《数字逻辑》
Digital Logic

组合逻辑（2）

北京工业大学软件学院
王晓懿

常用组合逻辑电路

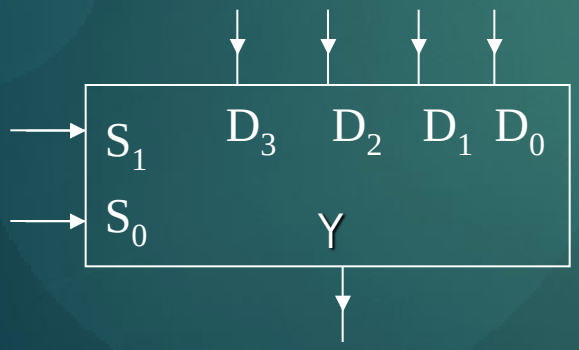
- 数据选择器(Data Selector)

数据选择器(Data Selector)

原理：在控制信号作用下，从多个输入中每次选中一个输出。因此又称多路开关(Multiplexer－MUX)。是计算机系统中使用最多的一类中规模器件。

例：4选1（4通道选1）数字选择器：

逻辑框图

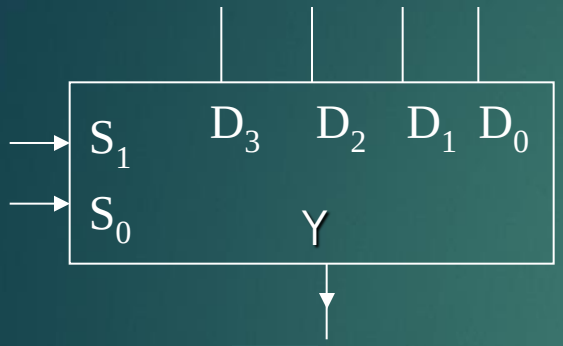


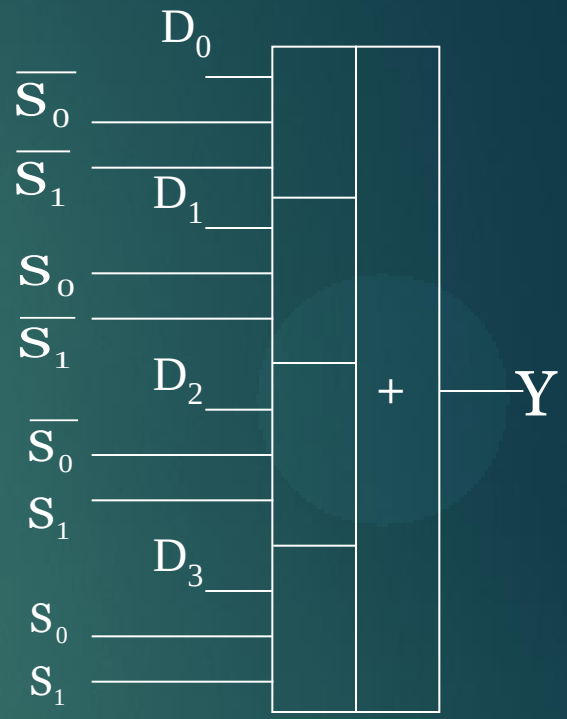
真值表

S_1	S_0	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

$$Y = \overline{S_0}\,\overline{S_1}D_0 + S_0\overline{S_1}D_1 + \overline{S_0}S_1D_2 + S_0S_1D_3$$

数据选择器的内部结构





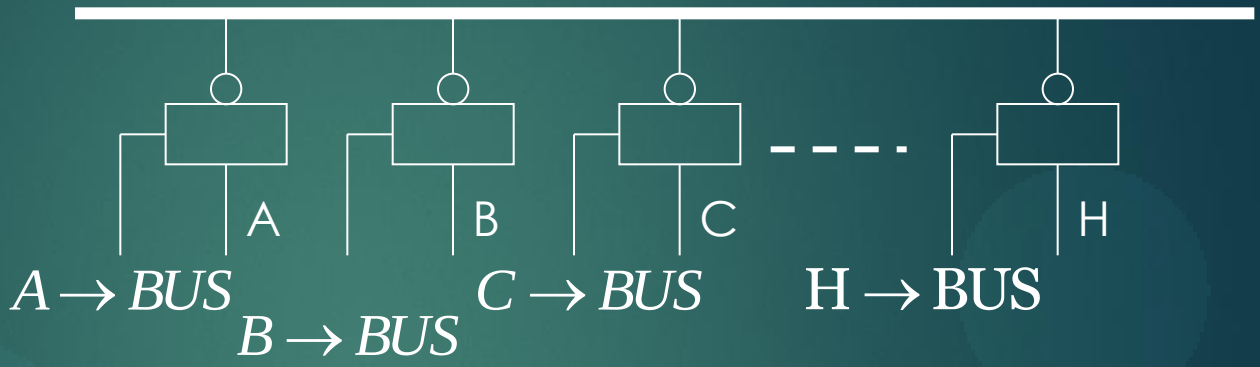
$$Y = \overline{S_0}\,\overline{S_1}D_0 + S_0\overline{S_1}D_1 + \overline{S_0}S_1D_2 + S_0S_1D_3$$

控制变量S_1 S_0的组合在同一时间内只能有一个为“1”，因此输出只能选中其中的一个

数据选择器用于总线发送控制

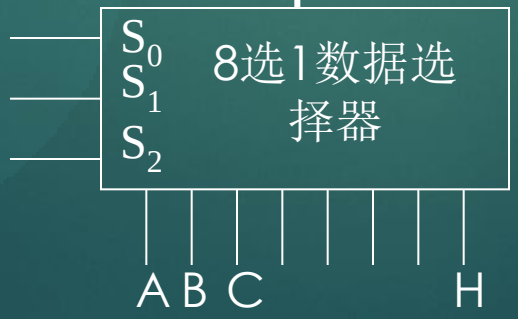
三态门

控制总线



数据选择器

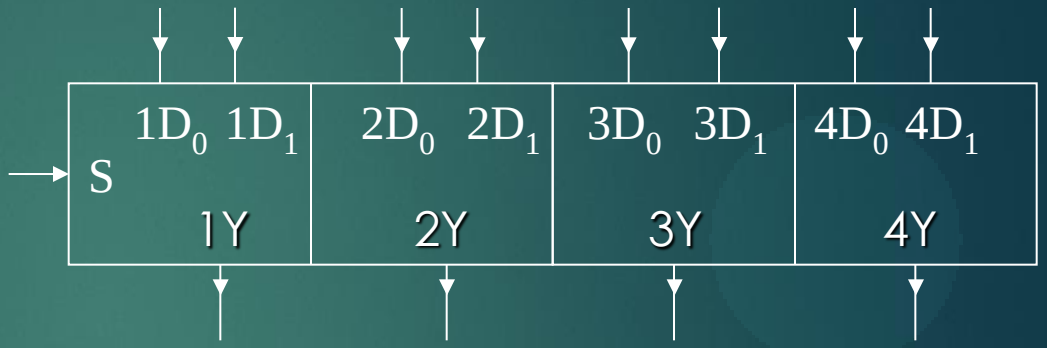
控制总线



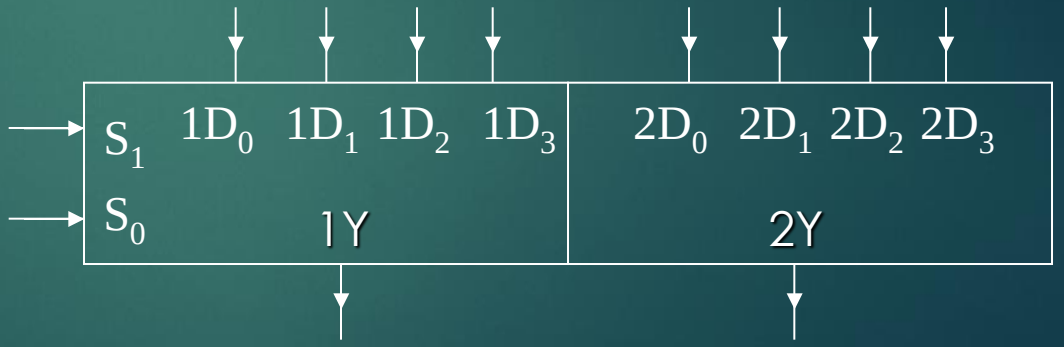
想想看：数据选择器还有哪些应用？

多位结构的数字选择器

4位2选1

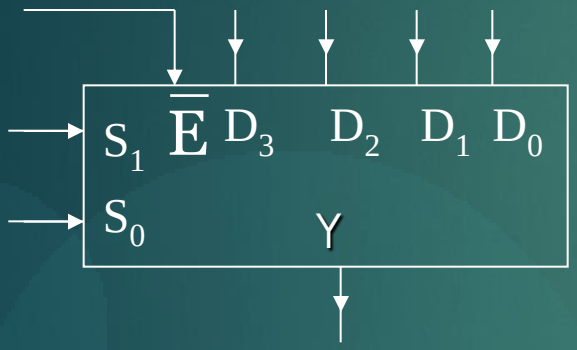


2位4选1



带控制端的数据选择器

逻辑框图



真值表

$\overline{E}$	S_1	S_0	Y
1	X	X	0
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3

