

数字电路与系统设计

第一章 数制与编码

宋娟
软件学
院



jsong84@qq.com



第一章 数制与编码

三、编码

按照一定的规律，对数据或信息进行编码为二进制代码，便于存储和传输。

1. BCD 码：用二进制代码表示十进制数字（0~9）

需要 4 位二进制代码表示十进制数，总共有 16 种组合，从中选择 10 个表示 0~9。

常用的 BCD 码包括 8421、5421、2421BCD 码，每位有固定的权重，称为有权码；

余 3BCD 码没有固定的权重，为无权码



几种常用的 BCD 代码

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



(4) 用 BCD 代码表示十进制

数 对于一个多位的十进制数，需要将每位十进制数用相应的 BCD 代码来表示。例如：

$$(463.5)_{10} = \left[\begin{array}{cccc} \textcircled{0}100 & 0110 & 0011 & . \ 0101 \\ 4 & 6 & 3 & 5 \end{array} \right] \text{8421BCD}$$

不能省略!

$$(863.2)_{10} = \left[\begin{array}{cccc} 1110 & 1100 & 0011 & . \ 001\textcircled{0} \\ 8 & 6 & 3 & 2 \end{array} \right] \text{2421BCD}$$

不能省略!



第一章 数制与编码

- 有权码的记忆方法
 - 8421BCD 码取得是自然二进制码的前 10 位
 - 三种 BCD 码的前 5 组是相同的;
 - 只需要记 5 的 BCD 编码, 后续的在 5 的基础上累加
- 余 3BCD 码: 不能按权重展开, 无权码
 - 余 3BCD 码 = 8421BCD 码 + 0011
- 例 1 : $(01001000.1011)_{\text{余}3} = (?)_{5421\text{BCD}}$



第一章 数制与编码

2. 可靠性编码

(1) 格雷码:

任何相邻的两个码组(包括首、尾两个码组)中,只有一个码元不同,即单位距离特性。

由于首尾两个码组也具有单位距离特性,因而格雷码也叫**循环码**。

□

典型的 Gray



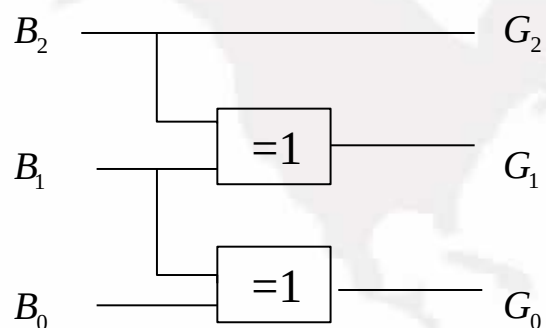
十进制数	二进制码	Gray 码
	$B_3B_2B_1B_0$	$G_3G_2G_1G_0$
0	0 0 0 0	0 0 0 <u>0</u>
1	0 0 0 1	0 0 0 <u>1</u>
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 <u>1</u> 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	<u>0</u> 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10	1 0 1 0	1 1 1 1
11	1 0 1 1	1 1 1 0
12	1 1 0 0	1 0 1 0
13	1 1 0 1	1 0 1 1
14	1 1 1 0	1 0 0 1
15	1 1 1 1	1 0 0 0

…一位反射对称轴

…二位反射对称轴

…三位反射对称轴

…四位反射对称轴



可以通过异或运算将 n 位自然码转换为 n 位格雷码

$$G_{n-1} = B_{n-1}$$

$$G_{n-2} = B_{n-1} \oplus B_{n-2}$$

$$\vdots$$

$$G_1 = B_2 \oplus B_1$$

$$G_0 = B_1 \oplus B_0$$



第一章 数制与编码

(2) 奇偶校验码

可以检测传输中错码的编码方式

奇校验码：数据位 +1 位校验码中“1”的个数为奇数

偶校验码：数据位 +1 位校验码中“1”的个数为偶数

例：数据位 11001011

那么奇校验编码结果为？

偶校验编码结果为？

接收端的校验方法：若采用奇校验，接收端检测“1”的个数，如果仍然为奇数，则认为无错，如果变为偶数，则认为有错

数字电路与系统设计

第二章 基本逻辑运算与 集成逻辑门

宋娟
软件学
院



jsong84@qq.com



内容概要

- 基本逻辑运算
 - 基本概念
 - 与、或、非逻辑运算的表示
- 复合逻辑运算
 - 与非、或非、与或非
 - 异或、同或运算
 - 逻辑运算的优先级与完备性
- 集成逻辑门
 - 集成逻辑门的参数
 - 集电极开路门
 - 三态门



二、复合逻辑运算

4. 异或 / 同或运算

(1) 异或：当输入两变量不同为 1，相同为 0

1) 真值表

<i>A</i>	<i>B</i>	<i>F</i>
0	0	0
0	1	1
1	0	1
1	1	0

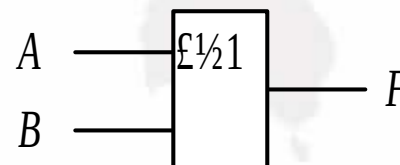
2) 逻辑表达式

$$F = A \oplus B = \bar{A}B + A\bar{B}$$

3) 逻辑符号



(b)



(c)



二、复合逻辑运算

4. 异或 / 同或运算

(2) 同或：当输入两变量相同为 1，不同为 0

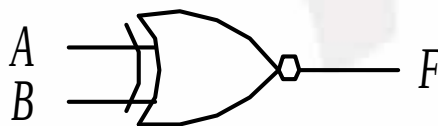
1) 真值表

<i>A</i>	<i>B</i>	<i>F</i>
0	0	1
0	1	0
1	0	0
1	1	1

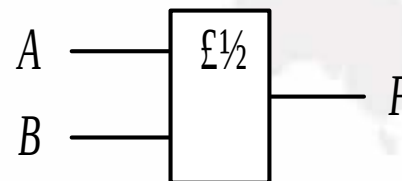
2) 逻辑表达式

$$F = A \odot B = \bar{A} \bar{B} + AB = \overline{A \oplus B}$$

3) 逻辑符号



(b)



(c)



二、复合逻辑运算

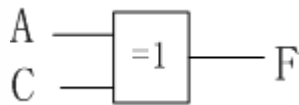
4. 异或 / 同或运算

常用逻辑表达式

$$0 \oplus 0 = 0 \quad 0 \oplus A = A \quad 0 \oplus 1 = 1 \quad A \oplus 1 = \bar{A}$$

$$1 \oplus 1 = 0 \quad A \oplus A = 0 \quad 1 \oplus 0 = 1 \quad A \oplus \bar{A} = 1$$

例：



当 $C=0$, $F=?$

当 $C=1$, $F=?$

可以作为控制电路，输出原变量或反变量！



二、复合逻辑运算

- 异或运算的奇偶特性

奇数个 1 相异或结果为

1

偶数个 1 相异或结果为

0

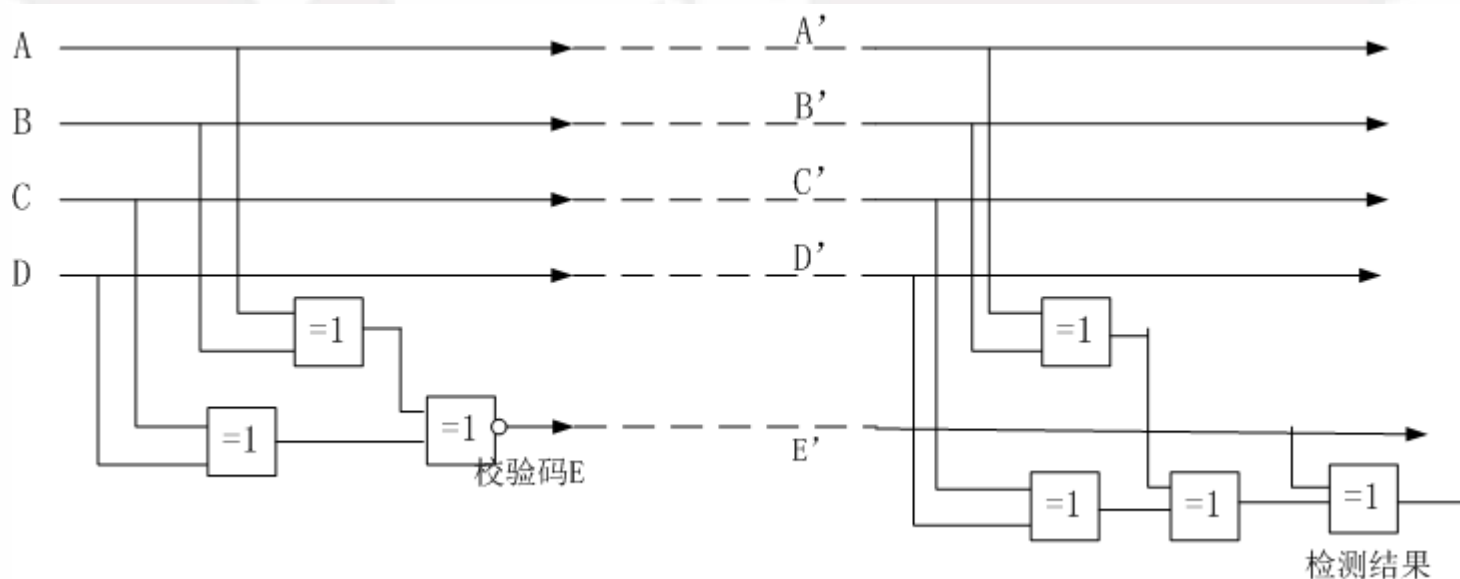
异或运算的奇偶特性

$$F = 1 \oplus 0 \oplus A \oplus A \oplus 1 \oplus 1 = ?$$



异或运算的奇偶特性

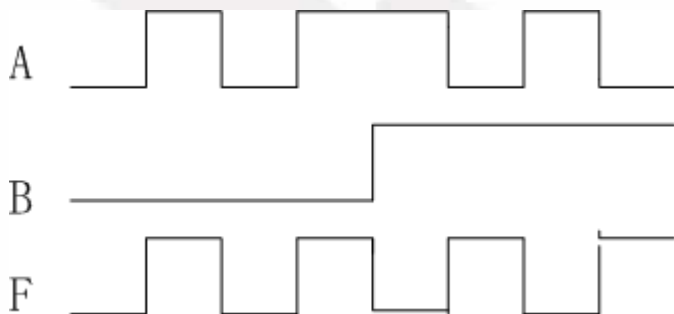
- 可以利用异或电路实现奇偶校验的产生和检测





二、复合逻辑运算

例 2. 已知输入 A、B 和输出 F 的波形，由此判断 F 与 A、B 的逻辑关系为 ()



