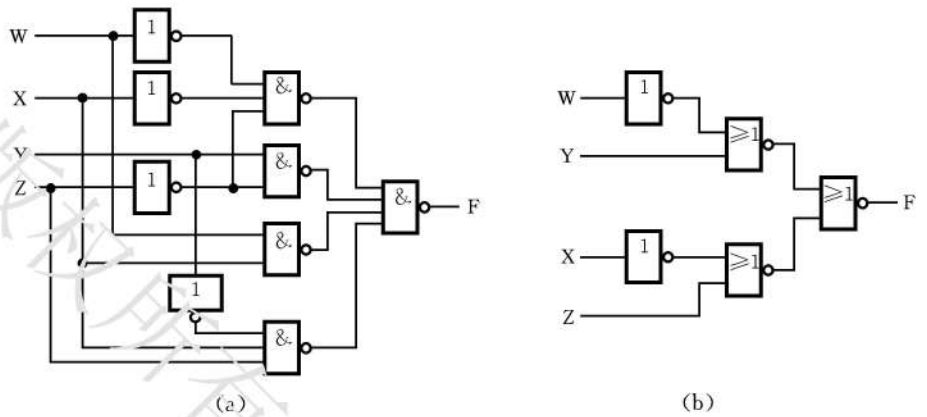


图 4.19　逻辑电路

分析上述设计过程不难发现，对该问题的逻辑描述与血型编码是直接相关的。为了减少逻辑表达式中反变量个数，进一步简化电路结构，可调整血型编码，如表4.10所示。

表 4.10　血型编码(2)

血　型	献		受	
	W	X	Y	Z
O	0	0	0	0
A	0	1	0	1
B	1	0	1	0
AB	1	1	1	1

根据新的编码方案和血型相容规则，可写出输出函数 F 的表达式：

$$\begin{aligned}F &= \overline{W}\,\overline{X}(\overline{Y}\,\overline{Z}+\overline{Y}Z+Y\,\overline{Z}+YZ)+\overline{W}X(\overline{Y}Z+YZ)+W\overline{X}(Y\,\overline{Z}+YZ)+WXYZ\\ &=\overline{W}\,\overline{X}+\overline{W}XZ+W\overline{X}Y+WXYZ\\ &=\overline{W}(\overline{X}+XZ)+WY(\overline{X}+XZ)\\ &=\overline{W}(\overline{X}+Z)+WY(\overline{X}+Z)\\ &=(\overline{W}+WY)(\overline{X}+Z)\\ &=(\overline{W}+Y)(\overline{X}+Z)\end{aligned}$$

该函数表达式中仅含两个反变量，假定采用或非门实现给定功能，则可将函数表达式变换成“或非-或非”表达式：

$$F=\overline{\overline{(\overline{W}+Y)(\overline{X}+Z)}}=\overline{\overline{\overline{W}+Y}+\overline{\overline{X}+Z}}$$

逻辑电路如图 4.19(b)所示，电路中只使用了 5 个逻辑门。

关于在无反变量提供时如何使组合电路达到最简的问题，至今尚无一种系统而有效的方法，只能由设计者根据具体问题进行灵活处理。

4.3　组合逻辑电路的险象

前面讨论组合逻辑电路时，只研究了输入和输出稳定状态之间的关系，而没有考虑信号传输中的时延问题，实际上，信号经过任何逻辑门和导线都会产生时间延迟，这就使得当电路所有输入达到稳定状态时，输出并不是立即达到稳定状态。

一般来说，延迟时间对数字系统是一个有害的因素。例如，使得系统操作速度下降，引起电路中信号的波形参数变坏，更严重的是在电路中产生竞争险象的问题。本节将专门针对后一个问题进行讨论。

4.3.1 险象的产生

在实际逻辑电路中，信号经过同一电路中的不同路径所产生的时延一般来说是各不相同的。各路径上延迟时间的长短与信号经过的门的级数有关，与具体逻辑门的时延大小有关，还与导线的长短有关。因此，输入信号经过不同路径到达输出端的时间也就有先有后，这就好像一场赛跑，各运动员到达终点的时间有先有后一样，这种现象称为**竞争现象**。在逻辑电路中，竞争现象是随时随地都可能出现的，我们可以更广义地把竞争现象理解为多个信号到达某一点有时差所引起的现象。

