第四章 组合逻辑电路

4.1 概述

组合逻辑电路是数字电路中最简单的一类逻辑电路,其基本特点是结构上无反馈、功能上无记忆,电路在任何时刻的输出都由该时刻的输入信号完全确定。图 4.1 是组合逻辑电路的示意框图。

图 4.1 组合逻辑电路示意框图

在图 4.1 中，X_0、X_1、…、X_{n-1}是输入逻辑变量，F_0、F_1、…、F_{m-1}是输出逻辑变量。任何时刻电路的输出，仅仅只取决于该时刻各个输入变量的取值，这样的逻辑电路就称为组合逻辑电路，简称为组合电路。输出变量和输入变量之间的逻辑关系一般表示为：

$$F_0 = f_0(X_0, X_1, \cdots, X_{n-1})$$
$$F_1 = f_1(X_0, X_1, \cdots, X_{n-1})$$
$$\vdots$$
$$F_{m-1} = f_{m-1}(X_0, X_1, \cdots, X_{n-1})$$

从电路结构来看,组合电路具有两个特点：

(1)电路由逻辑门电路组成,不包含任何记忆元件,没有记忆能力；

(2)输入信号是单向传输的,电路中不存在任何反馈回路。

组合逻辑电路需要讨论的两个基本问题是分析与设计。这两个问题实际就是逻辑门和逻辑代数的综合应用。

对于一个已知的逻辑电路,应用逻辑函数来描述它的工作、研究它的工作特性和逻辑功能称为分析。反过来,根据逻辑要求,或者描述逻辑功能的函数,确定用什么逻辑电路来实现其功能叫做设计。显然,分析和设计是两个相反的过程。

组合电路的应用十分广泛。它不但能独立完成各种功能复杂的逻辑操作,而且也是时序逻辑电路的重要组成部分,因此,它在逻辑电路中占有相当重要的地位。

另外，对于一些常用的组合逻辑电路，如加法器、比较器、编码器、译码器、数据选择器和数据分配器等，事实上并不需要我们用逻辑门来设计，因为它们有现成的集成模块。本章的另一个内容就是介绍各种常用的 MSI（）组合逻辑电路、实现原理和应用方法。

4.2 加法器

加法器是一种最基本的算术运算电路，其功能是实现二进制数的加法运算。计算机 CPU 中的运算器就包含了加法器单元。

4.2.1 半加器和全加器

一、半加器

只考虑本位两个一位二进制数 A_i 和 B_i 相加而不考虑相邻低位进位的加法器称为“半加器”（Half Adder）。

半加器的真值表如表 4.1 所示。表中 A_i 和 B_i 分别表示被加数和加数，S_i 为本位和输出，C_{i+1} 为向相邻高位的进位输出。

表 4.1 半加器的真值表

A_i	B_i	S_i	C_i
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

由真值表可以直接写出输出逻辑函数表达式：

$$S_i = \overline{A_i} B_i + A_i \overline{B_i} = A_i \oplus B_i$$

$$C_i = A_i B_i$$

半加器的逻辑电路图和逻辑符号如图 4.2 所示。

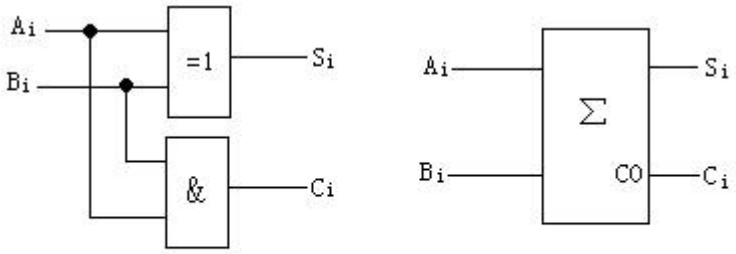


图 4.2 半加器

二、全加器

在实际的加法运算中，除了最低位外，其它各位都需要考虑低位向本位的进位。这种对两个一位二进制并考虑低位来的进位的加法运算称为“全加”。实现全加运算的电路叫全加器（Full Adder）。

加法器的真值表如表 4.2 所示。表中 A_i 和 B_i 分别表示被加数和加数，C_{i-1} 表示来自相邻低位的进位，S_i 为本位和输出，C_i 为向相邻高位的进位输出。

表 4.2 全加器的真值表

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

根据真值表可以写出 S_i 和 C_i 的输出逻辑函数表达式：

$$S_i = \overline{A_i}\,\overline{B_i}C_{i-1} + \overline{A_i}B_i\overline{C_{i-1}} + A_i\overline{B_i}\,\overline{C_{i-1}} + A_iB_iC_{i-1}$$

$$= \overline{A_i}(B_i \oplus C_{i-1}) + A_i(\overline{B_i \oplus C_{i-1}}) = A_i \oplus B_i \oplus C_i$$

$$C_i = A_i\overline{B_i}C_{i-1} + A_iB_i\overline{C_{i-1}} + B_iC_{i-1} = A_i(B_i \oplus C_{i-1}) + B_iC_{i-1}$$

全加器的逻辑电路图和逻辑符号如图 4.3 所示。

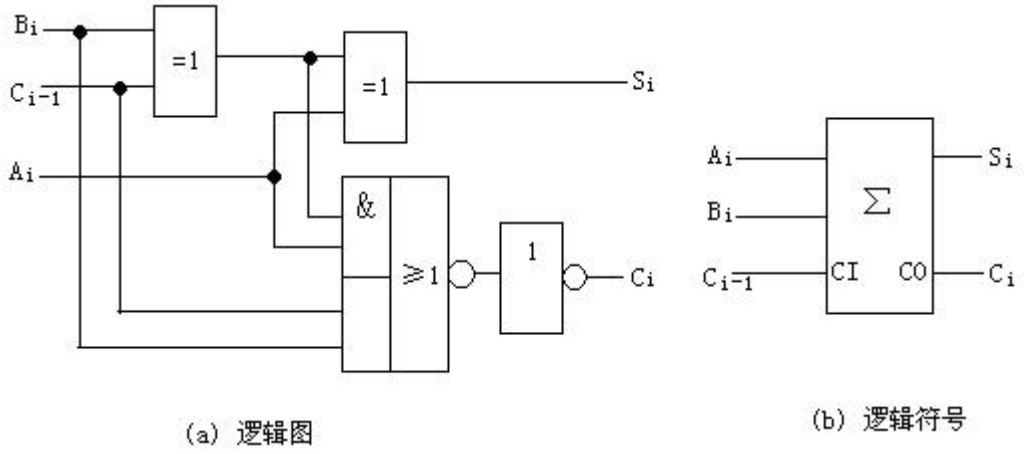


图 4.3 全加器

三、集成全加器

集成全加器 74183 的外引线排列图如图 4.4 所示。这种双全加器具有独立的全加和与进位输出，这样每个全加器既可单独使用，又可将两个全加器级连起来使用。

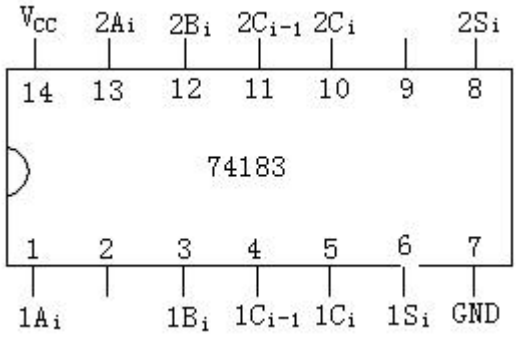


图 4.4

4.2.2 加法器模块

实现多位二进制数相加的电路称为加法器。根据进位方式不同，分为串行进位加法器和超前进位加法器。

一、4 位串行进位加法器

将四个全加器依次级连起来，就构成了 4 位串行进位加法器，电路如图 4.5 所示。这种加法器的最大优点是电路简单，连接方便。但最大缺点是运算速度太慢。从图中可以看到，被加数和加数的各位是同时加到各位的输入端，而各位全法器的进位输入则是按照由低向高逐级串行传送的，各进位形成一个进位链。由于每一位相加的和都与本位的进位输入有关，所以，最高位必须等到各低位全部完成相加并送来进位信号后才能产生运算结果。显然这种加法器的位数越多，运算速度就越慢。

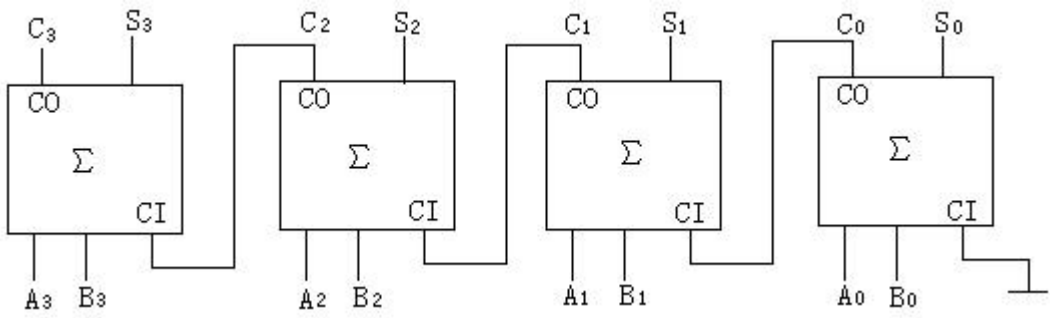


图 4.5 四位串行加法器

二、超前进位加法器

由串行加法器可知，为了提高运算速，必须设法减小或消除由于进位信号逐级传递所浪费的时间。这就要求各位的进位信号能事先知道。对于四位加法器，第一位全加器的进位信号表达式由表 4.2 可写为：

$$C_0 = A_0B_0 + A_0C_{0-1} + B_0C_{0-1} = A_0B_0 + (A_0 + B_0)C_{0-1}$$

第二位全加器的进位信号表达式可写为:

$$C_1 = A_1B_1 + (A_1 + B_1)C_0 = A_1B_1 + (A_1 + B_1)[A_0B_0 + (A_0 + B_0)C_{0-1}]$$

第三位全加器的进位信号表达式可写为:

$$C_2 = A_2B_2 + (A_2 + B_2)C_1$$

$$= A_2B_2 + (A_2 + B_2)\{A_1B_1 + (A_1 + B_1)[A_0B_0 + (A_0 + B_0)C_{0-1}]\}$$

第四位全加器的进位信号表达式可写为:

$$C_3 = A_3B_3 + (A_3 + B_3)C_2$$

$$= A_3B_3 + (A_3 + B_3)\{A_2B_2 + (A_2 + B_2)\{A_1B_1 + (A_1 + B_1)[A_0B_0 + (A_0 + B_0)C_{0-1}]\}\}$$

可见,只要 A_3 A_2 A_1 A_0、B_3 B_2 B_1 B_0 和 C_{0-1} 给出后,便可按以上表达式确定 C_3、C_2 、C_1、C_0。这样，如果用逻辑门实现上述逻辑函数表达式，并将结果送到相应全加器的进位输入端，则每一级的全加运算再也不需等待了，四位超前进位加法器就是由 4 个全加器和相应的进位逻辑电路构成。图 4.6 是四位二进制超前进位加法器 74283 的引线图。

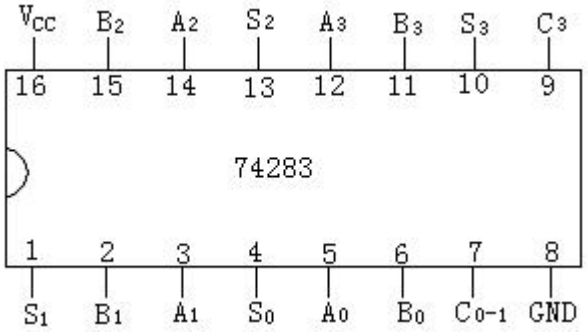


图 4.6 四位 2 进制超前进位加法器

4.2.3 加法器的应用

一、利用加法器实现特殊代码的转换

当需要转换的两种代码间存在某种数量上的关系时，利用加法器就可以很方便地实现它们之间的转换。

例 4.10 利用集成加法器 74283 实现 8421BCD 码与余 3 码之间的相互转换。

解：8421BCD 码和余 3 码的编码表如表 4.3 所示。从表中可见，余 3 码是在对应的 8421BCD 码的基础上加 0011。

表 4.3 8421BCD 码和余三码编码表

十进制数	8421BCD 码 A B C D	余 3 码 W X Y Z
0	0 0 0 0	0 0 1 1
1	0 0 0 1	0 1 0 0
2	0 0 1 0	0 1 0 1
3	0 0 1 1	0 1 1 0
4	0 1 0 0	0 1 1 1
5	0 1 0 1	1 0 0 0
6	0 1 1 0	1 0 0 1
7	0 1 1 1	1 0 1 0
8	1 0 0 0	1 0 1 1
9	1 0 0 1	1 1 0 0

可得出：

WXYZ=ABCD+0011

ABCD=WXYZ-0011

为了变减法为加法，可对-3 求 4 位二进制数的补码得 1101。所以有：

WXYZ=ABCD+0011

ABCD=WXYZ+1101

实现 8421BCD 码与余 3 码转换的电路图如图 4.7 所示。

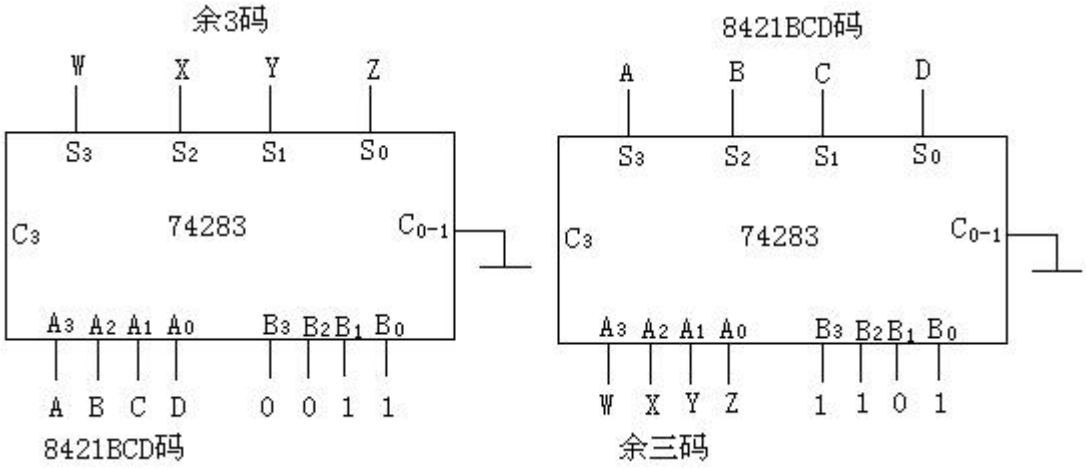


图 4.7 利用加法器实现 BCD 码与余三码之间的转换

二、利用二进制加法器实现十进制数加法运算

为了简便起见，这里只讨论一位十进制数的加法运算问题。两个 1 位 8421BCD 码相加，

实质上是两个小于等于 9（1001）的 4 位二进制数的相加。按 8421BCD 码的要求，相加后仍应是 8421BCD 码；而按二进制数相加其结果却是二进制数，这就需要进行校正处理。表 4.4 列出了按两种运算规则相加的结果。从表中可见，若相加的结果是 0~9 之间的数，则二者相同，若相加的结果是 10~19 之间的数，则二进制数相加和比对应的用 8421BCD 码表示的相加和等效的二进制值固定相差为 6（0110），所以，若要得出用 8421BCD 码表示的相加和，就需对实际的相加结果加 6（0110）进行校正。

表 4.4 两种运算规则相加的结果

十进制数	二进制数相加和					8421BCD 码相加				
N	C_3	S_3	S_2	S_1	S_0	C_{10}	S_8	S_4	S_2	S_1
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	1
2	0	0	0	1	0	0	0	0	1	0
3	0	0	0	1	1	0	0	0	1	1
4	0	0	1	0	0	0	0	1	0	0
5	0	0	1	0	1	0	0	1	0	1
6	0	0	1	1	0	0	0	1	1	0
7	0	0	1	1	1	0	0	1	1	1
8	0	1	0	0	0	0	1	0	0	0
9	0	1	0	0	1	0	1	0	0	1
10	0	1	0	1	0	1	0	0	0	0
11	0	1	0	1	1	1	0	0	0	1
12	0	1	1	0	0	1	0	0	1	0
13	0	1	1	0	1	1	0	0	1	1
14	0	1	1	1	0	1	0	1	0	0
15	0	1	1	1	1	1	0	1	0	1
16	1	0	0	0	0	1	0	1	1	0
17	1	0	0	0	1	1	0	1	1	1
18	1	0	0	1	0	1	1	0	0	0
19	1	0	0	1	1	1	1	0	0	1

根据上表分析，先用一片 74238 将两个一位的 8421BCD 码相加,并对所得的和数进行判断,决定是否需要校正。设校正标志函数为 Z,，并设相加和数大于 9（1001）时 Z=1，表示需要做加 6（0110）校正；否则 Z=0，表示不需要校正。由表 4.11 可知，需要校正的和数为 01010~10011。可见函数 $Z= C_3+S_3S_2+S_3S_1$。

在得出 Z 的函数式后，再用一片 74283 进行加 6（0110）校正运算。此时参加运算的两个数是 $S_3S_2S_1S_0$ 和 0ZZ0。相加的和作为 8421BCD 码的个位输出，而 Z 则作为 8421BCD 码的十位输出。电路如图 4.8 所示。

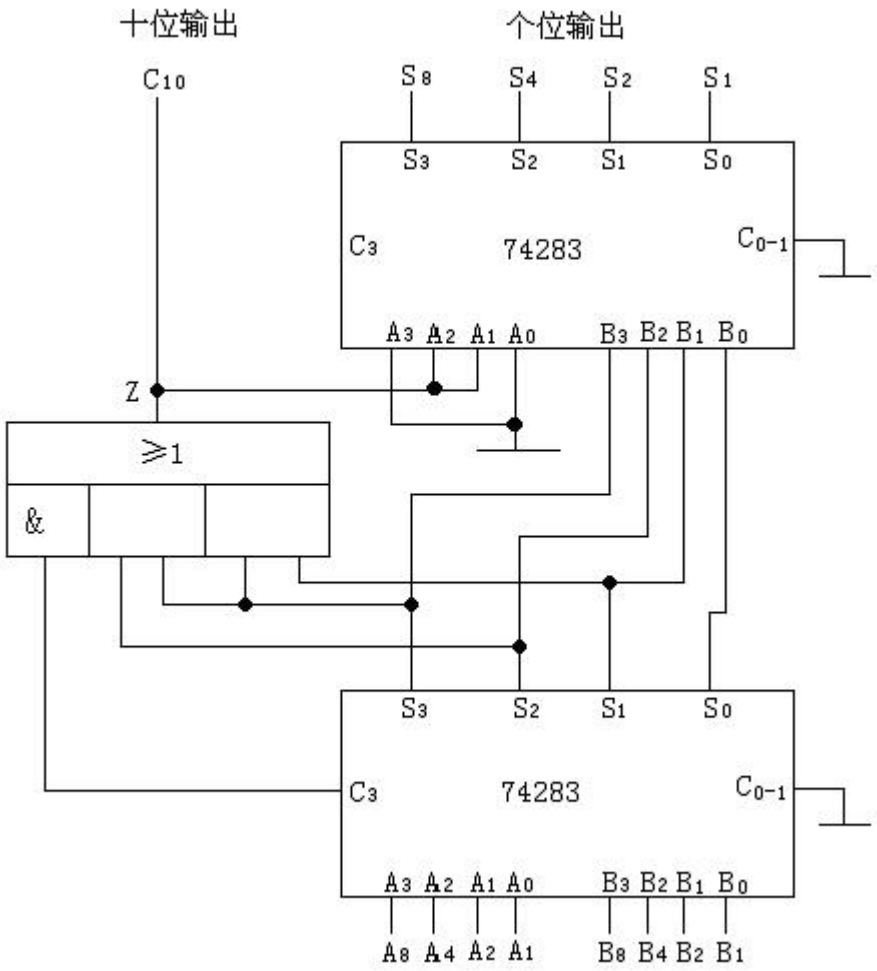


图 4.8 一位 8421BCD 码加法器电路

4.3 数值比较器

数值比较器是对两个位数相同的二进制数进行数值比较并判断其大小关系的算术运算电路。

4.3.1 一位数值比较器

一位数值比较器的输入是两个要进行比较的 1 位二进制数，这里用 A、B 表示，输出是比较的结果，有三种情况：A>B、A=B、A<B，现分别用 L、G、M 表示，并约定当 A>B 时 L=1，A=B 时 G=1，A<B 时 M=1。图 4.9 是一位比较器的示意框图。

