第四章　组合逻辑电路

4.1　概　述

主要要求：

- 掌握组合逻辑电路和时序逻辑电路的概念。
- 了解组合逻辑电路的特点与描述方法。

4.3.3 数据选择器

主要要求：

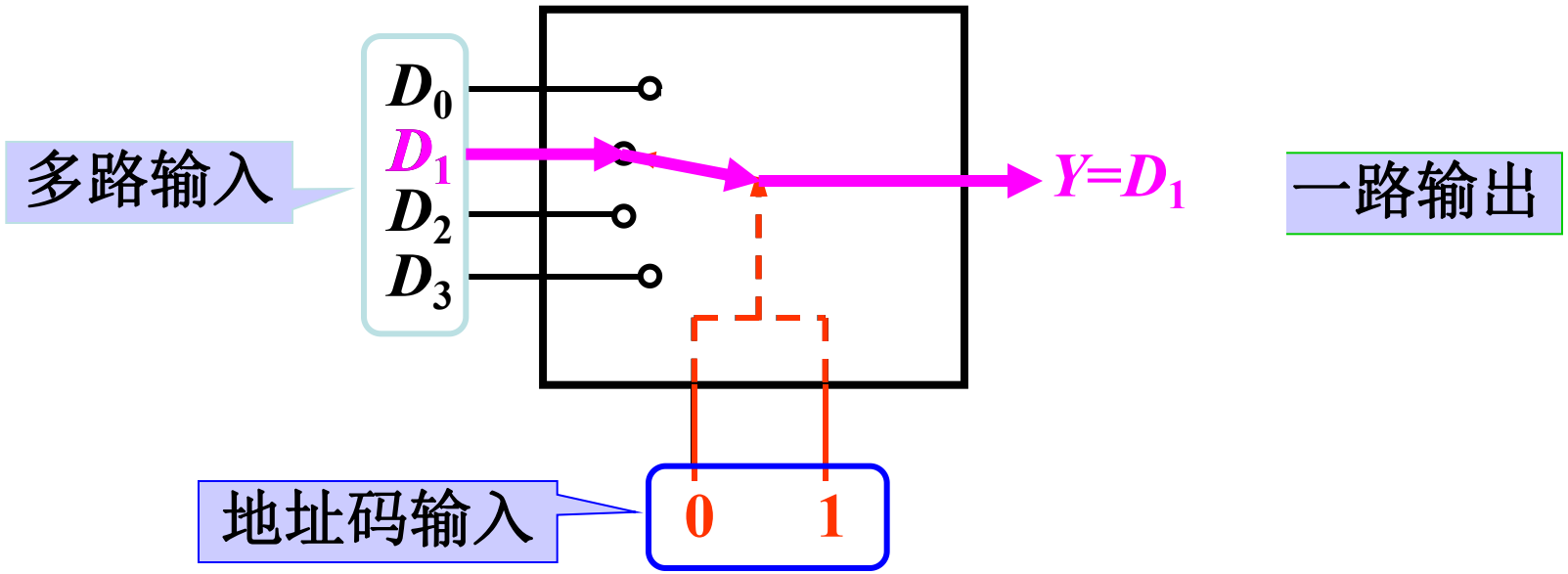
- 理解数据选择器和数据分配器的作用。
- 理解常用数据选择器的逻辑功能及其使用。
- 掌握用数据选择器实现组合逻辑电路的方法。

一、数据选择器和数据分配器的作用

数据选择器： 根据地址码的要求，从多路输入信号中选择其中一路输出的电路.