电路中竞争现象的存在，使得输入信号的变化可能引起输出信号出现非预期的错误输出，这一现象称为**险象**。并不是所有的竞争都会产生错误输出。通常，把不产生错误输出的竞争称为**非临界竞争**，而导致错误输出的竞争称为**临界竞争**。

组合电路中的险象是一种瞬态现象，它表现为在输出端产生不应有的尖脉冲，暂时地破坏正常逻辑关系。一旦瞬态过程结束，即可恢复正常逻辑关系。下面举例说明这一现象。

例如，图 4.20(a)所示的是由与非门构成的组合电路，该电路有 3 个输入，1 个输出，输出函数表达式为

$$F=\overline{\overline{AB}\cdot\overline{\overline{AC}}}=AB+\overline{A}C$$

假设输入变量 B=C=1，将 B、C 的值代入上述函数表达式，得

$$F=A+\overline{A}$$

由互补律可知，无论 A 怎样变化，该函数表达式 F 的值应恒为 1，即当 B=C=1 时，不论 A 是 0 还是 1，输出 F 的值都应保持 1 不变。然而，这是在一种理想状态下得出的结论。现在要讨论的是当考虑电路存在时间延迟时，该电路的实际输入/输出关系。更具体地说，当 B=C=1 时，A 的变化会使电路引起怎样的输出响应。假定每个门的延迟时间为 t_{pd}，则可用图 4.20(b)所示的时间图来说明。

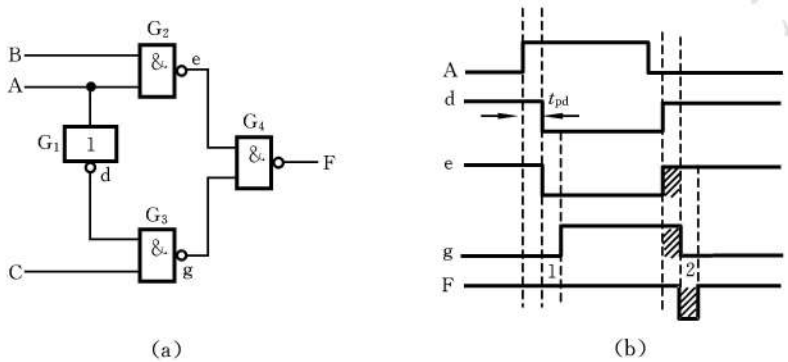


图 4.20　具有险象的逻辑电路及时间图(Ⅰ)

当 A 由低电平变到高电平以后，经过一个 t_{pd} 时间，反相器 G_1 的输出 d 由高电平变为低电平，同时与非门 G_2 的输出 e 也由高电平变为低电平，但要再经过一个 t_{pd}，与非门 G_3 的输出 g

才能由低电平变为高电平。最后到达门 G_4 输入端的是由同一个 A 信号经不同路径传输而得到的两个信号 e 和 g，e 和 g 的变化方向相反，并具有一个 t_{pd} 的时差。显然，这里（见图 4.20(b)中 1 处）存在一次竞争。但因门 G_4 是一个与非门，e 和 g 竞争的结果，使门 G_4 的输出保持为高电平，没有出现尖脉冲，即这里没有产生险象，所以这次竞争是一次非临界竞争。但当 A 由高电平变为低电平时，情况就不一样了。e 和 g 同样要在门 G_4 上发生竞争，且 e 和 g 在一个 t_{pd} 的时间内同时为高电平，根据门 G_4 的与非逻辑特性，输出 F 必然会出现一个负跳变的尖脉冲（见图 4.20(b)中 2 处）。也就是说，这次竞争的结果产生了险象，是一次临界竞争。

如果将图 4.20(a)中所示的与非门改为或非门，修改结果如图 4.21(a)所示，则根据修改后的电路可写出输出函数表达式为

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

假设输入变量 B=C=0，将 B、C 的值代入上述函数表达式，可得

$$F=A\cdot\overline{A}$$

由互补律可知，函数 $F=A\cdot\overline{A}$ 的值应恒为 0，即 B=C=0 时，无论 A 怎样变化，输出 F 的值都应保持 0 不变。然而，当考虑电路中存在的时间延迟时，可分析出电路输入/输出关系的时间图如图 4.21(b)所示。