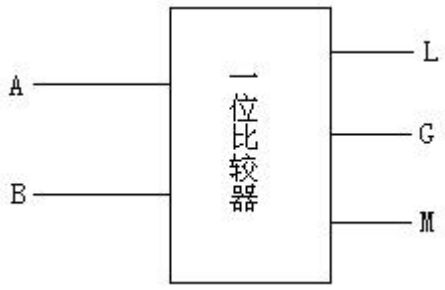


图 4.9 一位比较器框图

根据比较器的概念，设满足相应条件时输出为 1，不满足条件时输出为 0。则可得如表 4.5

所示的真值表。

表 4.5 一位数值比较器的真值表

A	B	L	G	M
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

由表 4.5 可直接得到：

$$L = A\overline{B}$$

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

$$M = \overline{A}B$$

所以可得以下一位数值比较器的逻辑电路图。

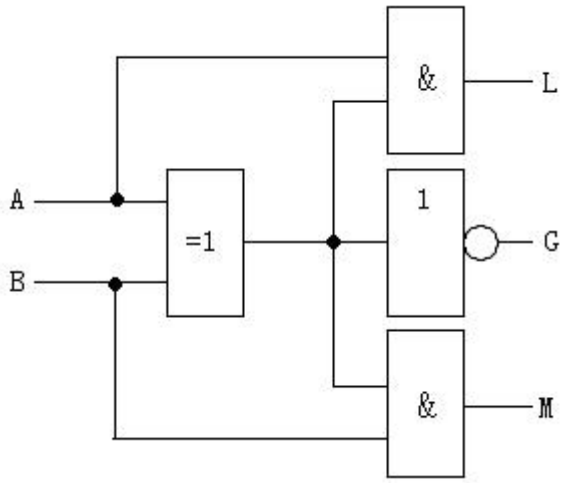


图 4.10 一位数值比较器

4.3.2 四位数值比较器

典型的 4 位集成数值比较器有 7485，图 4.11 是它的外引线图。

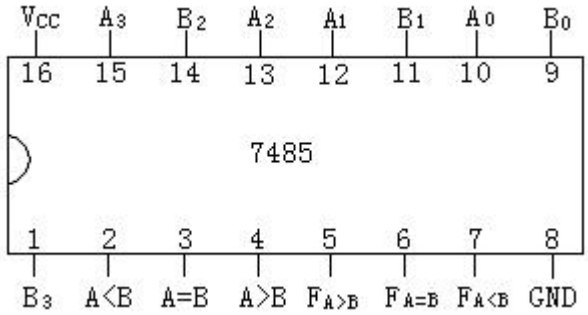


图 4.11 集成数值比较器 7485 外引线图

表 4.6 列出了该比较器的功能真值表。

表 4.6　四位比较器的功能真值表

比较器输入				级联输入			输出		
A_3 B_3	A_2 B_2	A_1 B_1	A_0 B_0	A>B	A<B	A=B	$F_{A>B}$	$F_{A<B}$	$F_{A=B}$
A_3> B_3	d	d	d	d	d	d	1	0	0
A_3< B_3	d	d	d	d	d	d	0	1	0
A_3= B_3	A_2>B_2	d	d	d	d	d	1	0	0
A_3= B_3	A_2<B_2	d	d	d	d	d	0	1	0
A_3= B_3	A_2= B_2	A_1>B_1	d	d	d	d	1	0	0
A_3= B_3	A_2= B_2	A_1<B_1	d	d	d	d	0	1	0
A_3= B_3	A_2= B_2	A_1= B_1	A_0>B_0	d	d	d	1	0	0
A_3= B_3	A_2= B_2	A_1= B_1	A_0<B_0	d	d	d	0	1	0
A_3= B_3	A_2= B_2	A_1= B_1	A_0=B_0	1	0	0	1	0	0
A_3= B_3	A_2= B_2	A_1= B_1	A_0=B_0	0	1	0	0	1	0
A_3= B_3	A_2= B_2	A_1= B_1	A_0=B_0	0	0	1	0	0	1

由功能真值表可见，要确定数 A 是大于还是小于 B，可从最高位开始逐位比较。若两数的最高位不相等，则最高位的比较结果就是最后的结果。如果两数的最高位相等，就进而比较次高位，直到发现两数的某位不相等为止。若 A、B 两数各位均相等，则输出状态取决于级联输入的状态。所以，在没有更低位参与比较时，芯片的级联输入端(A>B)(A<B)(A=B)应接 001，以便在 A、B 两数相等时，产生 A=B 的比较结果。这在使用时是应该注意的。

4.3.3 集成比较器的应用

一、8 位二进制数比较器

8 位二进制数比较器的电路如图 4.12 所示。

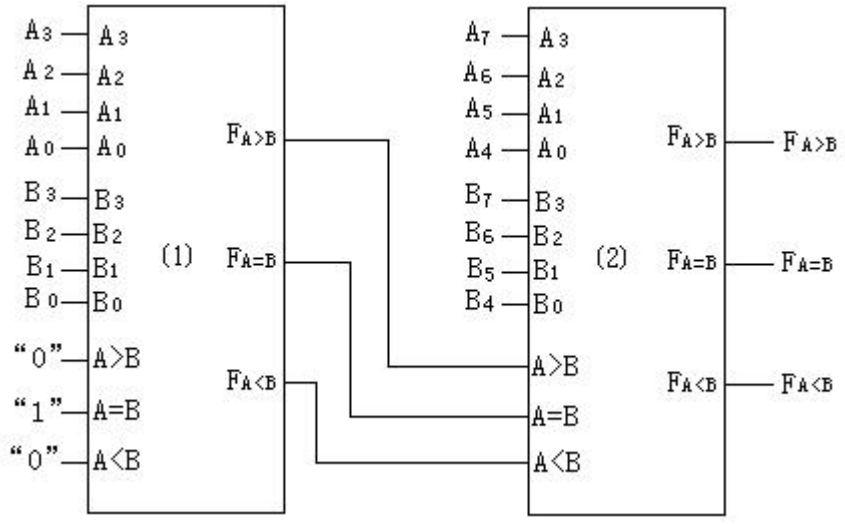


图 4.12　八位 2 进制比较器

图中芯片(1)对低 4 位进行比较，因没有更低的比较结果输入，其级联输入端接“010”。而芯片(2)对高 4 位进行比较，级联端接低位比较器的比较结果输出。当 $A_7A_6A_5A_4 \neq B_7B_6B_5B_4$ 时，8 位比较器的比较结果由高 4 位决定，芯片(1)的比较结果不产生影响；当 $A_7A_6A_5A_4=B_7B_6B_5B_4$ 时，8 位比较器的比较结果由低 4 位决定；当

$A_7A_6A_5A_4A_3A_2A_1A_0=B_7B_6B_5B_4B_3B_2B_1B_0$ 时，比较结果由芯片(1)的级联输入端决定。而当该级联输入是“010”时，最终比较结果为 A=B。

二、利用 4 位比较器和逻辑门设计输血指示器

设该输血指示器的输入是一对要求“输血-受血”的血型,当符合对应的输-受关系时，电路输出为“1”。在人类的四种基本血型中，O 型血可输给任意血型的人，但他自己只能接受 O 型；AB 型可接受任意血型，却只能输给 AB 型；A 型能输给 A 型或 AB 型，可接受 O 型和 A 型；B 型能输给 B 型或 AB 型，可接受 B 型和 O 型。

设用二进制数 00 表示 O 型血；01 表示 A 型血；10 表示 AB 型血；11 表示 B 型血。这样对应输血和受血就需要 4 个输入变量，设用 AB 代表输送血型，CD 代表接受血型。另外用 F 表示输出函数，并用 F=1 表示可输血；F=0 表示不可输血。由此可得如表 4.7 所示的真值表。

根据输血常识，可以输血的情况无非有三种：

(1) 只要血型相同就可以输，即只要 AB=CD,则 F=1；

(2) 只要输送的是 O 型血，就可以输，即 AB=00 时，F=1；

(3) 只要接受方是 AB 型血，就可以输，即 CD=10 时，F=1；

而其他情况均不可输血。

由此得出的用 4 位数码比较器及门电路设计的输血指示器如图 4.13 所示。

表 4.7 输血指示器真值表

输送血型		接受血型		输血指示
A	B	C	D	F
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

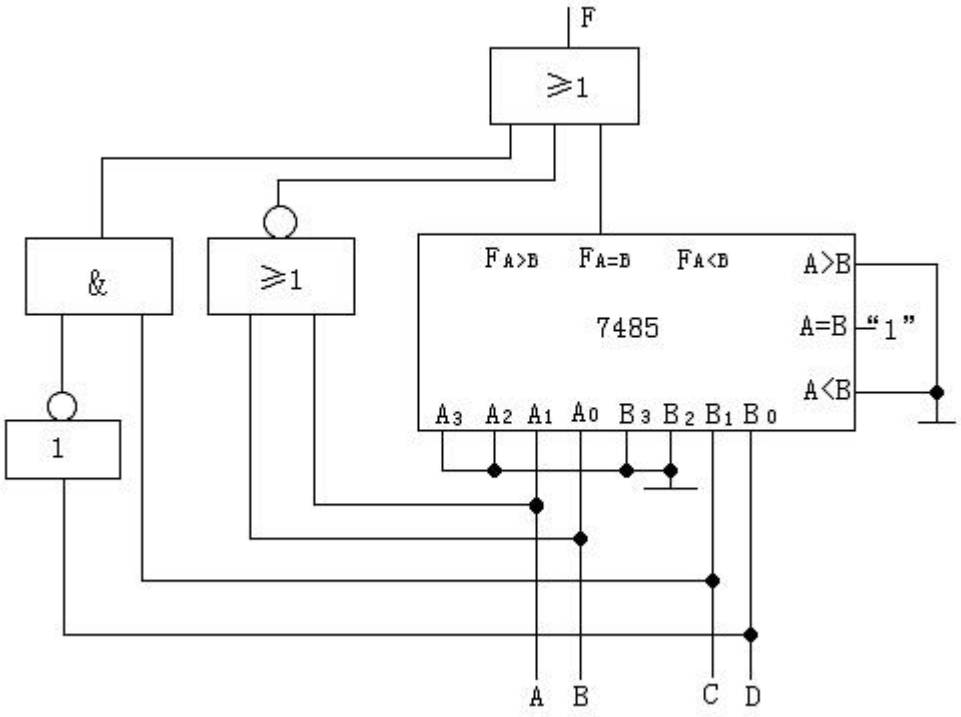


图 4.13 输血指示器电路

4.4 编码器和译码器

4.4.1 编码器

这里的编码特指对一组选定的二值代码赋于其特定的含义。如将人们熟悉的十进制数，用二进制码或 BCD 码表示出来。而在日常生活中，电话号码；人名等都是编码的特例。完成编码的电路称为编码器。

一、二进制编码器

用 n 位二进制代码对 2^n 个信号进行编码的电路称为二进制编码器。3 位二进制编码器的真值表和逻辑框图分别如表 4.8 和图 4.14 所示。这里输入 $\overline{I}_0 \sim \overline{I}_7$ 采用 8 中取 1 码，逻辑 0 有效。而输出 $Y_2Y_1Y_0$ 是 3 位二进制码。

表 4.8 三位二进制编码器真值表

$\overline{I}_0$	$\overline{I}_1$	$\overline{I}_2$	$\overline{I}_3$	$\overline{I}_4$	$\overline{I}_5$	$\overline{I}_6$	$\overline{I}_7$	Y_2	Y_1	Y_0
0	1	1	1	1	1	1	1	0	0	0
1	0	1	1	1	1	1	1	0	0	1
1	1	0	1	1	1	1	1	0	1	0
1	1	1	0	1	1	1	1	0	1	1
1	1	1	1	0	1	1	1	1	0	0
1	1	1	1	1	0	1	1	1	0	1
1	1	1	1	1	1	0	1	1	1	0
1	1	1	1	1	1	1	0	1	1	1

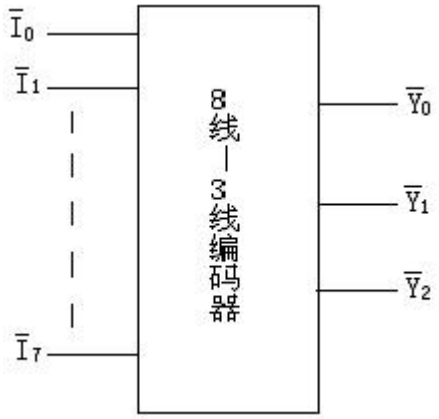


图 4.14 编码器框图

编码器输入有 8 个变量，故可能的组合有 256 种，这里只用了 8 种，其余的 248 种可视为约束条件。即任一时刻只允许一个输入为“0”。故可得：

$\overline{\overline{I_i}+\overline{I_j}}=\bar{1}$ $(i\neq j)$ 约束条件

$I_i\cdot I_j=0$

下面求该编码器的逻辑表达式

$$Y_2=\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3I_4\overline{I}_5\overline{I}_6\overline{I}_7+\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3\overline{I}_4I_5\overline{I}_6\overline{I}_7+\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3\overline{I}_4\overline{I}_5I_6\overline{I}_7+\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3I_4\overline{I}_5\overline{I}_6I_7$$

令： $A=\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3I_4\overline{I}_5\overline{I}_6\overline{I}_7$ $B=\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3\overline{I}_4I_5\overline{I}_6\overline{I}_7$

$C=\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3\overline{I}_4\overline{I}_5I_6\overline{I}_7$ $D=\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3\overline{I}_4\overline{I}_5\overline{I}_6I_7$

根据约束条件：

$$A=\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3I_4\overline{I}_5\overline{I}_6\overline{I}_7+I_0I_4+I_1I_4+I_2I_4+I_3I_4+I_5I_4+I_6I_4+I_7I_4$$

$$=I_4[\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3\overline{I}_5\overline{I}_6\overline{I}_7+(I_0+I_1+I_2+I_3+I_5+I_6+I_7)]$$

$$=I_4[\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3\overline{I}_5\overline{I}_6\overline{I}_7+\overline{\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3\overline{I}_5\overline{I}_6\overline{I}_7}]=I_4$$

同样可得：$B=I_5$ $C=I_6$ $D=I_7$

故 $Y_2=I_4+I_5+I_6+I_7=\overline{\overline{I}_4\overline{I}_5\overline{I}_6\overline{I}_7}$

用同样的方法可求得：

$Y_1=I_2+I_3+I_6+I_7=\overline{\overline{I}_2\overline{I}_3\overline{I}_6\overline{I}_7}$ $Y_0=I_1+I_3+I_5+I_7=\overline{\overline{I}_1\overline{I}_3\overline{I}_5\overline{I}_7}$

可见，用 3 个 4 输入与非门就可实现这种 8 线—3 线二进制编码器。

在该编码器中，对$\overline{I}_0$的编码是隐含编码。即当$\overline{I}_1 \sim \overline{I}_7$均处于无效状态时，编码器输出的就是对$\overline{I}_0$的编码。

二、优先编码器

在前面介绍的二进制编码器中，一次只允许一个输入信号有效，即有约束条件。而优先编码器允许一次有多个输入端有编码有效信号输入。但优先编码器对全部编码输入信号规定了不同的优先等级，当多个输入信号有效时，它能够根据事先安排好的优先顺序，只对优先级最高的有效输入信号进行编码。当优先级高的输入没有编码请求时，编码器才对低优先的输入进行编码。图 4.15 是优先编码器 74148 的电路图和逻辑符号，表 4.9 给出了 74148 的功能真值表。

表 4.9　优先编码器 74LS148 真值表

$\overline{ST}$	$\overline{I}_0$	$\overline{I}_1$	$\overline{I}_2$	$\overline{I}_3$	$\overline{I}_4$	$\overline{I}_5$	$\overline{I}_6$	$\overline{I}_7$	$\overline{Y}_2$	$\overline{Y}_1$	$\overline{Y}_0$	$\overline{Y}_{EX}$	Y_S
1	d	d	d	d	d	d	d	d	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	d	d	d	d	d	d	d	0	0	0	0	0	1
0	d	d	d	d	d	d	0	1	0	0	1	0	1
0	d	d	d	d	d	0	1	1	0	1	0	0	1
0	d	d	d	d	0	1	1	1	0	1	1	0	1
0	d	d	d	0	1	1	1	1	1	0	0	0	1
0	d	d	0	1	1	1	1	1	1	0	1	0	1
0	d	0	1	1	1	1	1	1	1	1	0	0	1
0	0	1	1	1	1	1	1	1	1	1	1	0	1

由真值表可以看出，编码器输入信号$\overline{I}_0 \sim \overline{I}_7$均为低电平有效，且$\overline{I}_7$的优先权最高，$\overline{I}_6$次之，$\overline{I}_0$最低。编码输出信号$\overline{Y}_2$、$\overline{Y}_1$和$\overline{Y}_0$则为反码输出。$\overline{ST}$为选通输入端（使能输入端）当$\overline{ST}$=0 时，编码器处于工作态；当$\overline{ST}$=1 时，输出$\overline{Y}_2$、$\overline{Y}_1$、$\overline{Y}_0$和$\overline{Y}_S$、$\overline{Y}_{EX}$均被封锁，编码器不工作，编码输出$\overline{Y}_2$、$\overline{Y}_1$和$\overline{Y}_0$全为 1。$Y_S$是选通输出端，级联使用时，高片位的$Y_S$端与低片位的$\overline{ST}$端连接起来，可以对优先编码器进行扩展。$\overline{Y}_{EX}$为优先扩展输出端，级联应用时可作为输出位的扩展端。

从真值表还可以看出，Y_S=0，表示编码器处于工作态，但无有效编码信号输入；如果Y_S为 1，则表示编码器或不工作，或工作且有有效编码信号输入。而$\overline{Y}_{EX}$=0，表示编码器处于工作态且有有效编码信号输入；如果$\overline{Y}_{EX}$=1，则表示编码器不工作，或工作，但无有效编码信号输入。利用使能输出端和扩展输出端，可以很方便地实现编码器的扩展。74LS148 优先编码器的函数表达式如下所示。