$$A.F = \overline{AB}$$

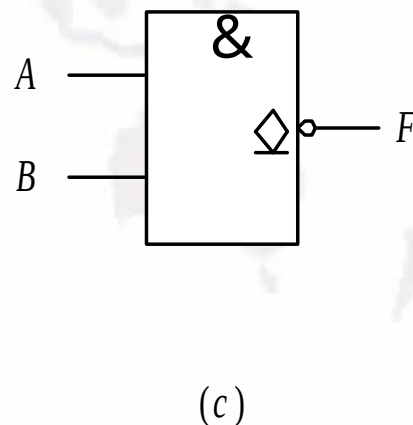
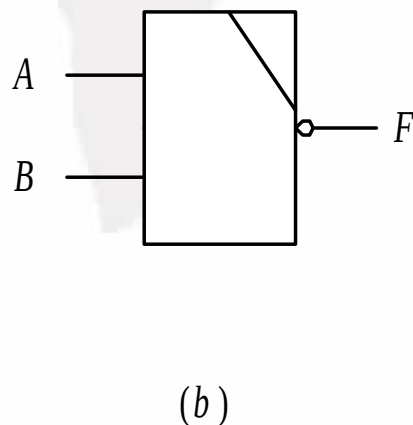
$$B.F = \overline{A+B}$$

$$C.F = A \oplus B$$

$$D.F = A \odot B$$

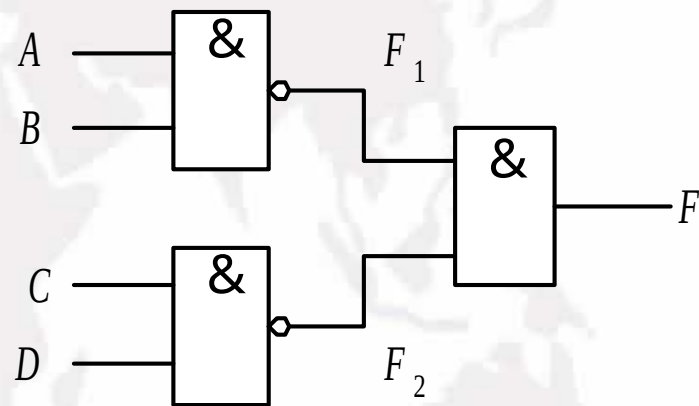
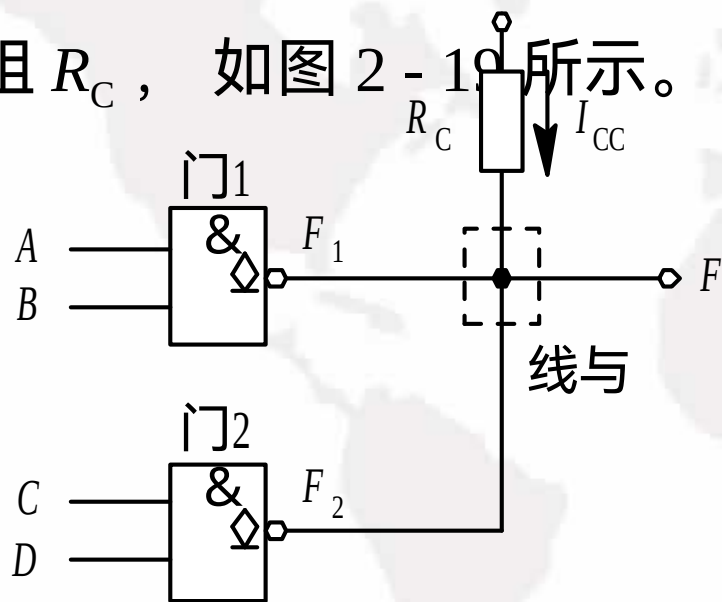


- OC 门和三态门是允许输出端直接并接在一起的两种 TTL 门
- 集电极开路门（OC 门）





(1) 电路结构及功能分析。OC 门的电路特点是其输出管的集电极开路。使用时，必须外接“上拉电阻 R_C ”和 $+U_{CC}$ 相连。多个 OC 门输出端相连时，可以共用一个上拉电阻 R_C ，如图 2 - 19 所示。



(a) 图 2 - 19 多个 OC 门并联

(b)

(a) 线与逻辑电路； (b) 等效逻辑图



- OC 门接入上拉电阻 R_C 后，与图 2 - 13 所示的与非门的差别仅在于用外接电阻 R_C 取代了由 V_3 和 V_4 构成的有源负载。
 - 当 F_1 、 F_2 输出低电平时， F 为低电平；
 - 当 F_1 、 F_2 输出高电平时， F 也为高电平；
 - 如果至少一个 OC 输出处在低电平（即逻辑 0），那么 R_C 会吸收电流、将输出线拉到低电平。

$$F = F_1 \square F_2$$

线与

因为 R_C 较大，无论两个 OC 门处于何种状态，在 V_{CC} 和地之间都不会出现低阻通路，电路可以安全工作

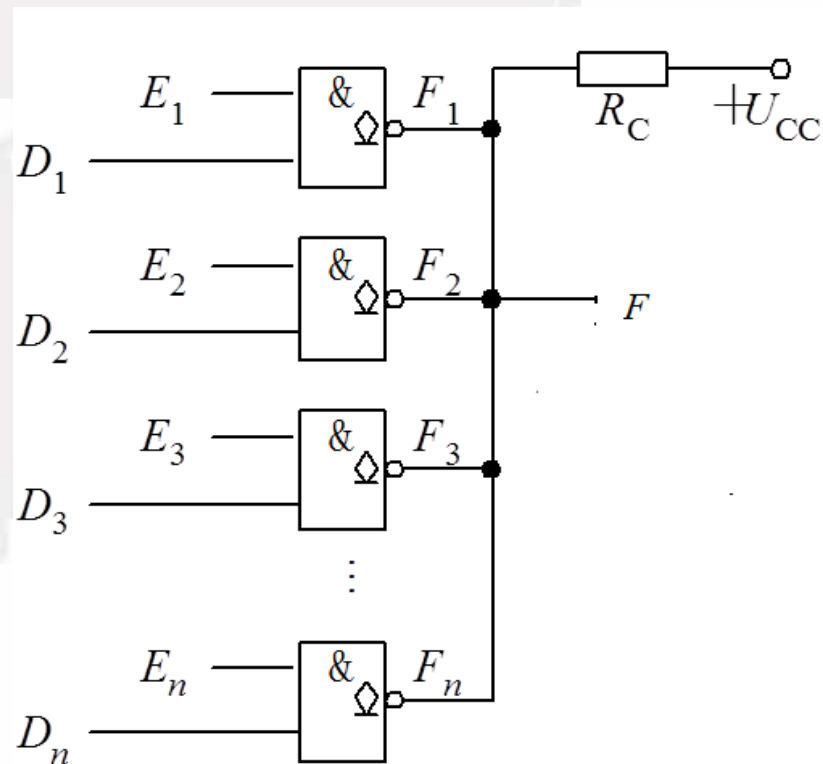


四、集成逻辑门

- 应用

- 实现多路信号在总线 (母线) 上的分时传输,
- 实现“与或非”逻辑运算

$$F = \overline{AB \square CD} = \overline{AB + CD}$$

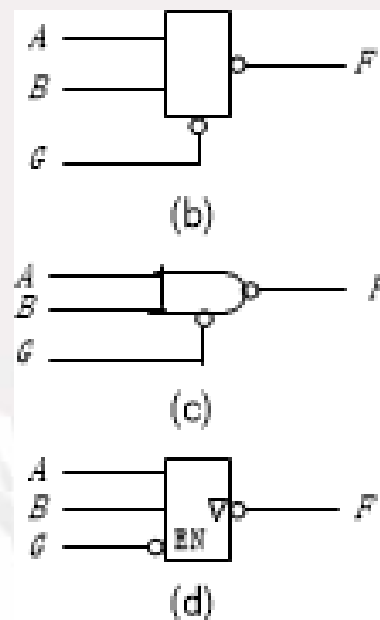




四、集成逻辑门

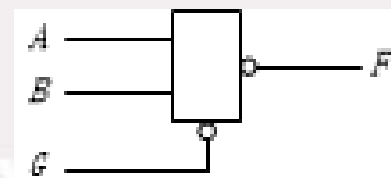
- 三态门

- 在很多系统中，采用总线结构，方便的增加或去掉某些外设，而不改变原有的结构。要求未被选中的外设接口挂在总线上的状态与总线脱钩，消除对总线的影晌
- 三态门出了有两个逻辑电平外，还有第三种状态：高阻态，相当于悬空，输出不会对其他相连的设备产生影响

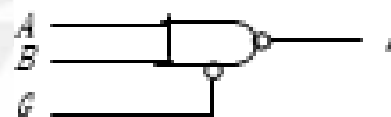




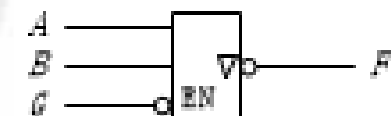
- 和普通 TTL 不同，多了一个控制端 G ，
- 当 $G=0$ 时, $F = \overline{AB}$ ，
- 当 $G=1$ 时，输出高阻态



(b)



(c)



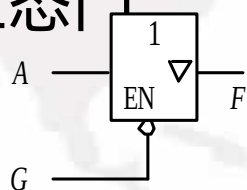
(d)



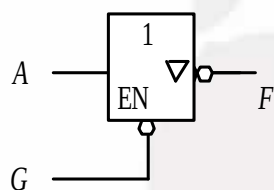
(2) 分类。三态门可以按如下的方式分类：

① 按逻辑功能分，如三态与非门、三态缓冲门、三态非门（三态倒相门）、三态与门。

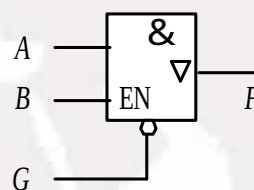
② 按控制模式分为两类，即低电平有效的三态门和高电平有效的三态门



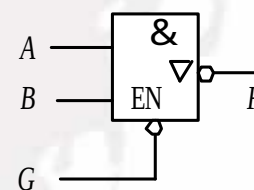
三态缓冲门



三态倒相门

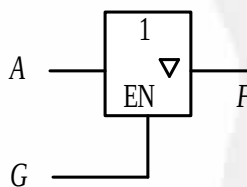


三态与门

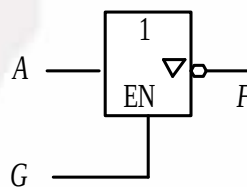


三态与非门

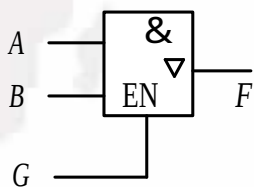
(a)