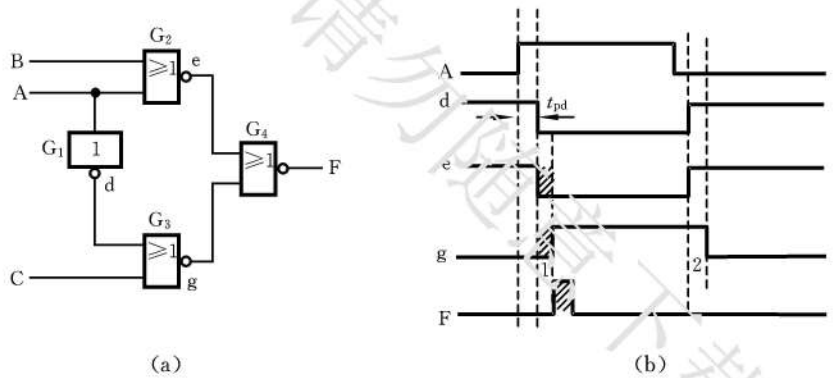


图 4.21　具有险象的逻辑电路及时间图（Ⅱ）

由图 4.21(b)可知，当 A 由低电平变为高电平时，将在输出端产生一个正跳变的尖脉冲信号，破坏了 F 的 0 信号输出，即发生了一次临界竞争。

通常按错误输出脉冲信号的极性将组合电路中的险象分为“0”型险象和“1”型险象。若错误输出信号为负脉冲，则称为“0”型险象；若错误输出信号为正脉冲，则称为“1”型险象。

4.3.2　险象的判断

判断一个电路是否有可能产生险象的方法有代数法和卡诺图法。

由前面对竞争和险象的分析可知，当某个变量 X 同时以原变量和反变量的形式出现在函数表达式中，且在一定条件下该函数表达式可简化成 $X+\overline{X}$ 或者 $X\cdot\overline{X}$ 的形式时，则与该函数表达式对应的电路在 X 发生变化时，可能由于竞争而产生险象。

代数法是根据描述电路的函数表达式来判断相应电路是否具有产生险象的条件。具体方法是：首先检查函数表达式中是否存在具备竞争条件的变量，即是否有某个变量 X 同时以原变量和反变量的形式出现在函数表达式中。若有，则消去函数表达式中的其他变量，即将这些

变量的各种取值组合依次代入函数表达式中，从而把它们从函数表达式中消去，而仅保留被研究的变量 X，再看函数表达式是否会变为 $X+\overline{X}$ 或者 $X\cdot\overline{X}$ 的形式，若会，则说明对应的逻辑电路可能产生险象。下面举例说明。

例 4.13 已知描述某组合逻辑电路的逻辑函数表达式为 $F=\overline{A}\,\overline{C}+\overline{A}B+AC$，试判断该逻辑电路是否可能产生险象。

解 观察函数表达式可知，变量 A 和 C 均具备竞争条件，所以应对这两个变量分别进行分析。先考察变量 A，为此将 B 和 C 的各种取值组合分别代入函数表达式中，可得到如下结果：

$$BC=00 \qquad F=\overline{A}$$
$$BC=01 \qquad F=A$$
$$BC=10 \qquad F=\overline{A}$$
$$BC=11 \qquad F=A+\overline{A}$$

由此可见，当 B=C=1 时，A 的变化可能使电路产生险象。类似地，将 A 和 B 的各种取值组合分别代入函数表达式中，可由代入结果判断出变量 C 发生变化时不会产生险象。

例 4.14 试判断函数表达式 $F=(A+B)(\overline{A}+C)(\overline{B}+C)$ 描述的逻辑电路中是否可能产生险象。

解 从给出的函数表达式可以看出，变量 A 和 B 均具备竞争条件。先考察变量 B，为此将 A 和 C 的各种取值组合分别代入函数表达式中，结果如下：

$$AC=00 \qquad F=B\cdot\overline{B}$$
$$AC=01 \qquad F=B$$
$$AC=10 \qquad F=0$$
$$AC=11 \qquad F=1$$