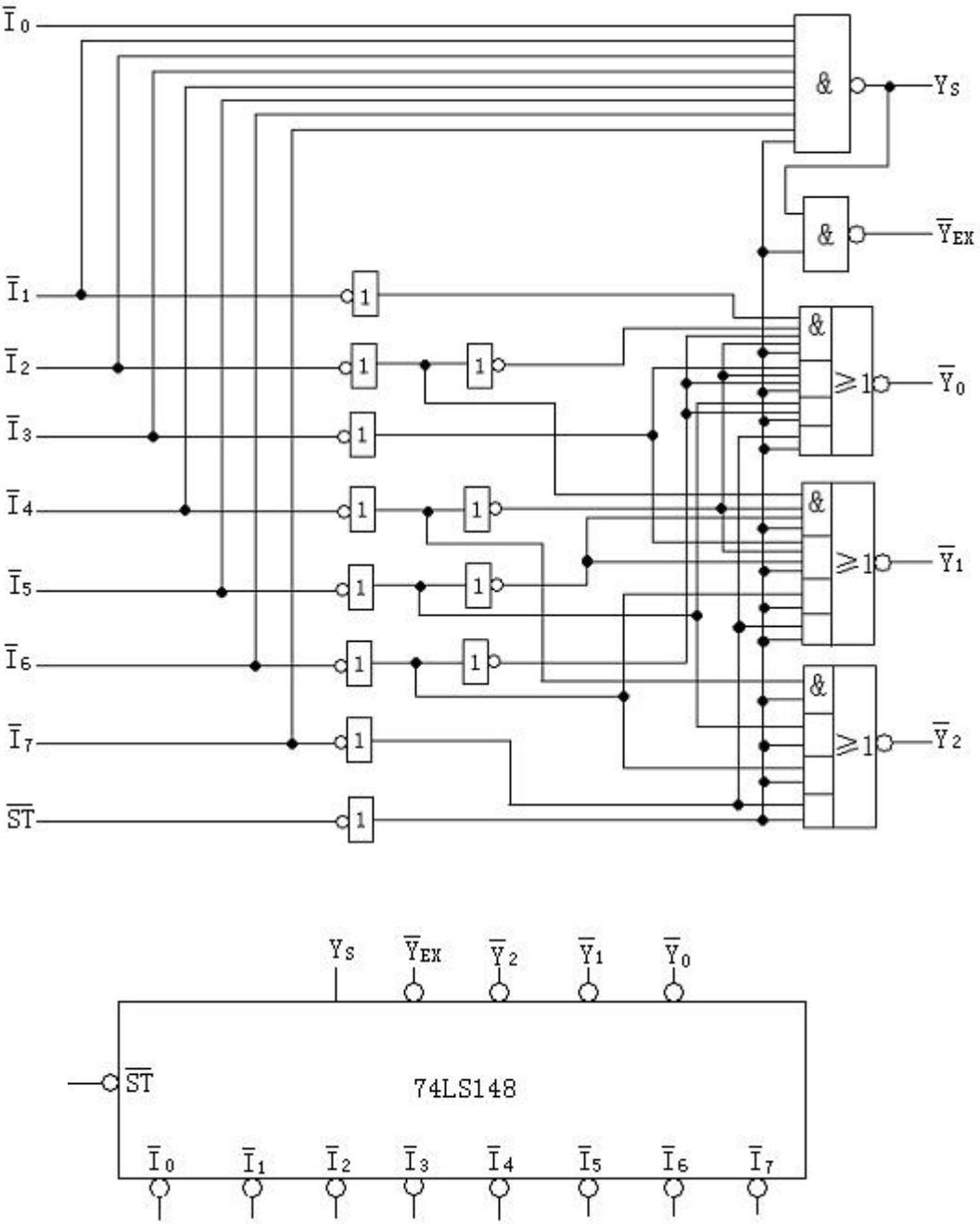


图 4.15 优先编码器 74148 的电路与逻辑符号

$$\overline{Y}_2 = \overline{ST(I_4 + I_5 + I_6 + I_7)}$$

$$\overline{Y}_1 = \overline{ST(I_2\overline{I}_4\overline{I}_5 + I_3\overline{I}_4\overline{I}_5 + I_6 + I_7)}$$

$$\overline{Y}_0 = \overline{ST(I_1\overline{I}_2\overline{I}_4\overline{I}_6 + I_3\overline{I}_4\overline{I}_6 + I_5\overline{I}_6 + I_7)}$$

$$Y_S = \overline{\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3\overline{I}_4\overline{I}_5\overline{I}_6\overline{I}_7 ST}$$

$$\overline{Y}_{EX} = \overline{\overline{\overline{I}_0\overline{I}_1\overline{I}_2\overline{I}_3\overline{I}_4\overline{I}_5\overline{I}_6\overline{I}_7 ST} \cdot ST}$$

4.4.2 编码器的应用

一、优先编码器的级联应用

将两片 74LS148 级联起来，就构成了 16 线—4 线优先编码器。其电路图如图 4.16 所示。它有 16 个编码信号输入端 $\overline{A}_{15} \sim \overline{A}_0$ 以及 4 个编码输出端 $\overline{Z}_3 \sim \overline{Z}_0$。第一片的编码信号输入端 $\overline{I}_7 \sim \overline{I}_0$ 作为 $\overline{A}_{15} \sim \overline{A}_8$ 输入；第二片的编码输入端 $\overline{I}_7 \sim \overline{I}_0$ 作为 $\overline{A}_7 \sim \overline{A}_0$，第一片的 $\overline{ST}$ 固定接“0”，处于工作态，而 Y_S 接第二片的 $\overline{ST}$ 输入端，控制第二片的工作。第一片的 $\overline{Y}_{EX}$ 作为输出 $\overline{Z}_3$。

当第一片有输入时，由于第二片的 $\overline{ST}$=1，故第二片不工作，且输出均为“1”。此时该级联编码器的输出就是 $\overline{Y}_{EX}=0$ 以及第一片 $\overline{Y}_2\overline{Y}_1\overline{Y}_0$ 的输出，即 $\overline{Z}_3\overline{Z}_2\overline{Z}_1\overline{Z}_0=\overline{Y}_{EX}\overline{Y}_2\overline{Y}_1\overline{Y}_0=0\overline{Y}_2\overline{Y}_1\overline{Y}_0$

当第一片没有输入信号时，$\overline{Y}_{EX}=1$，Y_S=0，使第二片的 $\overline{ST}$=0，第二片处于工作态。此时的输出就是 $\overline{Y}_{EX}$ =1 以及第二片 $\overline{Y}_2\overline{Y}_1\overline{Y}_0$ 的输出。即 $\overline{Z}_3\overline{Z}_2\overline{Z}_1\overline{Z}_0=\overline{Y}_{EX}\overline{Y}_2\overline{Y}_1\overline{Y}_0=1\overline{Y}_2\overline{Y}_1\overline{Y}_0$。

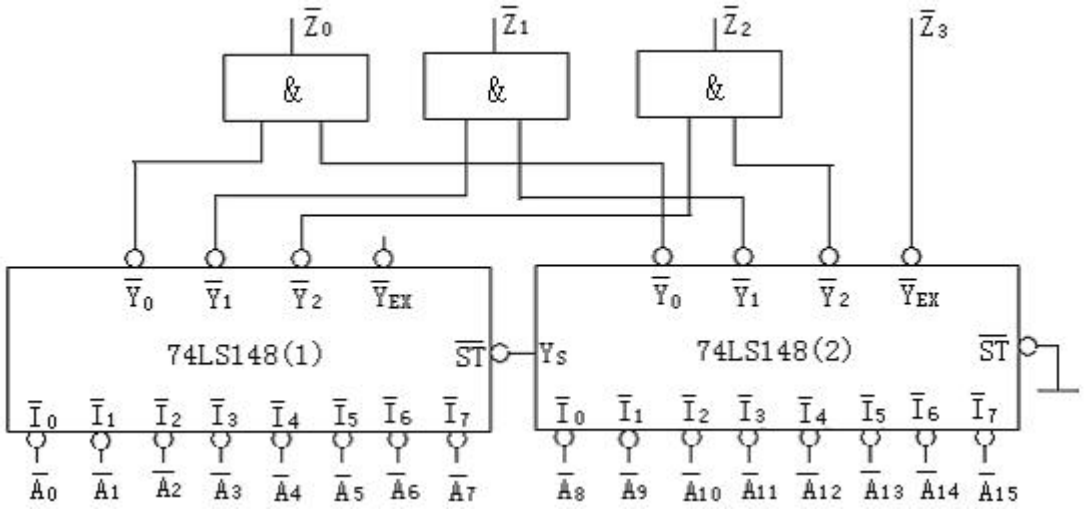


图 4.16　74LS148 的级联应用

二、用二进制优先编码器构成 8421BCD 码编码器

用二进制优先编码器 74LS148 构成的 8421BCD 码编码器电路如图 4.17 所示。当输入端 $\overline{A}_9$ 或 $\overline{A}_8$ 为低电平（9 或 8 有编码请求）时，与非门输出为 1，这使得 74LS148 的 $\overline{ST}$ 为 1，编码器不工作，使得 $\overline{Y}_2\overline{Y}_1\overline{Y}_0=111$。如果这时 $\overline{A}_8$=0，编码器输出 $Z_8Z_4Z_2Z_1$ 为 1000。

如果这时 $\overline{A}_9$=0，编码器输出 $Z_8Z_4Z_2Z_1$ 为 1001。当 $\overline{A}_9$、$\overline{A}_8$ 均为高电平时，与非门输出为 0，编码器处于工作态，74LS148 对输入的 $\overline{A}_7 \sim \overline{A}_0$ 进行编码。这时若 $\overline{A}_5$=0，则 $\overline{Y}_2\overline{Y}_1\overline{Y}_0$=010，编码输出 $Z_8Z_4Z_2Z_1$ 为 0101。既为 5 对应的 8421BCD 码。对其它位的输入可以以此类推。

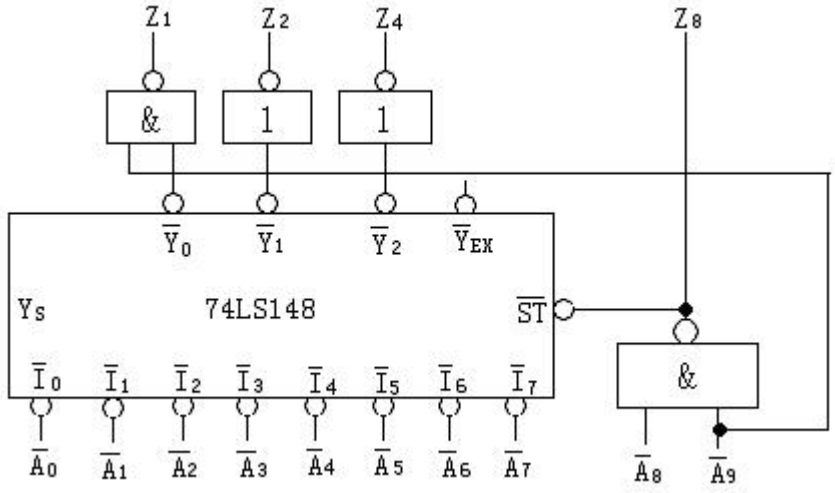


图 4.17　74LS148 构成的 8421BCD 码编码器

4.4.3 译码器

译码是编码的异过程，即译码器是将特定的二进制代码“翻译”成对应的信号。实现译码的电路称为译码器。下面将分别介绍二进制译码器、二—十进制译码器和显示译码器。

一、二进制译码器

把特定的不同二进制代码按其原意翻译成对应的输出信号的电路，叫二进制译码器。也称为全译码器，因为它把输入变量的取值全都翻译了出来。图 4.18 是它的示意框图 A_0、A_1、•••、A_{n-1} 是 n 位二进制代码，Y_0、Y_1 •••、Y_{m-1} 是 m 个输出信号，在二进制译码器中，$m = 2^n$。

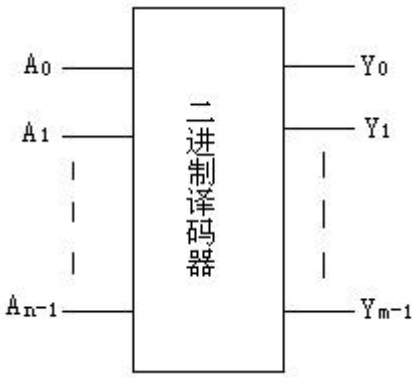


图 4.18 二进制译码器示意框图

二、三位二进制译码器

表 4.10 是 3 位二进制译码器的真值表，输入的是 3 位二进制代码 $A_2A_1A_0$，输出的是其状态译码 Z_7~Z_0。

表 4.10　3 位二进制译码器的真值表

A_2	A_1	A_0	Z_7	Z_6	Z_5	Z_4	Z_3	Z_2	Z_1	Z_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

从表可看出，三个输入 A_2~A_0 的八种组合中的每一种，都唯一地使 Z_0~Z_7 八个输入中的一个为“1”。由此可得该译码器的逻辑表达式为：

$$Z_0 = \overline{A}_2\,\overline{A}_1\,\overline{A}_0 \qquad Z_1 = \overline{A}_2\,\overline{A}_1 A_0$$

$$Z_2 = \overline{A}_2 A_1\,\overline{A}_0 \qquad Z_3 = \overline{A}_2 A_1 A_0$$

$$Z_4 = A_2\,\overline{A}_1\,\overline{A}_0 \qquad Z_5 = A_2\,\overline{A}_1 A_0$$

$$Z_6 = A_2 A_1\,\overline{A}_0 \qquad Z_7 = A_2 A_1 A_0$$

根据真值表和表达式，画出的 3 位二进制译码器如图 4.19 所示。图中每个与门的三个输入，分别接 A_2~A_0 的八种组合之一。对任意一种输入组合，Z_7~Z_0 中仅有一个为“1”。

如果把图 4.19 中的与门换成与非门，同时把输出信号写成反变量，则就得到了输出低电平有效的三位二进制译码器。如图 4.20 所示。

由此得到的逻辑表达式如下：

$$\overline{Z}_0 = \overline{\overline{A}_2\,\overline{A}_1\,\overline{A}_0} \qquad \overline{Z}_1 = \overline{\overline{A}_2\,\overline{A}_1 A_0}$$

$$\overline{Z}_2 = \overline{\overline{A}_2 A_1\,\overline{A}_0} \qquad \overline{Z}_3 = \overline{\overline{A}_2 A_1 A_0}$$

$$\overline{Z}_4 = \overline{A_2\,\overline{A}_1\,\overline{A}_0} \qquad \overline{Z}_5 = \overline{A_2\,\overline{A}_1 A_0}$$

$$\overline{Z}_6 = \overline{A_2 A_1\,\overline{A}_0} \qquad \overline{Z}_7 = \overline{A_2 A_1 A_0}$$

74LS138 是最常用的中规模集成电路的 3 线—8 线译码器，其真值表如表 4.11 所示。图 4.21 给出了 74LS138 的逻辑电路图和符号表示。

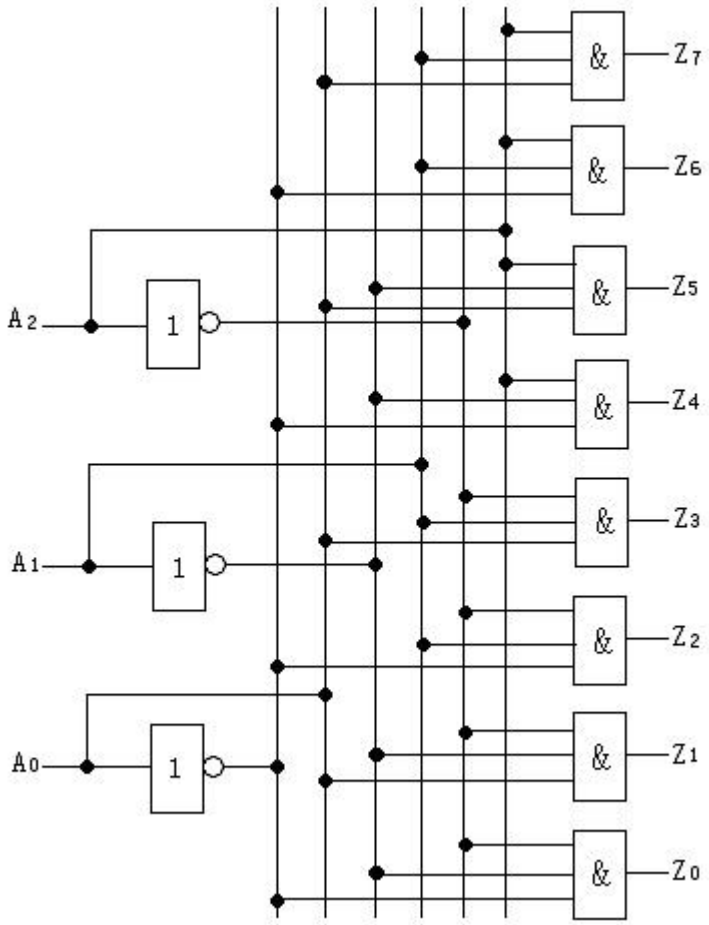


图 4.19 三位二进制译码器电路

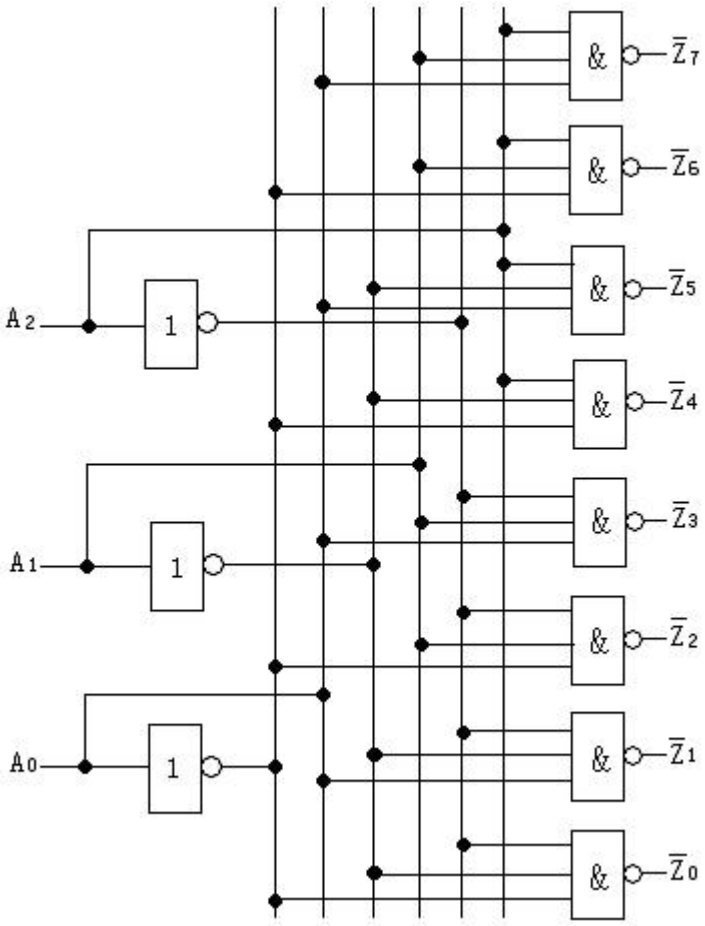


图 4.20 由与非门构成低电平有效的三位二进制译码器电路

表 4.11　74LS138 译码器的真值表

S_1	$\overline{S}_2+\overline{S}_3$	A_2	A_1	A_0	$\overline{Z}_0$	$\overline{Z}_1$	$\overline{Z}_2$	$\overline{Z}_3$	$\overline{Z}_4$	$\overline{Z}_5$	$\overline{Z}_6$	$\overline{Z}_7$
0	d	d	d	d	1	1	1	1	1	1	1	1
d	1	d	d	d	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1	1	1
1	0	1	0	0	1	1	1	1	0	1	1	1
1	0	1	0	1	1	1	1	1	1	0	1	1
1	0	1	1	0	1	1	1	1	1	1	0	1
1	0	1	1	1	1	1	1	1	1	1	1	0

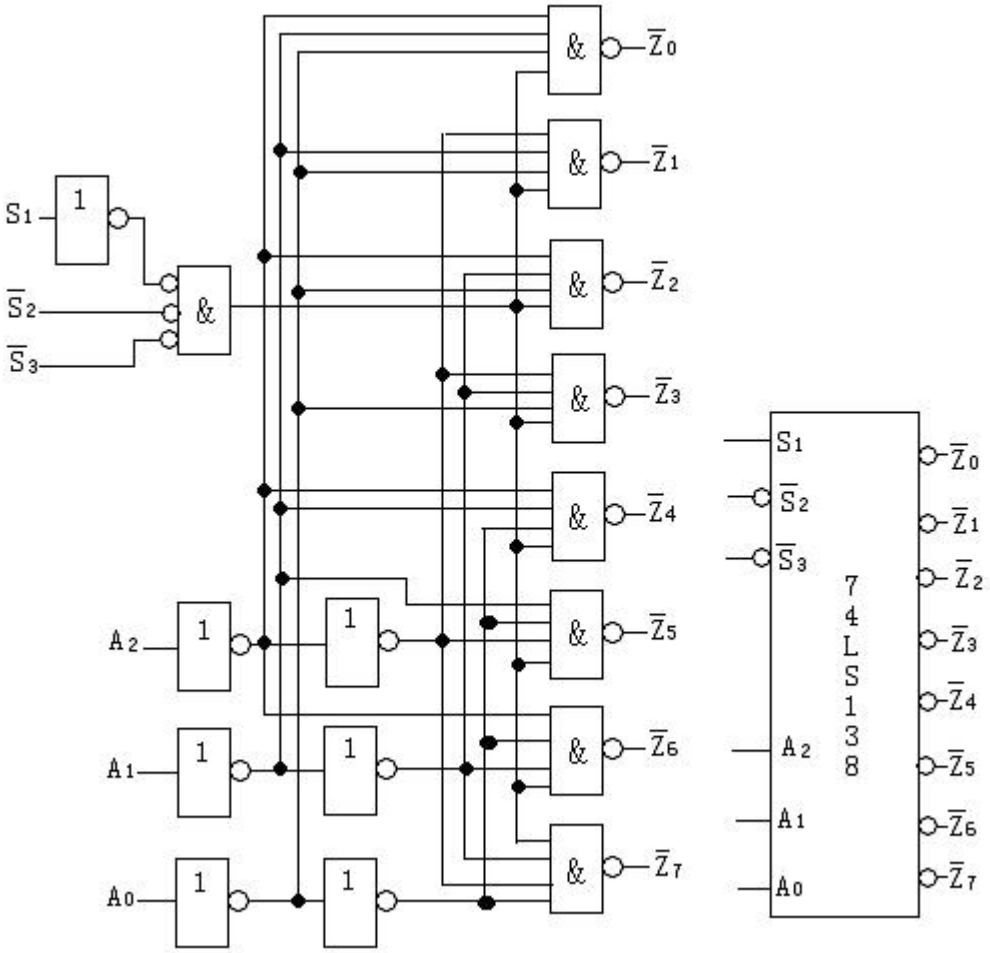


图 4.21　集成 3 线—8 线译码器 74LS138 逻辑电路图和符号

从逻辑图上可见，74LS138 和低电平有效的三位二进制译码器相同。但 74LS138 具有使能（Enable）控制端 S_1、$\overline{S}_2$、和$\overline{S}_3$，它们的组合用于控制译码器的“选通”和“禁止”。从逻辑图可以看出，$EN = S_1 \cdot \overline{\overline{S}_2} \cdot \overline{\overline{S}_3} = S_1 \cdot \overline{\overline{S}_2 + \overline{S}_3}$ 连接到所有与非门的一个输入端上，仅当 S_1=1 并且 $\overline{S}_2 = \overline{S}_3 = 0$ 时，EN 才为“1”，所有与非门开启，译码器被选通，处于工作状态，由输入 A2~A0 来确定 $\overline{Z}_7 \sim \overline{Z}_0$ 的状态。否则，EN=“0”，所有与非门的输出均为“1”，译码器处于“禁止”状态。