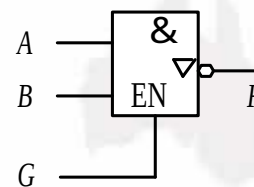
三态缓冲门



三态倒相门



三态与门



三态与非门

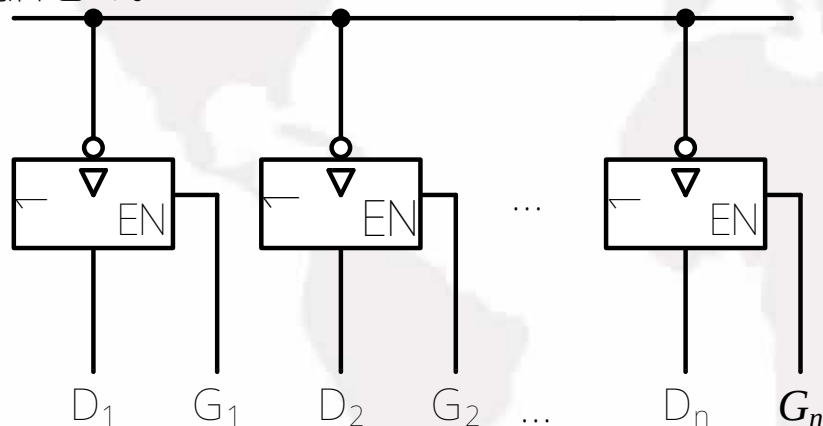
(b)



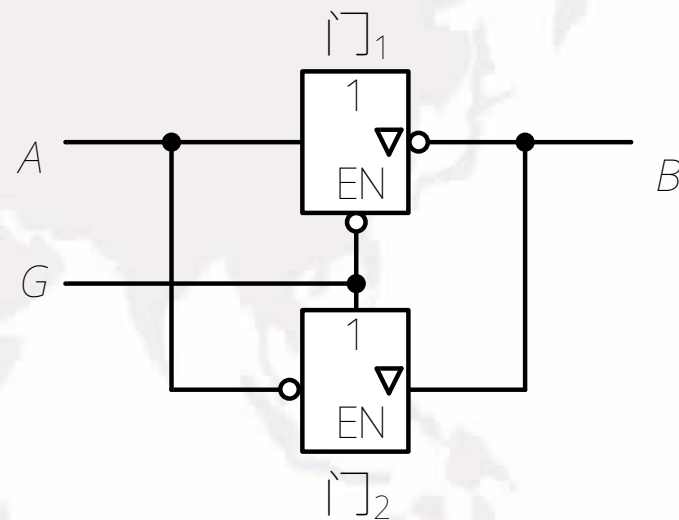
三态门的应用

- (1) 多路数据在总线上的分时传送
- (2) 实现数据的可控双向传送

数据总线



(a)



(b)

数字电路与系统设计

第三章 布尔代数与逻辑 函数化简

宋娟
软件学
院



jsong84@qq.com



内容概要

- 基本公式和三大法则
 - 基本公式
 - 三大法则
- 逻辑函数的代数法化简
 - 化简原则
 - 化简方法
- 卡诺图化简
 - 最小项标准式
 - 卡诺图表示与化简
 - 各种逻辑形式的化简



3.1 基本公式和法则

1. 基本公式

0-1 律 $0 \square A = 0$ $1 + A = 1$

自等律 $A \square 1 = A$ $A + 0 = A$

互补律 $A \square \bar{A} = 0$ $A + \bar{A} = 1$

等幂律 $A \square A = A$ $A + A = A$

交换律 $A \square B = B \square A$ $A + B = B + A$

结合律 $A \square (B \square C) = (A \square B) \square C$ $A + (B + C) = (A + B) + C$

分配律 $A \square (B + C) = AB + AC$ $A + B \square C = (A + B) \square (A + C)$



3.1 基本公式和法则

1. 基本公式

含有相同的变量，其中只有一项取值不同

吸收律 1：两个**逻辑相邻项**可以合并为一项，保留相同的变量，消去不同的变量

$$(A + B) \square (A + \bar{B}) = A \quad AB + A\bar{B} = A$$

在逻辑函数化简中使用最为广泛

吸收律 2 若式中包含某单因子项，则包含该因子的其余项为多余项。

$$A + AB = A \quad A \square (A + B) = A$$

当某一项完全包含了另外一项，包含的这一项就可以被消去



3.1 基本公式和法则

1. 基本公式

吸收律 3：若式中包含某单因子项，其他项含有该因子的反因子，则该反因子可消去

$$A(\bar{A} + B) = AB \quad A + \bar{A}B = A + B$$

证明：

$$A + \bar{A}B = A + AB + \bar{A}B = A + B$$



3.1 基本公式和法则

- 多余项定律： 当一个变量分别以原变量 / 反变量处在别的项中，那含有该两项余下的变量构成的项为多余项，可消去

$$AB + \bar{A}C + BC = AB + \bar{A}C$$

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

- 多余项定理的证明

$$AB + \bar{A}C + BC = AB + \bar{A}C + BC(A + \bar{A})$$

- 多余项定理的推广

$$AB + \bar{A}C + \overline{BCDE + CF} = AB + \bar{A}C$$



3.1 基本公式和法则

摩根律 / 反演律

$$\begin{cases} \overline{AB} = \bar{A} + \bar{B} \\ \overline{A + B} = \bar{A}\bar{B} \end{cases}$$

$A B$	$\overline{A+B}$	\overline{AB}	\overline{AB}	$\bar{A} + \bar{B}$
0 0	1	1	1	1
0 1	0	0	1	1
1 0	0	0	1	1
1 1	0	0	0	0



3.1 基本公式和三大法则

1. 代入法则：逻辑等式中的任何变量 A ，都可用另一函数 Z 代替，等式仍然成立。

代入法则可以扩大基本公式的应用范围。

例 1 证 $\overline{A+B+C} = \bar{A} \cdot \bar{B} \cdot \bar{C}$

明令 $Z = A + B$

同理可将摩根定律推广到 n 变量



3.1 基本公式和三大法则

2. 对偶法则：原式 F 成立，则其对偶式 G 也一定成立。

对偶式的求法：

将其中的“ $+$ ”换成“ \cdot ”，“ \cdot ”换成“ $+$ ”，“ 1 ”换成“ 0 ”，

“ 0 ”换成“ 1 ”，并保持原先的逻辑优先级，变量不变，两变量以上的非号不动。

可以减少公式的记忆量



3. 反演法则 □

由原函数求反函数，称为反演或求反。摩根定律是进行反演的重要工具。多次应用摩根定律，可以求出一个函数的反函数。

□ 例：用摩根律求 $F = A + B + \bar{C} + \overline{D + E}$ 的反函数

反演法则：将原函数 F 中的“ \cdot ”换成“ $+$ ”，“ $+$ ”换成“ \cdot ”；“ 0 ”换成“ 1 ”，“ 1 ”换成“ 0 ”；原变量换成反变量，反变量换成原变量，长非号即两个或两个以上变量的非号不变，即可得反函数。

例 $F = \bar{A}B + C\bar{D} \quad \overline{\overline{A}\bar{C}} + \overline{\overline{B}C}$



4. 综合例子

例 3：函数的 $F = AB + \bar{A}C + \bar{B}C$ 最简与或式为？

A. $AB + \bar{A}C + \bar{B}C$ B. $AB + \bar{A}C$

C. $\bar{A}C + \bar{B}C$ D. $AB + C$



3.3 卡诺图法化简

2. 最小项标准式

最小项：对于一个给定变量数目的逻辑函数，所有变量参加相“与”的项叫做最小项。在一个最小项中，每个变量只能以原变量或反变量出现一次。

两变量有 4 个最小项： $\bar{A}\bar{B}$, $\bar{A}B$, $A\bar{B}$, AB

三变量 8 个最小项： $\bar{A}\bar{B}\bar{C}$, $\bar{A}\bar{B}C$, $\bar{A}B\bar{C}$, ...

N 变量： 2^n 个最小项

最小项标准式：全是由最小项组成的“与或”式，便是最小项标准式（不一定由全部最小项组成）。



3. 卡诺图的结构

- 把图中的几何相邻和逻辑相邻关系一一对应起来

$A \backslash B$	0	1
0	$\bar{A}\bar{B}$ m_0	$A\bar{B}$ m_2
1	$\bar{A}B$ m_1	AB m_3

格雷码

$AB \backslash C$	00	01	11	10
0	$\bar{A}\bar{B}\bar{C}$ m_0	$\bar{A}B\bar{C}$ m_2	$AB\bar{C}$ m_6	$A\bar{B}\bar{C}$ m_4
1	$\bar{A}\bar{B}C$ m_1	$\bar{A}BC$ m_3	ABC m_7	$A\bar{B}C$ m_5

$AB \backslash CD$	00	01	11	10
00	$\bar{A}\bar{B}\bar{C}\bar{D}$ m_0	$\bar{A}B\bar{C}\bar{D}$ m_4	$AB\bar{C}\bar{D}$ m_{12}	$A\bar{B}\bar{C}\bar{D}$ m_8
10	$\bar{A}\bar{B}C\bar{D}$ m_1	$\bar{A}BC\bar{D}$ m_5	$ABC\bar{D}$ m_{13}	$A\bar{B}C\bar{D}$ m_9
11	$\bar{A}\bar{B}CD$ m_3	$\bar{A}BCD$ m_7	$ABCD$ m_{15}	$A\bar{B}CD$ m_{11}
10	$\bar{A}\bar{B}C\bar{D}$ m_2	$\bar{A}BC\bar{D}$ m_6	$ABC\bar{D}$ m_{14}	$A\bar{B}C\bar{D}$ m_{10}

几何相邻
还包括首
尾相邻



4 逻辑函数的卡诺图表示法

(1) 函数是最小项标准式

$$F = ABC + AB\bar{C} + A\bar{B}C + \bar{A}\bar{B}C = m_7 + m_6 + m_5 + m_1$$

(2) 不是最小项标准式，先整理成最小项标准式

$$F = B\bar{C} + C\bar{D} + \bar{B}CD + \bar{A}\bar{C}D + ABCD$$



利用卡诺图化简最简与或式

步骤:

- (1) 用卡诺图表示原始函数
- (2) 用卡诺图合并最小项, 圈住所有的“1”;
- (3) 将卡诺圈的结果相或, 得到最终的化简函数。

注意事项:

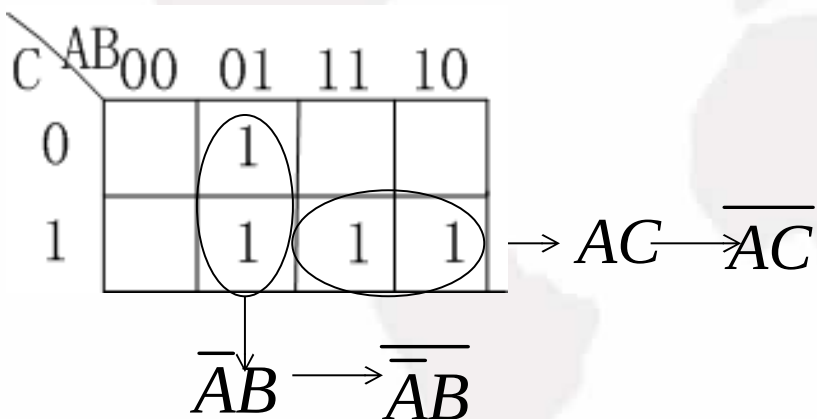
- (1) 要圈住所有的“1”;
- (2) 卡诺圈要尽可能的大, 圈越大, 输入变量越少;
- (3) 卡诺圈要尽可能的少, 圈越少, 与项越少;
- (4) 不要重复圈, 若某圈中的“1”均被其余圈圈过, 则是一个多余圈



其他逻辑形式的化简

(1) 最简与非式：将与或式二次取反，用摩根律展开求得与非式

例： $F = \bar{A}B + AC$ 化简为最简与非式



$$F = \bar{\bar{A}C} \cdot \bar{\bar{A}B}$$

步骤：圈“1”取反得每个与非项，将全部与非项再与非得到与非式

例： $F = \bar{B}CD + B\bar{C} + \bar{A}CD + A\bar{B}C$



其他逻辑形式的化简

(2) 最简与或非式：将反函数化为最简与或式，再求非一次得到 F 的最简与或非式

例： $F = \bar{A}B + AC$ 化简为最简与或非

式

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

$\bar{A}\bar{B}$

圈 0 得到反函数的最简与或

$$A\bar{C} \quad \text{或} = A\bar{C} + \bar{A}\bar{B}$$

再取反得到原函数 F 的最简与或非式

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



其他逻辑形式的化简

(3) 最简或与式：将最简与或非式用两次摩根律展开

例：式 $F = \overline{A}B + AC$ 化简为最简或与式

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

$$\overline{A}\overline{B} \longrightarrow \overline{\overline{A}\overline{B}} \longrightarrow A + B$$

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

$$\rightarrow AC \xrightarrow{\text{取反}} \overline{AC} \xrightarrow{\text{摩根律}} (\overline{A} + C)$$

圈 0 得到与或非式的与项

将每个与项取反

在用摩根律展开得到或项

最后将或项相与得到或与式



其他逻辑形式的化简

(4) 最简或非式：将最简或与式二次取反，用摩根律展开一次

例：式 $F = \bar{A}B + AC$ 化简为最简或非

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

$\bar{A}\bar{B} \longrightarrow \overline{A+B}$

$\bar{A}\bar{C} \xrightarrow{\text{摩根律}} \overline{(\bar{A}+C)}$

圈 0 得到与或非式的与项
将每个与项用摩根律展开得到或非项
最后将或非项相或非得到或非式



其他逻辑形式的化简

- 例求 $F(A, B, C, D) = \sum (0, 4, 5, 7, 8, 12, 13, 14, 15)$ 的最简与或式、与非式、与或非式、或与式和或非式



含有无关项逻辑函数的化简

- 实际的逻辑问题中，变量的某些取值组合不允许出现，或者是变量之间具有一定的制约关系。我们将这类问题称为非完全描述
 - 例如：8421BCD 编码
 - 例如：十字路口红绿灯
- 无关项：在逻辑函数中不允许出现的最小项，也称为约束项



含有无关项逻辑函数的化简

- 化简原则：若该无关项有利于函数化简，则圈进卡诺圈，若不利于函数化简，则不圈进卡诺圈。
- 原因：该无关项不可能出现，所以可以对它任意赋值，不影响函数的结果



含有无关项逻辑函数的化简

- 例：化简 $F(A, B, C, D) = \sum (0, 1, 4, 10, 11, 14)$ ，约束条件为 $BD=0$

数字电路与系统设计

第四章 组合逻辑电路

宋娟
软件学
院



jsong84@qq.com



内容概要

- 组合逻辑电路的分析
- 组合逻辑电路的设计
- 常用中规模组合逻辑器件的原理和应用
 - 半加器和全加器
 - 编码器和译码器
 - 数据选择器
 - 数据比较器
- 组合逻辑电路的竞争与冒险
 - 竞争与冒险
 - 判别
 - 消除



4.1 组合逻辑电路的分析

组合逻辑电路的分析过程如下：

- (1) 由给定的逻辑电路图， 写出输出端的逻辑表达式；
- (2) 列出真值表；
- (3) 从真值表概括出逻辑功能；
- (4) 对原电路进行改进设计， 寻找最佳方案（这一步不一定都要进行）。



4.2 组合逻辑电路的设计

电路设计的任务就是根据功能设计电路。一般按如下步骤进行：□

- (1) 将文字描述的逻辑命题变换为真值表 □
- (2) 进行函数化简，化简形式应依据选择什么门而定。
- (3) 根据化简结果和选定的门电路，画出逻辑电路。

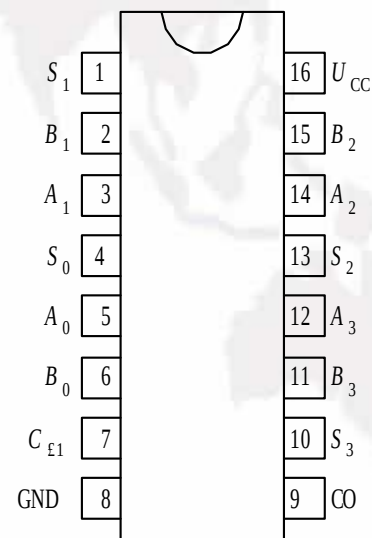
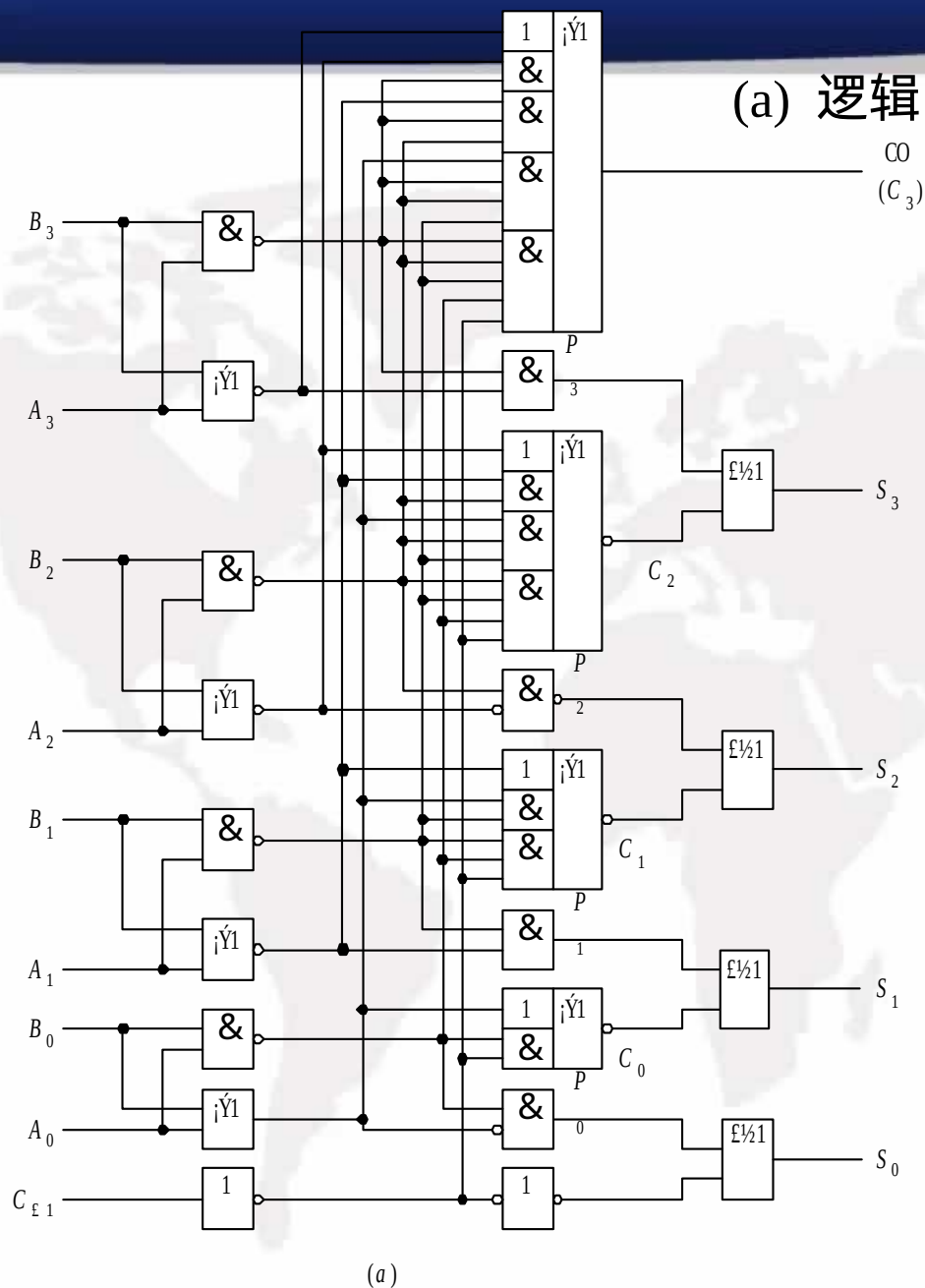