又称多路选择器(Multiplexer，简称MUX)或多路开关。

4选1数据选择器工作示意图

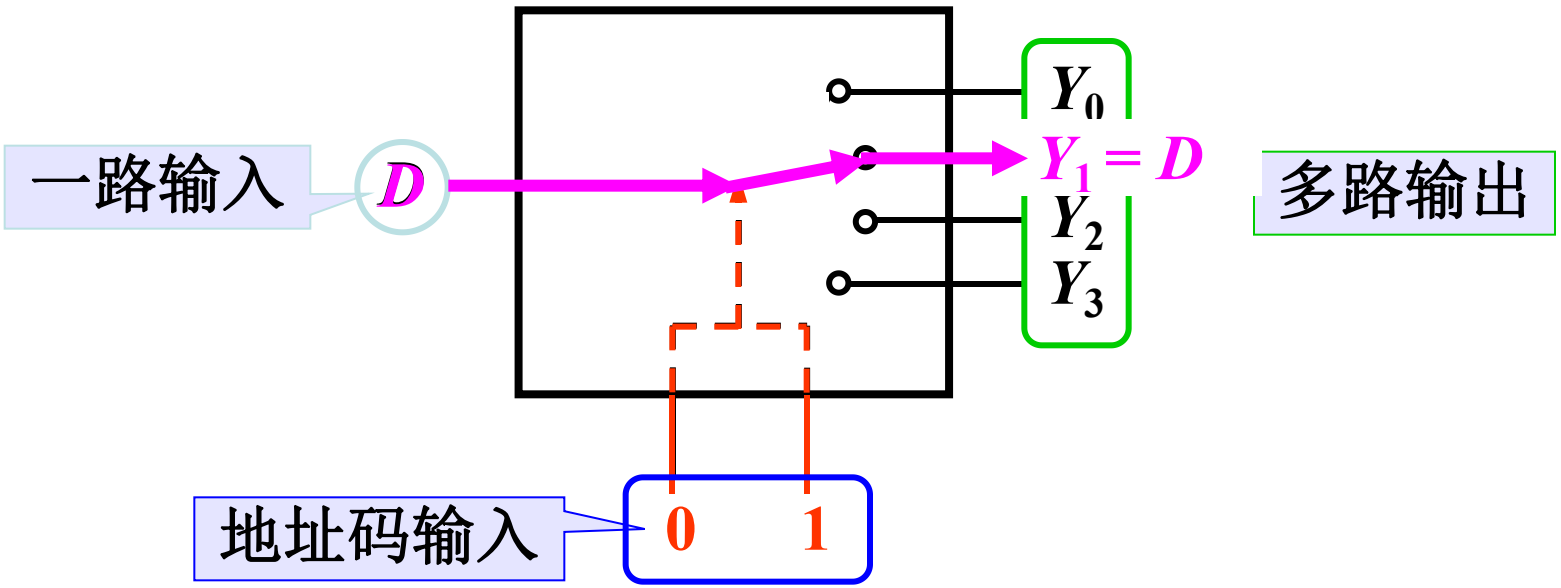


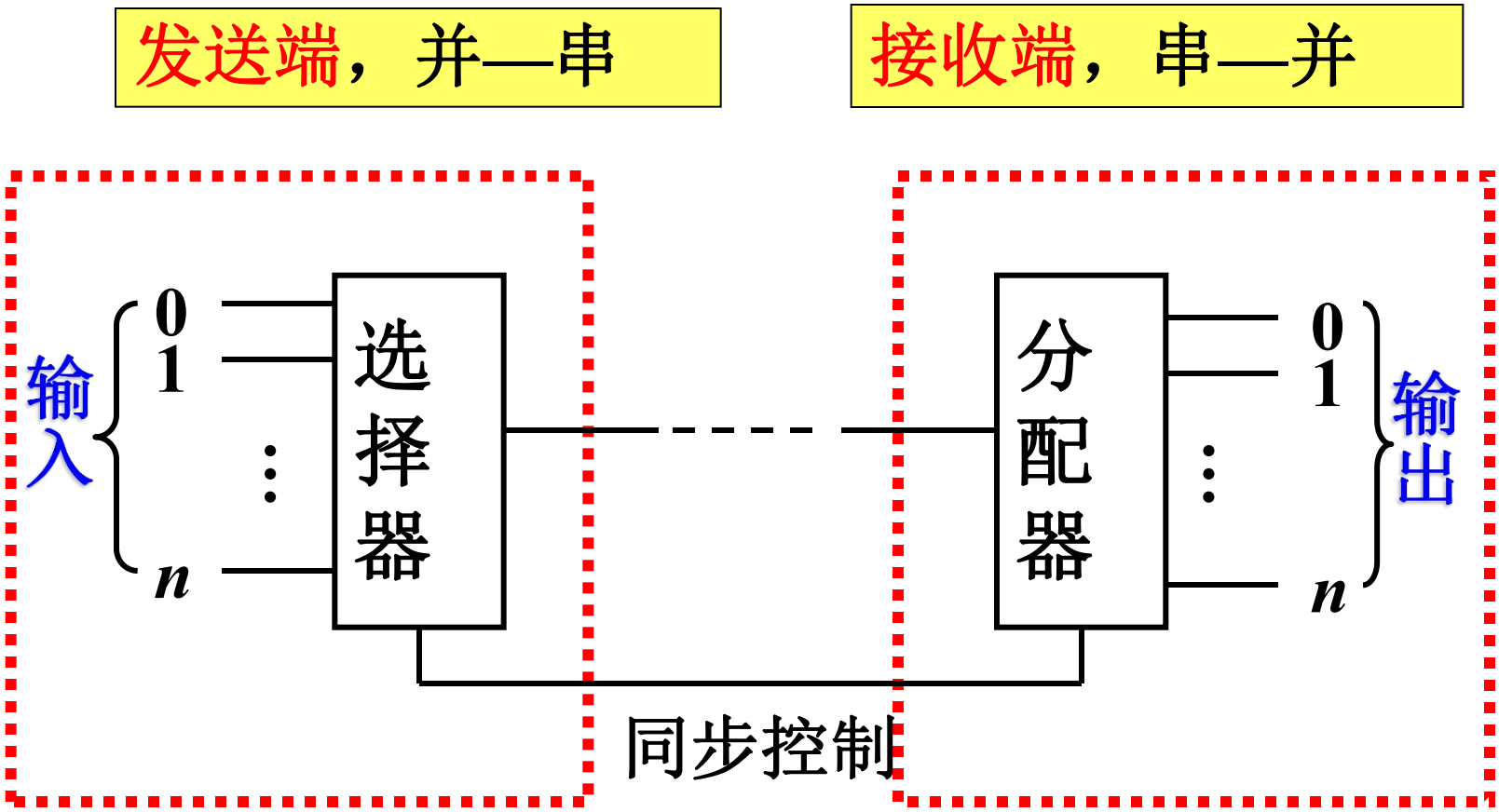
数据选择器的输入信号个数 N 与地址码个数 n 的关系为 $N=2^n$

数据分配器：根据地址码的要求，将一路数据分配到指定输出通道上去的电路。

Demultiplexer，简称DMUX

4路数据分配器工作示意图

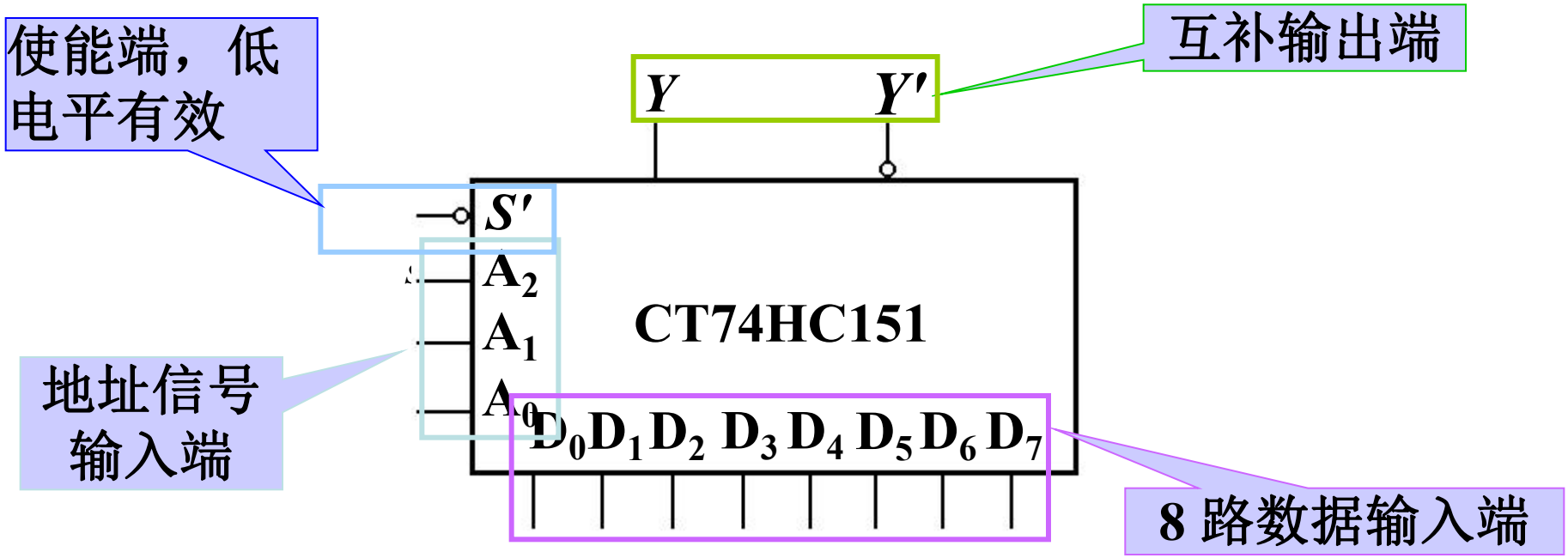




串行传输数据示意图

二、数据选择器的逻辑功能及其使用

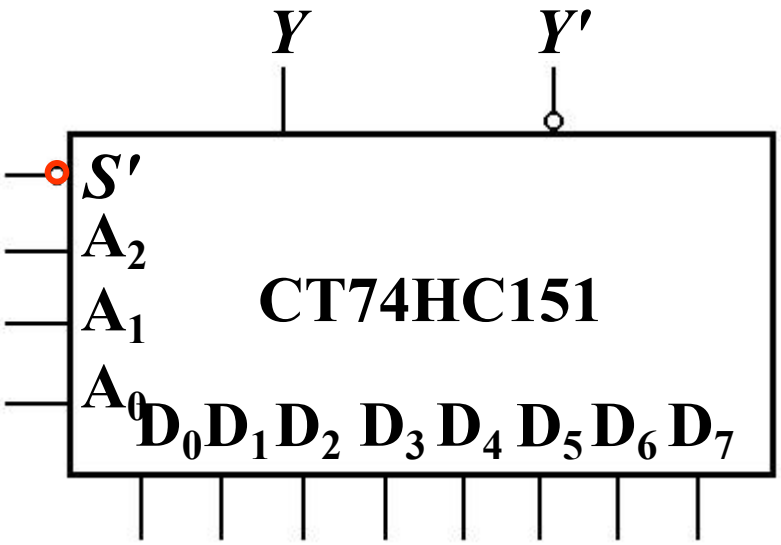
8 选 1 数据选择器 CT74HC151



CT74HC151的逻辑功能示意图

8 选 1 数据选择器 CT74HC151 真值表

输入				输出
S'	A_2	A_1	A_0	Y
1	×	×	×	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



CT74HC151 输出函数表达式

输入				输出
S'	A_2	A_1	A_0	Y
1	×	×	×	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

$$Y = A_2'A_1'A_0'D_0 + A_2'A_1'A_0D_1 +$$

$$A_2'A_1A_0'D_2 + A_2'A_1A_0D_3 +$$

$$A_2A_1'A_0'D_4 + A_2A_1'A_0D_5 +$$

$$A_2A_1A_0'D_6 + A_2A_1A_0D_7$$

$$= m_0D_0 + m_1D_1 + m_2D_2 + m_3D_3 +$$

$$m_4D_4 + m_5D_5 + m_6D_6 + m_7D_7$$

其输出端能提供地址输入变量的全部最小项。

能实现函数发生器吗？怎样实现？

三、用数据选择器实现组合逻辑函数

$$Y = m_0D_0 + m_1D_1 + m_2D_2 + m_3D_3 +$$

$$m_4D_4 + m_5D_5 + m_6D_6 + m_7D_7$$

$$Y = \sum_{i=0}^{2^k-1} m_iD_i$$

设：K为选择器的选择输入端数，N为逻辑函数的变量数

1、$N=K$

2、$N<K$

3、$N>K$

1、N=K

✹ ［例］试用数据选择器实现函数 $Y=AB+AC+BC$ 。

代 数 法 求 解　　选用 CT74HC151

解：

(1) 写出逻辑函数的最小项表达式

$$Y=AB+AC+BC=A'BC+AB'C+ABC'+ABC$$

(2) 写出数据选择器的输出表达式

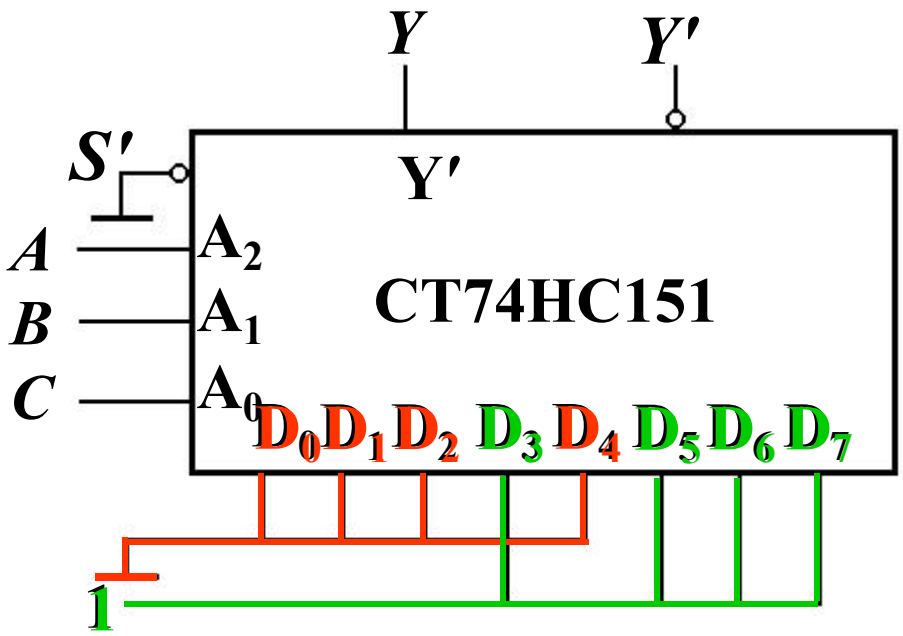
$$Y'=A'B'C'D_0+A'B'CD_1+A'BC'D_2+A'BCD_3$$
$$+AB'C'D_4+AB'CD_5+ABC'D_6+ABCD_7$$

