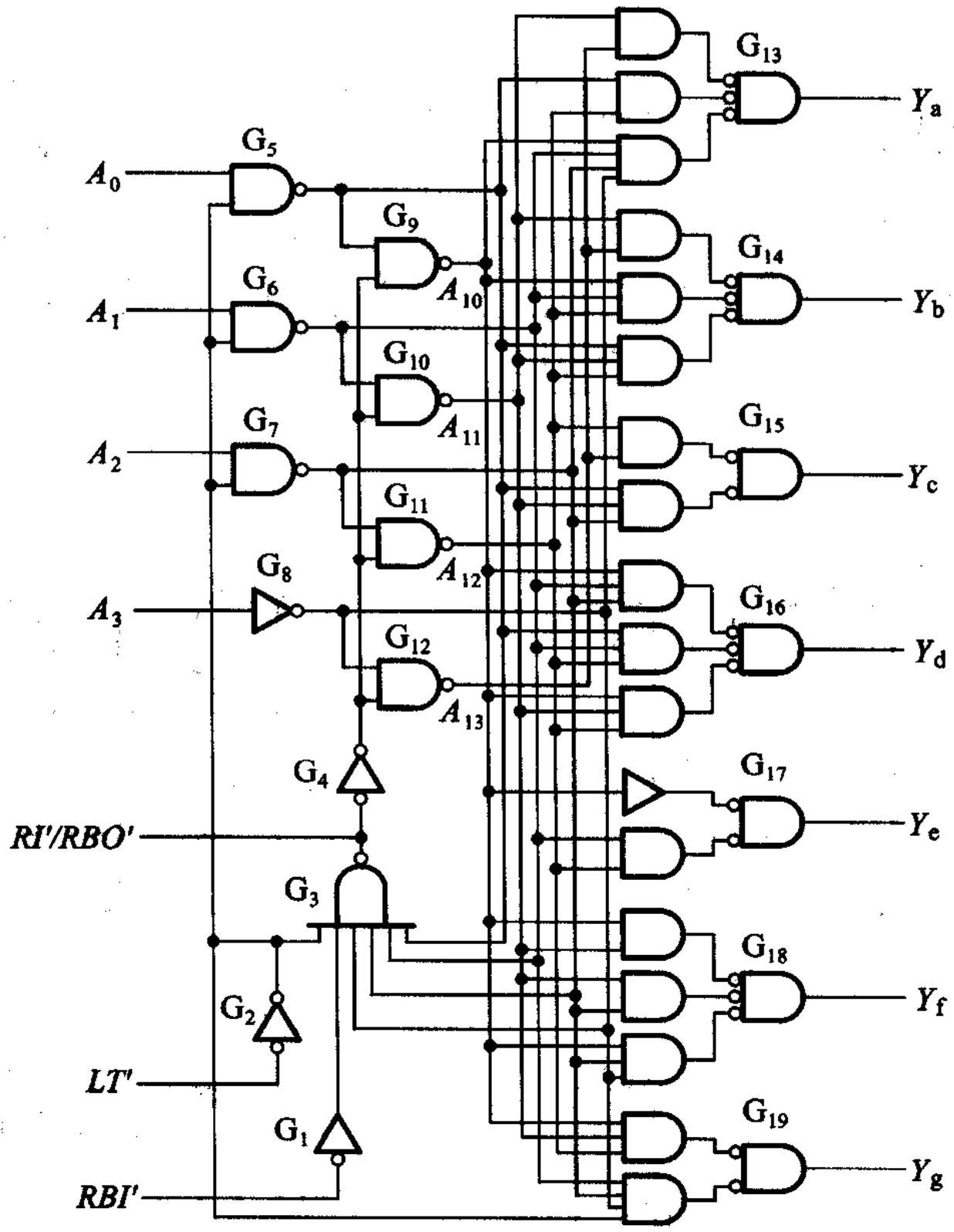


图 4.3.16　BCD－七段显示译码器 7448 的逻辑图

设置灭零输入信号*RBI'*的目的是为了能把不希望显示的零熄灭。例如，有一个 8 位的数码显示电路，整数部分为 5 位，小数部分为 3 位，在显示 13.7 这个数时将呈现 00013.700 字样。如果将前、后多余的零熄灭，则显示的结果将更加醒目。

由图 4.3.16 可知，当输入 $A_3=A_2=A_1=A_0=\mathbf{0}$ 时，本应显示出 **0**。如果需要将这个零熄灭，则可加入 $RBI'=\mathbf{0}$ 的输入信号。这时 G_3 的输出为低电平，并经过 G_4 输出低电平使 $A_{13}=A_{12}=A_{11}=A_{10}=\mathbf{1}$。由于 $G_{13}\sim G_{19}$ 每个**与或非**门都有一组输入全为高电平，所以 $Y_a\sim Y_g$ 全为低电平，使本来应该显示的 **0** 熄灭。

灭灯输入/灭零输出*BI'/RBO'*：

这是一个双功能的输入/输出端，它的电路结构如图 4.3.17(a)所示。

*BI'/RBO'*作为输入端使用时，称灭灯输入控制端。只要加入灭灯控制信号 $BI'=\mathbf{0}$，无论 $A_3A_2A_1A_0$ 的状态是什么，定可将被驱动数码管的各段同时熄灭。由