4.3 常用中规模组合逻辑部件原理与应用

- 半加器 / 全加器
- 编 / 译码器
- 数据选择器
- 数据比较器

图 4-16 74LS283 逻辑图与
引脚图

(a) 逻辑图; (b) 引脚图





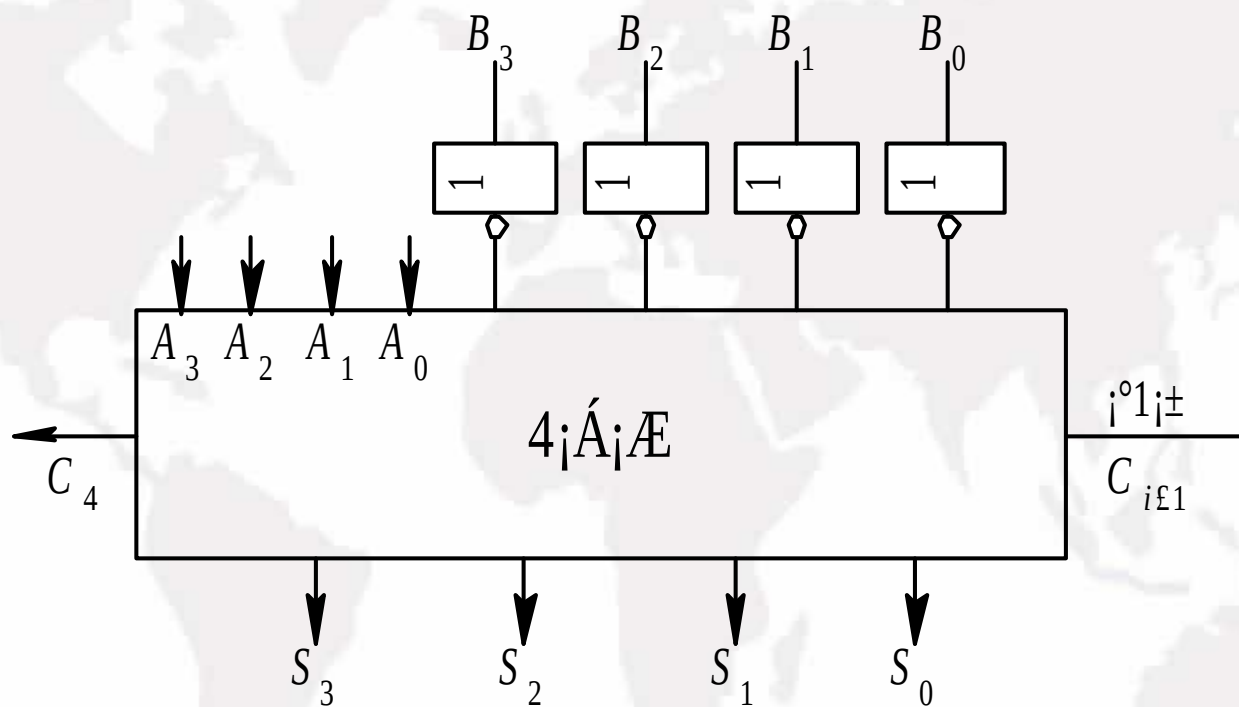
4. 全加器的应用

- 1) 实现多位二进制加法
 - 2) 利用全加器实现二进制减法
- 利用“加补”的概念，即可将减法用加法来实现



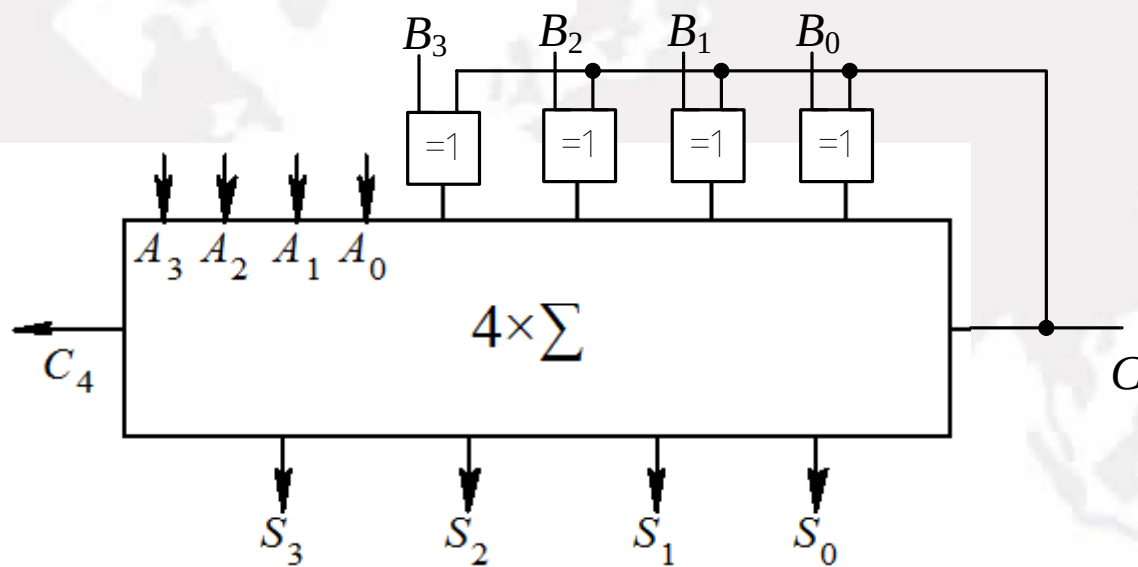
例 6 试用全加器构成二进制减法器。

□





例 7 利用一个 4 位全加器同时实现加法和减法电路： $c=0$ 时做加法， $c=1$ 时做减法 □

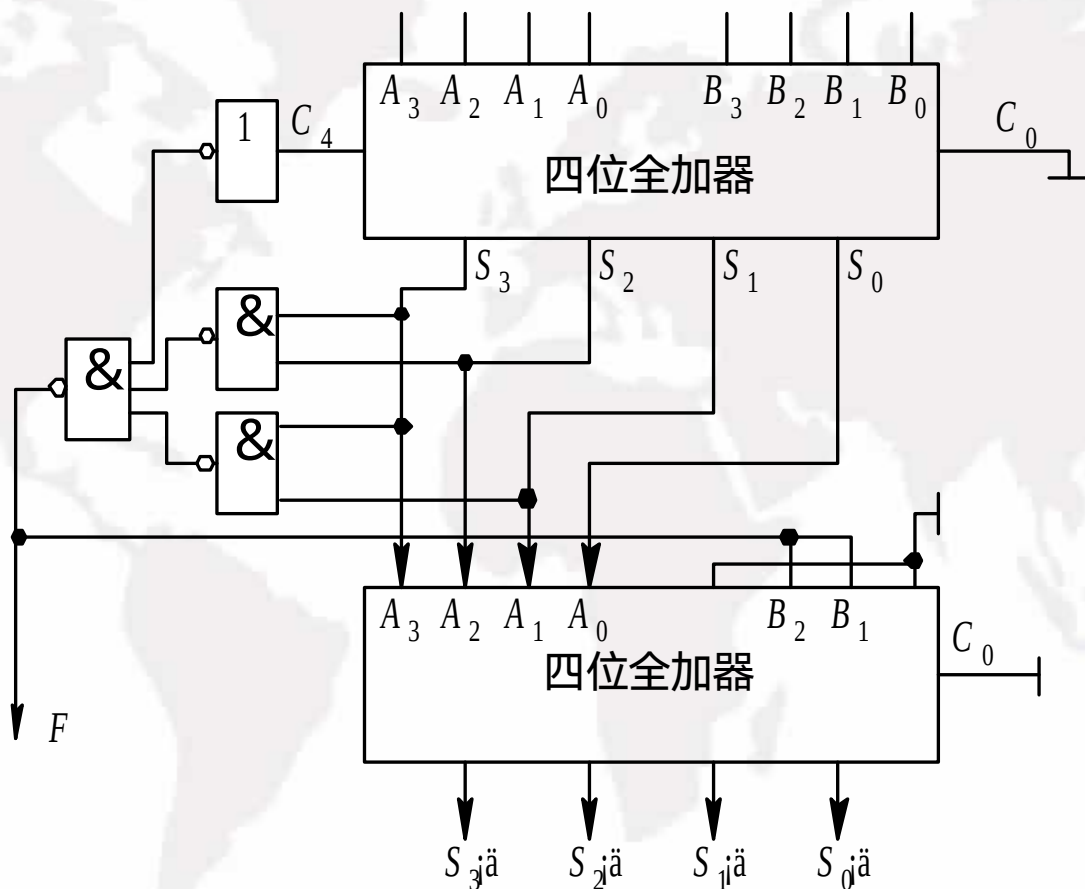




应用 3：完成十进制加法即 8421BCD 码的加法

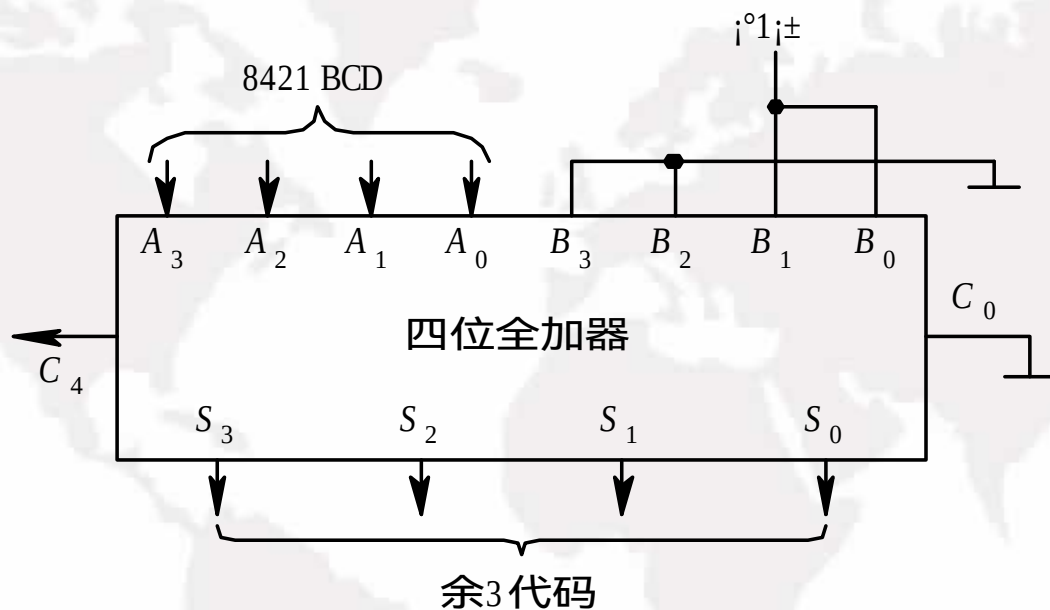
法

例 8 试用四位全加器构成一位 8421 码的加法电路。□





应用 5：完成码制转换



例：利用一个 4 位全加器完成 8421BCD 码和余 3BCD 码的双向转换



4.3.2 编码器与译码器

- **编码：**将特定的信息或数据用二进制代码表示
- **译码：**编码的逆过程，将输入的二进制代码转换成特定的信息。