使能（Enable）控制端

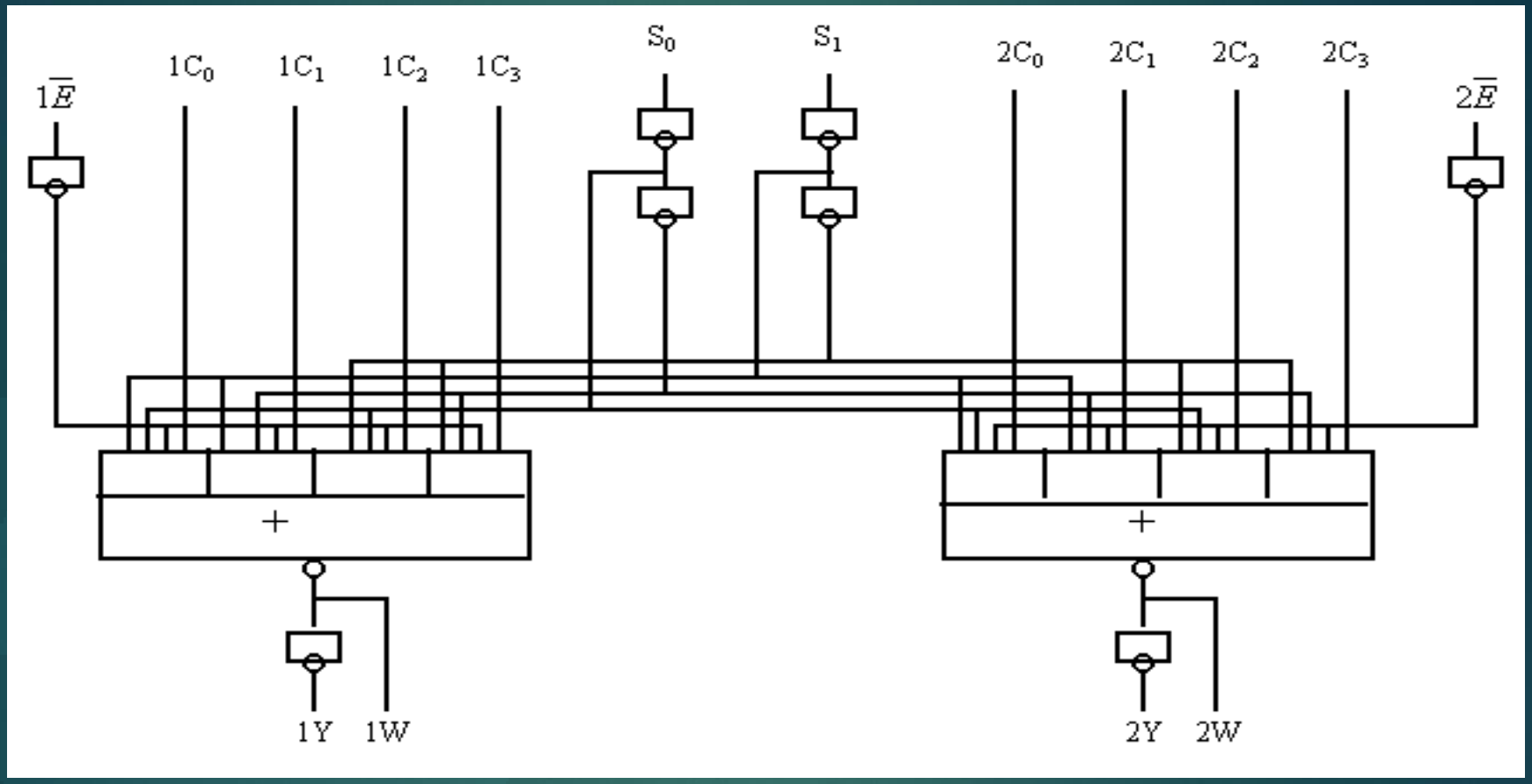
$\overline{E}=1$，选择器被禁止

$\overline{E}=0$，选择器输出$Y=D_i$

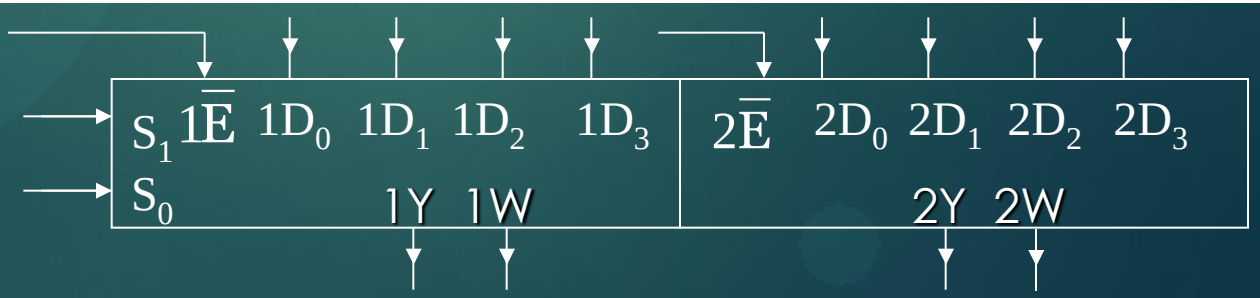
$$Y=\overline{E}(\overline{S_0}\,\overline{S_1}D_0+S_0\overline{S_1}D_1+\overline{S_0}S_1D_2+S_0S_1D_3)$$