(3) 比较 Y 和 Y' 两式中最小项的对应关系

令 $A=A_2$，$B=A_1$，$C=A_0$

为使 $Y=Y'$，应令 $\begin{cases} D_0=D_1=D_2=D_4=0 \\ D_3=D_5=D_6=D_7=1 \end{cases}$

(4) 画连线图



卡 诺 图 法 求 解

［例］ 试用数据选择器实现函数 $Y = AB + AC + BC$ 。

解：(1) 选择数据选择器　　选用 CT74HC151

(2) 画出 Y 和数据选择器输出 Y' 的卡诺图

Y 的卡诺图

A \ BC	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Y' 的卡诺图

A_2 \ A_1A_0	00	01	11	10
0	D_0	D_1	D_3	D_2
1	D_4	D_5	D_7	D_6

(3) 比较逻辑函数 Y' 和 Y 的卡诺图

设 $Y = Y'$、$A = A_2$、$B = A_1$、$C = A_0$

对比两张卡诺图后得 $\begin{cases} D_0 = D_1 = D_2 = D_4 = 0 \\ D_3 = D_5 = D_6 = D_7 = 1 \end{cases}$

(4) 画连线图　　与代数法所得图相同

用数据选择器实现逻辑函数，用8－1MUX实现Y(A,B,C)，真值表已知

1）列8-1MUX功能表
2）对比Y函数真值表和8-1MUX功能表
3）连接对应输入输出信号
4）给控制信号正确的值

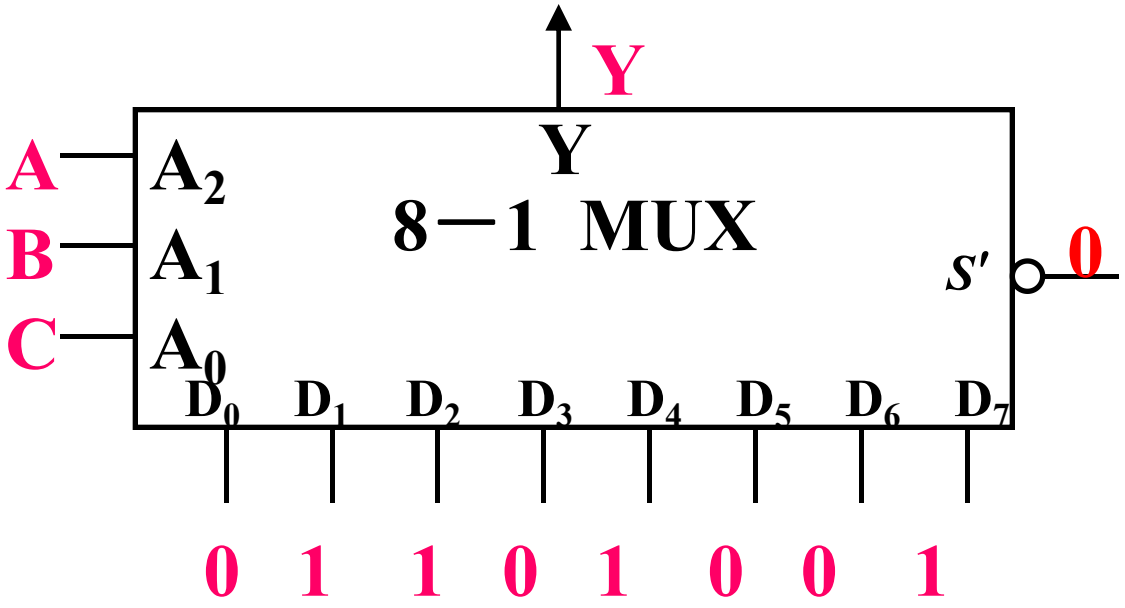
对比

Y函数真值表

ABC	Y
000	0
001	1
010	1
011	0
100	1
101	0
110	0
111	1

8-1MUX功能表

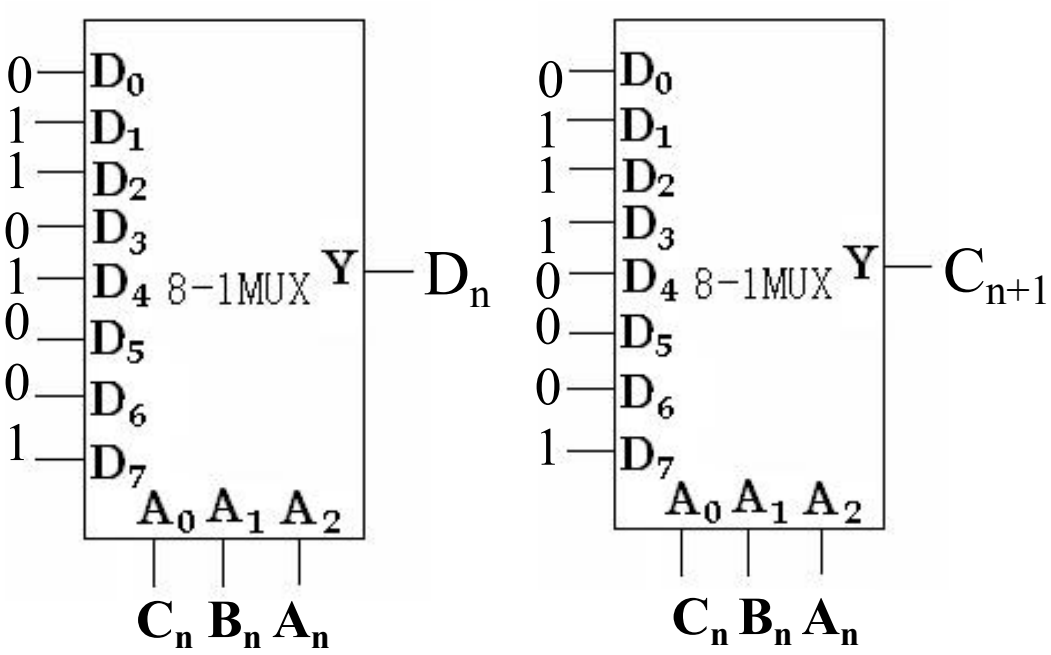
$A_2A_1A_0$	Y
000	D_0
001	D_1
010	D_2
011	D_3
100	D_4
101	D_5
110	D_6
111	D_7



例：用8－1MUX实现一位全减器

全减器真值表

输入			输出	
A_n	B_n	C_n	D_n	C_{n+1}
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



2、N<K

例　用8－1MUX实现Y=X1+X0

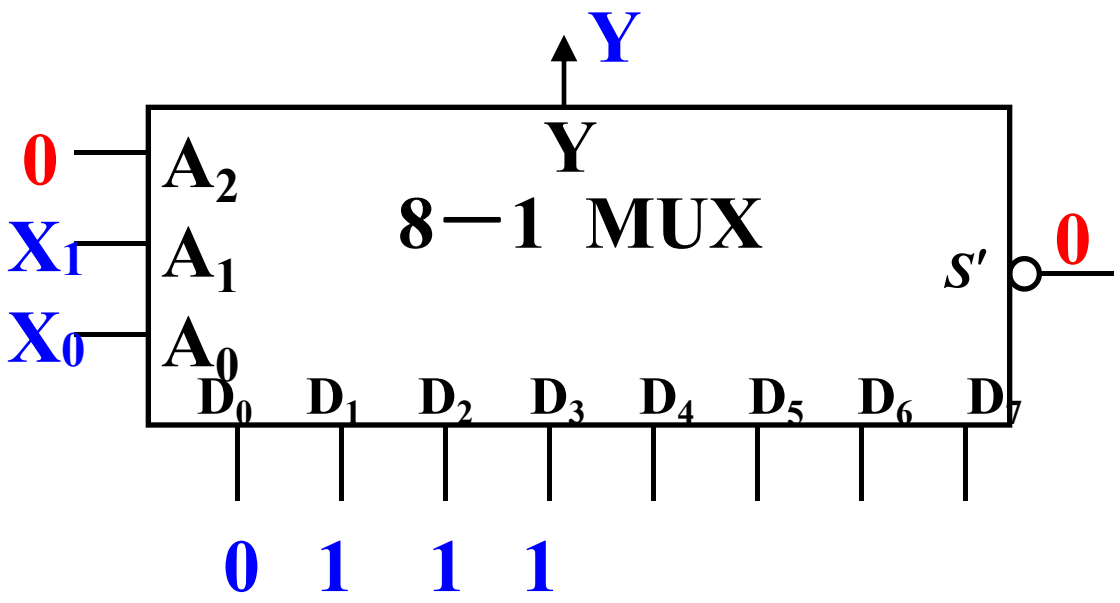
1）列函数Y的真值表
2）对比Y函数真值表和8-1MUX功能表
3）连接对应输入输出信号
4）给控制信号S及多余信号A2赋正确的值