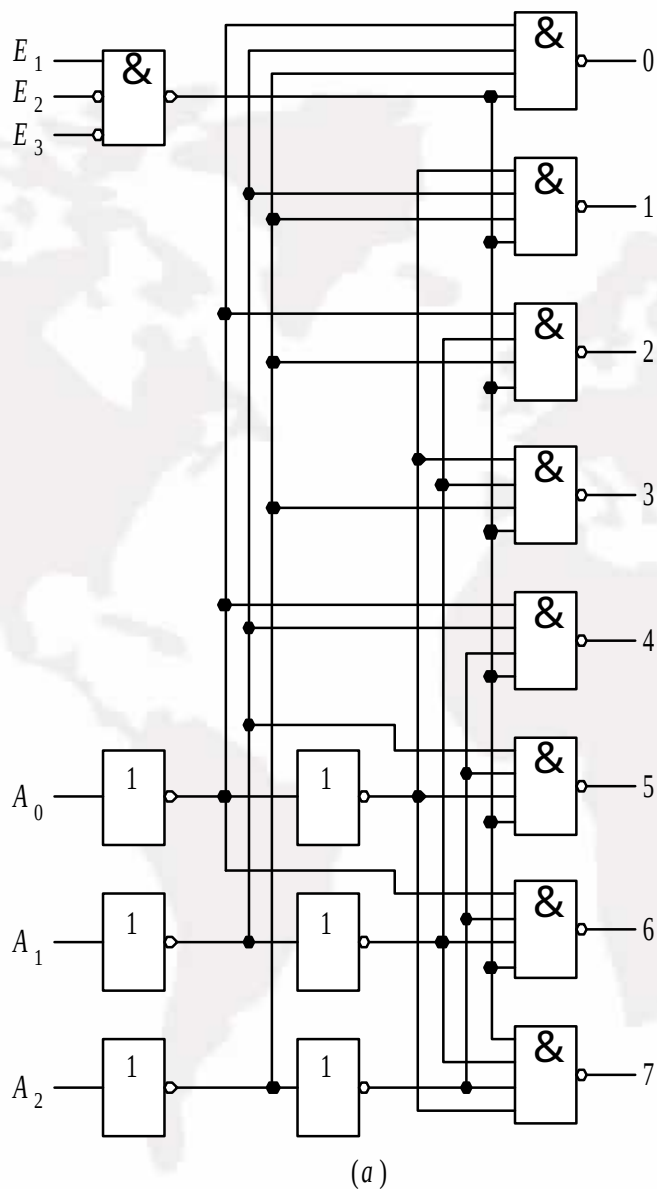
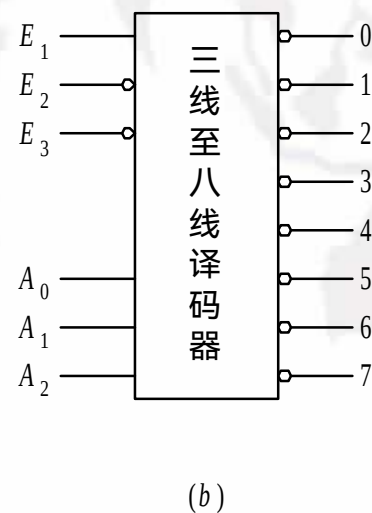


图 4-35 集成 3-8 译码器 (74LS138) 的电路图和逻辑符号





利用使能端进行功能扩展

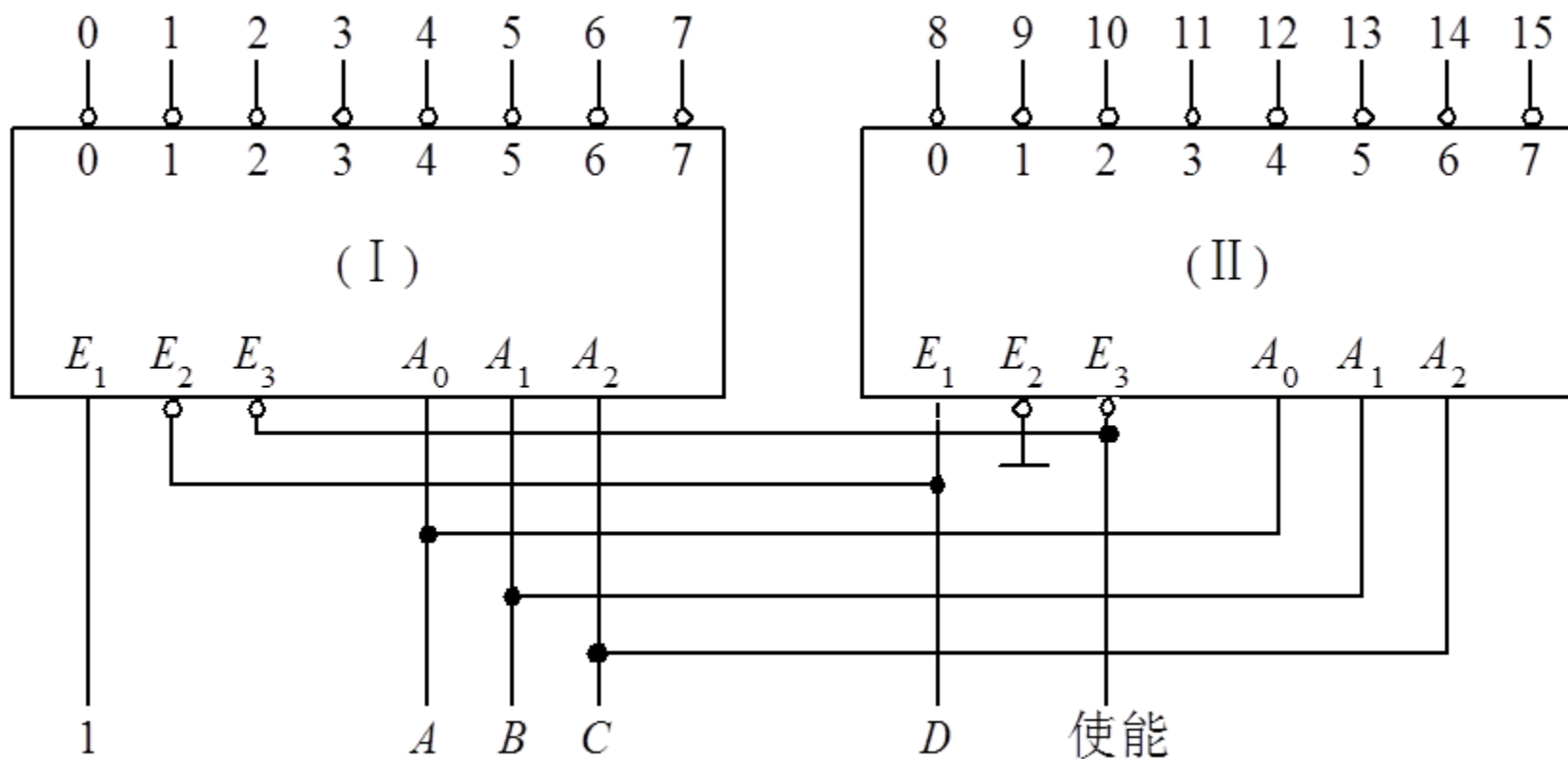


图 4-36 3 - 8 译码器扩大为 4 - 16 译码器



译码器应用

(1) 利用译码器实现组合逻辑函数，设计组合逻辑电路

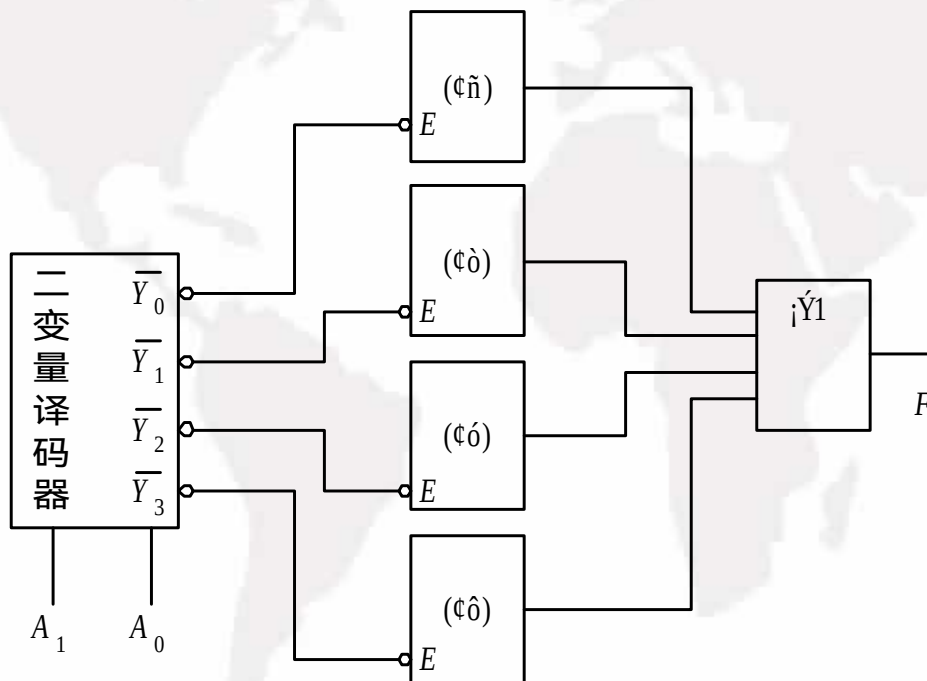
[例 1] 用译码器 74LS138 和若干与非门，设计实现一个代码转换器，要求将 3 位步进码 CBA 转换成二进制码 $F_3F_2F_1$ ，编码表如下。

输 入			输 出		
C	B	A	F_3	F_2	F_1
0	0	0	0	0	0
1	0	0	0	0	1
1	1	0	0	1	0
1	1	1	0	1	1
0	1	1	1	0	0
0	0	1	1	0	1



译码器的应用

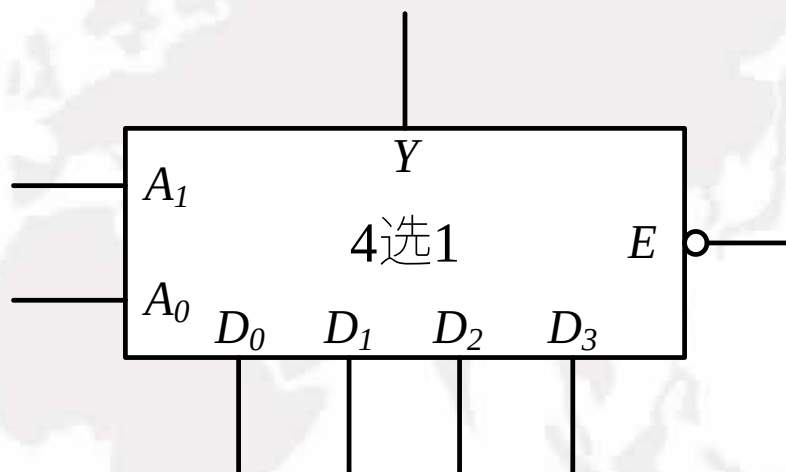
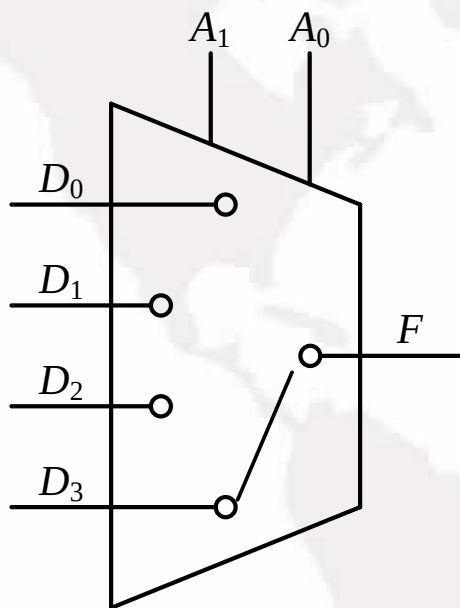
2) 译码器可以在数字系统中作为其它芯片的片选信号
译码器的输出连接到三态门的使能端，进行数据的分时传送等