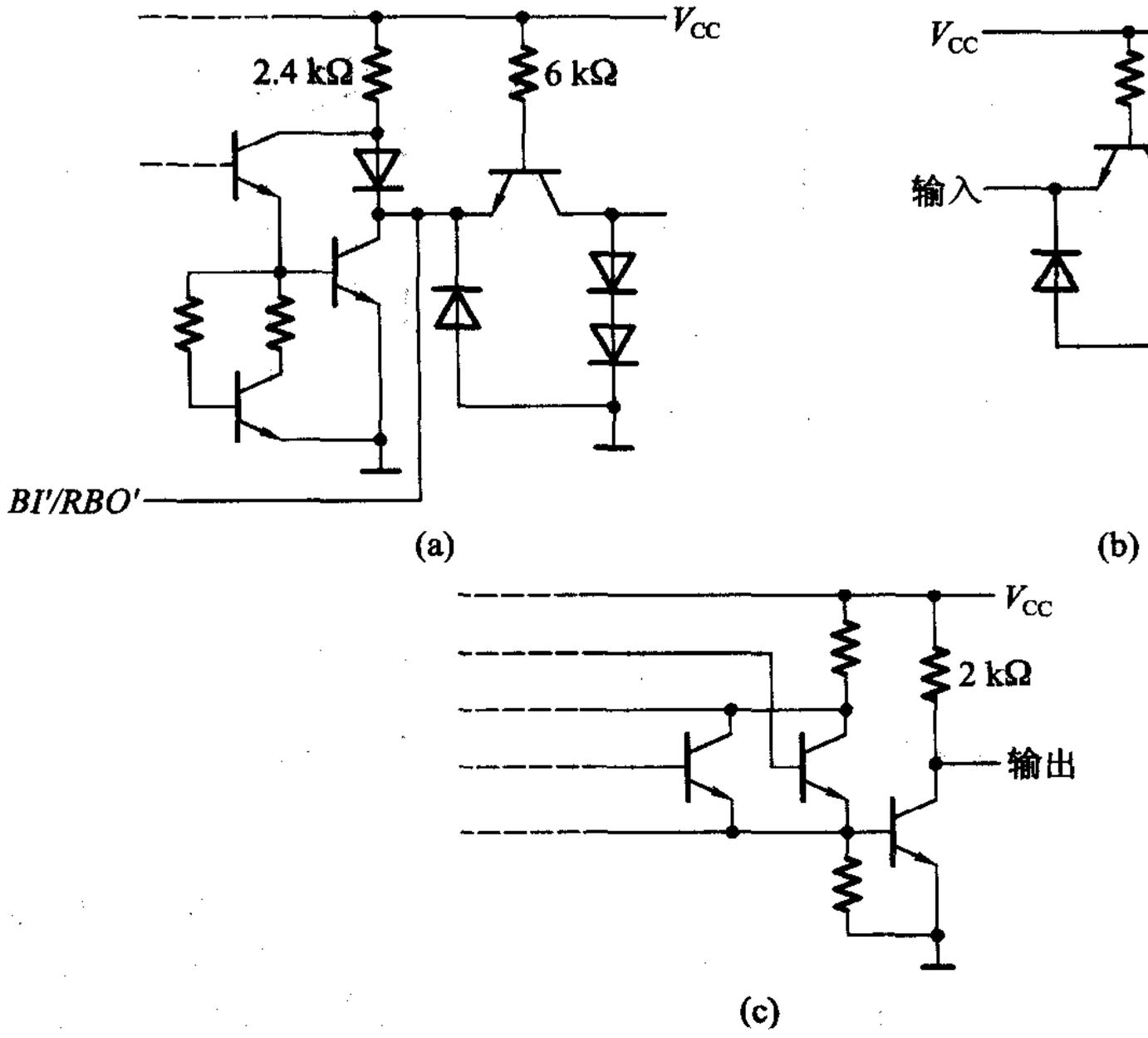


图 4.3.17 7448 的输入、输出电路

(a) *BI'/RBO'*端 (b) 输入端 (c) 输出端

图 4.3.16 可见,此时 G_4 肯定输出低电平,使 $A_{13}=A_{12}=A_{11}=A_{10}=\mathbf{1}$,$Y_a \sim Y_g$ 同时输出低电平,因而将被驱动的数码管熄灭。

*BI'/RBO'*作为输出端使用时,称为灭零输出端。由图 4.3.16 可得到

$$RBO'=(A_3' \cdot A_2' \cdot A_1' \cdot A_0' \cdot LT' \cdot RBI)' \qquad (4.3.12)$$

上式表明,只有当输入为 $A_3=A_2=A_1=A_0=\mathbf{0}$,而且有灭零输入信号($RBI'=\mathbf{0}$)时,*RBO'*才会给出低电平。因此,$RBO'=\mathbf{0}$ 表示译码器已将本来应该显示的零熄灭了。

用 7448 可以直接驱动共阴极的半导体数码管。由图 4.3.17(c)所示的 7448 输出电路可以看到,当输出管截止、输出为高电平时,流过发光二极管的电流是由 V_{CC}经 2 kΩ 上拉电阻提供的。当 $V_{CC}=5$ V 时,这个电流只有 2 mA 左右。如果数码管需要的电流大于这个数值时,则应在 2 kΩ 的上拉电阻上再并联适当的电阻。图 4.3.18 给出了用 7448 驱动 BS201A 半导体数码管的连接方法。

将灭零输入端与灭零输出端配合使用,即可实现多位数码显示系统的灭零控制。图 4.3.19 示出了灭零控制的连接方法。只需在整数部分把高位的*RBO'*与低位的*RBI'*相连,在小数部分将低位的*RBO'*与高位的*RBI'*相连,就可以把前、后多余的零熄灭了。在这种连接方式下,整数部分只有高位是零,而且被熄灭的