Y函数真值表

X_1X_0	Y
0 0	0
0 1	1
1 0	1
1 1	1

8-1MUX功能表

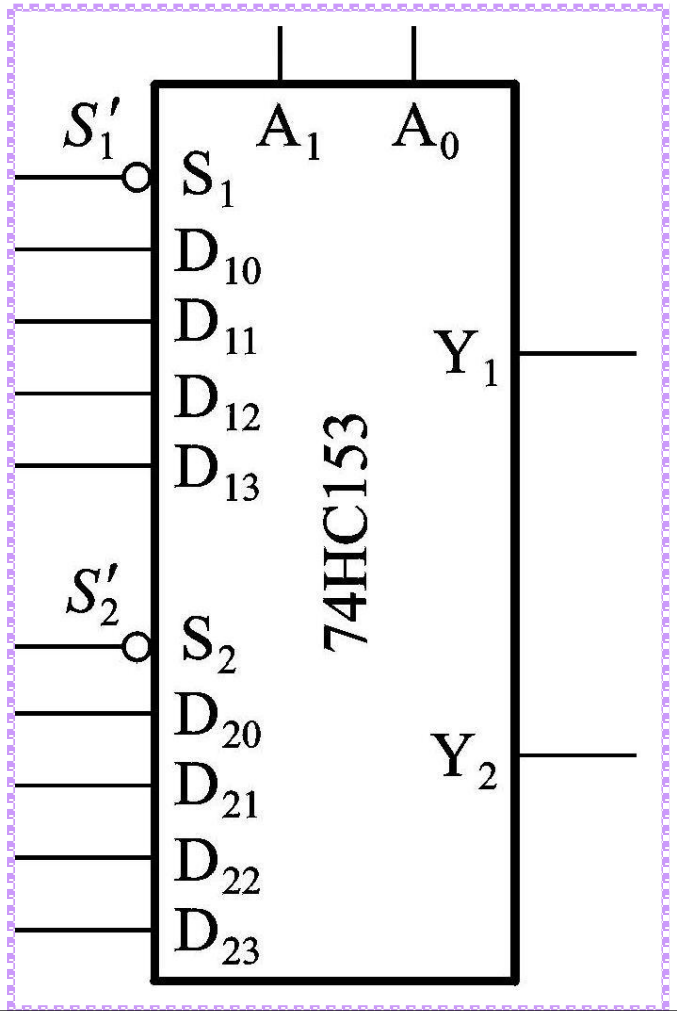
$A_2A_1A_0$	Y
0 0 0	D_0
0 0 1	D_1
0 1 0	D_2
0 1 1	D_3
1 0 0	D_4
1 0 1	D_5
1 1 0	D_6
1 1 1	D_7



3、N>K　　有两种方法：扩展法和降维法。

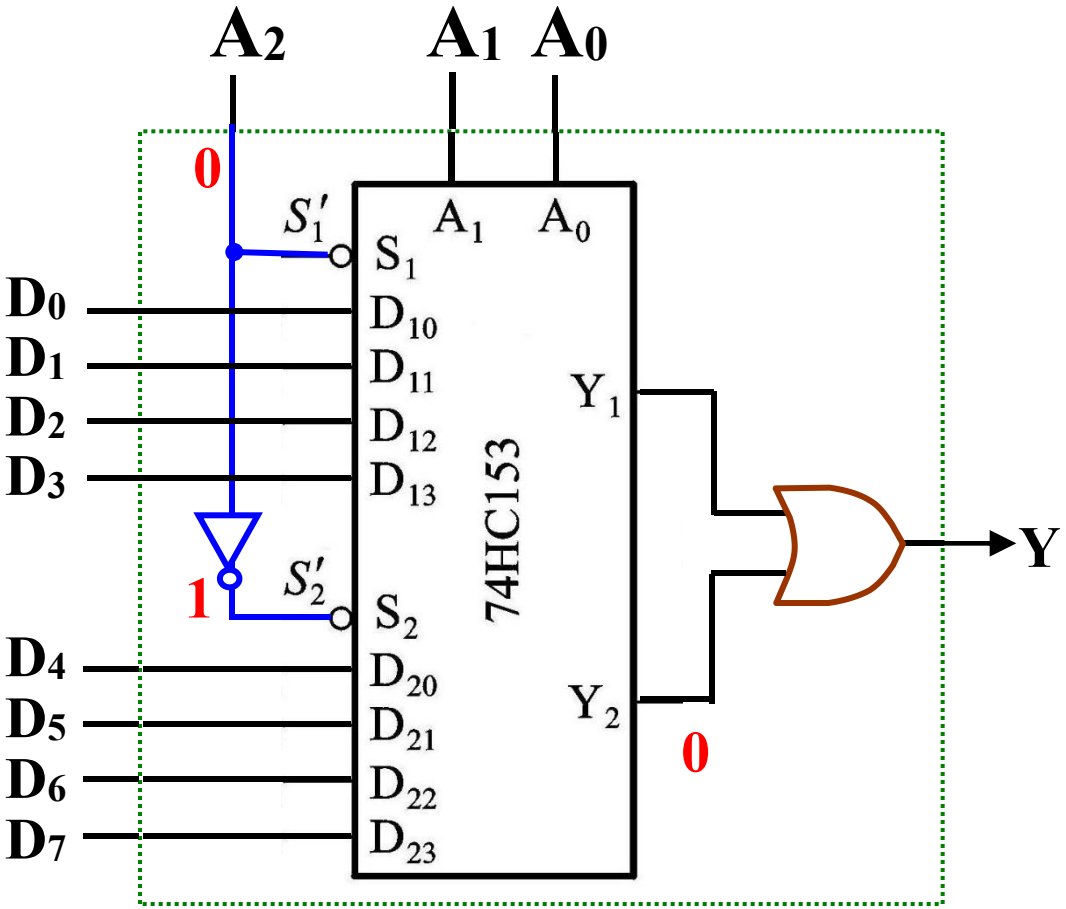
(1) 扩展法

实用芯片 74HC153,双4-1选通器(4-1MUX)



- 公共的地址输入端(A_1A_0)
- 独立的数据输入和输出端

例　用两个“4选1”接成一个“8选1”



8-1MUX 功能表

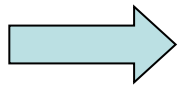
$A_2A_1A_0$	Y
0 0 0	D_0
0 0 1	D_1
0 1 0	D_2
0 1 1	D_3
1 0 0	D_4
1 0 1	D_5
1 1 0	D_6
1 1 1	D_7

$$Y=(A_2'A_1'A_0')D_0+(A_2'A_1'A_0)D_1+(A_2'A_1A_0')D_2+(A_2'A_1A_0)D_3$$
$$+(A_2A_1'A_0')D_4+(A_2A_1'A_0)D_5+(A_2A_1A_0')D_6+(A_2A_1A_0)D_7$$