4.3.3 数据选择

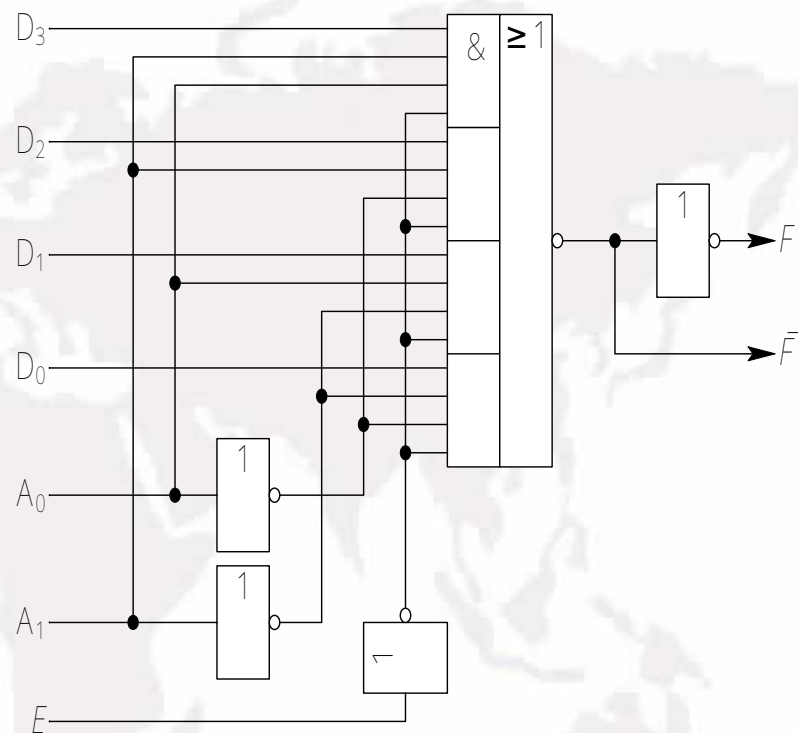
在地址信号的控制下，从多路输入数据中选择一路作为输出





功能表

地址		选通	数据	输出
A_1	A_0	E	D	F
×	×	1	×	0
0	0	0	$D_0 \sim D_3$	D_0
0	1	0	$D_0 \sim D_3$	D_1
1	0	0	$D_0 \sim D_3$	D_2
1	1	0	$D_0 \sim D_3$	D_3



$$F = (\bar{A}_1 \bar{A}_0 D_0 + \bar{A}_1 A_0 D_1 + A_1 \bar{A}_0 D_2 + A_1 A_0 D_3) \bar{E}$$