译码器工作时，若能使 EN 端在输入信号变化前为“0”，输入信号稳定后为“1”，则可以消除由于输入信号变化过程中的时延引起的险象干扰。

设置三个控制端的原因，除了更灵活、有效地控制译码器的工作状态外，还可以利用它们实现译码器的扩展。图 4.27 为由两片 74LS138 译码器芯片扩展而成的 4 线—16 线译码器连接图。当高位 A_3=0 时，片（1）的 $\overline{S}_2 = 0$ 处于工作态，片（2）的 S_1=0 被禁止，输出 $\overline{Z}_7 \sim \overline{Z}_0$ 是 $0A_2A_1A_0$ 的译码；当 A_3=1 时，片（1）的 $\overline{S}_2 = 1$ 被禁止，片（2）的 S_1=1 处于工作态，

输出 $\overline{Z}_7 \sim \overline{Z}_0$ 是 $1A_2A_1A_0$ 的译码。整个级连电路的使能端是 $\overline{S}$，当 $\overline{S}$=0 时级连电路工作，完成对输入 4 位二进制代码 $A_3A_2A_1A_0$ 的译码；当 $\overline{S}$=1 时级连电路被禁止，输出 $\overline{Z}_{15} \sim \overline{Z}_0$ 均为 1 状态。

由以上电路分析可知，低电平译码输出有效的每一个译码输出端都是一个最大项，因此，这种译码器是一个最大项发生器。而高电平译码输出有效的译码器的每一个译码输出端都是一个最小项，因此这种译码器是一个最小项发生器。译码器的这种特性，使得它可以用来实现任意的组合逻辑函数。

三、二—十进制译码器

1、8421BCD 码输入的 4 线—10 线译码器

将十进制数的二进制编码即 BCD 码翻译成对应的十个输出信号的电路，称为二—十进制译码器。因为 BCD 码一般都是由 4 位二进制代码组成，形成 4 个输入信号，故又把二—十进制译码器叫做 4 线—10 线译码器。二—十进制译码器的译码真值表如表 4.12 所示。

表 4.12　4 线—10 线译码器的真值表

A_3	A_2	A_1	A_0	Z_0	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	Z_7	Z_8	Z_9
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1
1	0	1	0	d	d	d	d	d	d	d	d	d	d
1	0	1	1	d	d	d	d	d	d	d	d	d	d
1	1	0	0	d	d	d	d	d	d	d	d	d	d
1	1	0	1	d	d	d	d	d	d	d	d	d	d
1	1	1	0	d	d	d	d	d	d	d	d	d	d
1	1	1	1	d	d	d	d	d	d	d	d	d	d

利用卡诺图对以上真值表进行化简，便可得二—十进制译码器的输出表达式。

$$Z_0 = \overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0 \qquad Z_1 = \overline{A}_3\overline{A}_2\overline{A}_1A_0$$

$$Z_2 = \overline{A}_2A_1\overline{A}_0 \qquad Z_3 = \overline{A}_2A_1A_0$$

$$Z_4 = A_2\overline{A}_1\overline{A}_0 \qquad Z_5 = A_2\overline{A}_1A_0$$

$$Z_6 = A_2A_1\overline{A}_0 \qquad Z_7 = A_2A_1A_0$$

$$Z_8 = A_3\overline{A}_0 \qquad\qquad Z_9 = A_3 A_0$$

根据以上表达式可得如图 4.22 所示的二—十进制译码器电路.

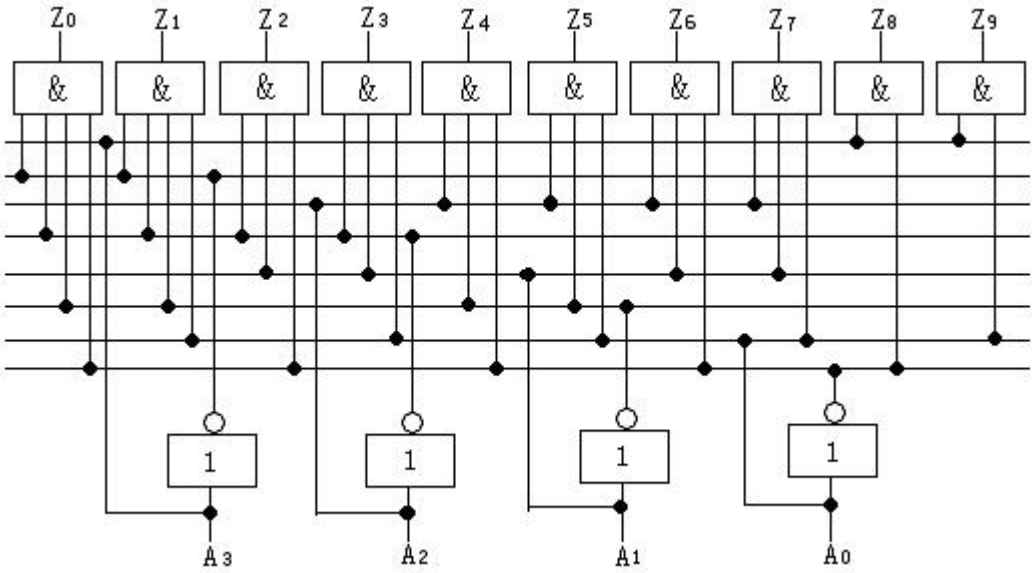


图 4.22 二—十进制译码器电路

2、集成 4 线—10 线译码器

74LS42 是一种集成 4 线—10 线译码器。因为 $m<2^n$，所以它是属于部分译码器。74LS42 译码器的逻辑符号以及真值表如图 4.23 和表 4.13 所示。

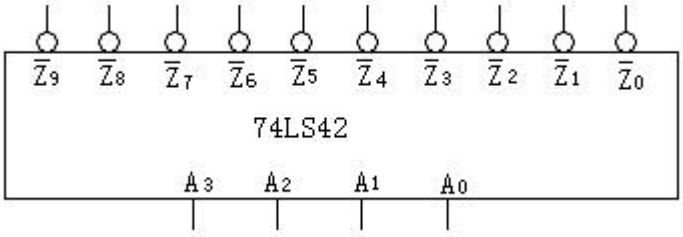


图 4.23 集成 4 线—10 线译码器 74LS42

从真值表可见，74LS42 译码器也是输出低电平有效，因此，它的每一个译码输出端都是一个最大项。此外，当输入为禁用码组 1010~1111 时，74LS42 译码器不会产生错误输出，这被称为拒绝伪输入译码。

四、数字显示译码器

在数字系统中，常常需要将数字、字母、符号的二进制代码翻译成人们习惯的形式，直观地显示出来，供人们读取或监视系统的工作情况。数字显示译码器还能够驱动发光二极管（LED)、荧光数码管、液晶数码管、气体放电管等显示器件将其直观地显示出来。在介绍数字显示译码器之前，首先简单介绍一下目前广泛使用的发光二极管 LED 构成的 7 段显示数码管的工作原理。

表 4.13　集成 4 线—10 线译码器 74LS42 真值表

A_3	A_2	A_1	A_0	$\overline{Z}_0$	$\overline{Z}_1$	$\overline{Z}_2$	$\overline{Z}_3$	$\overline{Z}_4$	$\overline{Z}_5$	$\overline{Z}_6$	$\overline{Z}_7$	$\overline{Z}_8$	$\overline{Z}_9$
0	0	0	0	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1	1	1
0	1	1	0	1	1	1	1	1	1	0	1	1	1
0	1	1	1	1	1	1	1	1	1	1	0	1	1
1	0	0	0	1	1	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0
1	0	1	0	1	1	1	1	1	1	1	1	1	1
1	0	1	1	1	1	1	1	1	1	1	1	1	1
1	1	0	0	1	1	1	1	1	1	1	1	1	1
1	1	0	1	1	1	1	1	1	1	1	1	1	1
1	1	1	0	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1

发光二极管是一种半导体显示器件，其基本结构是由磷化镓、砷化镓或磷砷化镓等材料构成的 PN 结。当 PN 结外加正向电压时，P 区的多数载流子空穴向 N 区扩散，N 区的多数载流子向 P 区扩散，当电子和空穴复合时就会释放能量，并发出一定波长的光。

单个 PN 结可以封装成发光二极管，多个 PN 结按分段式或点阵式可以构成半导体数码管。目前常用的有 7 段显示数码管。它是将 7 个发光二极管按一定的方式连接在一起，每段为一个发光二极管，7 段分别为 a、b、c、d、e、f、g，显示哪个字型，则对应段的发光二极管就发光。7 段显示数码管的结构如图 4.24（a）所示，图中的 dp 是小数点。

7 段数码显示器分为共阴极和共阳极两种。所谓共阴是指数码管 7 个发光二极管的阴极都连在一起，接到地。发光二极管的阳极经过限流电阻接到 7 段译码器相应的输出端（译码器输出应为高电平有效）；所谓共阳是指数码管 7 个发光二极管的阳极都连在一起，接到 Vcc，而发光二极管的阴极经过限流电阻接到 7 段译码器相应的输出端（译码器输出应为低电平有效）。改变限流电阻可改变发光二极管的亮度。共阴和共阳数码管的结构如图 4.24（b）和 4.24（c）所示。

7 段显示数码管的驱动信号 a~g 来自 4 线—7 线译码器。常用的中规模 4 线—7 线显示译码器有 74LS46~74LS49。它们的使用特性大致相同。以 74LS48 为例，它驱动的是共阴连接的数码管。具有集电极开路输出结构，并接有 2KΩ的上拉电阻。它将 8421BCD 码译成 a,b,c,d,e,f,g7 段输出并进行驱动，它同时还具有消隐和试灯的辅助功能。表 4.14 是 74LS48 的逻辑功能表，它有 4 个输入信号 A_3~ A_0，对应四位 8421BCD 码；有 7 个输出 a~ g，对应 7 段字形。当控制信号有效时，A_3~ A_0 输入一组 8421BCD 码，a~ g 输出端便有相应的输出，电路实现正常的译码。译码输出为 1 的端口，对应数码管的字段就点亮。例如当 $A_3A_2A_1A_0$=0101 时，只有 b 和 e 输出为 0，其余的都输出为 1，a 段、c 段、d 段、f 段、g 段点亮，显示数字“5”。图 4.25 是译码显示系统图。它由译码器 74LS48 和共阴极数码管 BS201A 组成。

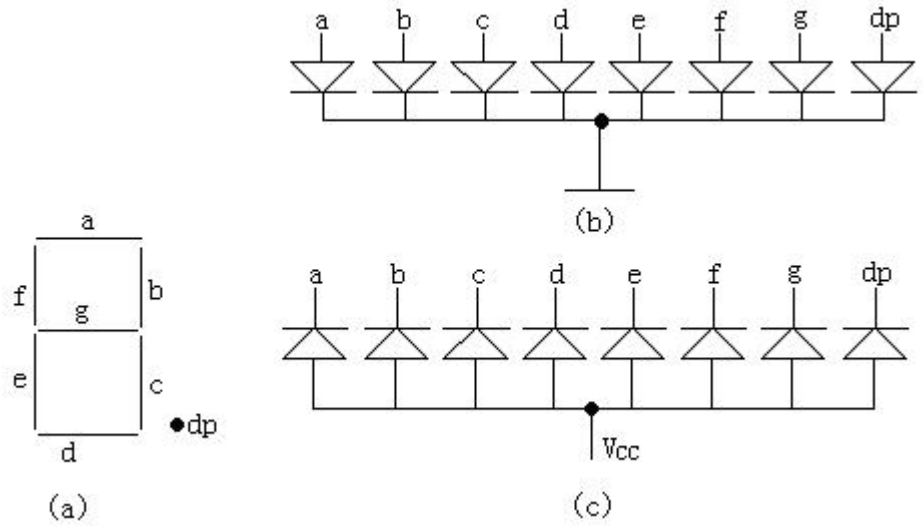


图 4.24　7 段显示数码管结构

表 4.14　74LS48 逻辑功能表

十进制	输入						入/出	输出						
或功能	$\overline{LT}$	$\overline{RBI}$	A_3	A_2	A_1	A_0	$\overline{BI}/\overline{RBO}$	a	b	c	d	e	f	g
0	1	1	0	0	0	0	1	1	1	1	1	1	1	0
1	1	d	0	0	0	1	1	0	1	1	0	0	0	0
2	1	d	0	0	1	0	1	1	1	0	1	1	0	1
3	1	d	0	0	1	1	1	1	1	1	1	0	0	1
4	1	d	0	1	0	0	1	0	1	1	0	0	1	1
5	1	d	0	1	0	1	1	1	0	1	1	0	1	1
6	1	d	0	1	1	0	1	0	0	1	1	1	1	1
7	1	d	0	1	1	1	1	1	1	1	0	0	0	0
8	1	d	1	0	0	0	1	1	1	1	1	1	1	1
9	1	d	1	0	0	1	1	1	1	1	0	0	1	1
10	1	d	1	0	1	0	1	0	0	0	1	1	0	1
11	1	d	1	0	1	1	1	0	0	1	1	0	0	1
12	1	d	1	1	0	0	1	0	1	0	0	0	1	1
13	1	d	1	1	0	1	1	1	0	0	1	0	1	1
14	1	d	1	1	1	0	1	0	0	0	1	1	1	1
15	1	d	1	1	1	1	1	0	0	0	0	0	0	0
灭灯	d	d	d	d	d	d	0	0	0	0	0	0	0	0
灭零	1	0	0	0	0	0	0	0	0	0	0	0	0	0
试灯	0	d	d	d	d	d	1	1	1	1	1	1	1	1

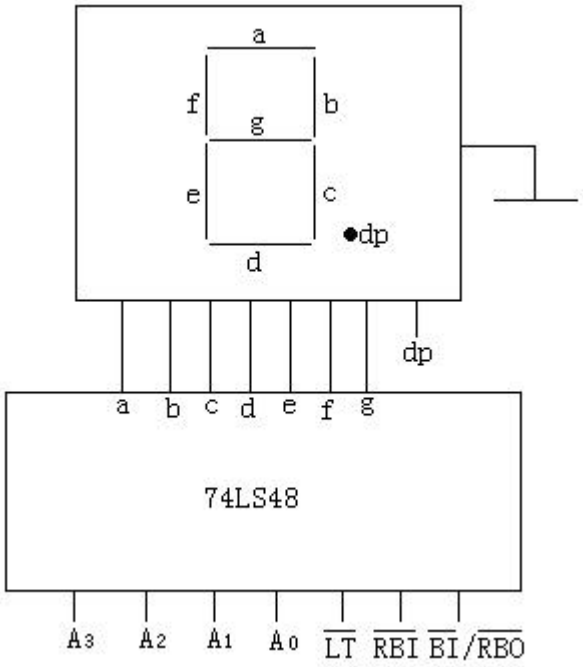


图 4.25　7 段数字显示系统框图

输入信号 $\overline{BI}$ 为熄灭信号。当 $\overline{BI}$ =0 时，无论 $\overline{LT}$ 和 $\overline{RBI}$ 以及数码输入 A_3~ A_0 状态如何，输出 Y_a~ Y_g 均为 0，7 段都处于熄灭状态。不显示数字。

输入信号 $\overline{LT}$ 为试灯信号，用来检查 7 段是否能正常显示。当 $\overline{BI}$ =1， $\overline{LT}$ =0 时，无论 A_3~ A_0 状态如何，输出 Y_a~ Y_g 均为 1，使显示器 7 段都点亮。

输入信号 $\overline{RBI}$ 为灭“0”信号，用来熄灭数码管显示的 0。当 $\overline{LT}$ =1， $\overline{RBI}$ =0 时，只有输入 A_3~ A_0=0000 时，输出 Y_a~ Y_g 均为 0，7 段都熄灭，不显示数字 0。但当输入 A_3~ A_0 为其它组合时，则可以正常显示。

电路输出 $\overline{RBO}$ 为灭 0 输出信号。当 $\overline{LT}$ =1， $\overline{RBI}$ =0，且 A_3~ A_0=0000 时，本片灭 0，同时输出 $\overline{RBO}$ =0。在多级译码显示系统中，这个 0 送到另一片译码器的 $\overline{RBI}$ 端，就可以使对应这两片译码器的数码管此刻的 0 都不显示。由于熄灭信号 $\overline{BI}$ 和灭 0 输出信号 $\overline{RBO}$ 是电路的同一点，共用一条引出线，故标为 $\overline{BI}$ / $\overline{RBO}$ 。

4.4.4 译码器的应用

一、在计算机系统中用作地址译码器

在计算机系统中，各种外部设备和接口电路如存储器、A/D 转换器、D/A 转换器、并行输入/输出接口、键盘、打印机等等都是通过地址总线 AB、数据总线 DB、和控制总线 CB 与 CPU 进行数据交换的。当 CPU 需要向某一设备或接口传送数据时，首先要选中该设备或接口。这可以通过简单的线选方式来实现，既用一根高位的地址线来选中该设备或接口，但当外扩的设备和接口较多时，高位的地址线就不够用了。这时就可以利用译码器来扩充地址线。如利用 74LS138 三线—八线译码器就可以将 3 条高位地址线扩充成 8 条地址线以实行对 8