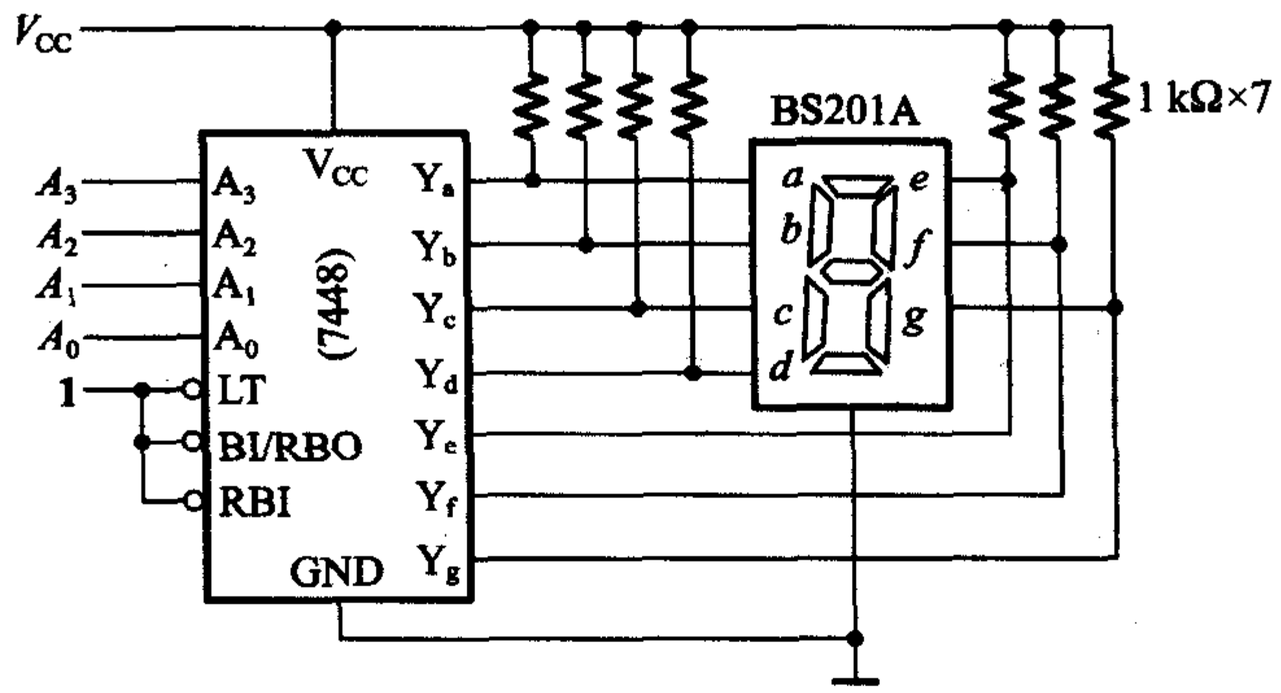


图 4.3.18　用 7448 驱动 BS201A 的连接方法

情况下，低位才有灭零输入信号。同理，小数部分只有在低位是零，而且被熄灭时，高位才有灭零输入信号。

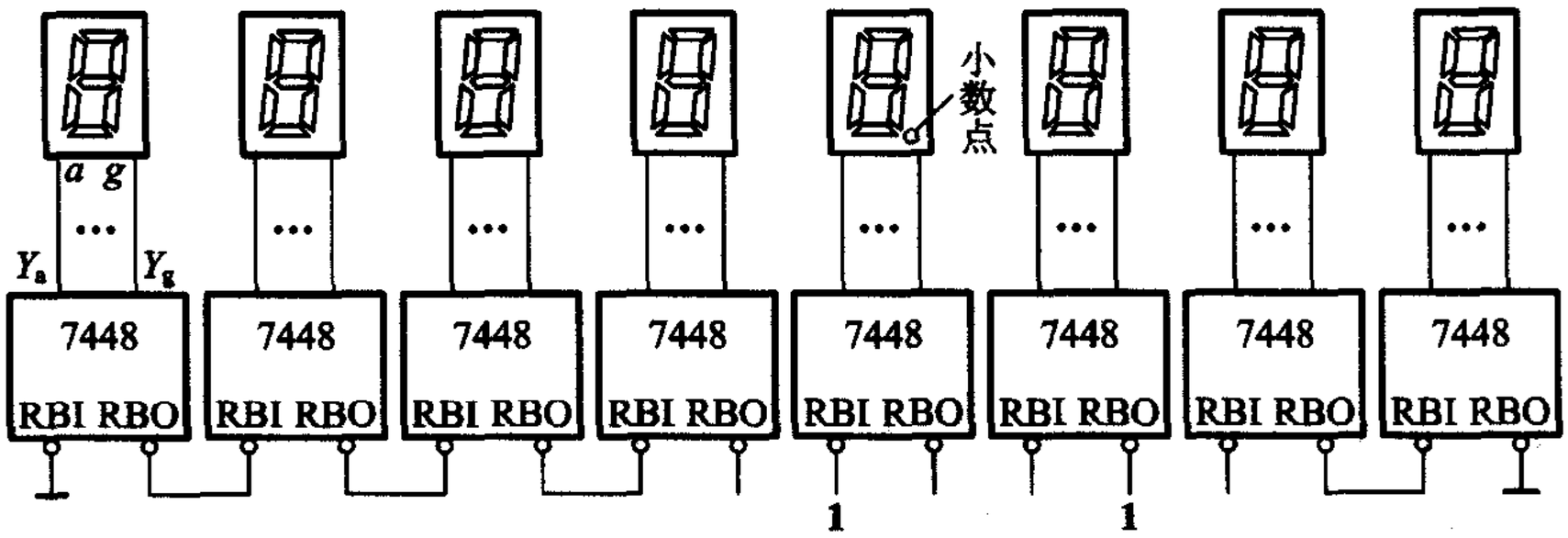


图 4.3.19　有灭零控制的 8 位数码显示系统

四、用译码器设计组合逻辑电路

前面已经详细介绍了二进制译码器的电路结构和工作原理。由图 4.3.8 所示的 3 线 -8 线译码器中可以看到，当控制端 $S=1$ 时，若将 A_2、A_1、A_0 作为 3 个输入逻辑变量，则 8 个输出端给出的就是这 3 个输入变量的全部最小项 $m'_0 \sim m'_7$，如式(4.3.7)所示。利用附加的门电路将这些最小项适当地组合起来，便可产生任何形式的三变量组合逻辑函数。

同理，由于 n 位二进制译码器的输出给出了 n 变量的全部最小项，因而用 n 变量二进制译码器和**或门**(当译码器的输出为原函数 $m_0 \sim m_{2^n-1}$ 时)或者**与非门**(当译码器的输出为反函数 $m'_0 \sim m'_{2^n-1}$ 时)定能获得任何形式输入变量数不大于 n 的组合逻辑函数。

【例 4.3.3】　试利用 3 线 -8 线译码器 74HC138 设计一个多输出的组合逻

辑电路。输出的逻辑函数式为

$$\begin{cases} Z_1 = AC' + A'BC + AB'C \\ Z_2 = BC + A'B'C \\ Z_3 = A'B + AB'C \\ Z_4 = A'BC' + B'C' + ABC \end{cases} \tag{4.3.13}$$