有使能端的2位4选1数据选择器

（带互补输出：$W=\overline{Y}$）

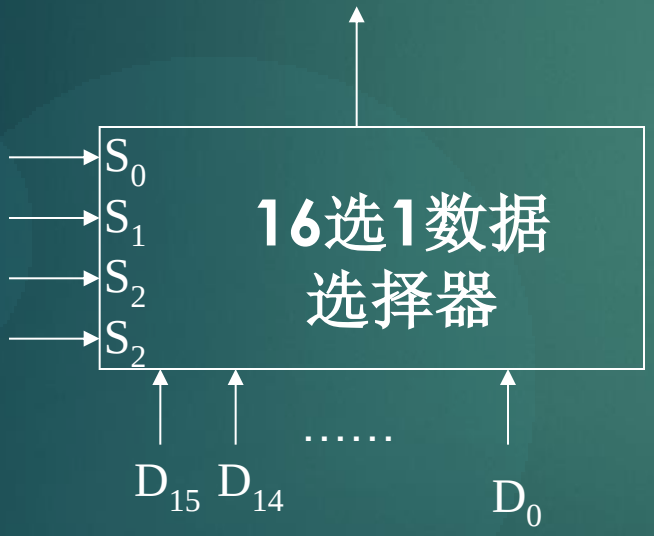


逻辑框图



选择器扩展：16选1选择器

中规模器件只有有限个输入，当需要更大规模电路时，就需要扩展。

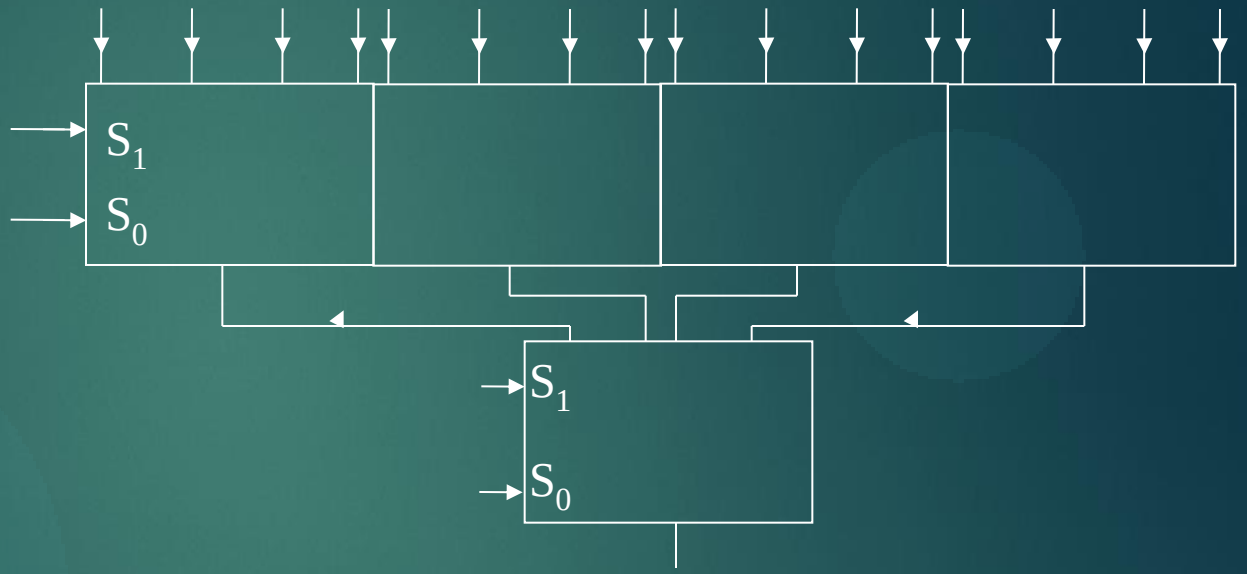


如果手中没有16选1的选择器，可以用8选1、4选1等扩展实现。

选择器扩展:用4选1选择器扩展成16选1选择器

16选1功能表

S_3	S_2	S_1	S_0	Y
0	0	0	0	Y_0
		0	1	Y_1
		1	0	Y_2
		1	1	Y_3
0	1	0	0	Y_4
		0	1	Y_5
		1	0	Y_6
		1	1	Y_7
1	0	0	0	Y_8
		0	1	Y_9
		1	0	Y_{10}
		1	1	Y_{11}
1	1	0	0	Y_{12}
		0	1	Y_{13}
		1	0	Y_{14}
		1	1	Y_{15}

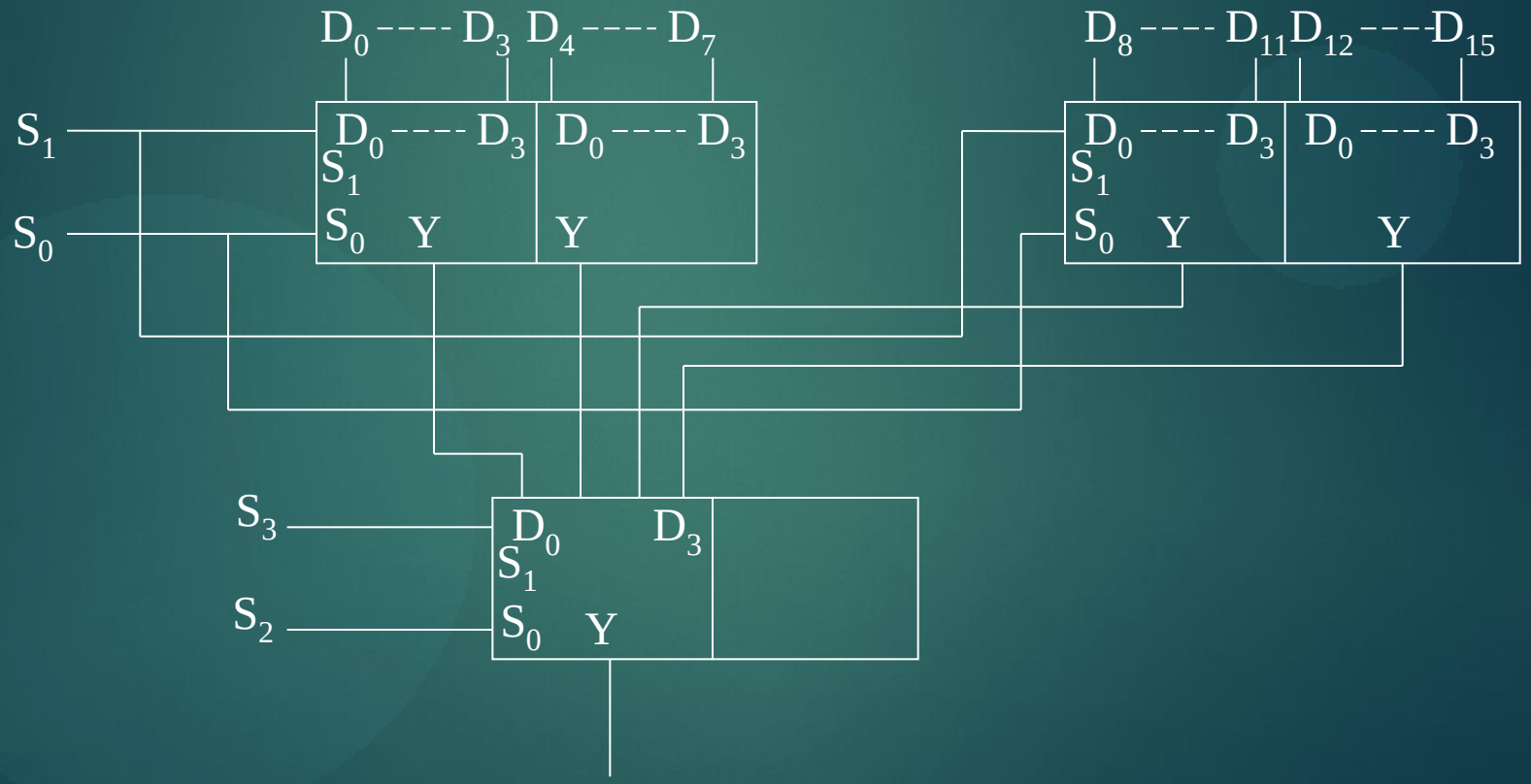


一定要选两次，分成4组，每组选出一个，再从4个中选择一次。

两种不同的扩展方案，从功能表上分析，可以先用低两位控制，也可以先用高两位控制。

选择器扩展： 用双4选1选择器(无$\overline{E}$)扩展成16选1选择器(1)

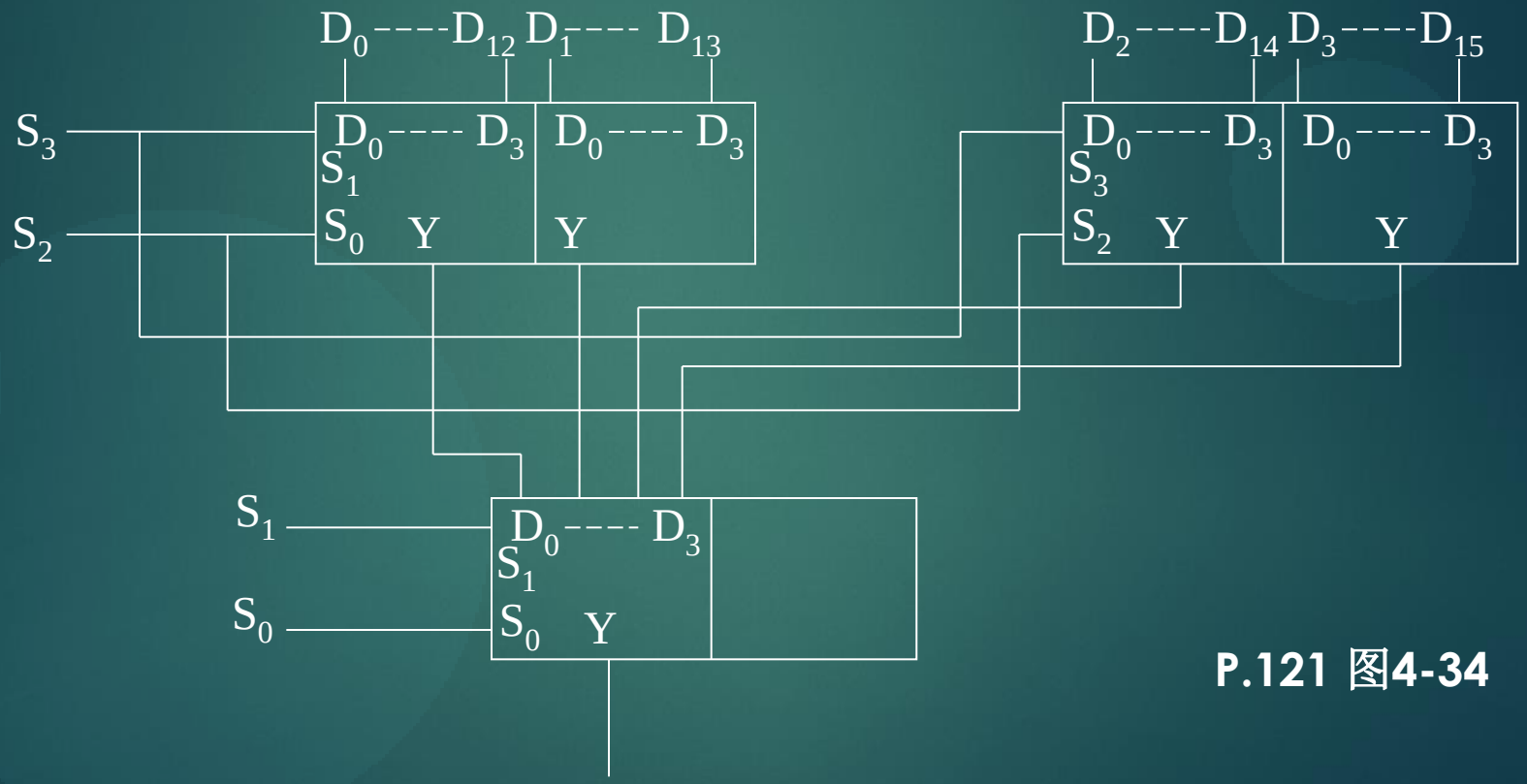
选择器无选通控制端，
只能用两级选择结构



逻辑结构：S_1 S_0控制第一层选择，S_3 S_2控制第二层选择。

选择器扩展： 用双4选1选择器(无$\overline{E}$)扩展成16选1选择器(2)