个外部设备或接口的选通。译码器在计算机系统扩展中的应用框图如图 4.26 所示。

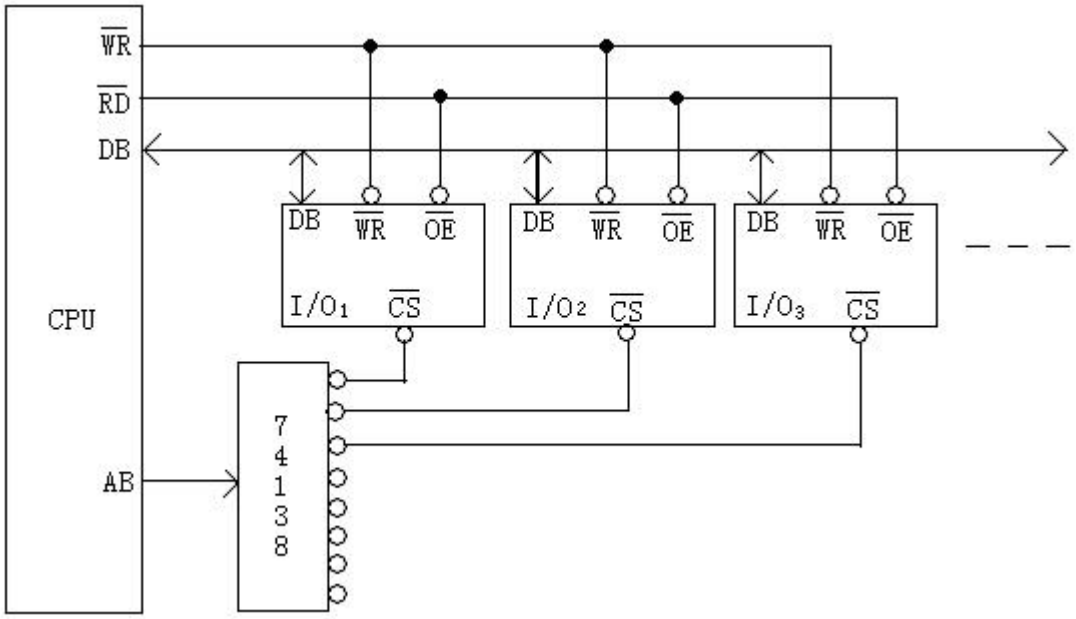


图 4.26 译码器在计算机系统扩展中的应用

二、译码器的扩展

当译码器的容量不能满足实际工作需要时，利用其的使能控制端，可以对其进行扩展。图 4.27 是用两片 74LS138 级连起来构成的 4 线—16 线译码器。

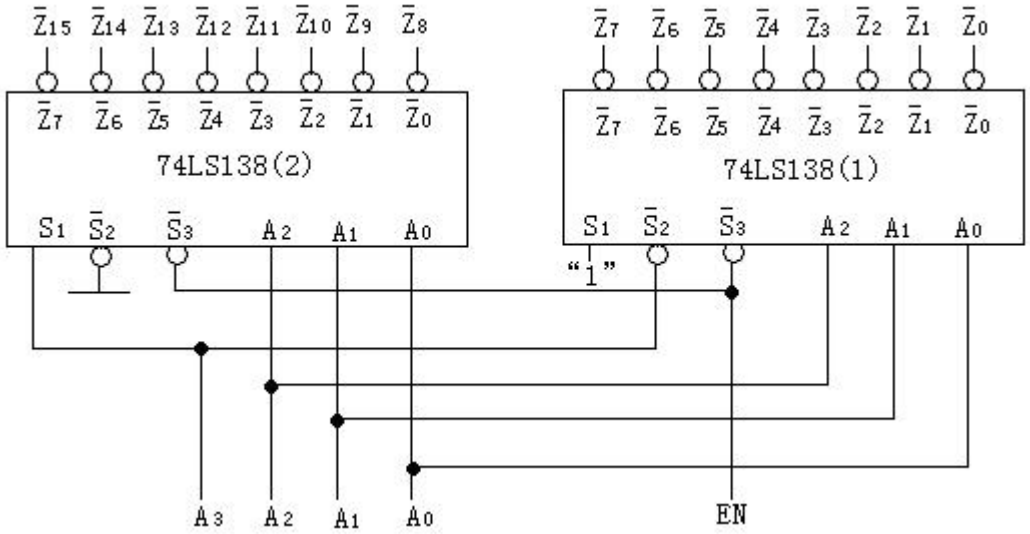


图 4.27　用 74LS138 构成的 4 线—16 线译码器

三、用译码器实现组合逻辑电路

前面我们已知道译码输出高电平有效的二进制译码器是一个最小项发生器，它的每一个译码输出端都是一个最小项。即：

$$Z_i = m_i = \overline{M}_i$$

译码输出低电平有效的二进制译码器是一个最大项发生器，它的每一个译码输出端都是一个最大项。即：

$$\overline{Z}_i = M_i = \overline{m_i}$$

而任一个逻辑函数表达式，都可以写成最小项之和或最大项之积。因此，用译码器可实现任何组合逻辑函数表达式。特别在实现多输出逻辑函数时，更显得方便。由此可知，对用最小项表示的逻辑函数，既可用输出高电平有效的译码器外加或门来实现，也可用输出低电平有效的译码器外加与非门来实现。而对用最大项表示的逻辑函数，既可用输出低电平有效的译码器外加与门来实现，也可用输出高电平有效的译码器外加或非门来实现。

例 4.1：用输出低电平有效的 3 线—8 线译码器实现逻辑函数 $F(A,B,C)=\sum m(0,1,3,7)$

解：$F(A,B,C)=\overline{\overline{m_0+m_1+m_3+m_7}}=\overline{\overline{m}_0\overline{m}_1\overline{m}_3\overline{m}_7}=\overline{\overline{Z}_0\overline{Z}_1\overline{Z}_3\overline{Z}_7}$

实现的电路如图 4.28 所示。

例 4.2：用输出高电平有效的 3 线—8 线译码器实现逻辑函数 $F(A,B,C)=\sum m(0,1,3,7)$

解：$F(A,B,C)=m_0+m_1+m_3+m_7=Z_0+Z_1+Z_3+Z_7$

实现的电路如图 4.29 所示。

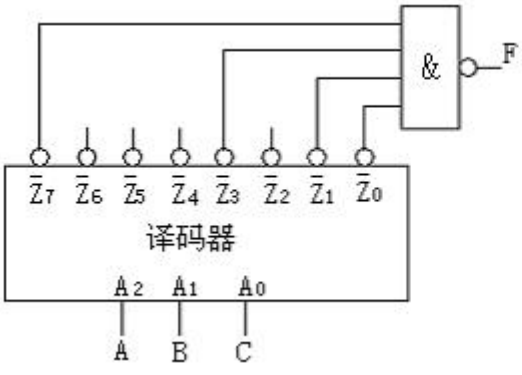


图 4.28 例 4.11 电路

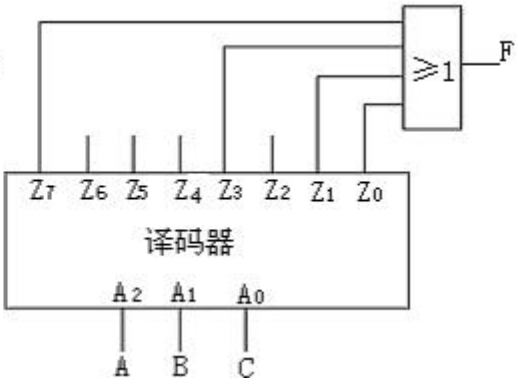


图 4.29 例 4.12 电路

例 4.3：试用 74LS138 设计一个一位二进制全加器。

解：74LS138 是输出低电平有效的译码器，而一位全加器的真值表如表 4.2 所示。从真值表可得一位全加器的和 S_i 及进位位 C_i 的最小项表达式为：

$$S_i(A_i,B_i,C_i)=\sum m(1,2,4,7)=\overline{\overline{Y}_1\overline{Y}_2\overline{Y}_4\overline{Y}_7}$$

$$C_i(A_i,B_i,C_i)=\sum m(3,5,6,7)=\overline{\overline{Y}_3\overline{Y}_5\overline{Y}_6\overline{Y}_7}$$

实现的电路如图 4.30 所示。

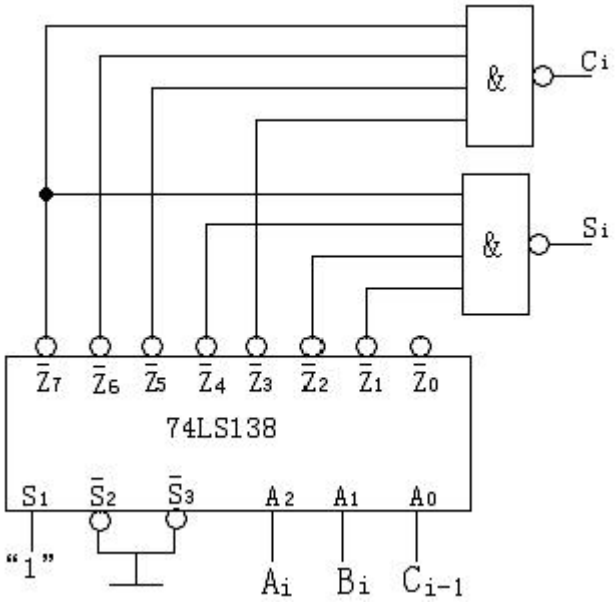


图 4.30　一位全加器电路

4.5 数据选择器和数据分配器

4.5.1 数据选择器

一、数据选择器的逻辑功能

数据选择器的逻辑功能是根据地址选择端的控制，从多路输入数据中选择一路数据输出。因此，它可实现时分多路传输电路中发送端电子开关的功能，故又称为复用器（Multiplexer），并用 MUX 来表示。

四选一数据选择器的真值表如表 4.22 所示，其中，D0、D1、D2、D3 是四路数据输入，A1、A0 为地址选择输入，Z 为数据选择器的输出，其逻辑符号如图 4.31 所示。根据表 4.15 的真值表可写出它的输出函数表达式为：

$$Z = \overline{A}_1 \overline{A}_0 D_0 + \overline{A}_1 A_0 D_1 + A_1 \overline{A}_0 D_2 + A_1 A_0 D_3$$

根据以上表达式，可画出如图 4.31 所示的逻辑电路图。

表 4.15　真值表

A_1　A_0	Z
0　0	D_0
0　1	D_1
1　0	D_2
1　1	D_3

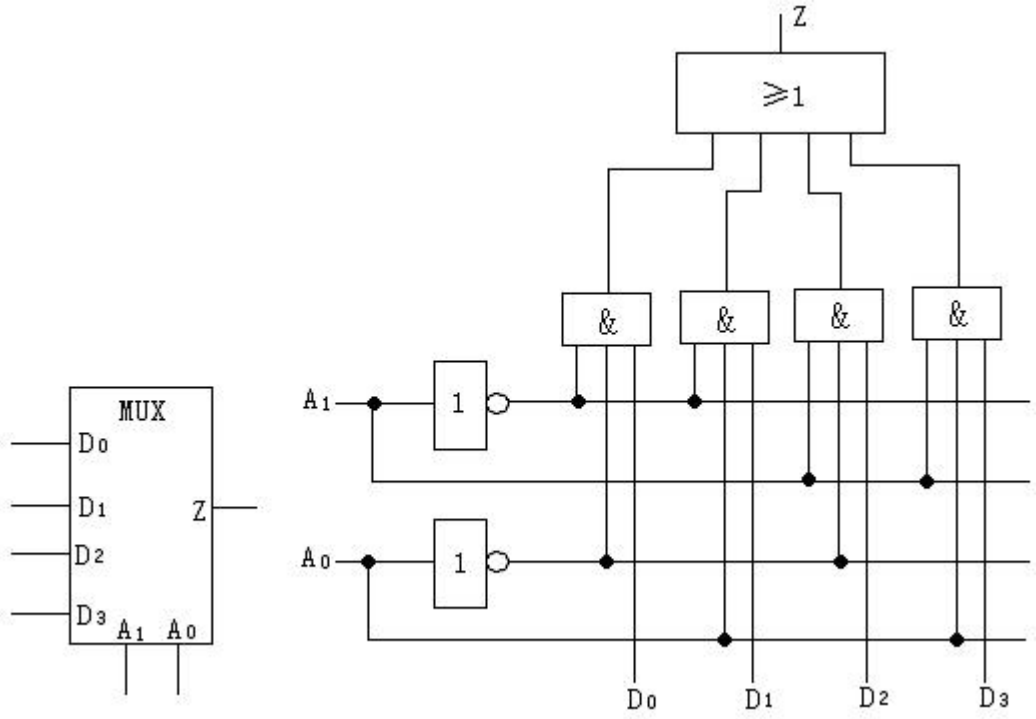


图 4.31 四选一数据选择器逻辑符号和逻辑电路图

二、集成数据选择器

集成数据选择器的规格和品种很多，这里以 74LS151 八选一数据选择器为例进行讨论。74LS151 的逻辑符号如图 4.32 所示，真值表如表 4.16 所示。

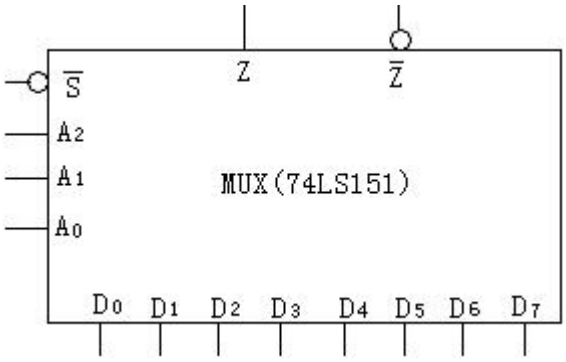


图 4.32 74LS151 数据选择器的逻辑符号

当使能端 $\overline{S}$=1 时，数据选择器输出与任何输入数据无关。使能端 $\overline{S}$=0 时，输出 Y 与输入数据 D_0~D_7 的逻辑关系为：

$$Y = \overline{A}_2\overline{A}_1\overline{A}_0D_0 + \overline{A}_2\overline{A}_1A_0D_1 + \overline{A}_2A_1\overline{A}_0D_2 + \overline{A}_2A_1A_0D_3 + A_2\overline{A}_1\overline{A}_0D_4 + A_2\overline{A}_1A_0D_5$$

$$+ A_2A_1\overline{A}_0D_6 + A_2A_1A_0D_7$$

表 4.16　74LS151 数据选择器真值表

$\overline{S}$	A_2	A_1	A_0	Z
1	d	d	d	0
0	0	0	0	D_0
0	0	0	1	D_1
0	0	1	0	D_2
0	0	1	1	D_3
0	1	0	0	D_4
0	1	0	1	D_5
0	1	1	0	D_6
0	1	1	1	D_7

若将 $A_2A_1A_0$ 看成三位逻辑变量，则以上逻辑表达式可写为：

$$Y = m_0D_0 + m_1D_1 + m_2D_2 + m_3D_3 + m_4D_4 + m_5D_5 + m_6D_6 + m_7D_7 = \sum_{i=0}^{7}(m_i \cdot D_i)$$

由数据选择器的逻辑表达式可以看出，当地址选择输入使某一个最小项 m_i 为“1”时，数据选择器的输出 Y 便为对应的输入数据 D_i，由此便实现了数据选择的功能。

常用的数据选择器还有双四选一数据选择器 74LS153、十六选一数据选择器 74LS150 等等。由上可知，若地址选择输入端有 n 位，便可实现 2^n 路数据的选择，即 2^n 选一，其输出为：

$$Y = \sum_{i=0}^{2^n-1}(m_i \cdot D_i)$$

数据选择器也可作函数发生器使用。数据选择器的输出表达式 $Y = \sum_{i=0}^{2^n-1}(m_i \cdot D_i)$ 本身就表示了一个与或函数，只要将适当的数据或变量赋给地址选择输入端和数据输入端，就可以实现特定的函数。

4.5.2 数据选择器的应用

一、用作多路数据选择

数据选择器的基本功能就是从多路输入的数据中选择一路输出。故数据选择器可用作多路

数据开关，实现多路数据通讯和路由选择。

二、数据选择器的扩展

利用数据选择器的选通控制端很容易实现数据选择器的扩展。例如用两片 74LS151 可扩展为十六选一数据选择器。扩展电路如图 4.33 所示。

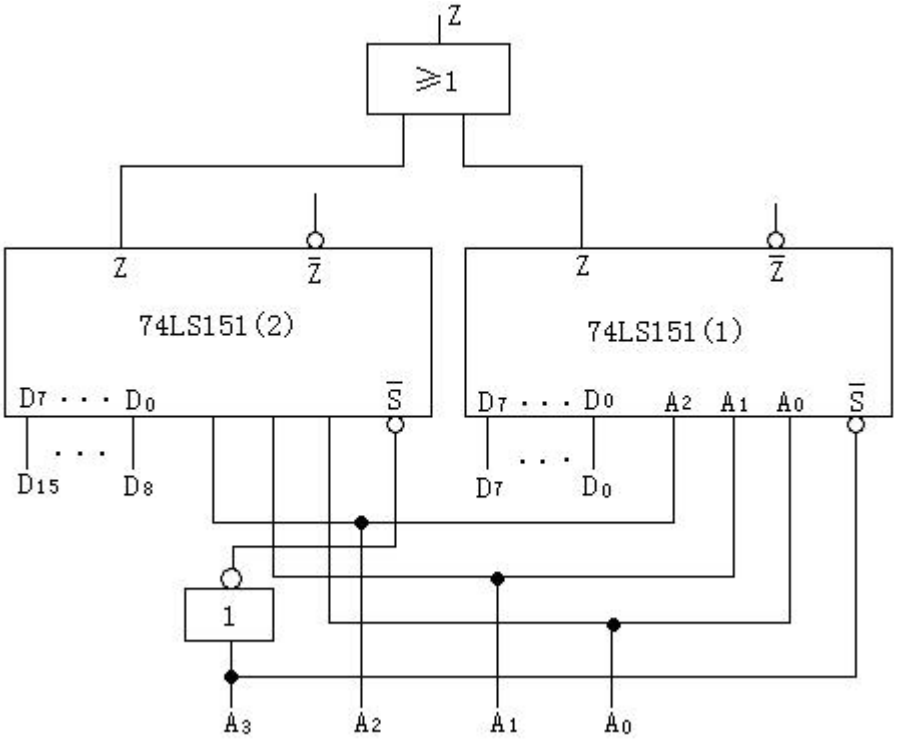


图 4.33 用 74LS151 扩展的十六选一数据选择器

三、实现组合逻辑函数

从输出表达式 $Y=\sum_{i=0}^{2^{n-1}}(m_i\cdot D_i)$ 可知，它是关于地址选择码的全部最小项和对应的各路输入数据的与或表达式。而任何组合逻辑函数都可表示为标准与或表达式，故数据选择器可用来实现任意的组合逻辑函数。

用数据选择器实现组合逻辑函数有两种方法，一种是真值表法，另一种是卡诺图法。

1、真值表法

所谓真值表法就是将要实现的逻辑函数用真值表的方法列出输入和输出之间的关系。然后用合适的数据选择器实现之。

例 4.4：利用八选一数据选择器实现下列逻辑函数。

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

根据该表达式列的真值表如表 4.17 所示。可见将真值表中的输入变量作为数据选择器的地址输入，而将真值表中的输出数据作为数据选择器的数据输入，则该选择器便实现了该逻辑函数。电路图如图 4.34 所示。

表 4.17 例 4.4 的真值表

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

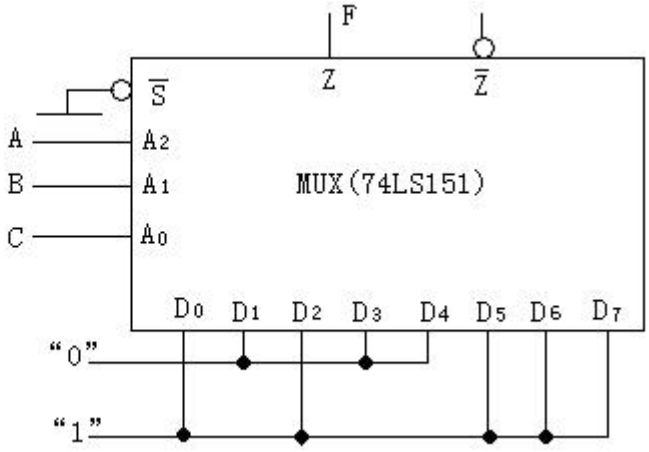


图 4.34 例 4.4 的电路

例 4.5：试用四选一数据选择器实现下列逻辑函数。

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

从表 4.17 可见，若选 A、B 作四选一数据选择器的地址选择码，则可通过比较变量 C 和输出 F 的关系可得如图 4.35 的电路图。

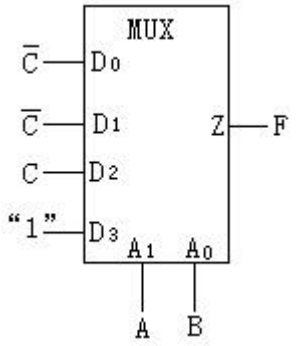


图 4.35 例 4.5 的电路

2、卡诺图法

所谓卡诺图法，就是利用卡诺图来确定数据选择器的地址选择变量和数据输入变量，最后得出实现的电路。下面介绍这种方法的实现步骤。

首先，将卡诺图画成与数据选择器相适应的形式。也就是说，所使用的数据选择器有几个地址选择码输入端，逻辑函数的卡诺图的某一边就应该有几个变量，且就将这几个变量作为数据选择器的地址选择码。