解： 首先将式(4.3.13)给定的逻辑函数化为最小项之和的形式，得到

$$\begin{cases} Z_1 = ABC' + AB'\ C' + A'BC + AB'C = m_3 + m_4 + m_5 + m_6 \\ Z_2 = ABC + A'BC + A'\ B'C = m_1 + m_3 + m_7 \\ Z_3 = A'BC + A'BC' + AB'C = m_2 + m_3 + m_5 \\ Z_4 = A'BC' + AB'C' + A'B'C' + ABC = m_0 + m_2 + m_4 + m_7 \end{cases} \tag{4.3.14}$$

由图 4.3.8 和式(4.3.7)可知，只要令 74HC138 的输入 $A_2 = A$、$A_1 = B$、$A_0 = C$，则它的输出 $Y_0' \sim Y_7'$ 就是式(4.3.14)中的 $m_0' \sim m_7'$。由于这些最小项是以反函数形式给出的，所以还需要将 $Z_1 \sim Z_4$ 变换为 $m_0' \sim m_7'$ 的函数式

$$\begin{cases} Z_1 = (m_3' \cdot m_4' \cdot m_5' \cdot m_6')' \\ Z_2 = (m_1' \cdot m_3' \cdot m_7')' \\ Z_3 = (m_2' \cdot m_3' \cdot m_5')' \\ Z_4 = (m_0' \cdot m_2' \cdot m_4' \cdot m_7')' \end{cases} \tag{4.3.15}$$

上式表明，只需在 74HC138 的输出端附加 4 个**与非**门，即可得到 $Z_1 \sim Z_4$ 的逻辑电路。电路的接法如图 4.3.20 所示。

如果译码器的输出为原函数形式($m_0 \sim m_7$)，则只要将图 4.3.20 中的**与非**门换成**或**门就行了。

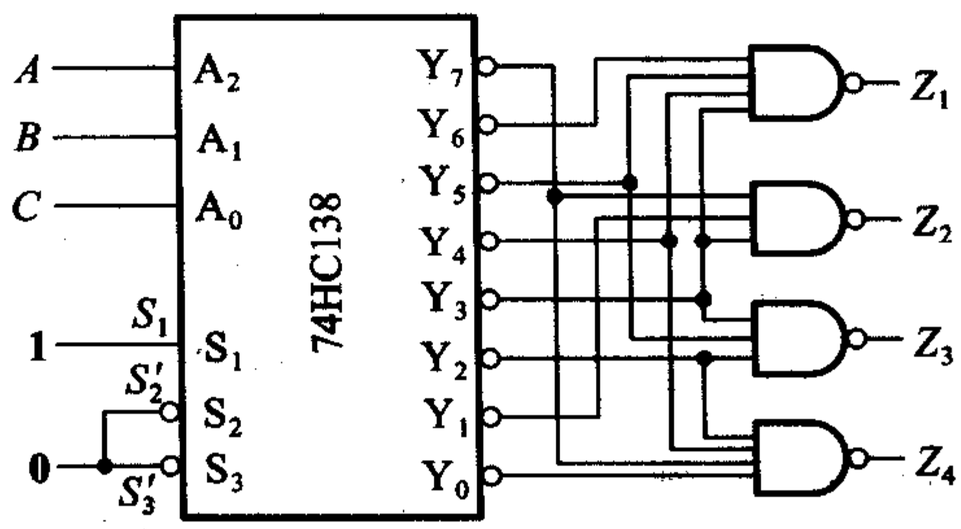


图 4.3.20 例 4.3.3 的电路

复习思考题

R4.3.2　用二－十进制译码器附加门电路(如图 4.3.20 所示的结构形式)能否得到任何形式的四变量逻辑函数？为什么？

R4.3.3　用 4 线－16 线译码器(输入为 A_3、A_2、A_1、A_0，输出为 $Y'_0 \sim Y'_{15}$)能否取代图 4.3.20 中的 3 线－8 线译码器？如果可以取代，那么电路应如何连接？

4.3.3　数据选择器

一、数据选择器的工作原理

在数字信号的传输过程中，有时需要从一组输入数据中选出某一个来，这时就要用到一种称为数据选择器(Data Selector)或多路开关(Multiplexer)的逻辑电路。

现以双 4 选 1 数据选择器 74HC153 为例，说明它的工作原理。图 4.3.21 是 74HC153 的逻辑图，它包含两个完全相同的 4 选 1 数据选择器。两个数据选择器有公共的地址输入端，而数据输入端和输出端是各自独立的。通过给定不同的地址代码(即 A_1A_0 的状态)，即可从 4 个输入数据中选出所要的一个，并送至输出端 Y。图中的 S'_1 和 S'_2 是附加控制端，用于控制电路工作状态和扩展功能。