选择器无选通控制端，
只能用两级选择结构

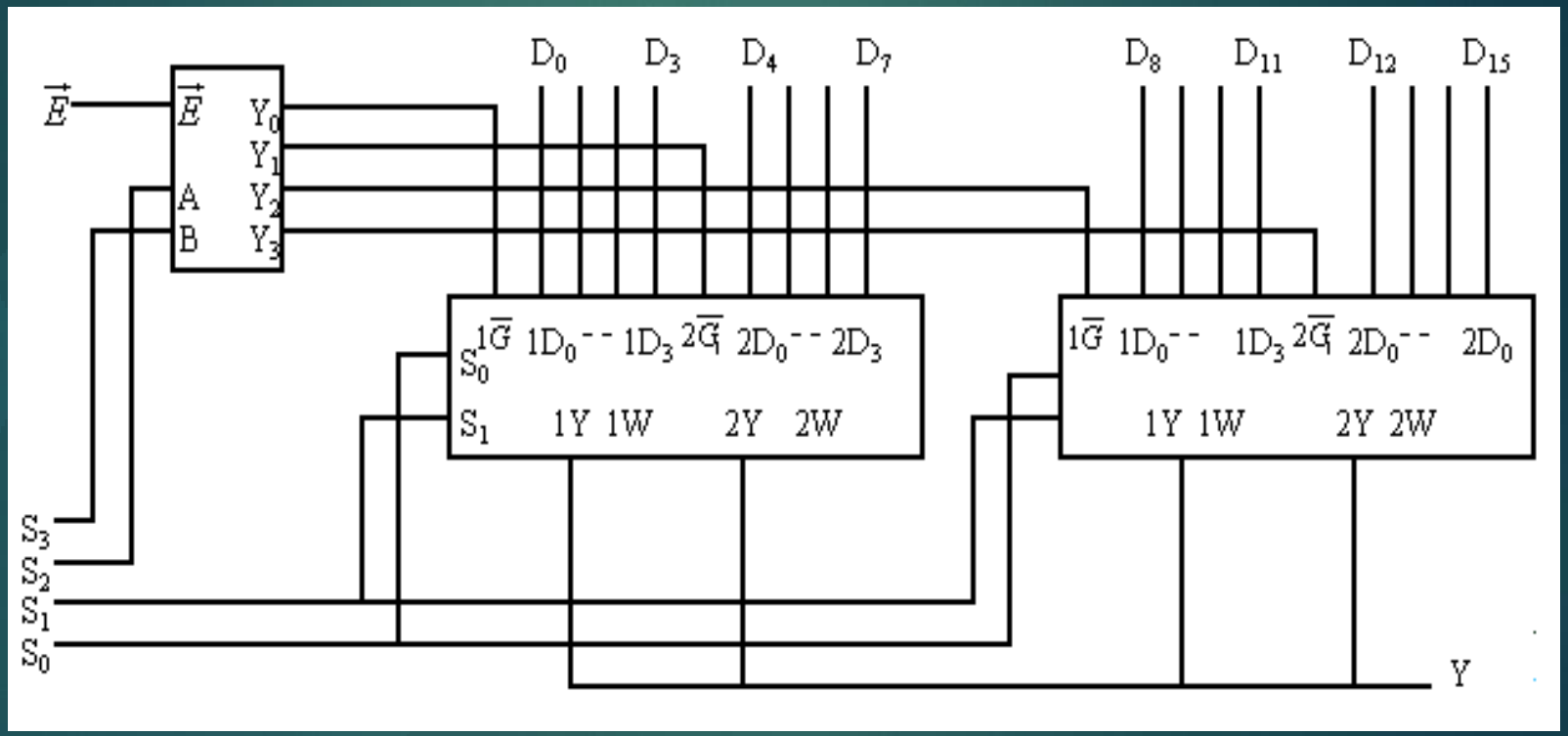


P.121 图4-34

逻辑结构：S_3 S_2控制第一层选择，S_1 S_0控制第二层选择。

选择器扩展： 带$\overline{E}$的双4选1选择器扩展成16选1选择器

用译码器＋带$\overline{E}$的数据选择器，一级选择就可以。



高两位控制端经译码后分别控制数据选择器的使能端，以实现扩展。输出级是三态门，因此可以“线与”。

译码器与数据选择器的比较

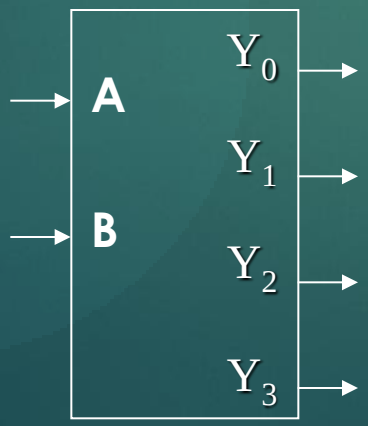
译码器

A	B	Y_0	Y_1	Y_2	Y_3
0	0	0	1	1	1
1	0	1	0	1	1
0	1	1	1	0	1
1	1	1	1	1	0

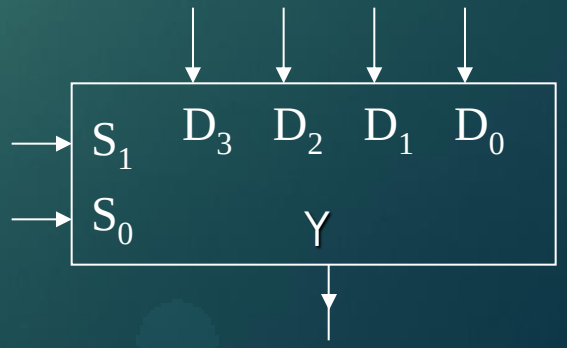
数据选择器

S_1	S_0	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

$$\begin{cases} Y_0 = \overline{\overline{A}\,\overline{B}} \\ Y_1 = \overline{\overline{A}B} \\ Y_2 = \overline{A\overline{B}} \\ Y_3 = \overline{AB} \end{cases}$$



$$Y = \overline{S_0}\,\overline{S_1}D_0 + S_0\overline{S_1}D_1 + \overline{S_0}S_1D_2 + S_0S_1D_3$$

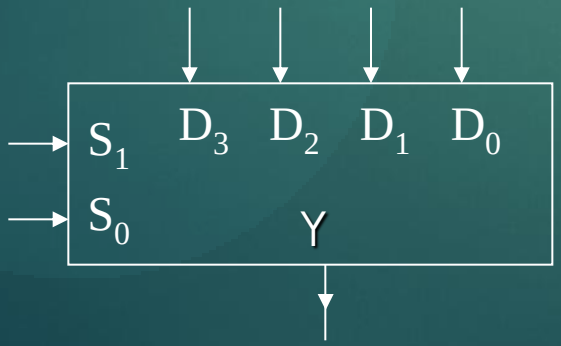


数据选择器与数据分配器

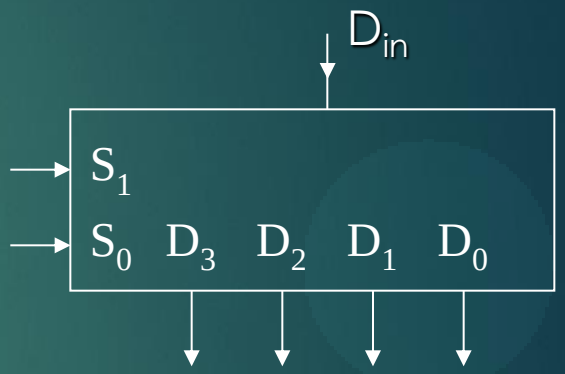
数据选择器

S_1	S_0	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

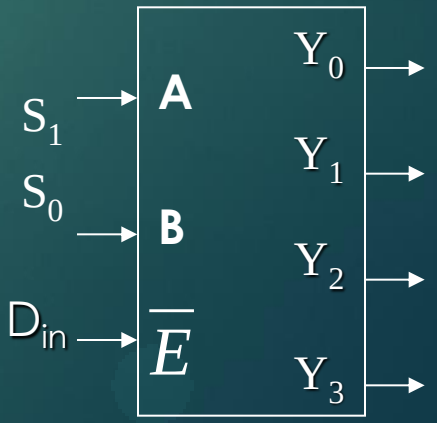
$$Y = \overline{S_0}\,\overline{S_1}D_0 + S_0\overline{S_1}D_1 + \overline{S_0}S_1D_2 + S_0S_1D_3$$