(2) 降维法（引入变量卡诺图）

一个逻辑函数卡诺图的变量数称为卡诺图的维数。如果把某些变量也作为卡诺图小方格内的值，则会减少卡诺图的维数，这种卡诺图称为降维卡诺图。

A \ BC	00	01	11	10
0	1	0	0	0
1	1	1	1	0



A \ B	0	1
0	C'	0
1	1	C

三维降两维

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

四维降三维

A \ BC	00	01	11	10
0				
1				

三维降两维

A \ B	0	1
0		
1		

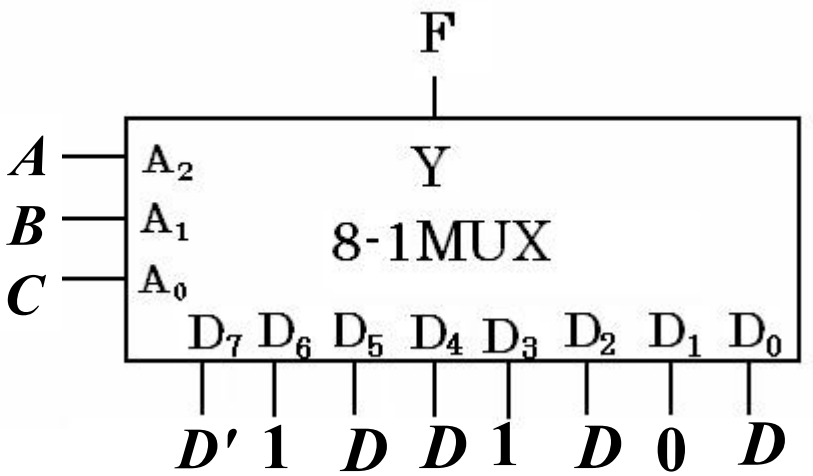
例: 用一个8－1MUX实现

$F(A,B,C,D)=\sum m(1,5,6,7,9,11,12,13,14)$

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

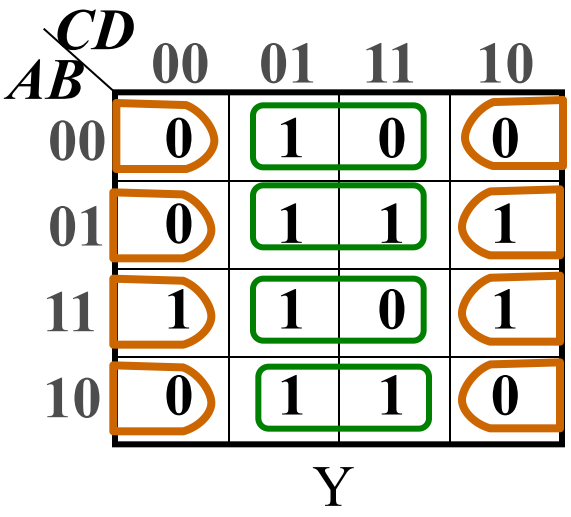
A \ BC	00	01	11	10
0	D	0	1	D
1	D	D	D′	1

如果用4－1MUX实现呢？

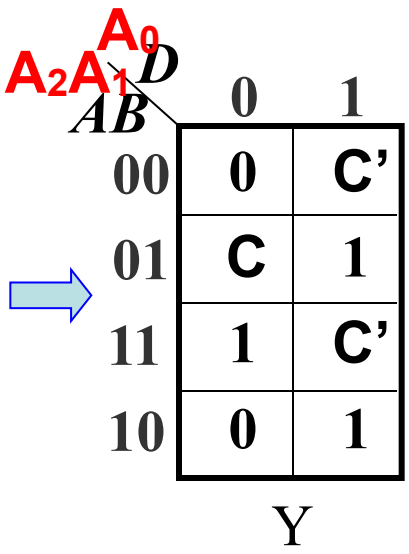


用一片8-1 MUX 实现Y(A,B,C,D) =∑m(1,5,6,7,9,11,12,13,14)，A2A1A0已经连接好，要求填写D0~D7的输入端

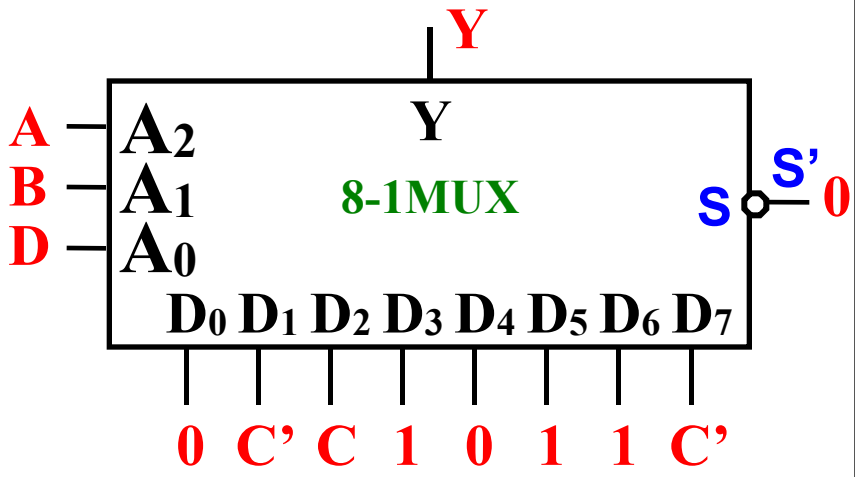
1) Y函数K图



2)Y函数 K图降维



3）对应8-1MUX的K图,连接信号线



A →A2
B →A1
D →A0
Y →Y

0 → D0
C' → D1
C → D2
1 → D3
0 → D4
1 → D5
1 → D6
C' → D7

例：用4－1MUX实现$F(A,B,C)$，其真值表已知。

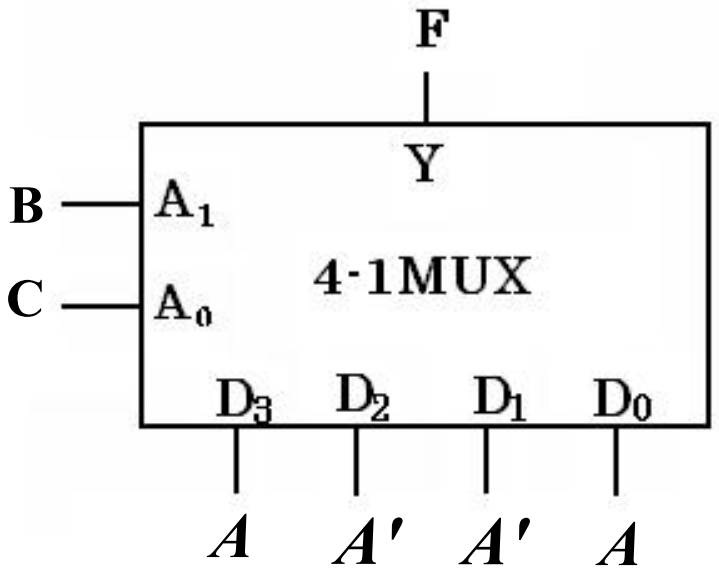
A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

利用卡诺图降维法

A \ BC	00	01	11	10
0	0	1	0	1
1	1	0	1	0

B \ C	0	1
0	A	A'
1	A'	A

F



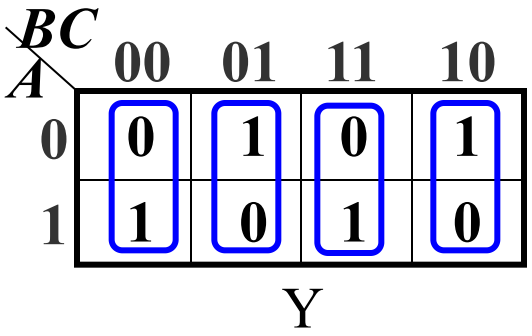
用一片4-1 MUX 实现函数Y(A,B,C)，真值表已知

Y函数
真值表

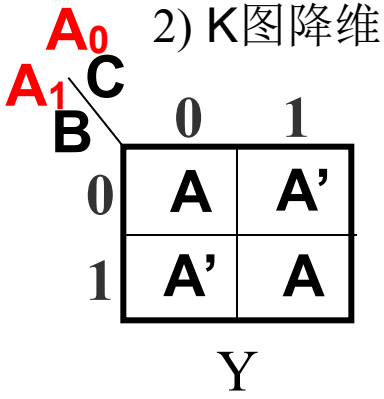
ABC	Y
000	0
001	1
010	1
011	0
100	1
101	0
110	0
111	1