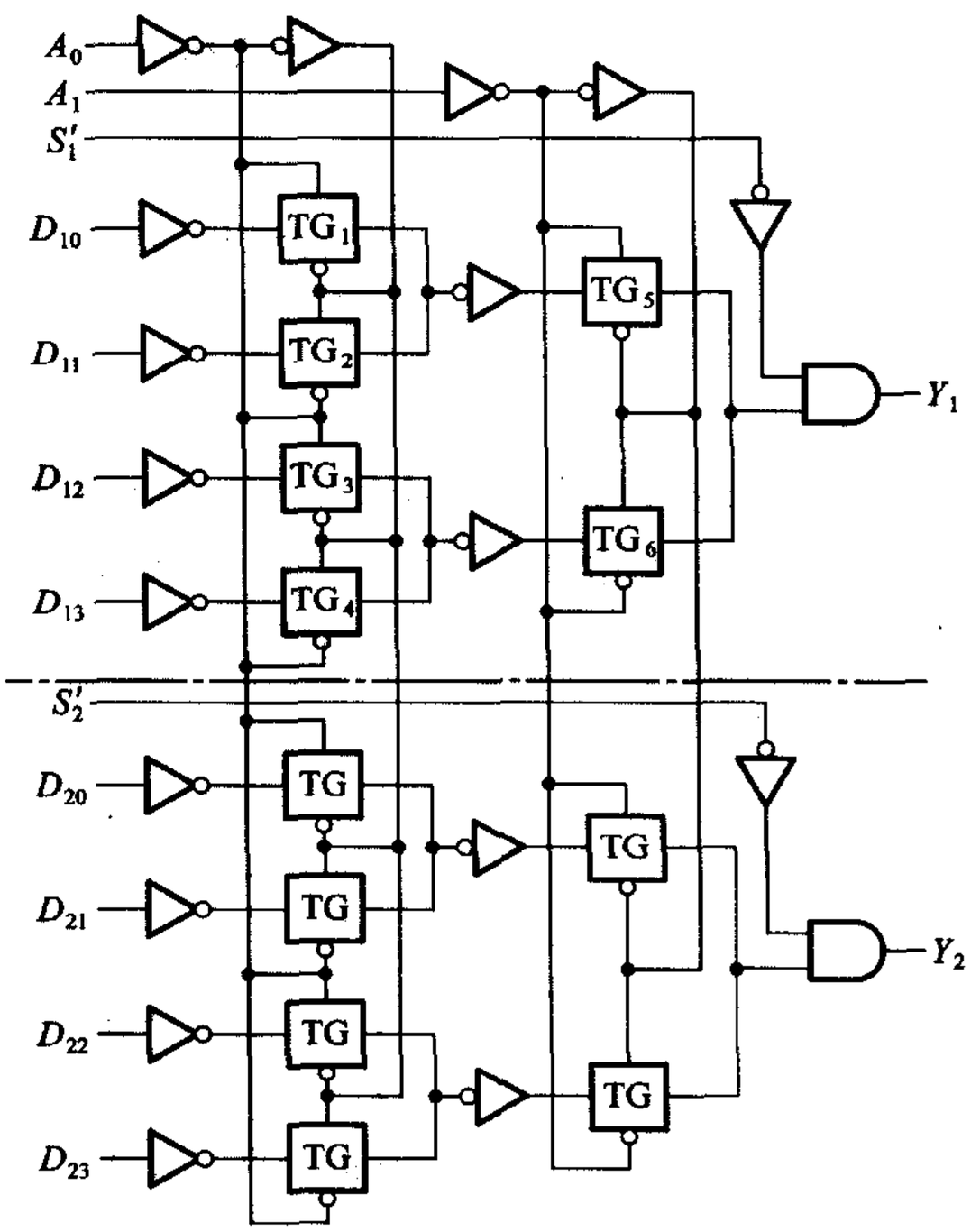


图 4.3.21　双 4 选 1 数据选择器 74HC153

由图 4.3.21 可见，当 $A_0 = \mathbf{0}$ 时传输门 TG_1 和 TG_3 导通，而 TG_2 和 TG_4 截止。当 $A_0 = \mathbf{1}$ 时 TG_1 和 TG_3 截止，而 TG_2 和 TG_4 导通。同理，当 $A_1 = \mathbf{0}$ 时 TG_5 导通、TG_6 截止。而 $A_1 = 1$ 时 TG_5 截止、TG_6 导通。因此，在 A_1A_0 的状态确定以后，$D_{10} \sim D_{13}$ 当中只有一个能通过两级导通的传输门到达输出端。例如，当 $A_1A_0 = \mathbf{01}$ 时，第一级传输门中的 TG_2 和 TG_4 导通，第二级传输门的 TG_5 导

通，只有 D_{11} 端的输入数据能通过传输门 TG_2 和 TG_5 到达输出端 Y_1。

输出的逻辑式可写成

$$Y_1 = [D_{10}(A_1'A_0') + D_{11}(A_1'A_0) + D_{12}(A_1A_0') + D_{13}(A_1A_0)] \cdot S_1' \qquad (4.3.16)$$

同时，上式也表明 $S' = 0$ 时数据选择器工作，$S' = 1$ 时数据选择器被禁止工作，输出被封锁为低电平。

【例 4.3.4】 试用两个带附加控制端的 4 选 1 数据选择器组成一个 8 选 1 数据选择器。

解： 如果使用两个 4 选 1 数据选择器，可以有 8 个数据输入端，是够用的。为了能指定 8 个输入数据中的任何一个，必须用 3 位输入地址代码，而 4 选 1 数据选择器的输入地址代码只有两位。第三位地址输入端只能借用控制端 S'。

用一片 74HC153 双 4 选 1 数据选择器，将输入的低位地址代码 A_1 和 A_0 接到芯片的公共地址输入端 A_1 和 A_0，将高位输入地址代码 A_2 接至 S_1'，而将 A_2' 接至 S_2'，同时将两个数据选择器的输出相加，就得到了图 4.3.22 所示的 8 选 1 数据选择器。