可见，当 A=C=0 时，B 的变化可能使电路输出产生险象。用同样的方法考察 A，可发现当 B=C=0 时，A 的变化也可能产生险象。

判断险象的另一种方法是卡诺图法。当描述电路的逻辑函数为与-或表达式时，采用卡诺图来判断险象比代数法更为直观、方便。其具体方法是：首先作出函数卡诺图，并画出和函数表达式中各与项对应的卡诺圈；然后观察卡诺图，若发现某两个卡诺圈存在“相切”关系，即两卡诺圈之间存在不被同一卡诺圈包含的相邻最小项，则该电路可能产生险象。下面举例说明。

例 4.15 已知某逻辑电路对应的函数表达式为 $F=\overline{A}D+\overline{A}C+AB\overline{C}$，试判断该电路是否可能产生险象。

解 首先作出给定函数的卡诺图，并画出函数表达式中各与项对应的卡诺圈，如图 4.22 所示。

观察该卡诺图可发现，包含最小项 m_1，m_3，m_5，m_7 的卡诺圈和包含最小项 m_{12}，m_{13} 的卡诺圈之间存在相邻最小项 m_5 和 m_{13}，且 m_5 和 m_{13} 不被同一卡诺圈所包含，所以这两个卡诺圈“相切”。这说明相应电路可能产生险象。这一结论可用代数法进行验证，即假定 B=D=1，C=0，代入函数表达式 F 之后可得 $F=A+\overline{A}$，可见相应电路可能由于 A 的变化而产生险象。

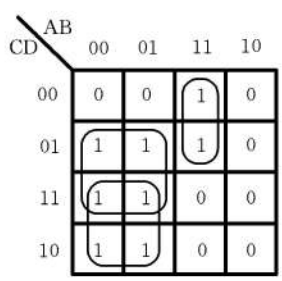


图 4.22 卡诺图

4.3.3　险象的消除

为了使一个电路可靠地工作，设计者应当设法消除或避免电路中可能出现的险象。下面介绍几种常用的方法。

1. 用增加冗余项的方法消除险象

增加冗余项的方法是，通过在函数表达式中“或”上多余的与项或者“与”上多余的或项，使原函数不可能在某种条件下化成 $X+\overline{X}$ 或者 $X \cdot \overline{X}$ 的形式，从而消除可能产生的险象。具体冗余项的选择可以采用代数法或者卡诺图法。

例 4.16　用增加冗余项的方法消除图 4.20(a)所示电路中可能产生的险象。

解　图 4.20(a)所示函数表达式为

$$F=AB+\overline{A}C$$

在前面分析过，当 B=C=1 时，输入 A 的变化使电路输出可能产生“0”型险象，即在输出应该为 1 的情况下产生了一个瞬间的 0 信号。解决问题的思路是，如何保证当 B=C=1 时，输出保持为 1。显然，若函数表达式中包含有与项 BC，则可达到这一目的。由逻辑代数的定理 8 可知，若某变量以原变量和反变量的形式出现在与-或表达式的某两个与项中，则由该两项的其余因子组成的第三项是冗余项。因此，BC 是上述函数的一个冗余项，将 BC 加入函数表达式中并不影响原函数的逻辑功能。加入冗余项 BC 后的函数表达式为

$$F=AB+\overline{A}C+BC$$

增加冗余项后的逻辑电路如图 4.23 所示。该电路不再产生险象。

冗余项的选择也可以通过在函数卡诺图上增加多余的卡诺圈来实现。其具体方法是：若卡诺图上某两个卡诺圈“相切”，则用一个多余的卡诺圈将它们之间的相邻最小项圈起来，与多余卡诺圈对应的与项就是要加入函数表达式中的冗余项。