数据分配器



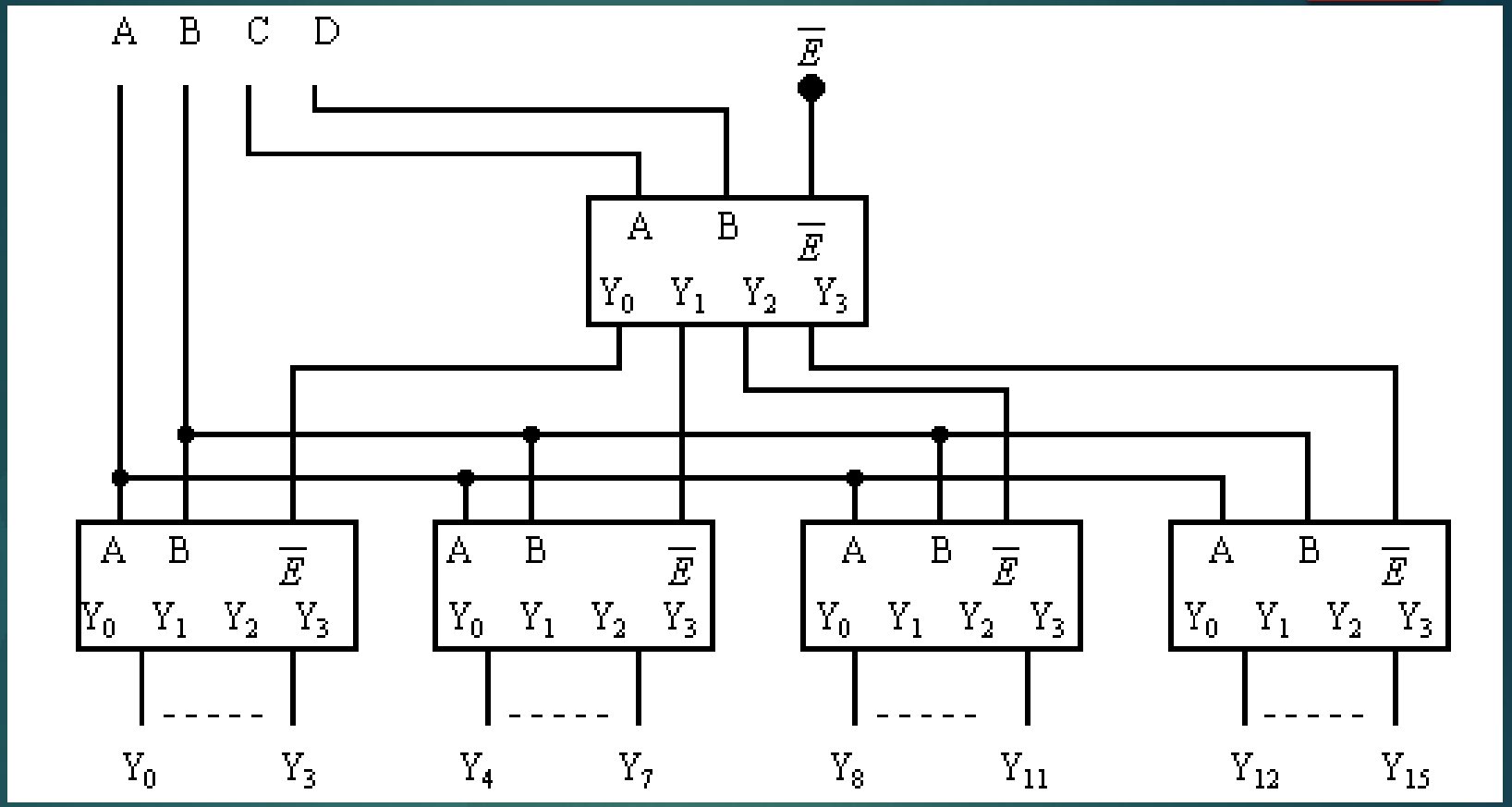
译码器实现数据分配器



译码器与数据选择器的比较

- 都是与非，与或非逻辑的全组合
- 都可以有$\overline{E}$控制端，用于扩展和选通
- 都可以实现逻辑函数（最小项的全组合）
- 扩展应用：译码器都要用$\overline{E}$，选择器可以不用$\overline{E}$（扩展时要先画真值表，找规律）
- 选择器的输出结构可以带有三态输出、OC门输出等结构，因此选择器可以用于总线发送器。

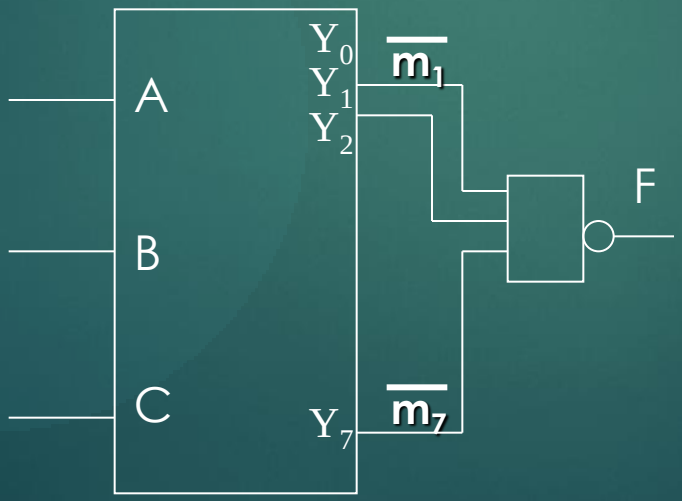
$\overline{E}$ 用作扩展（译码器）



5片2－4译码器构成4－16译码器。第一层的一个译码器用作选片。E=0时，C D＝00时选中左边一片，译出$Y_0 \ldots Y_3$;依此类推。

译码器实现逻辑函数

- 译码器输出可以看成是N个输入变量组成的2^N个最小项,再经一级与非门,组成“与非-与非”逻辑，既可表达“与-或”表达式。
- 例如：$F=\overline{A}\overline{B}C+\overline{A}B\overline{C}+AB\overline{C}=m_1+m_2+m_7$

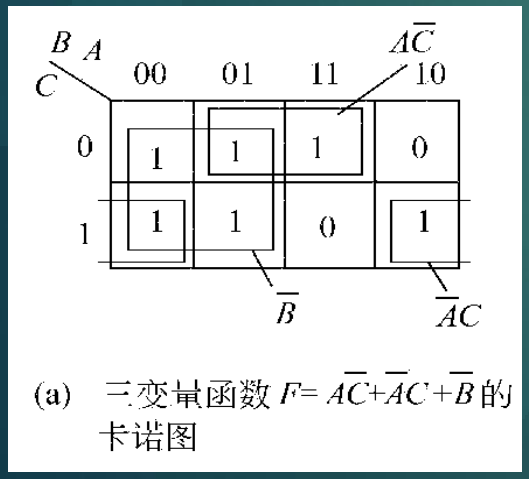


数据选择器实现逻辑函数

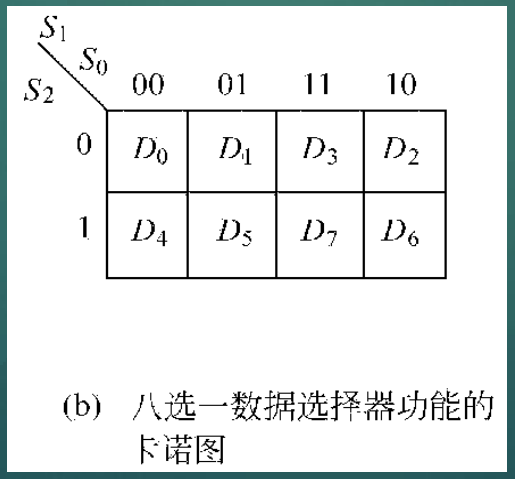
数据选择器: 逻辑结构就是与-或表达式。

数据选择器可以看成是N个控制端选择2^N个最小项组成的“与-或”表达式。选择某些输入为“1”，就是选中这些最小项组成逻辑函数。逻辑变量接到选择控制端，逻辑表达式中包含的最小项取“1”，其余取“0”。