$$= \left(\sum_{i=0}^3 m_i D_i \right) \bar{E}$$



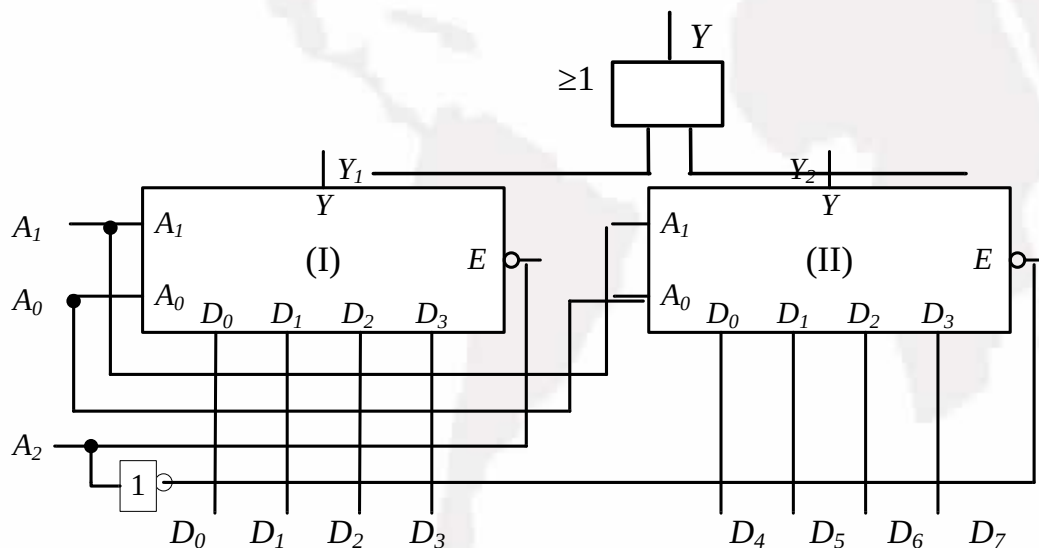
常用集成数据选择器有如下几种：□

(1) 四选一数据选择器 74LS153；□

(2) 八选一数据选择器 74LS151；□

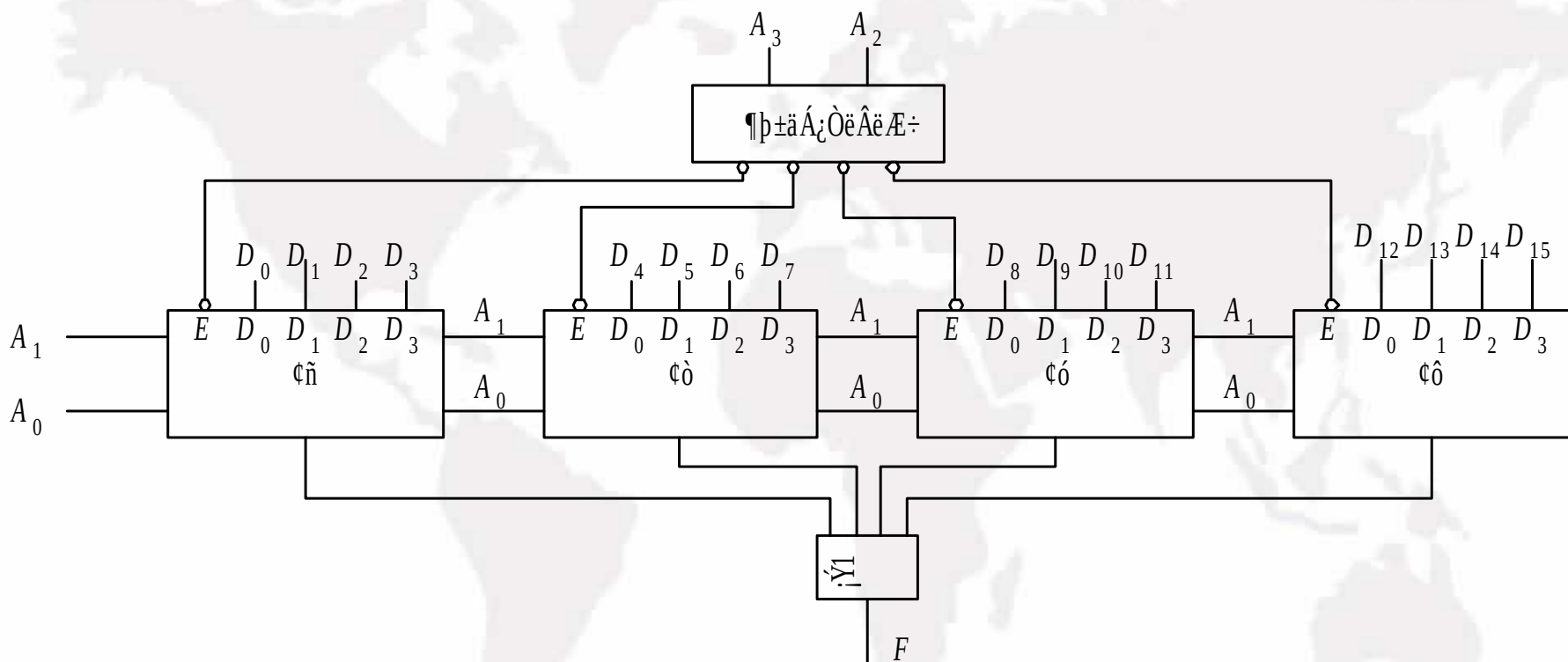
数据选择器的功能扩展

例 15 将四选一数据选择器扩为八选一数据选择器。



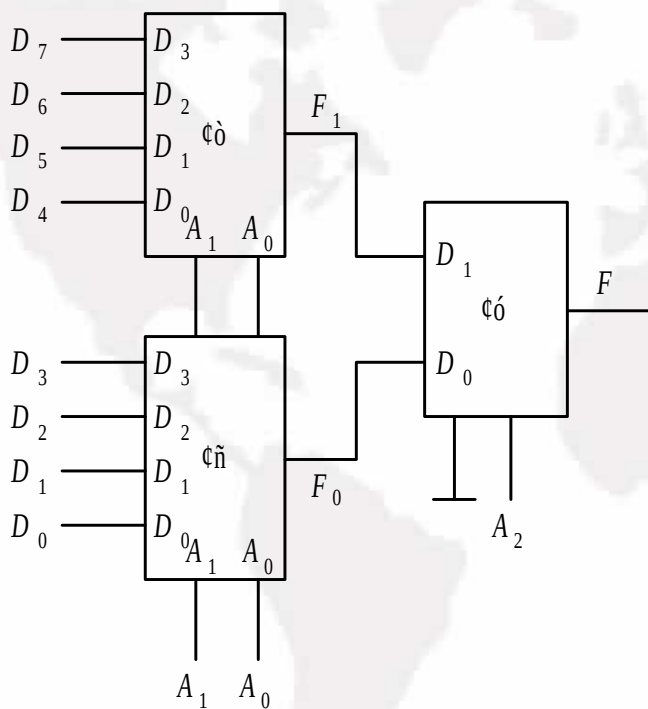


例 16 将四选一数据选择器扩大为十六选一数据选择器。

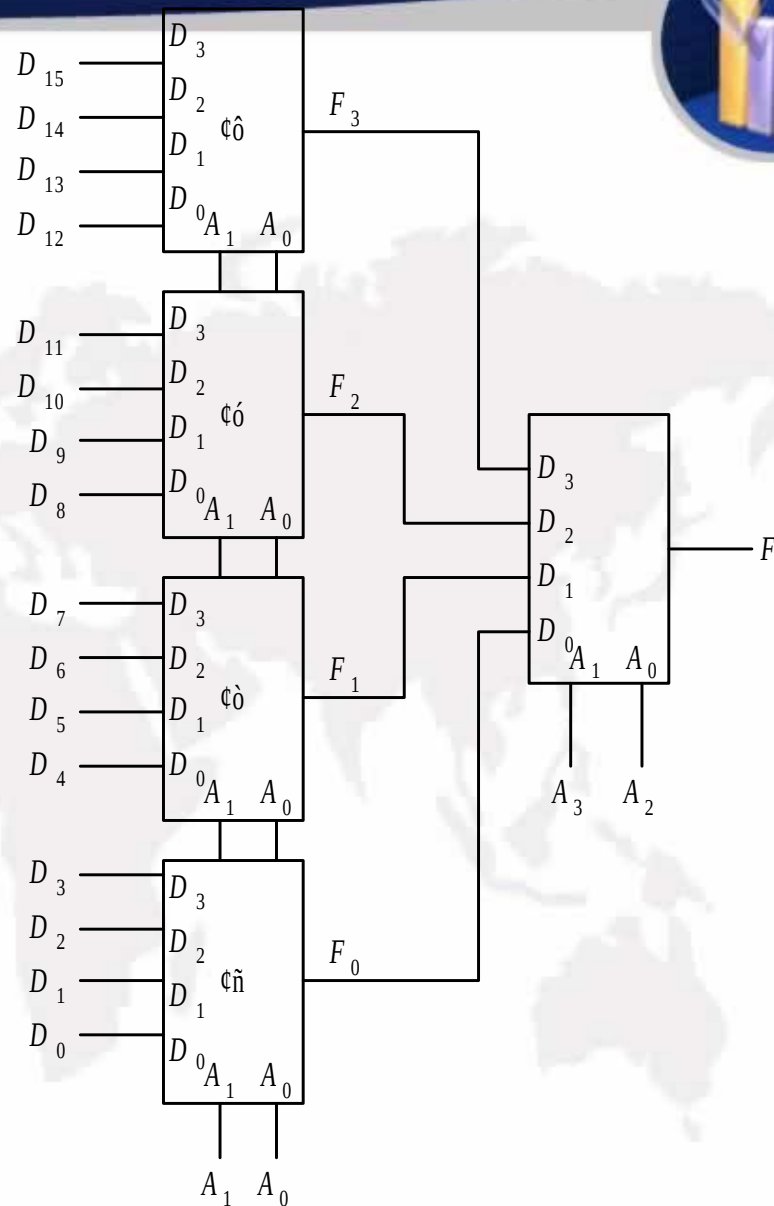




不利用使能端进行扩展



(a)



(b)



3.2 数据选择器的应用

1) 利用数据选择器实现组合逻辑函数

对于具有 n 个地址输入端的数据选择器，表达式为

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

其中 m_i 是由地址变量 A_{n-1} 、 \dots 、 A_1 、 A_0 组成的地址最小项。

任何一个具有 l 个输入变量的逻辑函数 F 也可以用输入变量的部分

最小项之和来表示： $F = \sum_{\text{部分}} m_i$

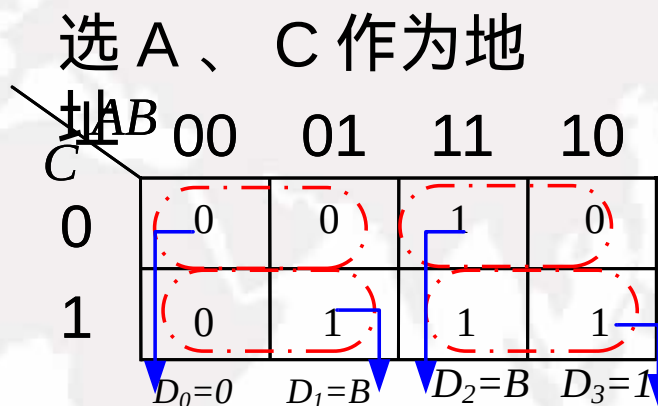
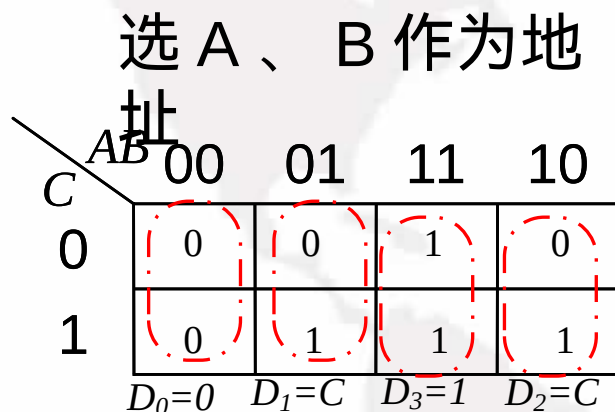
将函数的输入变量加到选择器的地址输入端，并根据函数 F 的需要选择适当的 D_i 值，使 $F=Y$ ，就可以用选择器实现逻辑函数 F 。



3.2 数据选择器实现逻辑函数

卡诺图法

首先在卡诺图上圈出地址变量的控制范围，在此范围内，确定相应数据输入端 D_i ，具体过程如下：



和代数法的结果一致！



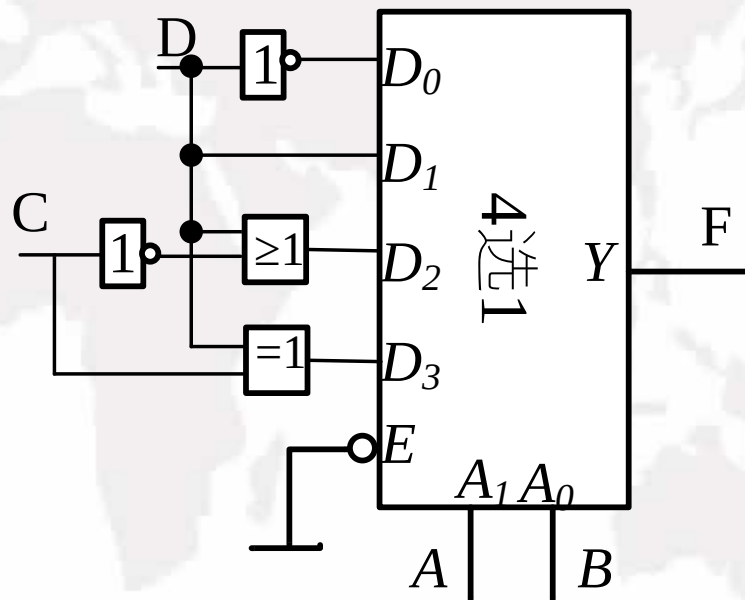
3.2 数据选择器实现逻辑函数

例 3：请用 4 选 1 实现逻辑函

数：
$$F(A, B, C, D) = \sum m(0, 2, 5, 7, 8, 9, 11, 13, 14)$$

解：如选择 $AB = A_1 A_0$ ，则 D_i 为变量 C 、 D 的函数。

$AB \backslash CD$	00	01	11	10
00	1	0	0	1
01	0	1	1	1
11	0	1	0	1
10	1	0	1	0
	D_0	D_1	D_3	D_2





2. 集成数字比较器

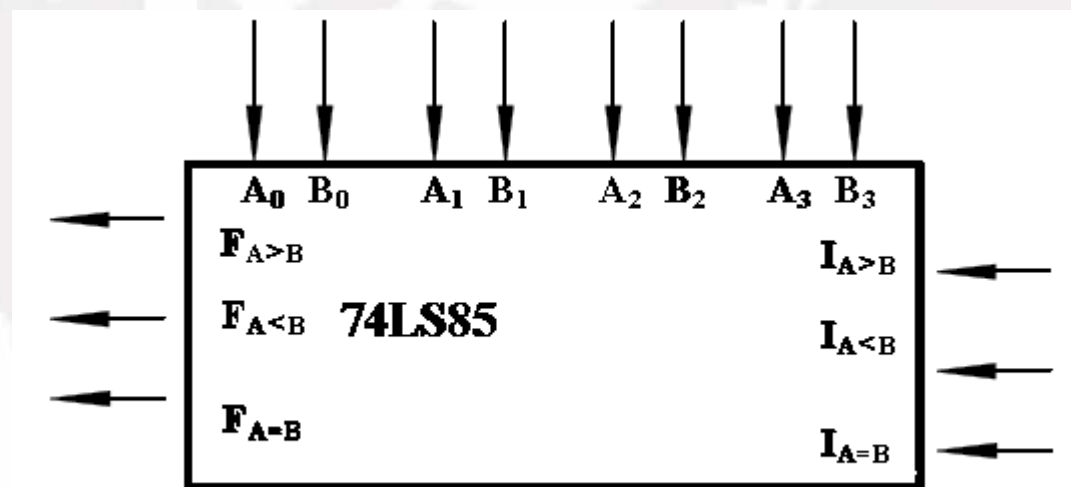
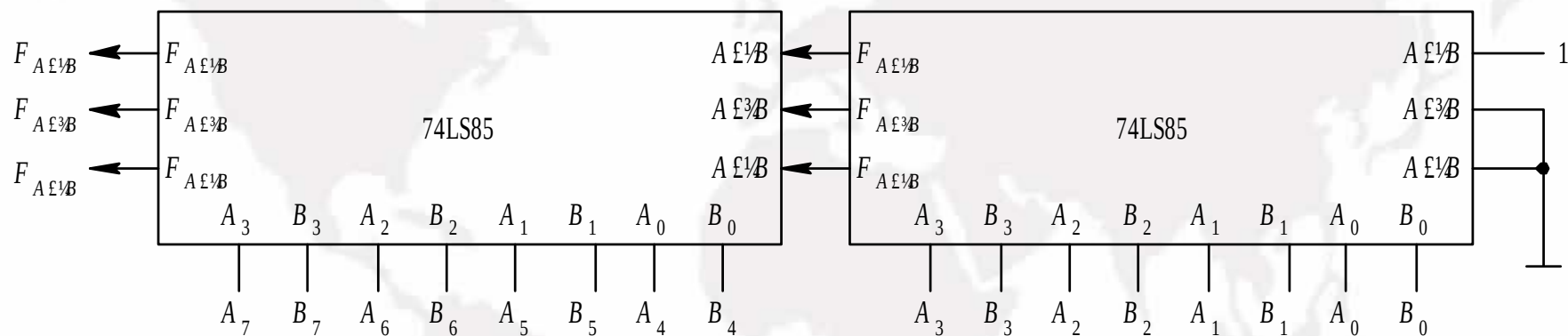


图 4-61 四位比较器 74LS85 逻辑符号



集成比较器的功能扩展

串联方式扩展





并联方式扩展