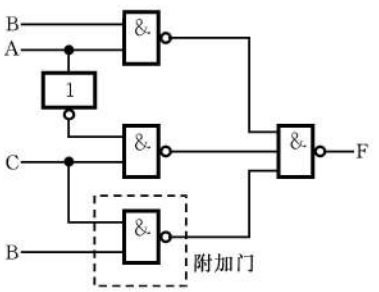


图 4.23　增加冗余项后的逻辑电路

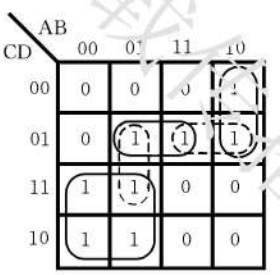


图 4.24　卡诺图

例 4.17　已知描述某组合电路的函数表达式为 $F=\overline{A}C+B\overline{C}D+A\overline{B}\overline{C}$，试用增加冗余项的方法消除该电路中可能产生的险象。

解　首先作出给定函数的卡诺图，如图 4.24 所示。该卡诺图中，包含最小项 m_2，m_3，m_6，m_7 的卡诺圈和包含最小项 m_5，m_{13} 的卡诺圈“相切”，其相邻最小项为 m_7 和 m_5；包含最小项 m_5，m_{13} 的卡诺圈和包含最小项 m_8，m_9 的卡诺圈“相切”，其相邻最小项为 m_9 和 m_{13}。可见，该电路可能由于竞争而产生险象。为了消除险象，可以在卡诺图上增加两个多余卡诺圈，分别把最小项 m_5，m_7 和 m_9，m_{13} 圈起来，如图 4.24 中虚线圈所示。由此得到函数表达式

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

式中，$\overline{A}BD$ 和 $A\overline{C}D$ 为冗余项。读者可用代数法验证，该函数表达式所对应的逻辑电路不再存在险象。

2. 增加惯性延时环节

在实际电路中用来消除险象的另一种方法是在组合电路输出端连接一个惯性延时环节。通常采用 RC 电路作惯性延时环节，如图 4.25(a)所示。由电路知识可知，图中的 RC 电路实际上是一个低通滤波器。由于组合电路的正常输出是一个频率较低的信号，而由竞争引起的险象都是一些频率较高的尖脉冲信号，因此，险象在通过 RC 电路后能基本被滤掉，保留下来的仅仅是一些幅度极小的毛刺，它们不再对电路的可靠性产生影响。图 4.25(b)表明了这种方法的效果。

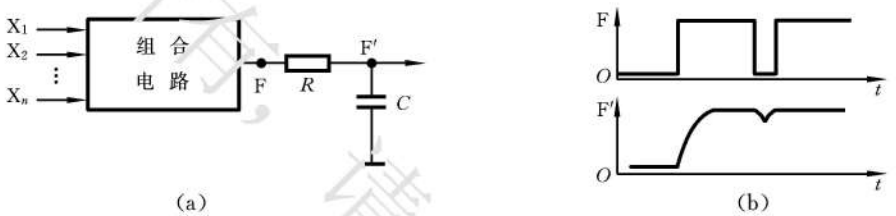


图 4.25 惯性延时环节

要注意的是：采用这种方法时，必须适当选择惯性环节的时间常数($\tau=RC$)，一般要求 τ 大于尖脉冲的宽度，以便能将尖脉冲“削平”；但也不能太大，否则将使正常输出的信号产生不允许的畸变。

3. 选通法

上面介绍了用增加冗余项或增加惯性延时环节消除险象的方法。这两种方法的缺点是要增加器件。对于组合电路中的险象除了用上述方法消除外，还可以采用另外一种完全不同的方法，那就是避开险象而不是消除险象。选通法不必增加任何器件，仅仅是利用选通脉冲的作用，从时间上加以控制，以避开险象脉冲。

由于组合电路中的险象总是发生在输入信号发生变化的过程中，且险象总是以尖脉冲的形式输出。因此，只要对输出波形从时间上加以选择和控制，利用选通脉冲选择输出波形的稳定部分，而有意避开可能出现的尖脉冲，便可获得正确的输出。

例如，图 4.26 所示与非门电路的输出函数表达式为

$$F=\overline{\overline{A\cdot 1}\cdot\overline{\overline{A}\cdot 1}}=A+\overline{A}$$

当 A 发生变化时，可能产生“0”型险象。

为了避开险象，可采用选通脉冲对该电路的输出门加以控制。在选通脉冲到来之前，选通控制线为低电平，门 G_4 关闭，电路输出被封锁，使险象脉冲无法输出。当选通脉冲到来后，门 G_4 开启，使电路送出稳定输出信号。