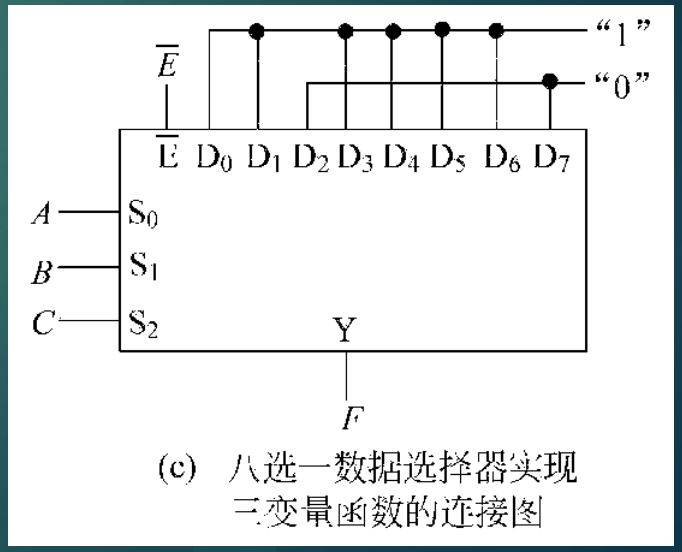
例如，用八选一数据选择器实现函数: $F = \overline{B} + \overline{A}C + A\overline{C}$



(a) 三变量函数 $F = A\overline{C} + \overline{A}C + \overline{B}$ 的卡诺图



(b) 八选一数据选择器功能的卡诺图



(c) 八选一数据选择器实现三变量函数的连接图

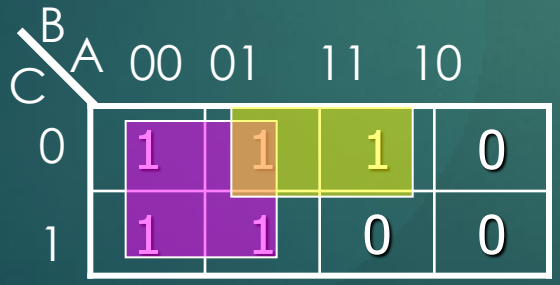
数据选择器实现逻辑函数

8选1数据选择器可以实现4变量函数，
3个变量用在选择控制端，1个变量在数据输入。

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

$F'=A\overline{C}+\overline{B},F''=A\overline{C}+\overline{A}C$

$F=F'\overline{N}+F''N$

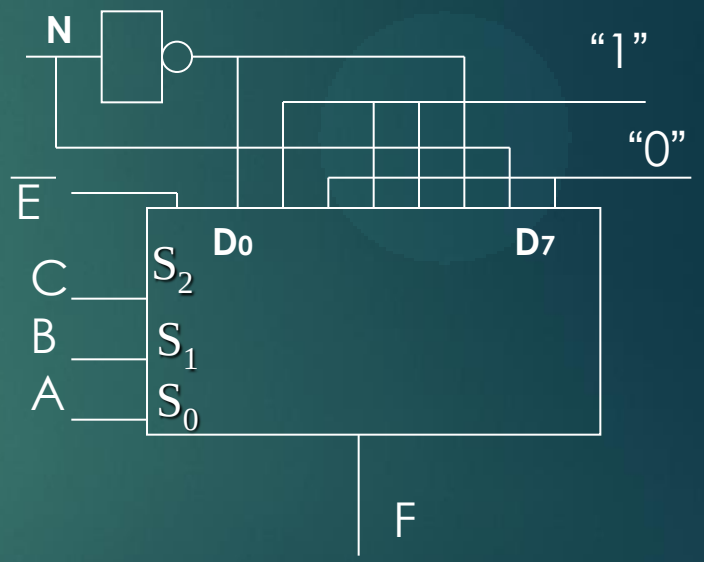


$F'=A\overline{C}+\overline{B},$

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

8选1数据选择器实现4变量函数

S_0 S_1 S_2	输入	F'	F''	函数F的值
0 0 0	D_0	1	0	$\overline{N}\bullet 1+N\bullet 0=\overline{N}$
1 0 0	D_1	1	1	$\overline{N}\bullet 1+N\bullet 1=1$
0 1 0	D_2	0	0	$\overline{N}\bullet 0+N\bullet 0=0$
1 1 0	D_3	1	1	$\overline{N}\bullet 1+N\bullet 1=1$
0 0 1	D_4	1	1	$\overline{N}\bullet 1+N\bullet 1=1$
1 0 1	D_5	1	0	$\overline{N}\bullet 1+N\bullet 0=\overline{N}$
0 1 1	D_6	0	1	$\overline{N}\bullet 0+N\bullet 1=N$
1 1 1	D_7	0	0	$\overline{N}\bullet 0+N\bullet 0=0$



常用组合逻辑电路

- 加法器

运算器

- 加法器
  - 1位加法器
  - 四位串行进位加法器
  - 快速加法器
  - 16位加法器

加法器

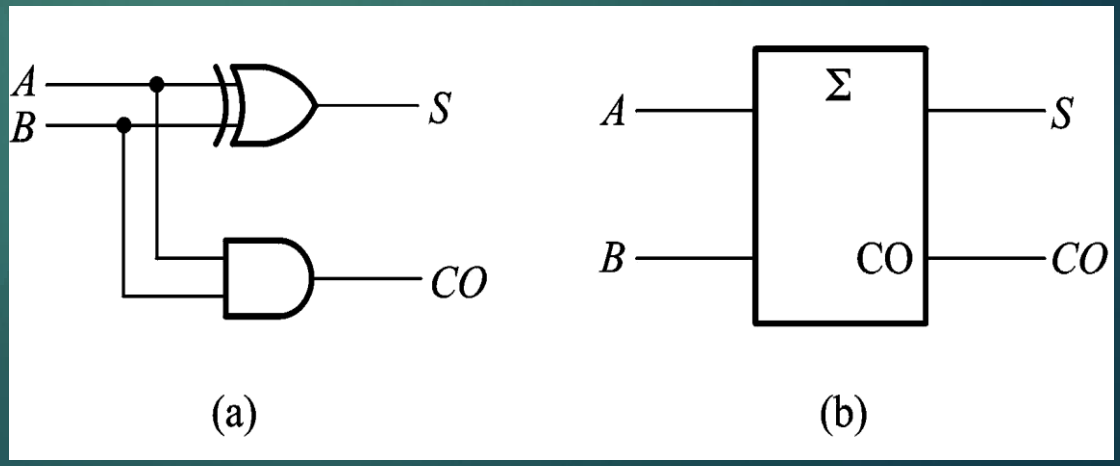
一、1位加法器

半加器：不考虑来自低位的进位，将两个1位的二进制数相加

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

$$S = A \oplus B$$

$$CO = AB$$



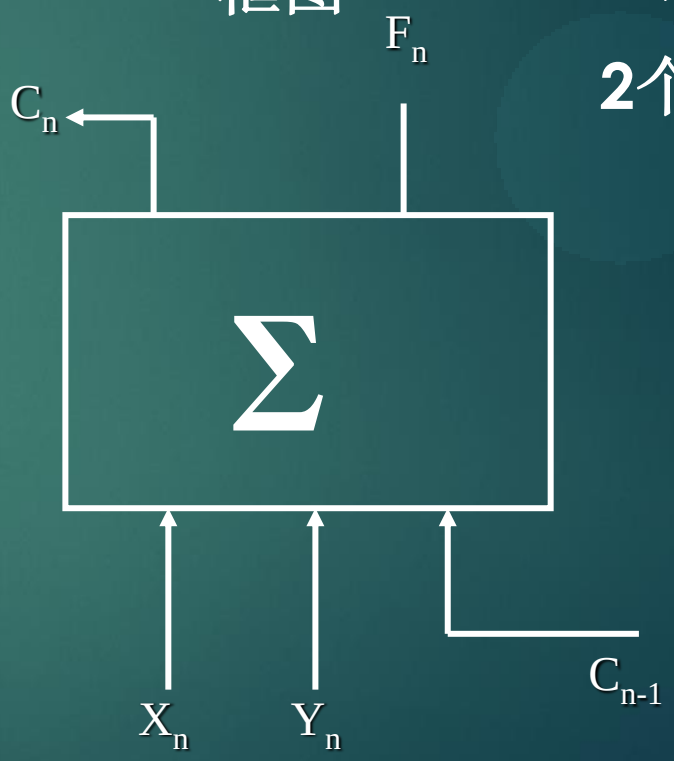
1位二进制全加器的设计与实现

（不考虑进位叫半加，考虑进位叫全加）

真值表

输入			输出	
X_n	Y_n	C_{n-1}	F_n	C_n
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