其次，将要实现的逻辑函数填入卡诺图并画卡诺圈。因为数据选择器输出函数是与或型表达式且包含地址选择码的全部最小项，故在画圈时不仅要圈最小项和随意项，而且还只能顺着地址选择码的方向圈，保证地址选择变量不被化简掉。

然后是读出所圈的结果。注意，地址选择码不读出，只读出其它变量的化简结果，这些结果就是地址选择码所选择的数据输入值。

最后，根据地址选择码和数据输入值，画出用数据选择器实现的逻辑电路。需要注意的是当读出的数据输入 D 的表达式包含两个或两个以上变量时，需要在数据选择器的基础上外

加门电路才行。

例 4.6：试用四选一数据选择器实现下列逻辑函数。

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

根据该函数画出的卡诺图和实现的电路如图 4.36 所示。图中 A、B 作为地址选择码，由此可得到的函数式为：

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

即 $D_0=\overline{C}$　　$D_1=1$　　$D_2=\overline{C}$　　$D_3=C$

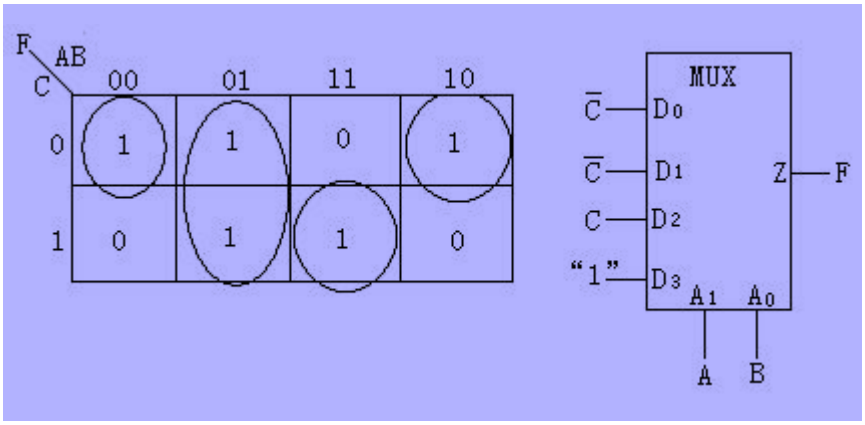


图 4.36 例 4.6 卡诺图和实现的电路

例 4.7：试用四选一数据选择器实现下列逻辑函数。

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

根据该函数画出的卡诺图和实现的电路如图 4.37 所示。图中 A、B 作为地址选择码，由此得到的函数式为：

$$F(A,B,C,D)=\overline{A}\cdot\overline{B}\cdot(C+D)+\overline{A}B\cdot\overline{C}+AB\cdot D+A\overline{B}\cdot 1$$

即 $D_0=C+D$　　$D_1=\overline{C}$　　$D_2=1$　　$D_3=D$

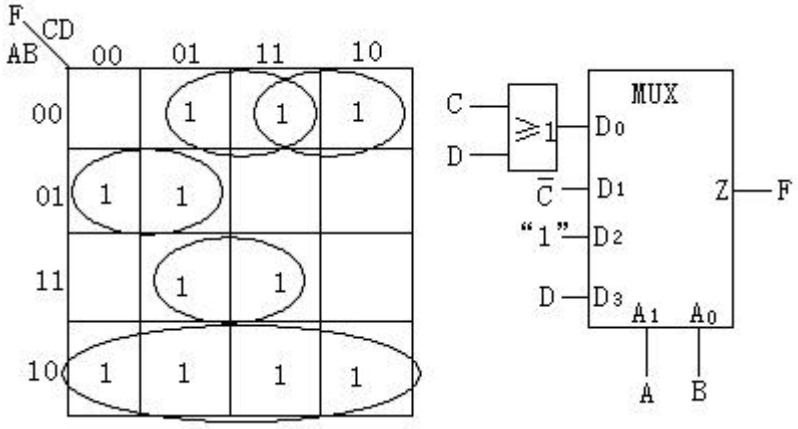


图 4.37 例 4.7 卡诺图和实现的电路

4.5.3 数据分配器

一、数据分配器的功能

数据分配器的逻辑功能是将一路输入数据根据地址选择码分配给多路数据输出中的某一路输出。其逻辑功能正好和数据选择器相反。它实现的是时分多路传输电路中接收端电子开关的功能。故又称为解复器（Demultiplexer），并用 DMUX 来表示。下面以四路数据分配器为例进行讨论。

四路数据分配器的真值表和逻辑符号如表 4.18 和图 4.38 所示，其中 D 为一路数据输入，Z_0~Z_3 为四路数据输出，A_1、A_0 为地址选择码输入端。其输出函数表达式为：

$$Z_0 = \overline{A}_1 \overline{A}_0 \cdot D \qquad Z_1 = \overline{A}_1 A_0 \cdot D \qquad Z_2 = A_1 \overline{A}_0 \cdot D \qquad Z_3 = A_1 A_0 \cdot D$$

从表达式可见，可以用逻辑门实现四路数据选择器的功能。

表 4.18 四路数据分配器的真值表

输入			输出			
D	A_1	A_0	Z_0	Z_1	Z_2	Z_3
0	0	0	D	0	0	0
0	1	0	0	D	0	0
1	0	0	0	0	D	0
1	1	0	0	0	0	D

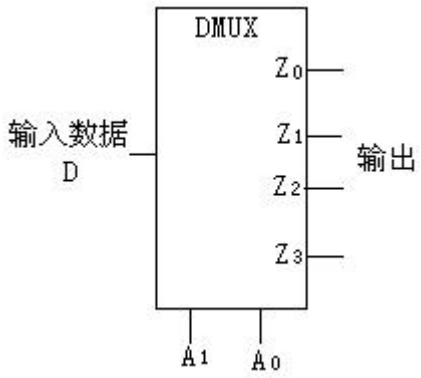


图 4.38 四路数据分配器的逻辑符号

二、用通用二进制译码器实现数据分配器

从数据分配器的真值表和函数表达式可看出，译码器可实现数据分配的功能。故工程上都是将通用二进制译码器作为数据分配器使用。

例 4.8：用译码器 74LS138 实现八路数据分配器。

用 74LS138 实现数据八路数据分配的电路如图 4.39 所示。下面以 D_0 为例说明输出的确定方法。根据数据分配的定义，当 $A_2A_1A_0$=000 时与 D 一致的输出是 D_0。现在 $A_2A_1A_0$=000 且 D=1 时，$\overline{Z}_0 = 1$；若 D=0 时，$\overline{Z}_0 = 0$。可见，$\overline{Z}_0$ 与 D 一致。

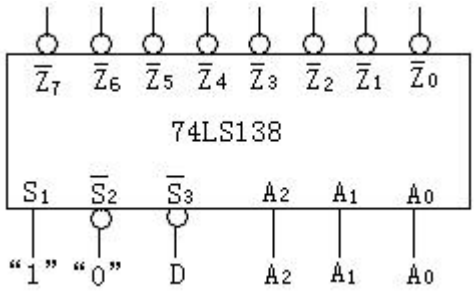


图 4.39 用 74LS138 构成八路数据分配器

4.6 组合逻辑电路的分析

本节主要讨论组合逻辑的分析方法,虽然目前中大规模集成电路发展较快,人们也已不再单纯地用门电路来组成复杂的数字系统,但是门电路是构成中大规模集成电路的基础,而且门电路作为一种基本的逻辑元件,仍是各种数字系统中不可缺少的组成部分。因此,详细讨论由门电路组成的组合逻辑电路仍然是很有必要的。

4.6.1 组合逻辑电路的分析步骤

组合逻辑电路分析的基本目的,就是要通过分析,导出给定电路输出变量与输入变量之间的逻辑关系,并确定电路的逻辑功能。对于逻辑门构成的组合逻辑电路,其分析过程通常包含以下几个步骤:

(l)根据给定的逻辑图,写出输出函数的逻辑表达式。

(2)根据已写出的输出函数逻辑表达式,列出真值表。

(3)根据逻辑表达式或真值表,判断电路的逻辑功能。

在第一步中根据所给电路写出的函数表达式可能并不是最简表达式，为了便于分析，通常要对逻辑函数进行化简。

以上是组合逻辑电路的一般分析步骤，在实际工作过程中，有时对一些复杂的电路还需借助分析者的实际工作经验。所以只有多接触实际，见多识广之后，分析起来才能得心应手。

4.6.2 组合逻辑电路的分析举例

例 4.9 试分析图 4.40 所示电路。

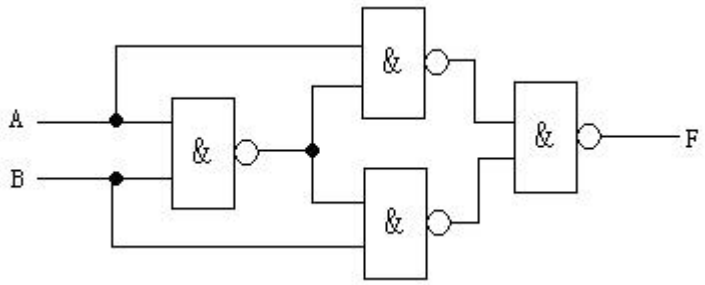


图 4.40 例 4.9 的逻辑电路图

解：该电路有两个输入变量 A、B 和一个输出变量 F。F 的逻辑表达式为

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

进行化简得

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

由化简后的表达式可见，该电路实现异或门的功能。

例 4.10 试分析图 4.41 所示电路

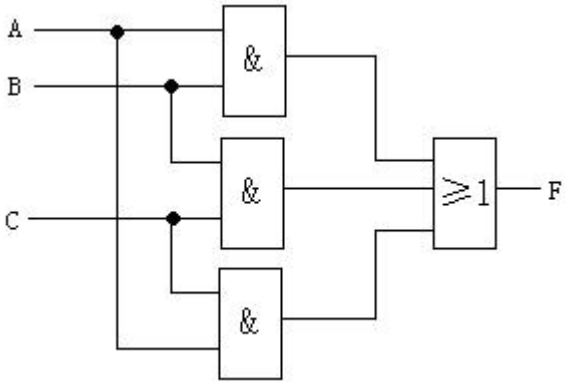


图 4.41 例 4.10 的逻辑电路图

解：电路有三个输入变量 A、B、C 和一个输出变量 F。F 的逻辑表达式为

F=AB+BC+AC

为了分析其逻辑功能，作真值表如表 4.19 所示。

表 4.19 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

考察该真值表可知，三个输入变量中，只要有两个或两个以上的输入变量为 1，则输出 F=1，否则 F=0。可见电路的功能实质上是对“多数”作判决。如果将 A、B、C 分别看作是三个人对某一个提案的表决，“1”表示赞成，“0”表示反对；将函数 F 看作是对该提案的表决结果，“1”表示该提案获得通过，“0”表示该提案未获得通过，则该电路实现了“表决功能”。所以该电路称为“多数表决电路”。

例 4.11 试分析图 4.42 所示电路

解：电路有三个输入变量 A、B、C 和一个输出变量 F。F 的逻辑表达式为

$$F=\overline{Z_1Z_2Z_3}$$

$$= \overline{\overline{AY} \cdot \overline{BY} \cdot \overline{CY}}$$

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

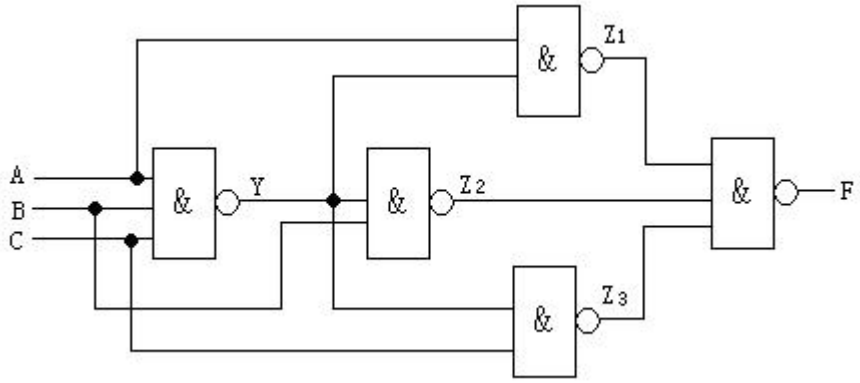


图 4.42 例 4.11 的逻辑电路图

为了便于分析，利用摩根定律先将 F 展成最小项表达式

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

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

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

再根据最小项表达式列出真值表，如表 4.20 所示。

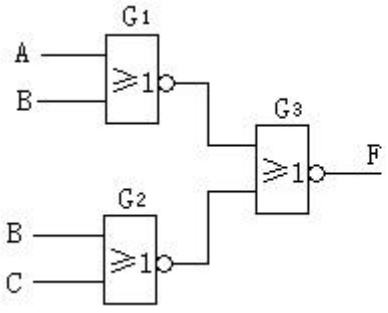
表 4.20 例 4.11 真值表

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

由真值表可看出，ABC=000 或 ABC=111 时，F=0；而 A、B、C 取值不全相同时，F=1。故这种电路称为“不一致”电路。

注意：分析组合逻辑电路的目的是确定它的逻辑功能。而一个组合逻辑电路的逻辑功能可以用多种形式来描述，如逻辑函数表达式、真值表、时序图、和文字叙述等。其中以真值表反映逻辑功能最直观、最全面，所以组合逻辑电路分析的最后一步在没有特殊要求的情况下通常是列出真值表。

例 4.12 已知图 4.43（a）电路的输出 F=(A+B)(B+C)，图（b）是该电路的真值表，但经安装后，测量的结果是 F′，试诊断其故障所在。



（a）电路

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

（b）真值表

图 4.43 例 4.12 电路的故障分析

解：由真值表可见，测得的 F′ 有 0 有 1，故可判断门 G3 输出端是无故障的。

当输入 ABC 为 001 时，F′ 为 1，而不是 F 应得到的 0，说明电路存在故障，从输出反推，因 F′ =1 说明门 G3 的两个输入均为 0；但从输入看，门 G2 的输出是 0，而门 G1 的输出应为 1。所以可以认为是门 G1～G3 的连线可能发生了短路到地的故障。

例 4.13 试分析图 4.44 由四选一数据选择器所示电路

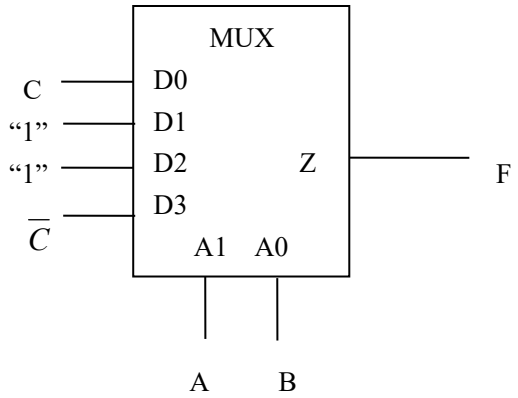


图 4.44 例 4.13 的逻辑电路

解：由四选一数据选择器的函数表达式可得：

$$F = \overline{A}\cdot\overline{B}C + \overline{A}B\cdot 1 + A\overline{B}\cdot 1 + AB\overline{C} = \overline{A}\cdot\overline{B}C + \overline{A}B + A\overline{B} + AB\overline{C}$$

将该逻辑函数代入真值表可见该电路实现“不一致”的功能。

例 4.14 试分析图 4.45 由译码器和逻辑门构成的电路。设输入 $X=X_2X_1X_0$，输出 $Y=Y_2Y_1Y_0$ 均为三位二进制数。

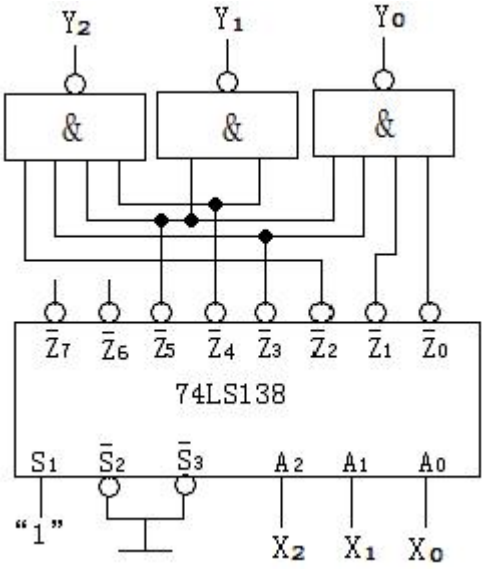


图 4.45 例 4.14 的逻辑电路

解：由图 4.45 可列出 Y_2、Y_1、Y_0 的逻辑函数表达式为：

$$Y_2(X_2,X_1,X_0)=\sum m(2,3,5,6)$$
$$Y_1(X_2,X_1,X_0)=\sum m(4,5)$$
$$Y_0(X_2,X_1,X_0)=\sum m(0,1,3,5)$$

根据函数表达式列出的真值表如表 4.21 所示

表 4.21 例 4.14 的真值表

X2	X1	X0	Y2	Y1	Y0
0	0	0	0	0	1
0	0	1	0	0	1
0	1	0	1	0	0
0	1	1	1	0	1
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	0	0	0

由真值表可见：电路完成的功能是：

（1） 当 X＜2 时，Y=1；

（2） 当 2≤X≤5 时，Y=X+2；

（3） 当 X＞5 时，Y=0。

4.7 组合逻辑电路设计

组合逻辑电路的设计是分析的逆过程，它是从给定的逻辑功能要求出发，经过一定的设计步骤，画出实现该功能的逻辑电路。对电路设计的基本要求是：功能正确，所用元件最少。

4.7.1 设计步骤

用逻辑门设计组合逻辑电路时，一般需要经过以下几个步骤：

一、　进行逻辑抽象

1、分析设计要求，确定输入、输出以及它们之间的因果关系；

2、用英文字母表示输入变量和输出变量；

3、状态赋值，即用 0 和 1 表示输入和输出的有关状态；

4、根据功能要求列出待设计电路的真值表。

二、　进行化简

根据实际情况用卡诺图法或公式法进行化简。

三、　画逻辑电路

1、根据要求使用的门电路类型，将输出函数表达式转化成与之适应的形式。

2、根据最后得到的函数表达式画出逻辑电路图。

4.7.2 设计举例

例 4.15 试设计一个组合电路，其输入为 8421BCD 码。当输入对应的十进制数 3≤X≤6 时，电路有指示。分别用与非门和或非门实现。允许有反变量输入。

解：设 8421BCD 码输入变量为 A、B、C、D，其中 A 为最高有效位。设输出指示变量为 F，且规定 3≤X≤6 时 F=1，否则 F=0。根据题意，可列出该电路的真值表如表 4.22 所示。

F 的卡诺图如图 4.46 和图 4.47 所示。由于用“与非”门实现需圈“1”，而用“或非”门实现需圈“0”。故用两个卡诺图给出示范。求出最简与或式和最简或与式后，可以用摩根定律将其变换为“与非-与非”式和“或非-或非”式，然后就可以用相应的逻辑门来实现。

表 4.22 例 4.15 的真值表

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

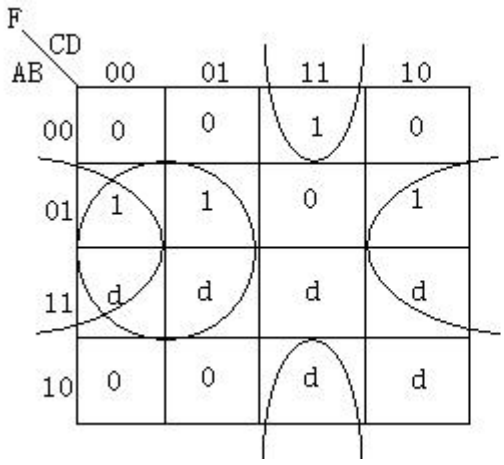


图 4.46

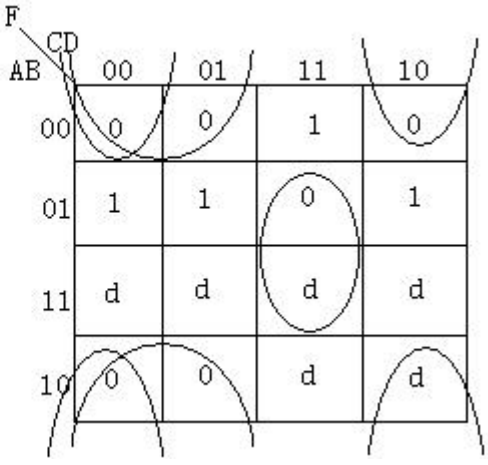


图 4.47

从图 4.46 可读出 F 的最简与或式为：$F = \overline{B}CD + B\overline{C} + B\overline{D}$

利用摩根定律对其变换得：$F = \overline{\overline{\overline{B}CD + B\overline{C} + B\overline{D}}} = \overline{\overline{\overline{B}CD} \cdot \overline{B\overline{C}} \cdot \overline{B\overline{D}}}$