方法I

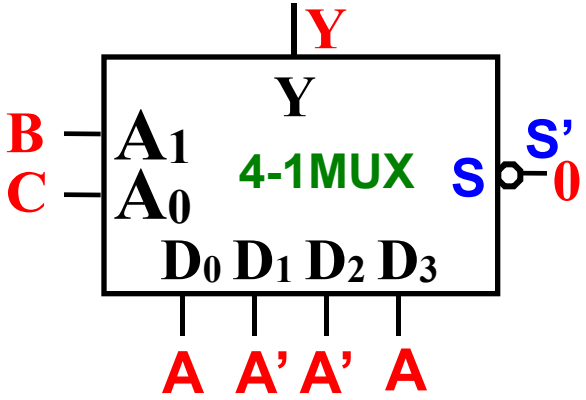
1) Y函数K图



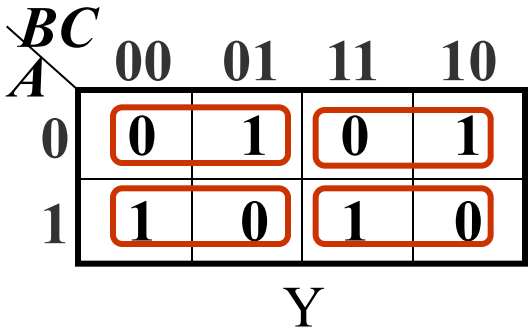
2) K图降维

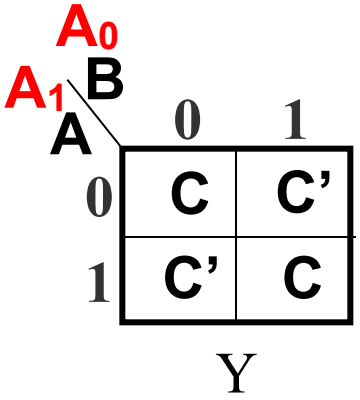


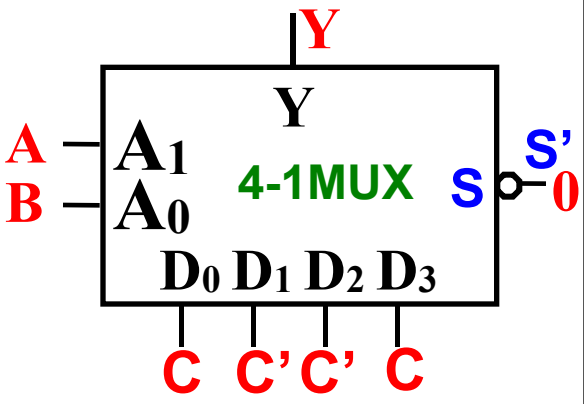
3）对应连接信号线



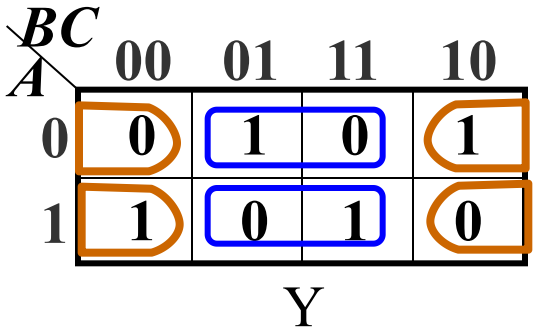
方法II

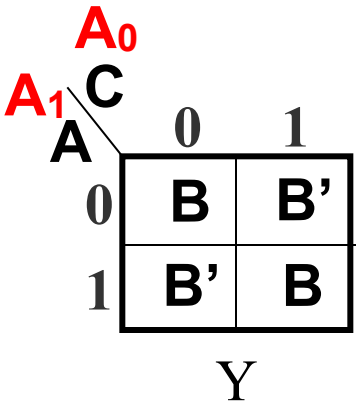


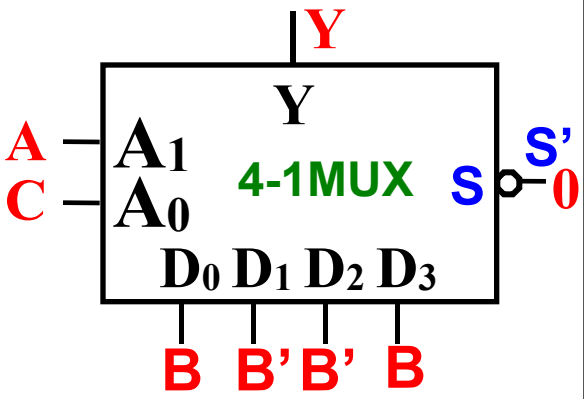




方法III

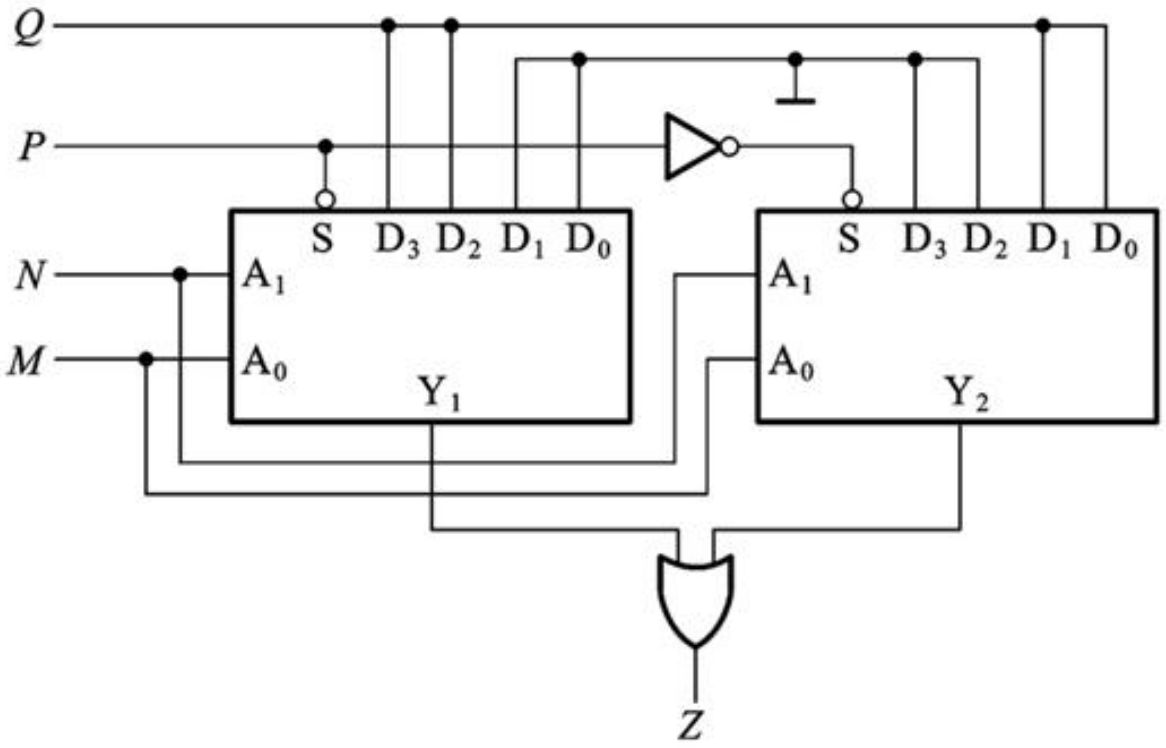




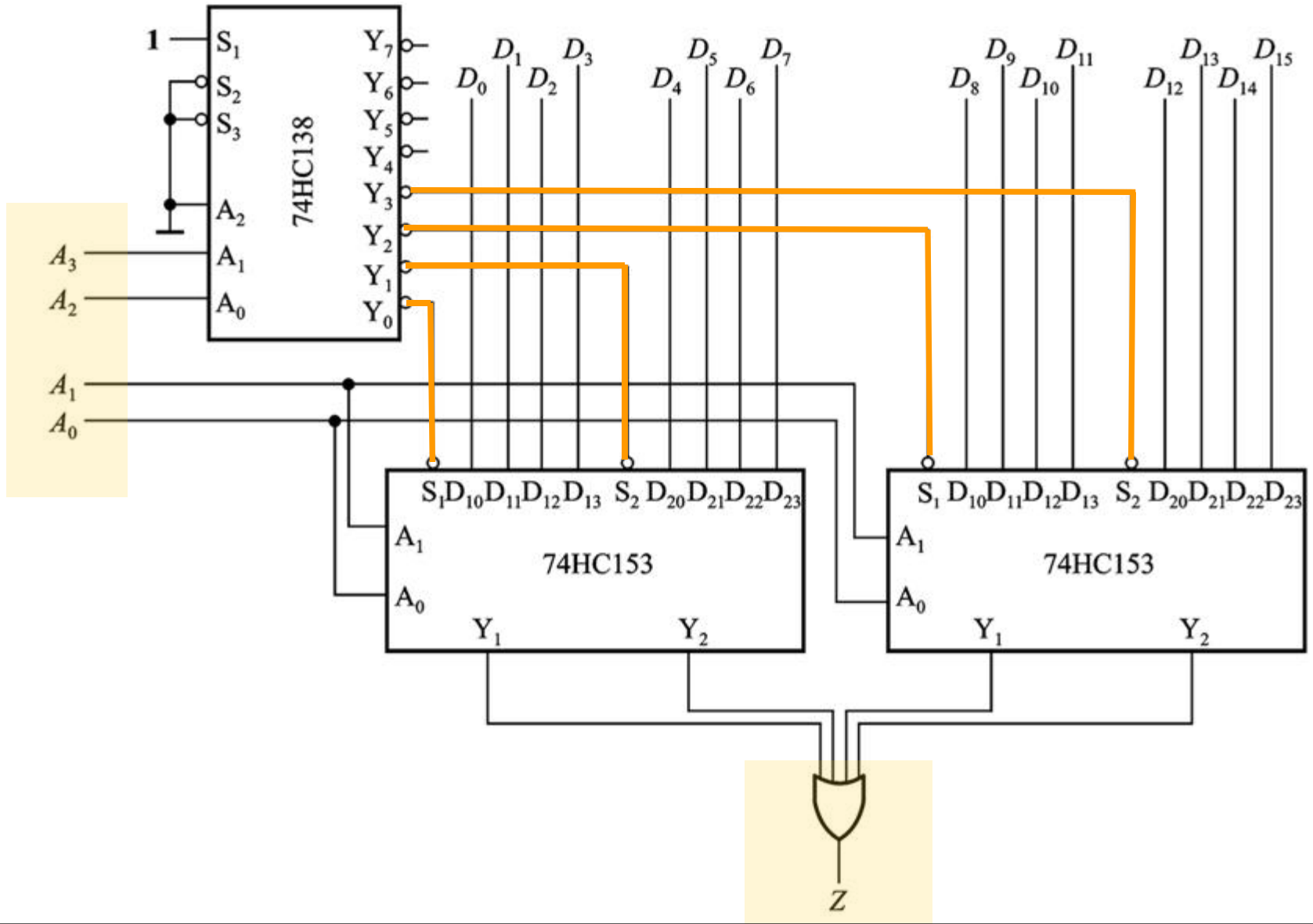


讨论： 图示是用两个4选1数据选择器组成的逻辑电路，试写出输出端Z与输入M、N、P、Q之间的逻辑函数式。已知数据选择器的逻辑函数式为：

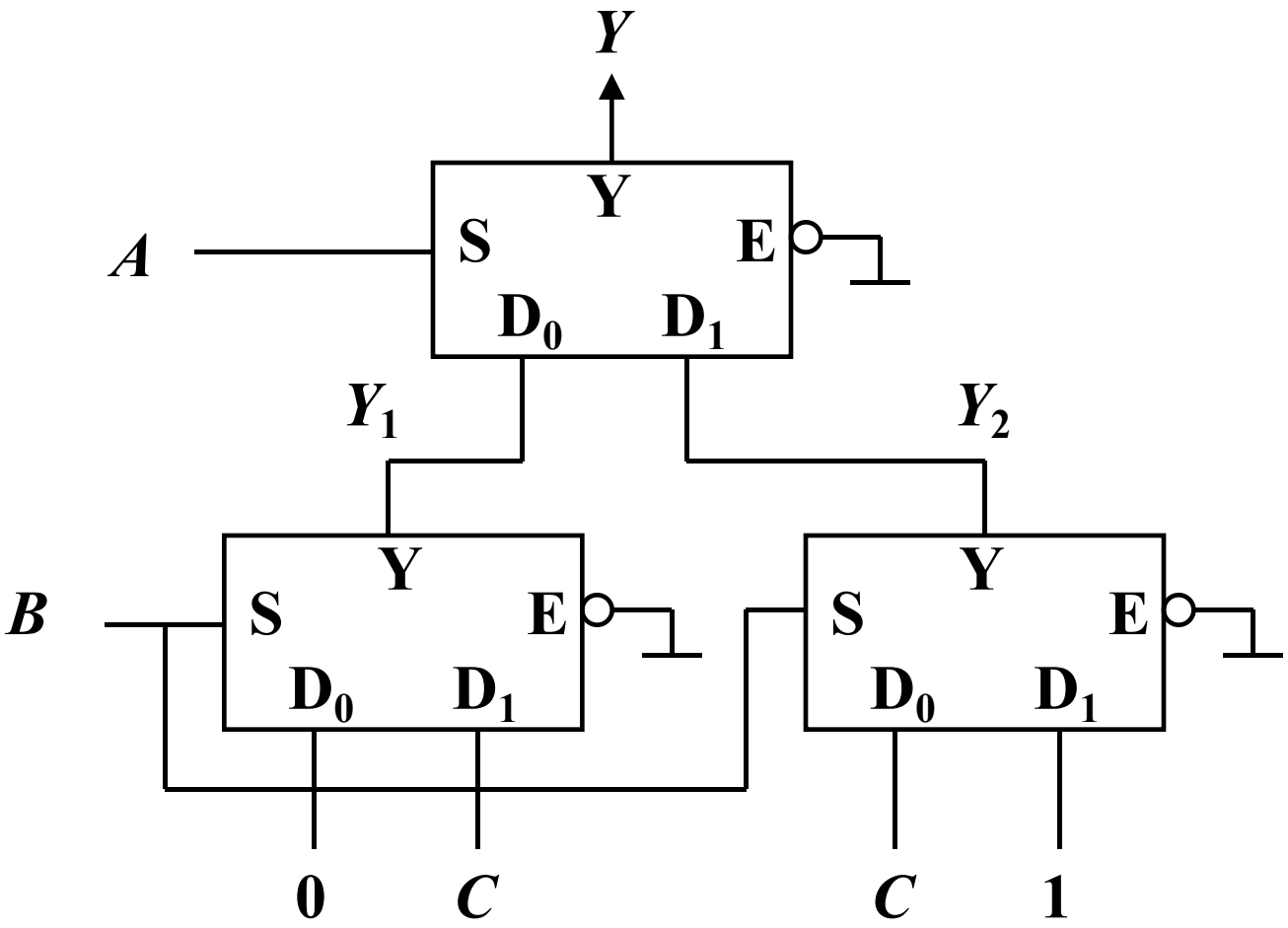
$$Y = (D_0 A_1' A_0' + D_1 A_1' A_0 + D_2 A_1 A_0' + D_3 A_1 A_0)$$



讨论 试用两片双4选1数据选择器74HC153和3线－8线译码器74HC138接成16选1的数据选择器。