框图

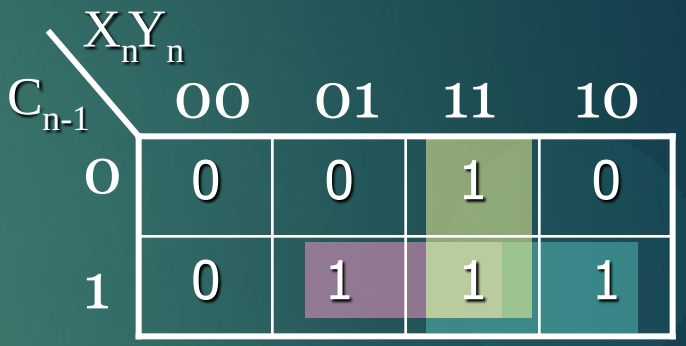


3个输入

2个输出

全加器逻辑的卡诺图化简

F_n



C_n

$$C_n = X_nY_n + X_nC_{n-1} + Y_nC_{n-1} = X_nY_n + (X_n + Y_n)C_{n-1}$$

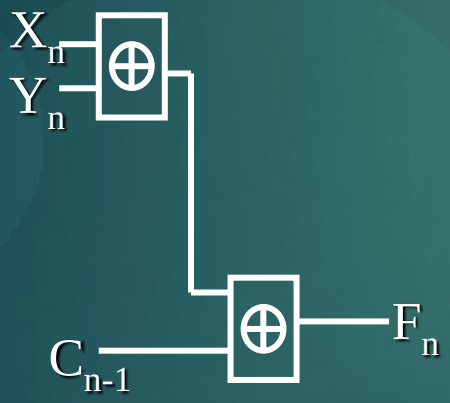
$$F_n = \overline{X}_n\overline{Y}_nC_{n-1} + \overline{X}_nY_n\overline{C}_{n-1} + X_n\overline{Y}_n\overline{C}_{n-1} + X_nY_nC_{n-1}$$

$$= \overline{C}_{n-1}(\overline{X}_nY_n + X_n\overline{Y}_n) + C_{n-1}(\overline{X}_n\overline{Y}_n + X_nY_n)$$

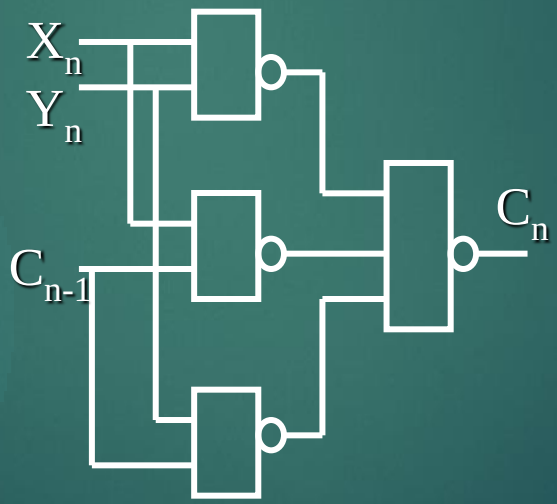
$$= C_{n-1} \oplus (X_n \oplus Y_n)$$

实现方案一

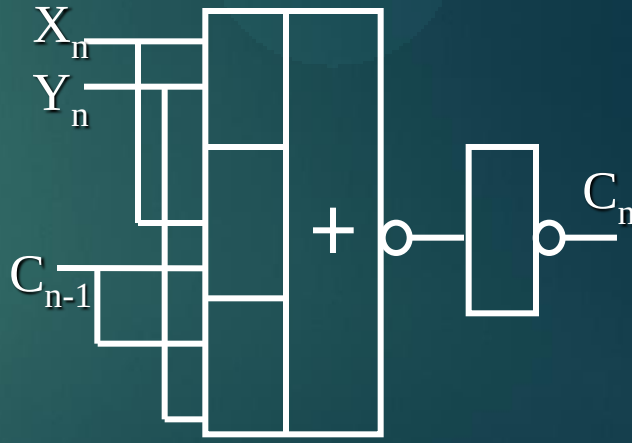
F_n用异或门实现



C_n用”与非”门实现

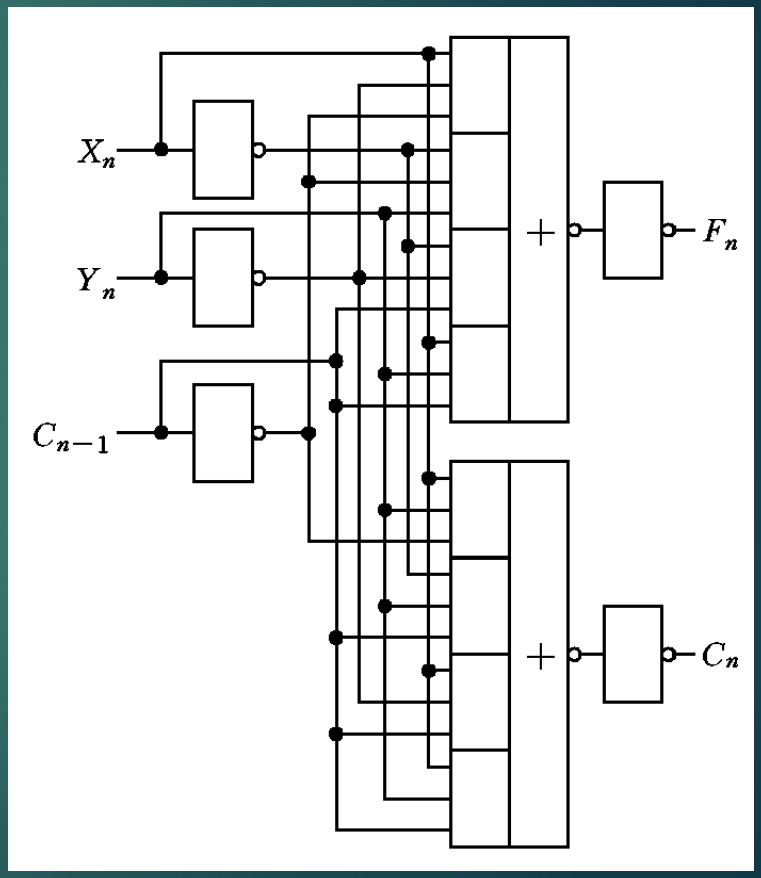


C_n用”与或非”门实现



实现方案二

直接由最小项用与或非门实现，
需要3级门



实现方案三

写$\overline{F}$ $\overline{C}$的表达式

$$\overline{F} = \overline{X_n}\,\overline{Y_n}\,\overline{C_{n-1}} + \overline{X_n}\,Y_n C_{n-1} + X_n\,\overline{Y_n}\,C_{n-1} + X_n Y_n\,\overline{C_{n-1}}$$