由此得到用与非门实现的电路如图 4.48 所示。

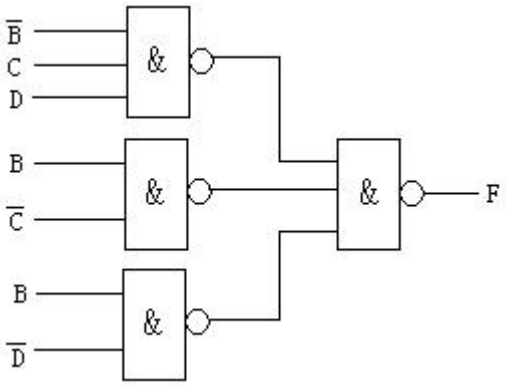


图 4.48 用与非门实现

从图 4.47 可读出 F 的最简或与式为：

$F = (B + D)(B + C)(\overline{B} + \overline{C} + \overline{D})$ ，利用摩根定律对其变换得：

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

由此得到用或非门实现的电路如图 4.49 所示。

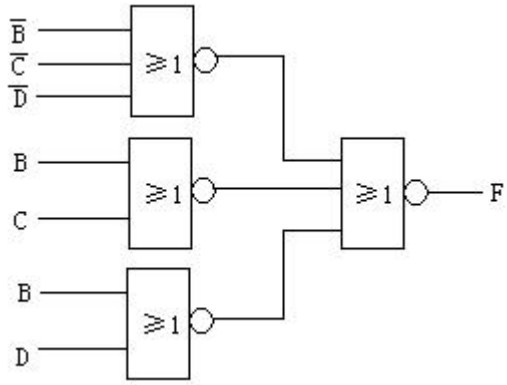


图 4.49 用或非门实现

例 4.16 某飞机有三台发动机，当其中任何一台运转时，用一盏绿灯 G 点亮来指示。当其中任意二台同时运转时，用一盏红灯 R 点亮来指示。当三台同时运转时,用红、绿灯均点亮来指示。试设计一个组合电路来实现其功能。要求用最少的逻辑门。

解：设用 A、B、C 表示三台发动机，当发动机运转时用“1”表示，不运转用“0”表示。灯亮用“1”表示，灯灭用“0”表示。由此列出电路的真值表如表 4.23 所示。

表 4.23 例 4.16 的真值表

A	B	C	G	R
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

G 和 R 的卡诺图如图 4.50 所示。由于需用最少的门实现，故圈卡诺图时要综合考虑。

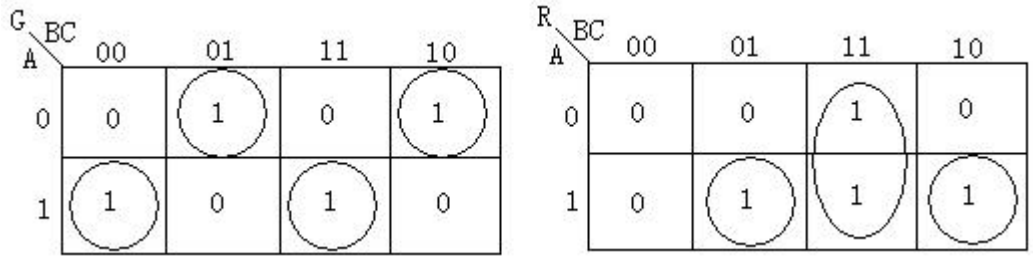


图 4.50 例 4.16 的卡诺图

从图 4.50 可读出 G、R 为：

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

$$R = BC + A\overline{B}C + AB\overline{C} = BC + A(B \oplus C)$$

由此得到的电路图如图 4.51 所示：

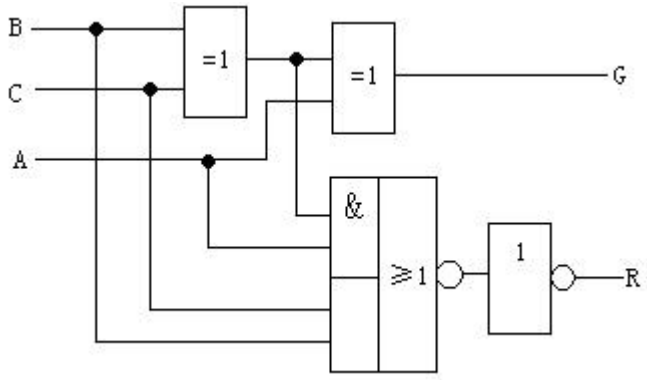


图 4.51 例 4.16 的电路图

例 4.17 某多功能逻辑运算电路的功能表如表 4.24 所示。该电路具有功能选择开关 K_1、K_0、两个输入变量 A、B，一个输出变量 F。在 K_1、K_0 的控制下按表 4.24 所示功能进行逻辑运算。试用逻辑门设计该电路。

表 4.24 例 4.17 的功能表

K_1	K_0	F
0	0	A+B
0	1	AB
1	0	$A \oplus B$
1	1	$\overline{AB}$

根据功能表可得对应的真值表如表 4.25 所示：

表 4.25 例 4.17 的真值表

K_1	K_0	A	B	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

根据真值表得到的卡诺图如图 4.52 所示：

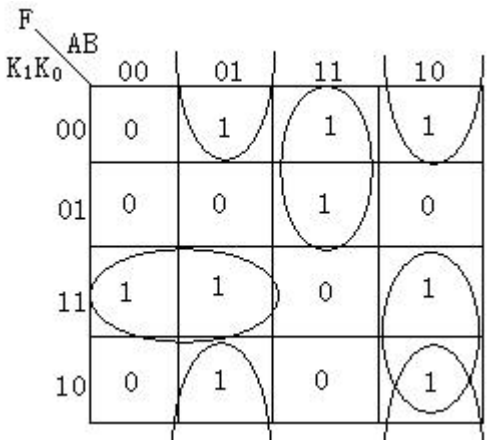


图 4.52

经化简后可得：

$$F = \overline{K_0}\,\overline{A}B + \overline{K_1}AB + \overline{K_0}A\overline{B} + K_1A\overline{B} + K_1K_0\,\overline{A}$$

据此可得电路图如图 4.53 所示：

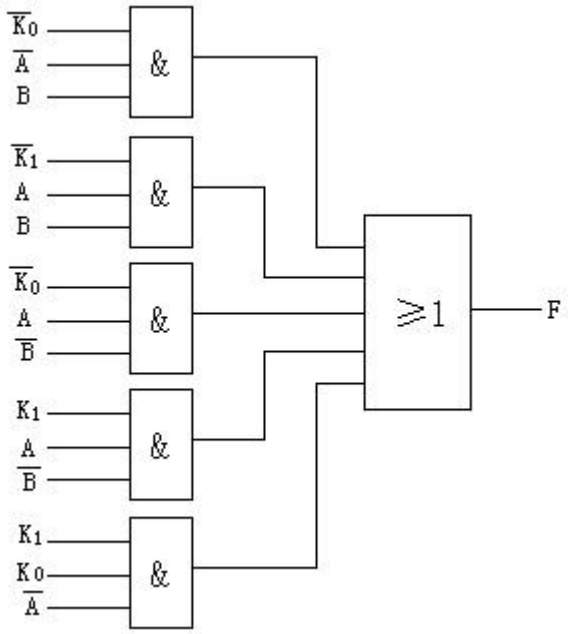


图 4.53 例 4.17 的电路图

例 4.18 试设计一个对三位二进制数的的判别电路，当对应的二进制数 3≤X≤6 时，函数 F=1，否则 F=0。输入无反变量提供，试用最少的与非门实现该电路。

解：根据题意列出的真值表如表 4.26 所示：

表 4.26 例 4.18 的真值表

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

根据真值表得到的卡若图如图 4.54 所示：

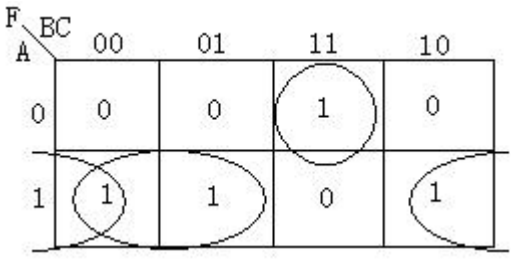


图 4.54

经化简后可得：

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

合并前两项中的头部（原变量部分）得：

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

将头部（原变量部分）插入尾部（反变量部分）中可得：

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

将等式化为可用与非门实现的形式：

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

据此得到的电路如图 4.55 所示：

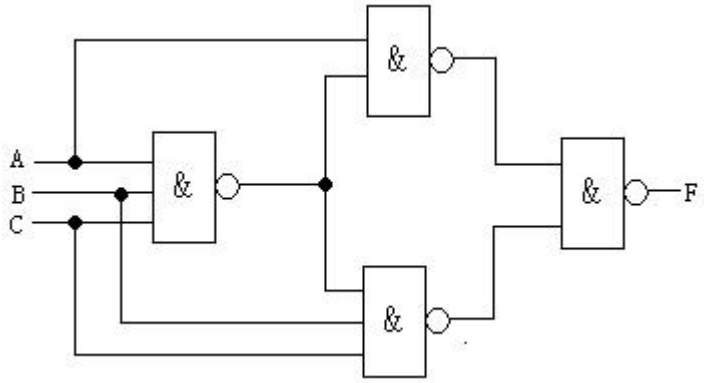


图 4.55 例 4.18 的电路

根据上例我们可以总结出不能提供反变量时由与非门实现逻辑函数的方法为：

（1）求出函数的最简与或表达式；

（2）合并头部（原变量部分）相同的逻辑与项；

（3）将头部插入尾部（逻辑与项中的反变量部分），得到合适的尾部因子；

（4）求得的尾部因子最好能被多个头部所共享（如上例中的 $\overline{ABC}$）

（5）利用摩根定律将函数变换为“与非—与非”形式。

同样，我们可以得到不能提供反变量时由或非门实现逻辑函数的方法：

（1）求出函数的最简或与表达式；

（2）合并头部（原变量部分）相同的逻辑或项；

（3）将头部插入尾部（逻辑或项中的反变量部分），得到合适的尾部因子；

（4）求得的尾部因子最好能被多个头部所共享；

（5）利用摩根定律将函数变换为“或非—或非”形式。

例 4.19 试用最少的或非门实现下列函数，输入端不能提供反变量。

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

合并前两项中的头部（原变量部分）得：

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

将头部插入尾部得到适当的尾部因子

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

将等式化为可用或非门实现的形式：

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

由此得到图 4.56 所示电路。

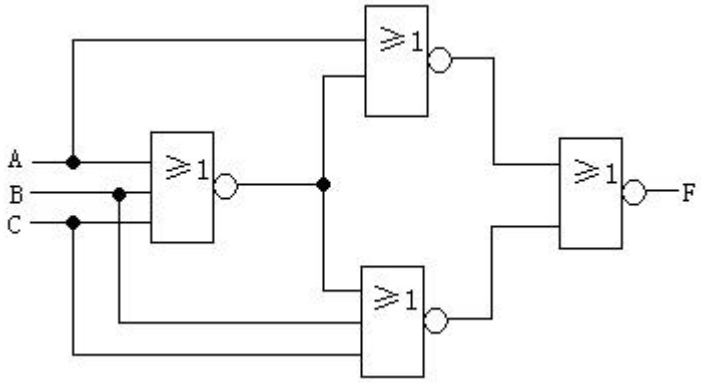


图 4.56 例 4.19 的电路

例 4.20 试用 74LS138 3 线—8 线译码器和适当的逻辑门设计一个 1 位的二进制全减器

1 位二进制全减器的真值表如表 4.27 所示，其中 A_i、B_i 分别为被减数和减数输入，G_{i-1} 为相邻低位的借位输入，D_i 为本位差输出，G_i 为本位向相邻高位的借位输出。

表 4.27 全加器的真值表

A_i	B_i	G_{i-1}	D_i	G_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

由真值表可以直接写出差输出 D_i 和借位输出 G_i 的最小项表达式：

$$D_i(A_i,B_i,G_{i-1})=\sum m(1,2,4,7)=\overline{\overline{Z}_1\overline{Z}_2\overline{Z}_4\overline{Z}_7}$$

$$G_i(A_i,B_i,G_{i-1})=\sum m(1,2,3,7)=\overline{\overline{Z}_1\overline{Z}_2\overline{Z}_3\overline{Z}_7}$$

实现的电路如图 4.57 所示

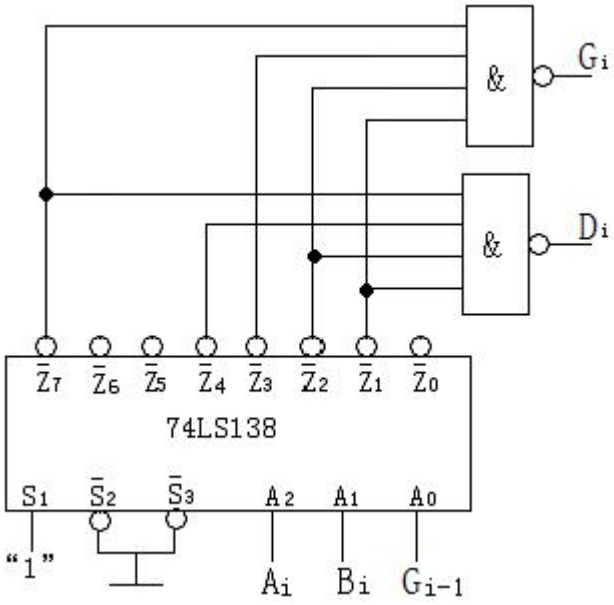


图 4.57 例 4.20 的电路

4.8 组合逻辑电路中的竞争与冒险

前面讨论组合电路时，只研究了输入和输出稳定状态之间的关系，均未考虑信号在传输过程中的延迟现象。实际上，信号经过任何逻辑门和线路都会产生时间延迟，这就使得当电路所有输入都达到稳定状态时，输出并不是立即达到稳定状态。而是经过一段过渡时间才能达到稳定状态。

一般来说，延迟时间对数字系统是有害的。它会使系统速度下降，引起电路中信号的波形畸变，更严重的是在电路中产生竞争冒险的问题。下面讨论电路中的竞争和冒险现象。

4.8.1 竞争和冒险现象

一、竞争与冒险

在组合逻辑电路中，当一个输入信号（含这个信号的非）经过多条路径传送后到达某一逻辑门的输入时，由于传送路径不同，使这个信号和这个信号的“非”到达门的输入时间有先有后，这一现象称为竞争。

不产生错误输出的竞争称为非临界竞争，产生错误输出的竞争称为临界竞争。临界竞争产生的错误输出即波形上的毛刺，有可能引起后级电路的错误动作，所以该错误输出称为冒险。

二、冒险现象的种类

组合电路中的险象和根据输入变化前后输出是否相等而分为静态险象和动态险象。如果在输入变化而输出不应该发生变化的情况下，输出端产生了瞬间的错误输出，即产生了险象，则这种险象称为静态险象。如果在输入变化而输出应该发生变化的情况下，输出在变化过程中产生了瞬间的错误输出，则称这种险象为动态险象。

除了按输出是否应该变化，可分为静态险象和动态险象外，还可以按错误输出脉冲信号的极性分为“1”型险象和“0”型险象。若错误输出信号为正脉冲，称为“1”型险象，反之，若错误输出信号为负脉冲，则称为“0”型险象。图 4.57 给出了静态“1”型险象、静态“0”型险象、动态“1”型险象和动态“0”型险象的波形。

需要指出的是，组合电路中的动态险象一般都是由静态险象引起的。所以，消除了静态险象也就消除了动态险象。

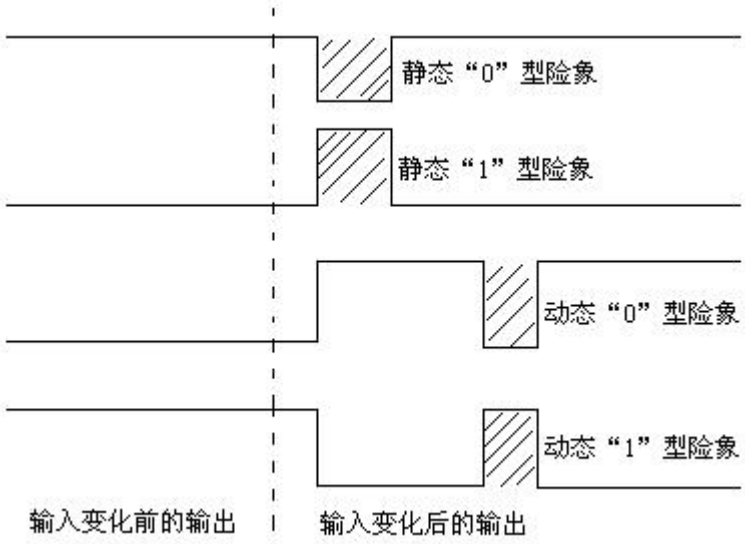


图 4.58 组合电路中可能产生的险象种类

4.8.2 怎样判定电路中有无险象

判定一个电路中有无险象的方法有代数法和卡诺图法。

一、代数法

由前面对竞争和冒险的分析可知，当某个变量 A 同时以原变量和反变量的形式出现在函数表达式中，且令除了变量 A（含 $\overline{A}$）以外的其他变量为某个恒定值（0 或 1）后，若出现 Y=A+ $\overline{A}$，则存在“0”型险象；若出现 Y= $A \cdot \overline{A}$，则存在“1”型险象。

例 4.21： 判断 $F = AC + \overline{A}B$ 是否存在险象。

解：令 B=C=1，则得 $F = A + \overline{A}$ 存在“0”型险象。图 4.59 是实现该函数式的电路和波形。

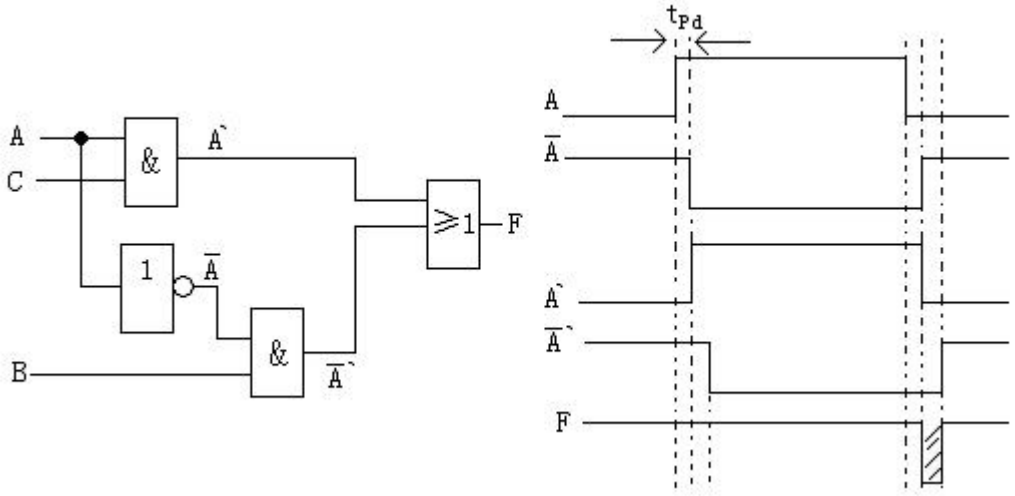


图 4.59 例 4.21 的电路和波形

二、卡诺图法

在逻辑函数的卡诺图中，函数式的每一个积项（或和项）对应了卡诺图上的一个卡诺圈。

如果两个卡诺圈存在着相切部分，且相切部分又未被其它卡诺圈圈住，则该电路必然存在险象。

例 4.22：用卡诺图法判断函数 $F(A,B,C,D)=\overline{A}\overline{B}D+A\overline{B}\overline{C}+BCD$ 是否存在险象。