讨论：图示电路是由三个2－1MUX组成的电路，试分析其逻辑功能。



作 业

4.16、 4.17、4.19、4.21

4.3.4　加法器

主要要求：

- 掌握加法器的逻辑功能及应用。

一、 半加器和全加器

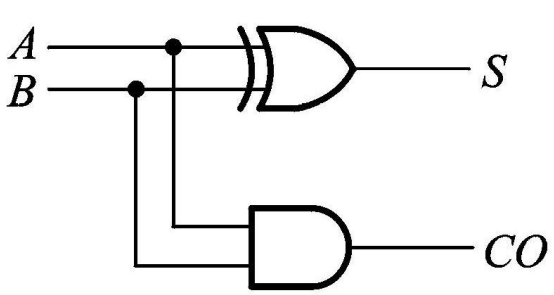
- 半加器

Half Adder，简称 HA。它只将两个 1 位二进制数相加，而不考虑低位来的进位。

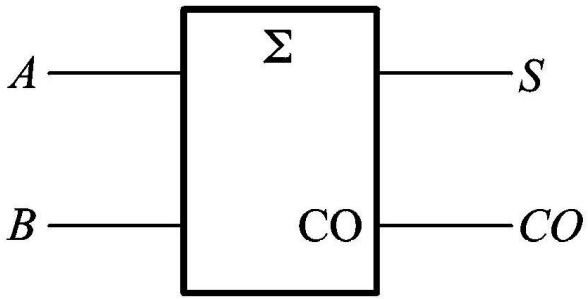
输 入		输 出	
A	B	S	CO
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$S = AB' + A'B = A \oplus B$$

$$CO = AB$$



(a)



(b)

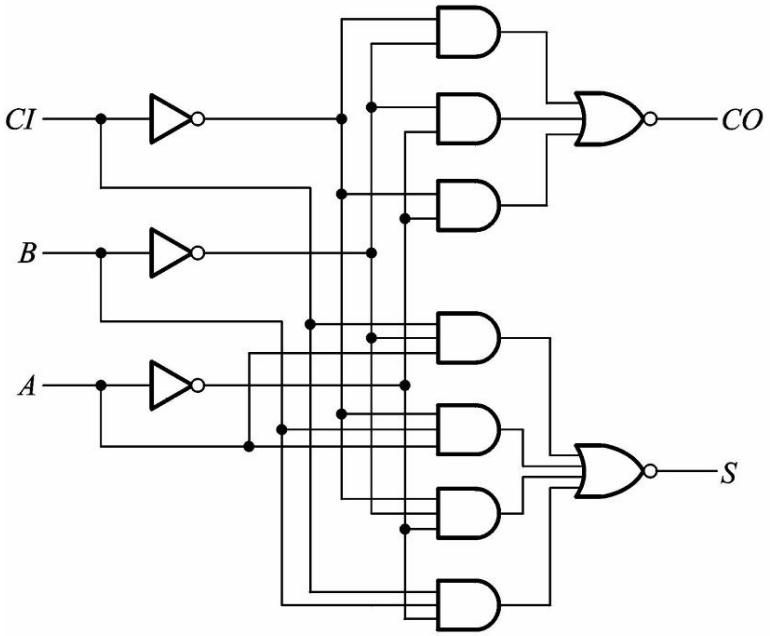
全加器

Full Adder，简称FA。能将本位的两个二进制数和相邻低位来的进位数进行相加。

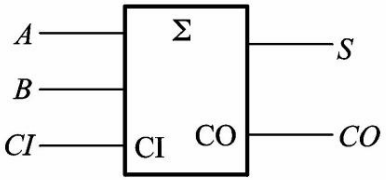
输入			输出	
A	*B*	*CI*	*S*	*CO*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$S=(A'B'CI'+A'B\cdot CI+AB'CI+ABCI')'$$

$$CO=(A'B'+B'CI'+A'CI')'$$



(a)



(b)