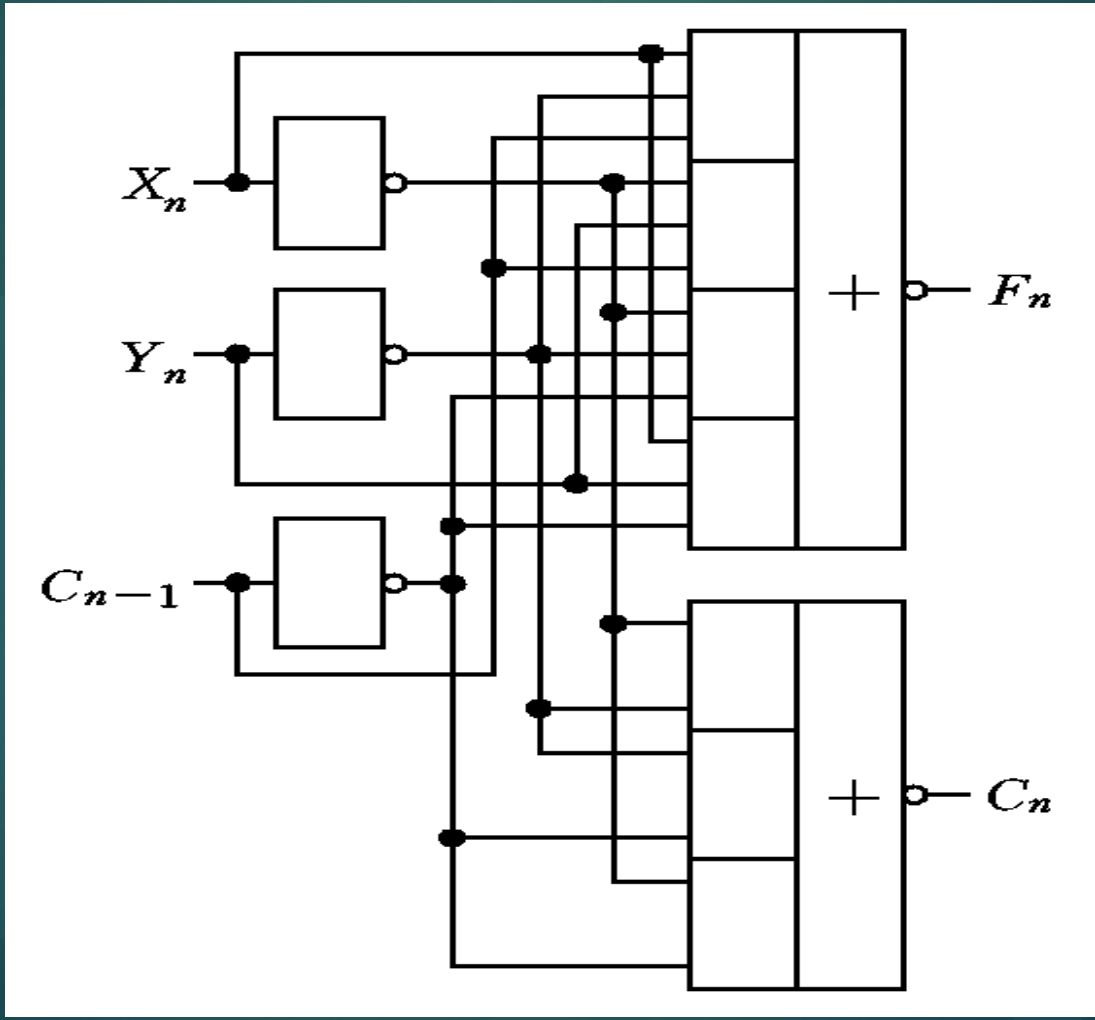
$$F = \overline{\overline{F}} = \overline{\overline{X_n}\,\overline{Y_n}\,\overline{C_{n-1}} + \overline{X_n}\,Y_n C_{n-1} + X_n\,\overline{Y_n}\,C_{n-1} + X_n Y_n\,\overline{C_{n-1}}}$$

$$\overline{C_n} = \overline{X_n}\,\overline{Y_n} + \overline{X_n}\,C_{n-1} + \overline{Y_n}\,C_{n-1}$$

$$C_n = \overline{\overline{C_n}} = \overline{\overline{X_n}\,\overline{Y_n} + \overline{X_n}\,C_{n-1} + \overline{Y_n}\,C_{n-1}}$$

经变换后只要2级门。

实现方案三（续）

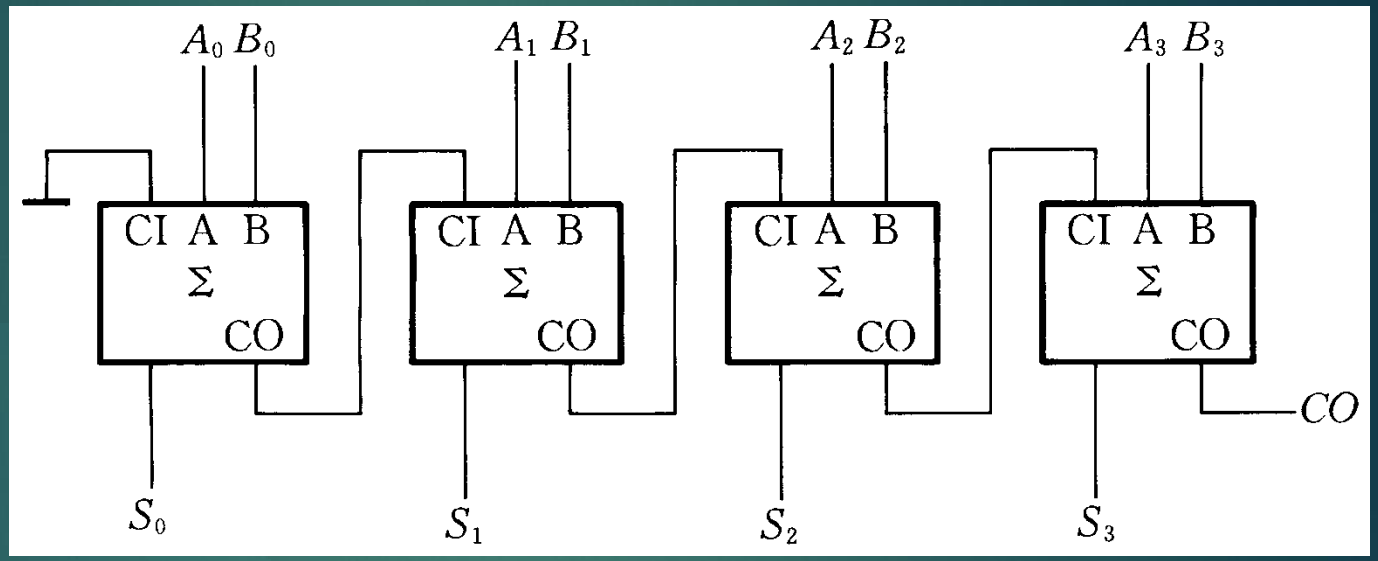


多位加法器

1. 4位串行进位加法器

优点：简单

缺点：慢



$$(CI)_i = (CO)_{i-1}$$

$$S_i = A_i \oplus B_i \oplus (CI)_i$$

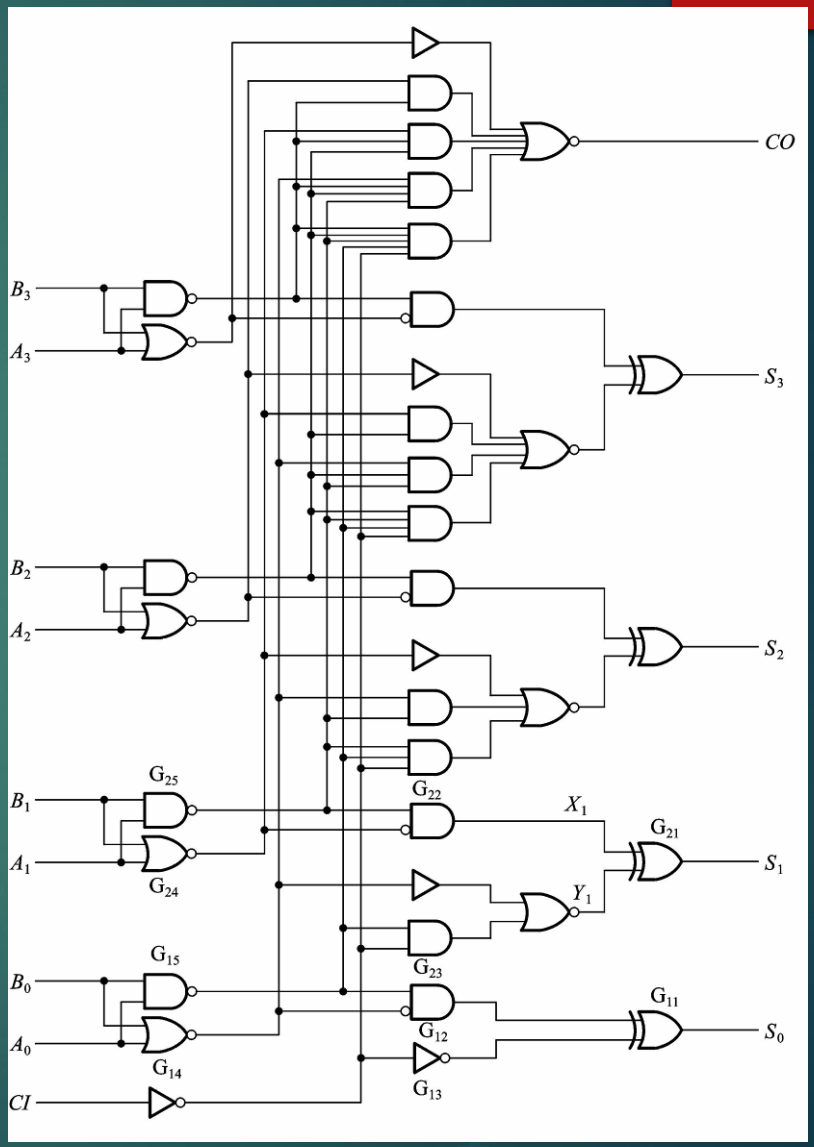
$$(CO)_i = A_i B_i + (A_i + B_i)(CI)_i$$

2. 并行进位（超前进位)加法器

基本原理：加到第i位的进位输入信号是两个加数第i位以前各位（0～j-1）的函数，可在相加前由A,B两数确定。

优点：快，每1位的和及最后的进位基本同时产生。

缺点：电路复杂。



74LS283