通常把这种在时间上让信号有选择地通过的方法称为选通法。

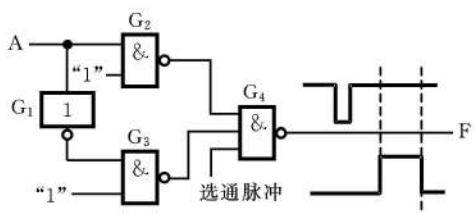


图 4.26 用选通法避开险象原理图

习 题 四

4.1 分析图 4.27 所示的组合逻辑电路，说明电路功能，并画出其简化逻辑电路图。

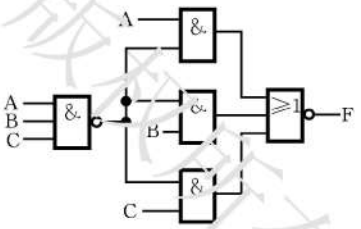


图 4.27 组合逻辑电路

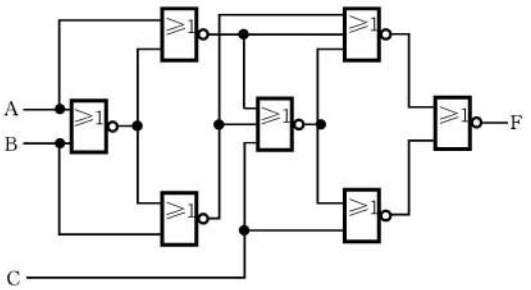


图 4.28 组合逻辑电路

4.2 分析图 4.28 所示的组合逻辑电路：(1) 指出在哪些输入取值下，输出 F 的值为 1；(2) 改用异或门实现该电路的逻辑功能。

4.3 分析图 4.29 所示组合逻辑电路，列出真值表，说明该电路的逻辑功能。

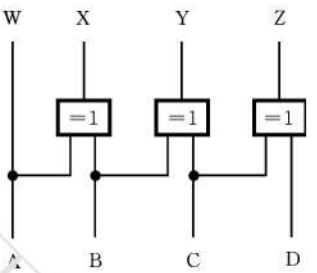


图 4.29 组合逻辑电路

4.4 设计一个组合逻辑电路，该电路输入端接收两个 2 位二进制数 $A=A_2A_1$，$B=B_2B_1$。当 A>B 时，输出 Z=1，否则 Z=0。

4.5 设计一个代码转换电路，将 1 位十进制数的余 3 码转换成 2421 码。

4.6 假定 X=AB 代表一个 2 位二进制数，试设计满足如下要求的逻辑电路(Y 也用二进制数表示)：

(1) $Y=X^2$ (2) $Y=X^3$

4.7 用与非门设计一个组合逻辑电路，该电路输入为 1 位十进制数的 2421 码，当输入的数字为素数时，输出 F 为 1，否则 F 为 0。

4.8 设计一个“四舍五入”电路。该电路输入为 1 位十进制数的 8421 码，当其值大于或等于 5 时，输出 F 的值为 1，否则 F 的值为 0。

4.9 设计一个检测电路，检测 4 位二进制码中 1 的个数是否为偶数。若为偶数个 1，则输出为 1，否则输出为 0。

4.10 设计一个加/减法器，该电路在 M 控制下进行加、减运算。当 M=0 时，实现全加器功能；当 M=1 时，实现全减器功能。

4.11 在输入不提供反变量的情况下，用与非门组成实现下列函数的最简电路。

(1) $F=A\overline{B}+\overline{A}C+B\overline{C}$ (2) $F=A\overline{B}\overline{C}+BC\overline{D}+A\overline{C}\overline{D}+\overline{B}CD$

4.12 下列函数描述的电路是否可能发生竞争？竞争结果是否会产生险象？在什么情况下产生险象？若产生险象，试用增加冗余项的方法消除。

(1) $F_1=AB+A\overline{C}+\overline{C}D$ (2) $F_2=AB+\overline{A}CD+BC$ (3) $F_3=(A+\overline{B})(\overline{A}+\overline{C})$