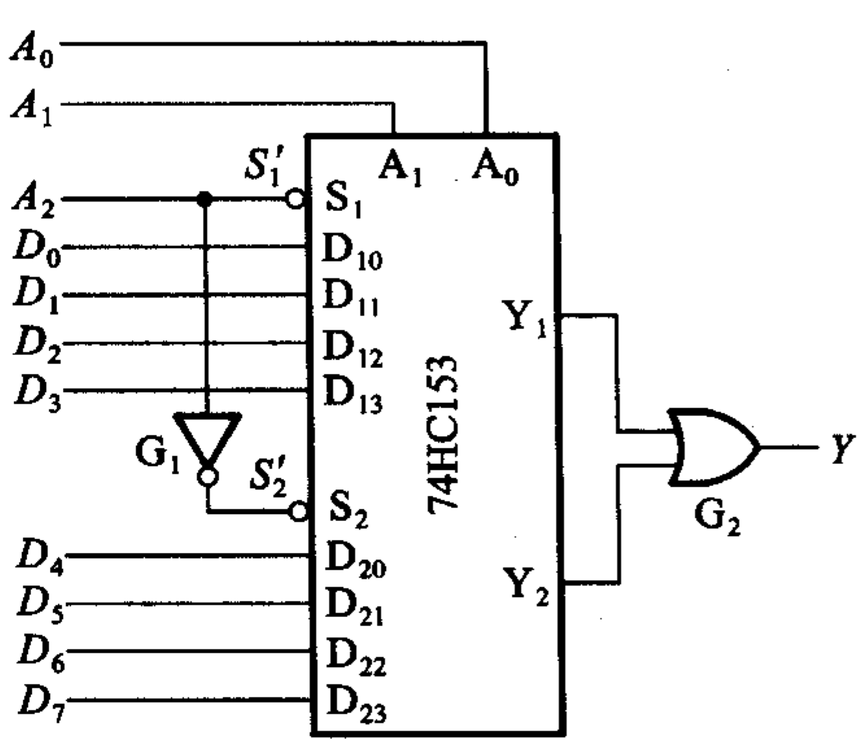


图 4.3.22 用两个 4 选 1 数据选择器接成的 8 选 1 数据选择器

当 $A_2 = 0$ 时上边一个数据选择器工作，通过给定 A_1 和 A_0 的状态，即可从 $D_0 \sim D_3$ 中选中某一个数据，并经过门 G_2 送到输出端 Y。反之，若 $A_2 = 1$，则下边一个 4 选 1 数据选择器工作，通过给定 A_1 和 A_0 的状态，便能从 $D_4 \sim D_7$ 中选出一个数据，再经过门 G_2 送到输出端 Y。

如果用逻辑函数式表示图 4.3.22 所示电路输出与输入间的逻辑关系，则得到

$$\begin{aligned} Y = {} & (A_2'A_1'A_0')D_0 + (A_2'A_1'A_0)D_1 + (A_2'A_1A_0')D_2 + (A_2'A_1A_0)D_3 \\ & + (A_2A_1'A_0')D_4 + (A_2A_1'A_0)D_5 + (A_2A_1A_0')D_6 + (A_2A_1A_0)D_7 \end{aligned} \qquad (4.3.17)$$

在需要对接成的 8 选 1 数据选择器进行工作状态控制时，只需在门 G_2 上增加一个控制输入端就够了（图中未画出）。

常见的数据选择器产品除“4 选 1”这种以外，还有“2 选 1”、“8 选 1”、“16 选 1”几种类型。它们的工作原理和上面所讲的 4 选 1 数据选择器类似，只是数据输入端和地址输入端的数目各不相同而已。

二、用数据选择器设计组合逻辑电路

由式(4.3.16)可见，具有两位地址输入 A_1、A_0 的 4 选 1 数据选择器在 $S = 1$

时输出与输入间的逻辑关系可以写成

$$Y = D_0(A_1'A_0') + D_1(A_1'A_0) + D_2(A_1A_0') + D_3(A_1A_0) \qquad (4.3.18)$$

若将 A_1、A_0 作为两个输入变量，同时令 $D_0 \sim D_3$ 为第三个输入变量的适当状态（包括原变量、反变量、**0** 和 **1**），就可以在数据选择器的输出端产生任何形式的三变量组合逻辑函数。

同理，用具有 n 位地址输入的数据选择器，可以产生任何形式输入变量数不大于 $n+1$ 的组合逻辑函数。

【例 4.3.5】 试用 4 选 1 数据选择器实现例 4.2.2 的交通信号灯监视电路。

解： 已知例 4.2.2 要求产生的逻辑函数为式(4.2.2)，即

$$Z = R'A'G' + R'AG + RA'G + RAG' + RAG \qquad (4.3.19)$$

将上式稍加变换即可化成与式(4.3.18)完全对应的形式

$$Z = R'(A'G') + R(A'G) + R(AG') + \mathbf{1} \cdot (AG) \qquad (4.3.20)$$

将式(4.3.20)与式(4.3.18)对照一下便知，只要令数据选择器的输入为

$$A_1 = A \qquad D_0 = R'$$

$$A_0 = G \qquad D_1 = D_2 = R$$

$$D_3 = \mathbf{1}$$

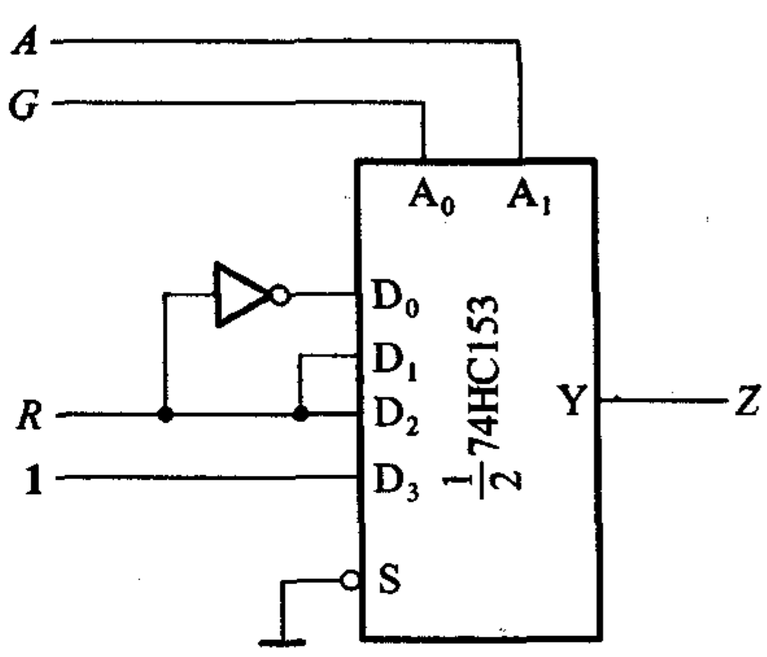


图 4.3.23　例 4.3.5 的电路

如图 4.3.23 所示，则数据选择器的输出就是式(4.2.2)所要求的逻辑函数 Z。

【例 4.3.6】 试用 8 选 1 数据选择器产生三变量逻辑函数

$$Z = A'\,B'\,C' + AC + A'BC \qquad (4.3.21)$$

解： 8 选 1 数据选择器有 3 位地址输入（$n=3$），能产生任何形式的四变量以下的逻辑函数，故定可生成式(4.3.21)的三变量逻辑函数。

图 4.3.24 中虚线框内部分是 8 选 1 数据选择器 74HC151 的逻辑图，在控制端输入 $S' = \mathbf{0}$（$S = \mathbf{1}$）的情况下，输出的逻辑式为

$$\begin{cases} Y = D_0(A_2'A_1'A_0') + D_1(A_2'A_1'A_0) + D_2(A_2'A_1A_0') + D_3(A_2'A_1A_0) + \\ \qquad D_4(A_2A_1'A_0') + D_5(A_2A_1'A_0) + D_6(A_2A_1A_0') + D_7(A_2A_1A_0) \\ W = Y' \end{cases} \qquad (4.3.22)$$