解：F 的卡诺图如图 4.60 所示。从图中可见，代表 $\overline{A}\overline{B}D$ 的卡诺圈和代表 BCD 的卡诺圈相切、代表 $\overline{A}\overline{B}D$ 的卡诺圈和代表 $A\overline{B}\overline{C}$ 的卡诺圈相切，且相切部分都未被其它卡诺圈圈住，故前者相邻情况下，在 A=0，C=D=1 时，B 若变化，电路会出现险象。后者相邻情况下，在 B=0，C=0，D=1 时，若 A 变化，则电路会出现险象。

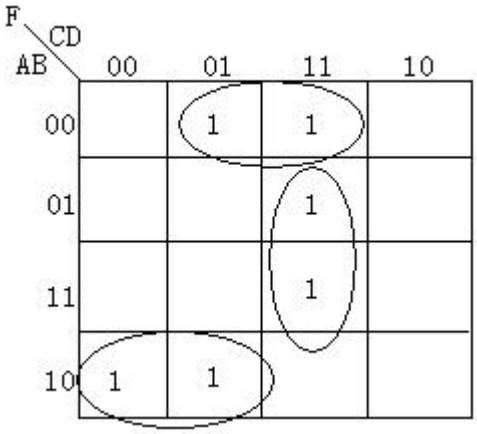


图 4.60 例 4.22 的卡诺图

4.8.3 险象的消除和减弱

当组合电路存在险象时，可以采用增加冗余项、引入封锁脉冲、增加输出端滤波等多种方法来消除和减弱险象。由于引入封锁脉冲大大增加了电路的复杂性，很少使用，故这里介绍用冗余项和输出端加滤波的方法。

一、增加冗余项

当竞争和险象是由单个变量改变状态引起的时，用增加冗余项的方法很方便。

例 4.23：$F=AB+\overline{A}C$

从函数式不难发现，当 B=C=1 时，有 $F=A+\overline{A}$。若 A 从 1 变为 0（或从 0 变为 1），则在输出门的输入端就会发生竞争，故输出就会发生险象。若在函数式中增加冗余项 BC，则函数式变为：$F=AB+\overline{A}C+BC$。增加冗余项后的电路见图 4.61，不难看出，当 B=C=1 时，由于 BC=1，封锁了输出门，故“0”型险象被抑制。

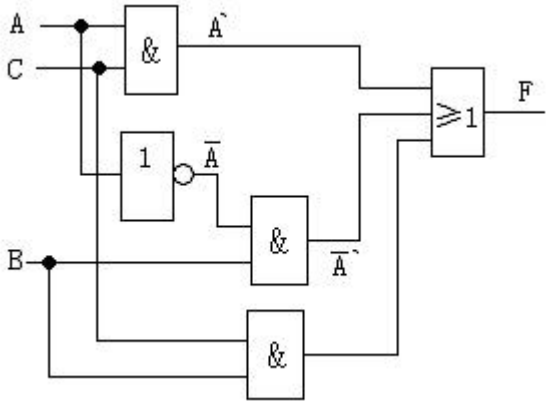


图 4.61 增加冗余项消除险象

二、接入滤波电容

因为竞争冒险所产生的干扰脉冲一般都比较窄，所以可以在电路的输出端并接一个小电容来减弱干扰脉冲。图 4.62 就是加了滤波电容后的电路。由于干扰脉冲通常和门电路的传输时间属同一个量级，所以在 TTL 电路中，电容的容量只要在几百微微法就可将干扰脉冲减弱到开门电平以下。

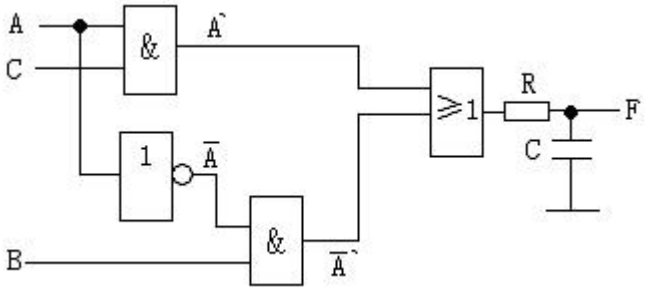


图 4.62 用滤波电容来减弱险象

4.9 组合逻辑电路的应用

4.9.1 用全加器将 2 位 8421BCD 码变换成二进制代码

设两位 8421BCD 码 $D = A_{80}A_{40}A_{20}A_{10}\ \ A_8A_4A_2A_1$（下标表示各位的权重）

将权重按 2 的幂展开，可得：

$$D = A_{80}\times 80 + A_{40}\times 40 + A_{20}\times 20 + A_{10}\times 10 + A_8\times 8 + A_4\times 4 + A_2\times 2 + A_1\times 1 =$$

$$= A_{80}\times(64+16) + A_{40}\times(32+8) + A_{20}\times(16+4) + A_{10}(8+2) + A_8\times 8 + A_4\times 4$$
$$+ A_2\times 2 + A_1\times 1$$

$$= A_{80}\times 2^6 + A_{40}\times 2^5 + (A_{80}+A_{20})\times 2^4 + (A_{40}+A_{10}+A_8)\times 2^3 + (A_{20}+A_4)\times 2^2$$

$$+ (A_{10}+A_2)\times 2^1 + A_1\times 2^0$$

根据上式可见，8421BCD 码变换为二进制代码可以用 BCD 码的各位码元，按照上式幂相加而得，电路如图 4.63 所示。图中，因 8421BCD 码和二进制代码的最低位是一样的，用一条直线连接即可，而其余各位可通过图 4.63 的加法器获得。

8421 BCD 码输入　　二进制码输出

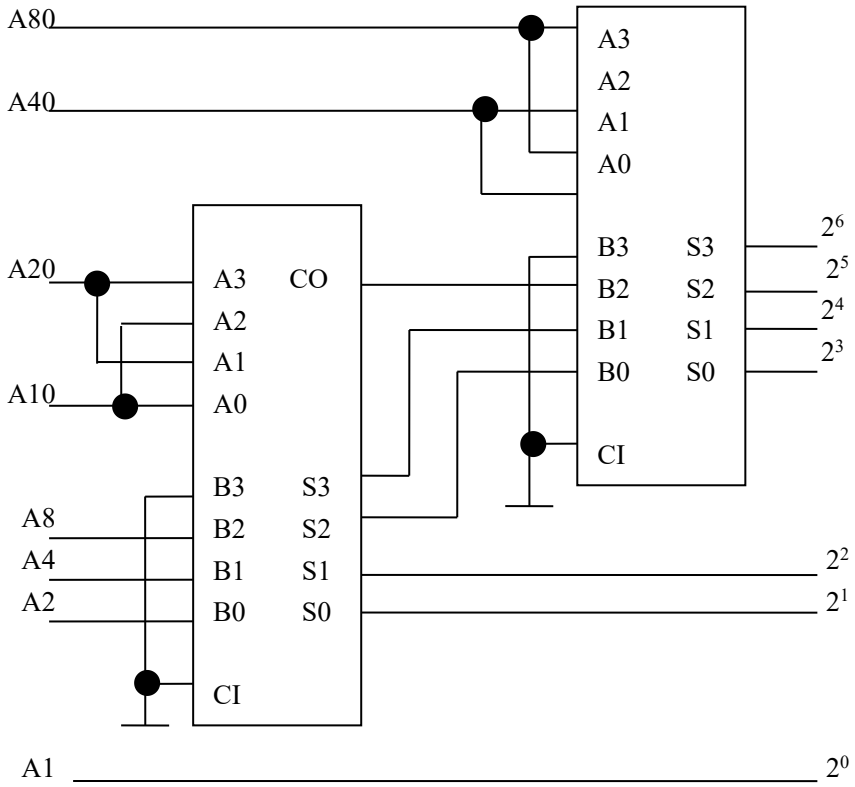


图 4.63 用 2 片 74LS283 全加器实现 8421BCD 码变换为二进制代码

4.9.2 数据传输系统

图 4.64 是由 16 选 1 数据选择器和 4 线到 16 线译码器构成的总线数据传输系统，它能将 16 位并行数据转换为串行数据进行传送，到达终端后又还原为并行数据输出。

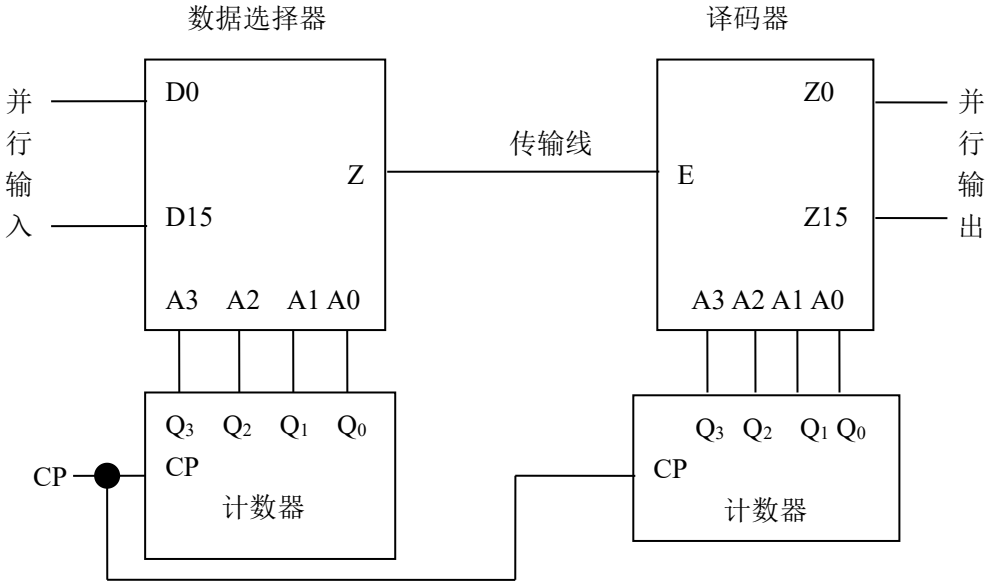


图 4.64 数据传输系统

4.10 习题

1. 试分析图 4.65 所示组合逻辑电路，说明电路功能。

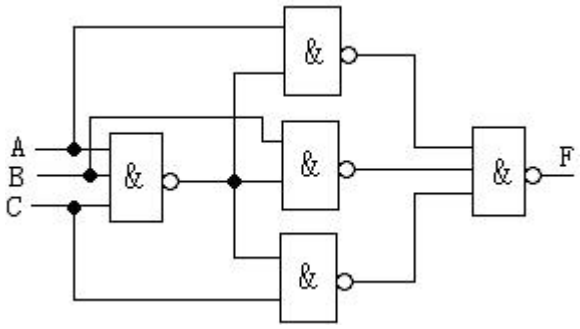


图 4.65

2. 试分析图 4.66 所示组合逻辑电路，并改用最少的与非门实现电路的功能。

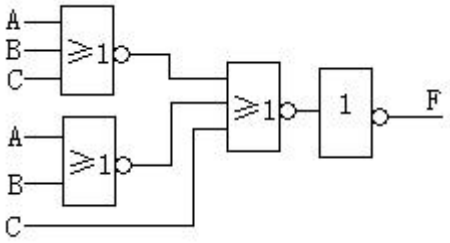


图 4.66

3.试分析图 4.67 所示组合逻辑电路，其中 S_4，S_3，S_2，S_1 为控制输入端，列出真值表，说明 F 与 A、B 之间的逻辑关系。

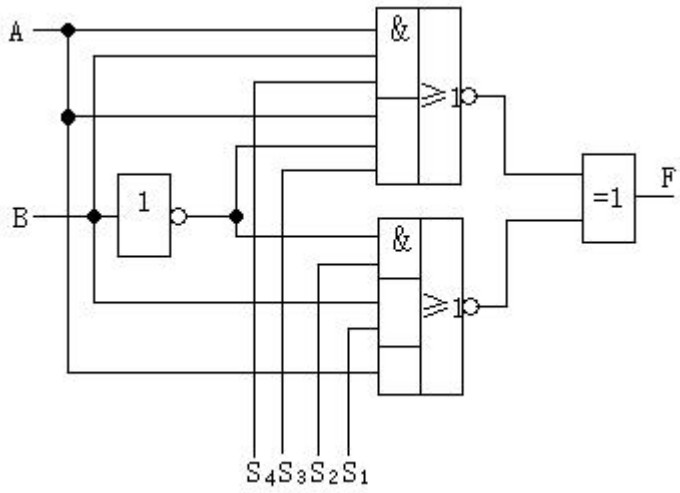


图 4.67

4.试分析图 4.68 所示组合逻辑电路，说出电路的逻辑功能，并改用异或门实现该电路的逻辑功能。

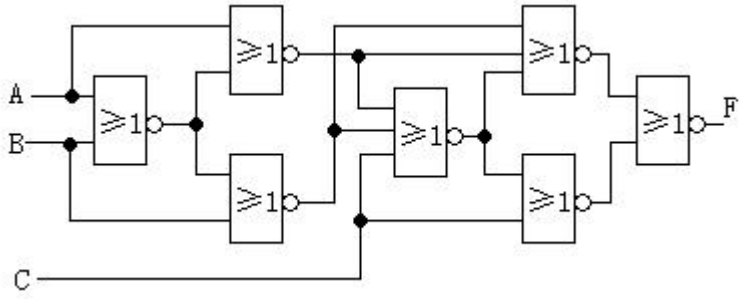


图 4.68

5 试分析图 4.69 所示组合逻辑电路，说出电路的逻辑功能。

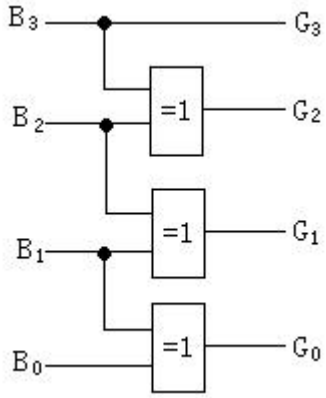


图 4.69

6 试分析图 4.70 所示组合逻辑电路，并用与非门实现该电路的逻辑功能。

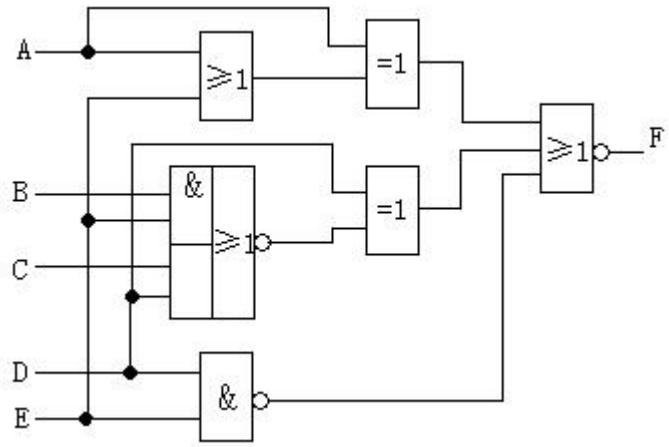


图 4.70

7 试分析图 4.71 所示组合逻辑电路，说出电路的逻辑功能。

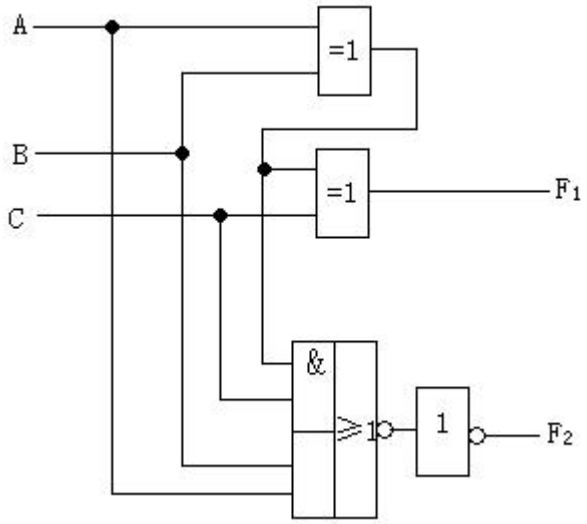


图 4.71

8 试设计一个代码转换电路，将一位 8421BCD 码转换为余三码。

9 用与非门设计一个组合电路，输入的是一位 8421BCD 码，当输入的数字为素数时，输出为 1，否则输出为 0。

10 试用或非门设计一个 8421BCD 码检测电路，当输入的数字 3≤X≤7 时，输出为 1，否则输出为 0。

11 试设计一个加/减法器，该电路在控制信号 X 的控制下进行加、减运算，当 X=0 时，实现全减器功能；当 X=1 时，实现全加器功能。

12 某栋楼有一盏路灯，同住该栋楼的三家住户要求在各自的家门口安装开关均能独立地控制路灯的开和关。请用最少的逻辑门设计出满足该要求的控制电路。

13 用逻辑门为医院设计一个血型配对指示器，当供血和受血血型不符合表题 4.28 时，输出为“1”，指示灯亮。

表题 4.28

供血血型	受血血型
A	A、AB
B	B、AB
AB	AB
O	A、B、AB、O

14 用与或非门设计一个两位二进制数 A、B 的比较电路，要求有 A<B、A=B 和 A>B 三种比较结果输出。

15 分别用与非门设计能实现下列功能的组合电路。

（1）三变量的表决电路（输出与多数变量的状态一致）；

（2）三变量的不一致电路（三个变量状态不相同时输出为 1，相同时输出为 0）；

（3）三变量判奇电路（三个变量中有奇数个 1 时输出为 1，否则输出为 0）；

（4）三变量判偶电路（三个变量中有偶数个 1 时输出为 1，否则输出为 0）。

16 某药店常用中药有 50 种，编号为 1～50。在中药配方时必须遵守下列配方规定：

（1） 第 6 号与第 9 号不能同时使用；

（2）第 22 号与第 38 号不能同时使用；
（3）第 1 号、第 43 号、第 48 号不能同时使用；
（4）用第 5 号时必须同时配用第 8 号；
（5）当第 33 号和第 42 号一起使用时，必须配用第 2 号。

试设计一个组合逻辑电路，要求在违反上述任一项规定时，输出 F=1，否则输出 F=0。

17 试设计一个将四位二进制代码转换为相应的补码的码制转换电路。

18 用与非门实现下列逻辑函数，要求不会产生险象。

（1） $F_1(A,B,C,D)=\sum m(2,3,5,7,8,10,13)$

（2） $F_2(A,B,C,D)=\sum m(0,2,3,4,8,9,14,15)$

（3） $F_3(A,B,C,D)=\sum m(1,5,6,7,11,12,13,15)$

19 函数 F= $AC\cdot B\overline{C}$，试分析当实现该电路的逻辑门的平均延迟时间为 10nS，会不会产生险象？什么条件下产生险象？画出无险象产生的改进电路图。

20 试用 74LS138（3 线—8 线译码器）和与非门实现下列逻辑函数。

(1)F_1(A,B,C)= $A\overline{C}+\overline{B}C+\overline{A}B$

(2) F_2(A,B,C)= $\overline{A}\cdot\overline{C}+A\overline{B}C$

21 试用 74LS138（3 线—8 线译码器）和与非门设计一个全减器。

22 试用四路数据选择器实现余三码到 8421BCD 码的转换。

23 试用一片 4 线—16 线译码器和适当的逻辑门设计一个一位十进制数 8421BCD 码的奇偶位产生电路（假定采用偶检验）。

24 当 4 路数据选择器的选择控制变量 A_1、A_0 接变量 A、B，数据输入端 D_0、D_1、D_2、D_3 依次接 C、1、1 $\overline{C}$ 时，电路实现什么功能？

25 试用 4 位二进制加法器设计下列十进制代码转换器：
(1) 8421BCD 码转换为余三码；
(2) 余三码转换为 8421BCD 码。

26 试用输出高电平有效的 4 线—16 线译码器和逻辑门实现两个二位二进制数的乘积。

27 分别用四选一和八选一数据选择器实现下列逻辑函数。
(1) F(A,B,C,D)= Σm(1,2,5,6,7,10,15)
(2) F(A,B,C,D)= Σm(0,3,8,9,10,11)+ Σd (1,2,5,14,15)
(3)F(A,B,C,D)=ΠM(1,2,8,9,10,12,14)・ΠD (0,3,5,6,11,13)

28 用一片四位全加器和必要的逻辑门,设计一个可控的四位加/减法器，当控制信号 M=0 时，进行 A 加 B；当 M=1 时，进行 A 减 B，此时 A≥B。