将式(4.3.21)化成与式(4.3.22)中 Y 对应的形式得到

$$\begin{aligned} Z &= A'\,B'\,C' + AC + A'BC \\ &= \mathbf{1} \cdot (A'\,B'\,C') + \mathbf{0} \cdot (A'\,B'\,C) + \mathbf{0} \cdot (A'\,B\,C') + \mathbf{1} \cdot (A'\,B\,C) + \\ &\quad \mathbf{0} \cdot (A\,B'\,C') + \mathbf{1} \cdot (A\,B'\,C) + \mathbf{0} \cdot (A\,B\,C') + \mathbf{1} \cdot (ABC) \end{aligned} \qquad (4.3.23)$$

将以上两式对照一下可知，只要令数据选择器的输入为

$$A_2 = A$$
$$A_1 = B \qquad D_0 = D_3 = D_5 = D_7 = \mathbf{1}$$
$$A_0 = C \qquad D_1 = D_2 = D_4 = D_6 = \mathbf{0}$$

则数据选择器的输出 Y 就是所需要的逻辑函数 Z。电路的接法如图 4.3.24 所示。

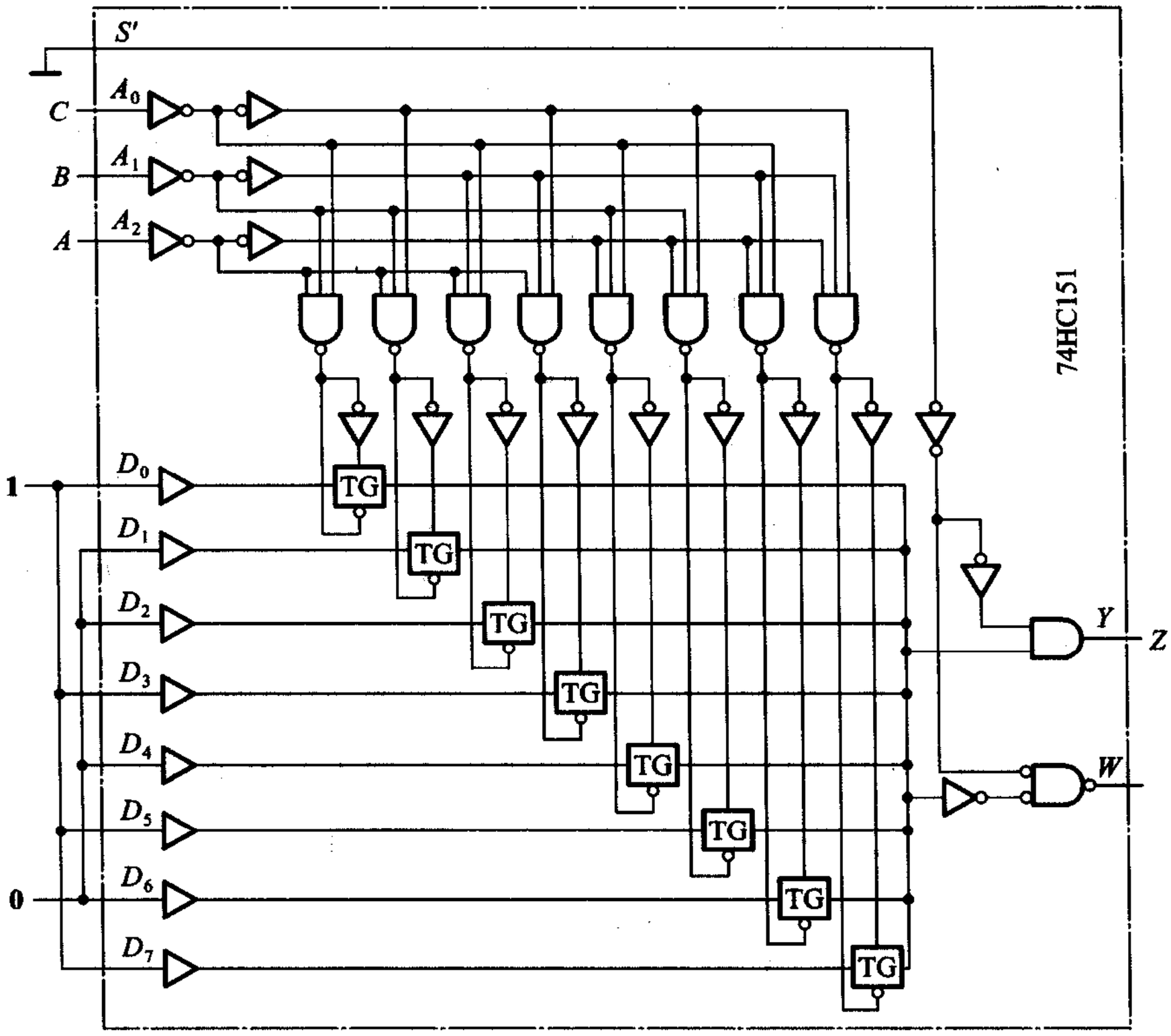


图 4.3.24 例 4.3.6 的电路

复习思考题

R4.3.4 数据选择器输入数据的位数和输入地址的位数之间应满足怎样的定量关系？

R4.3.5 如果用同样的一个 4 选 1 数据选择器产生同样的一个三变量逻辑函数，电路接法是否是唯一的？

4.3.4　加法器

两个二进制数之间的算术运算无论是加、减、乘、除，目前在数字计算机中都是化做若干步加法运算进行的。因此，加法器是构成算术运算器的基本单元。

一、1 位加法器

1. 半加器

如果不考虑有来自低位的进位将两个 1 位二进制数相加，称为半加。实现半加运算的电路称为半加器。

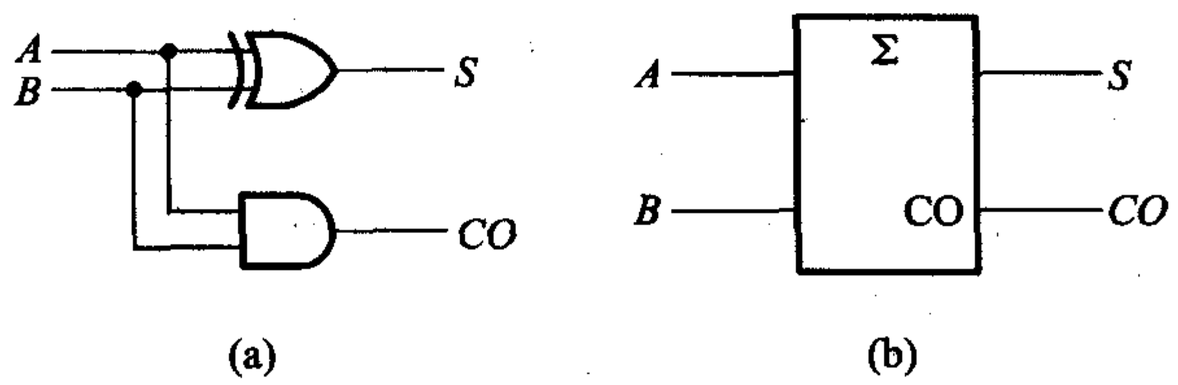
按照二进制加法运算规则可以列出如表 4.3.8 所示的半加器真值表，其中 A、B 是两个加数，S 是相加的和，CO 是向高位的进位。将 S、CO 和 A、B 的关系写成逻辑表达式则得到

$$\begin{cases} S = A'B + AB' = A \oplus B \\ CO = AB \end{cases} \tag{4.3.24}$$

表 4.3.8　半加器的真值表

输　入		输　出	
A	B	S	CO
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

因此，半加器是由一个**异或**门和一个**与**门组成的，如图 4.3.25 所示。



(a)　　(b)

图 4.3.25　半加器

（a）逻辑图　　（b）符号

2. 全加器

在将两个多位二进制数相加时，除了最低位以外，每一位都应该考虑来自低位的进位，即将两个对应位的加数和来自低位的进位 3 个数相加。这种运算称为全加，所用的电路称为全加器。

根据二进制加法运算规则可列出 1 位全加器的真值表，如表 4.3.9 所示。

表 4.3.9 全加器的真值表

输入			输出	
CI	*A*	*B*	*S*	*CO*
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

画出图 4.3.26 所示 *S* 和 *CO* 的卡诺图，采用合并 **0** 再求反的化简方法得到

$$\begin{cases} S = (A'B'CI' + AB'CI + A'BCI + AB\ CI')' \\ CO = (A'B' + B'CI' + A'CI')' \end{cases} \tag{4.3.25}$$

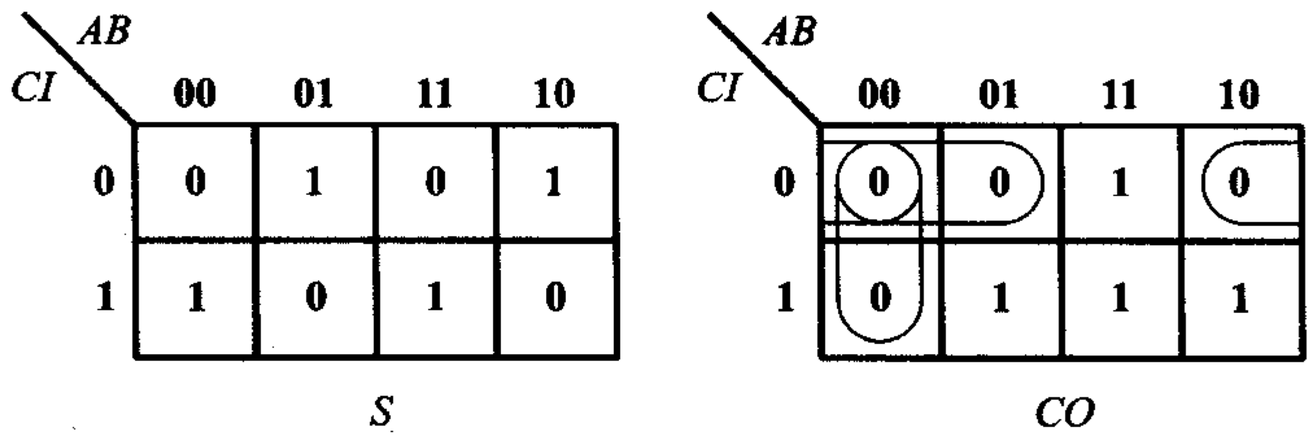


图 4.3.26 全加器的卡诺图

图 4.3.27(a)所示双全加器 74LS183 的逻辑图就是按式(4.3.25)组成的。全加器的电路结构还有多种其他形式，但它们的逻辑功能都必须符合表 4.3.9 给出的全加器真值表。

二、多位加法器

1. 串行进位加法器

两个多位数相加时每一位都是带进位相加的，因而必须使用全加器。只要依次将低位全加器的进位输出端 *CO* 接到高位全加器的进位输入端 *CI*，就可以构成多位加法器了。

图 4.3.28 就是根据上述原理接成的 4 位加法器电路。显然，每一位的相加结果都必须等到低一位的进位产生以后才能建立起来，因此将这种结构的电路称为串行进位加法器(或称为行波进位加法器)。

这种加法器的最大缺点是运算速度慢。在最不利的情况下，做一次加法运算需要经过 4 个全加器的传输延迟时间(从输入加数到输出状态稳定建立起来