二、多位加法器

串行进位加法器

其低位进位输出端依次连至相邻高位的进位输入端，最低位进位输入端接地。因此，高位数的相加必须等到低位运算完成后才能进行，这种进位方式称为串行进位。运算速度较慢。

超前进位加法器

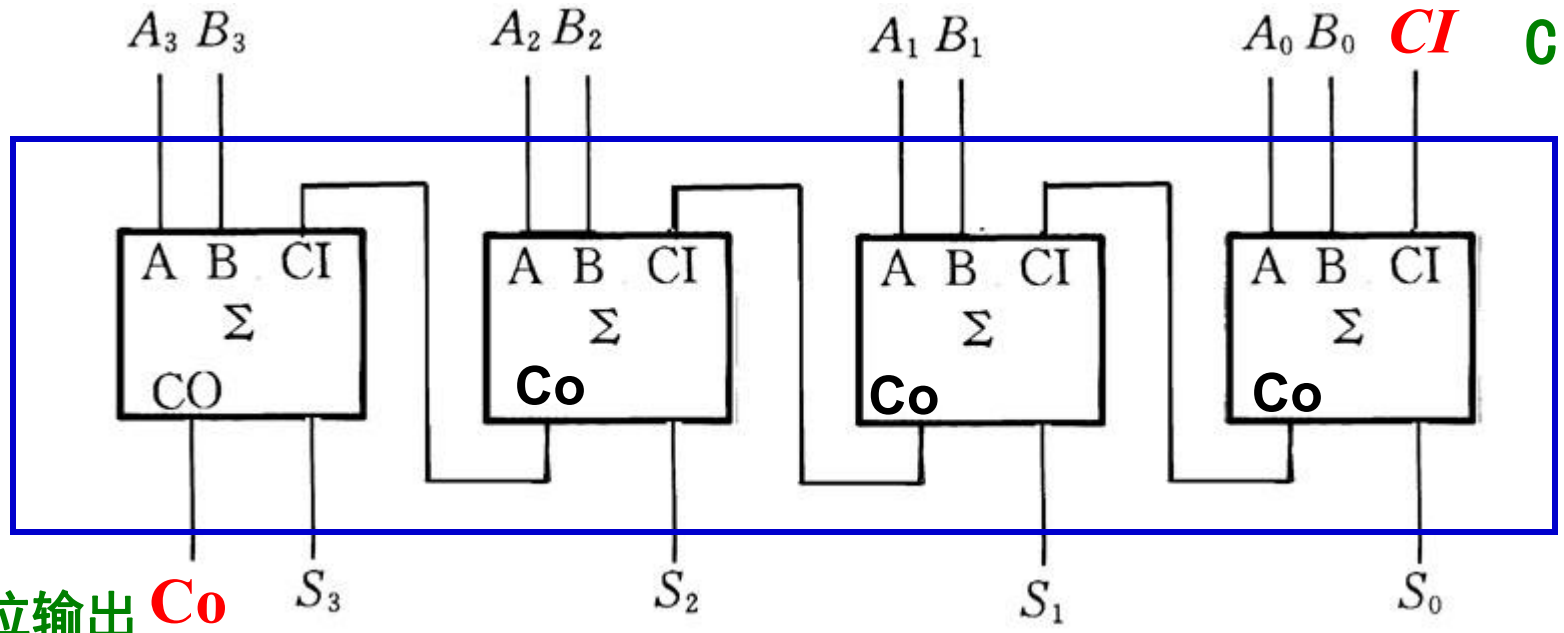
其进位数直接由加数、被加数和最低位进位数形成。各位运算并行进行。运算速度快。

1. 四位串行进位加法器

$A_3A_2A_1A_0$
$B_3B_2B_1B_0$
$C_OS_3S_2S_1S_0$

构成：把*n*位全加器串联起来，低位全加器的进位输出连接到相邻的高位全加器的进位输入。

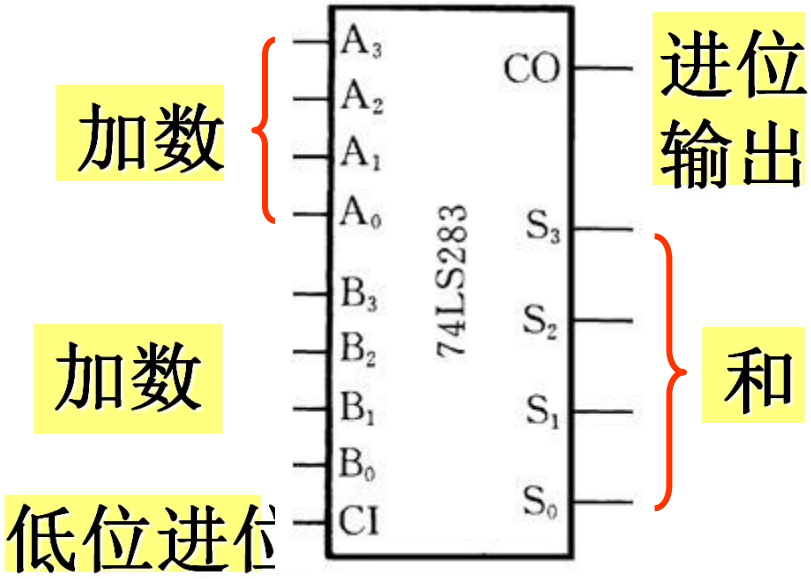
注意：
$CI_0=0$



优点：简单
缺点：慢

4位串行进位加法器

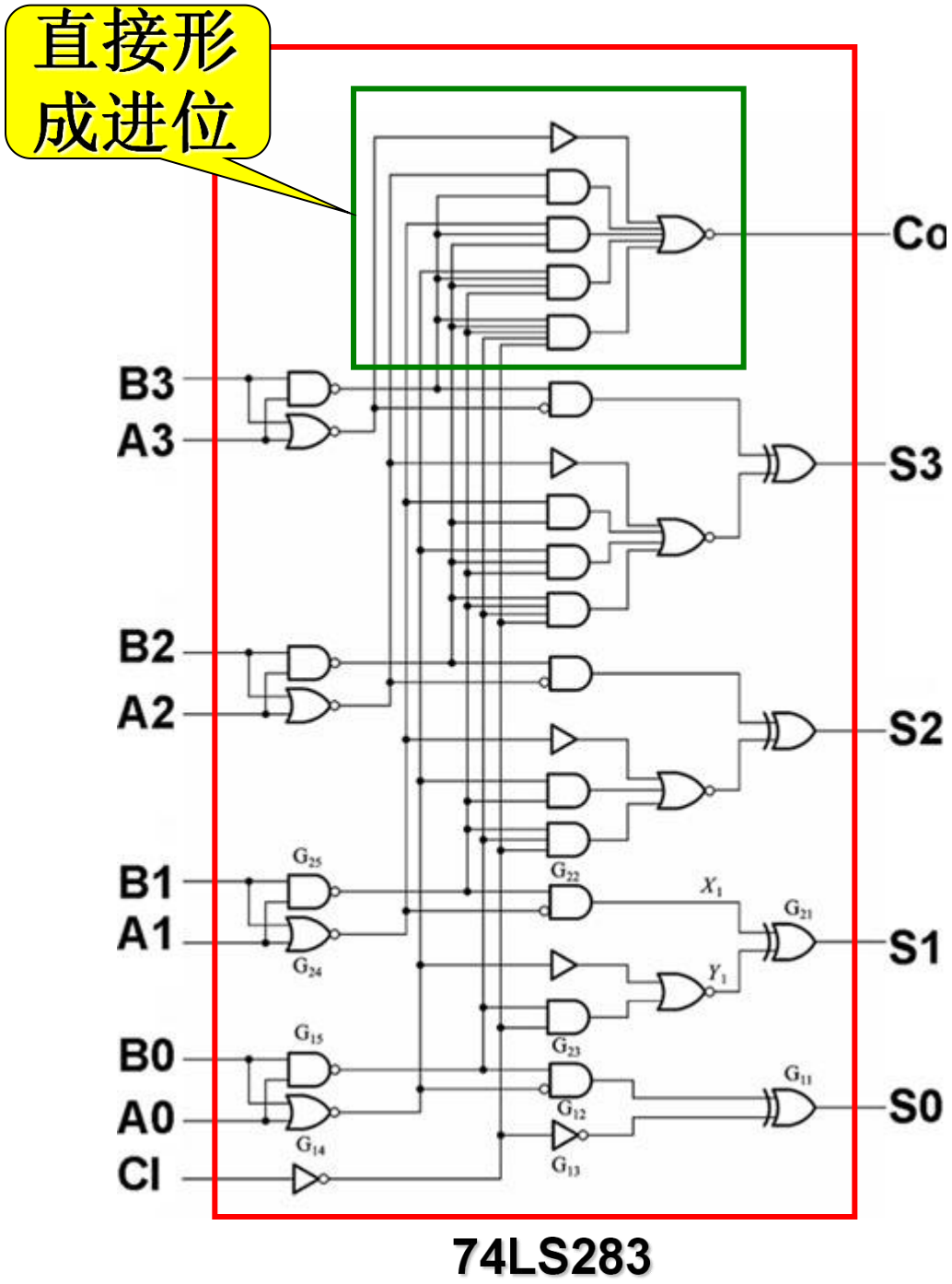
2. 超前进位加法器



4位超前进位加法器
74LS283

优点：快，每一位的和及最后的进位基本同时产生。

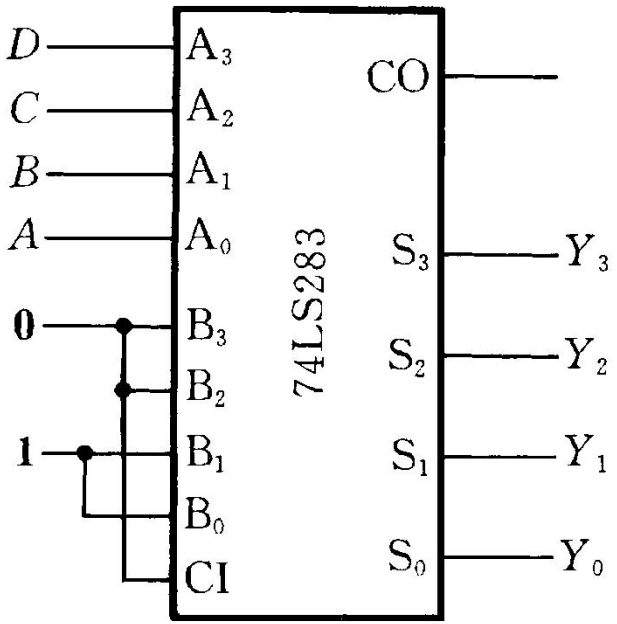
缺点：电路复杂



用加法器设计组合逻辑电路

例：将BCD8421码转换为余3码。

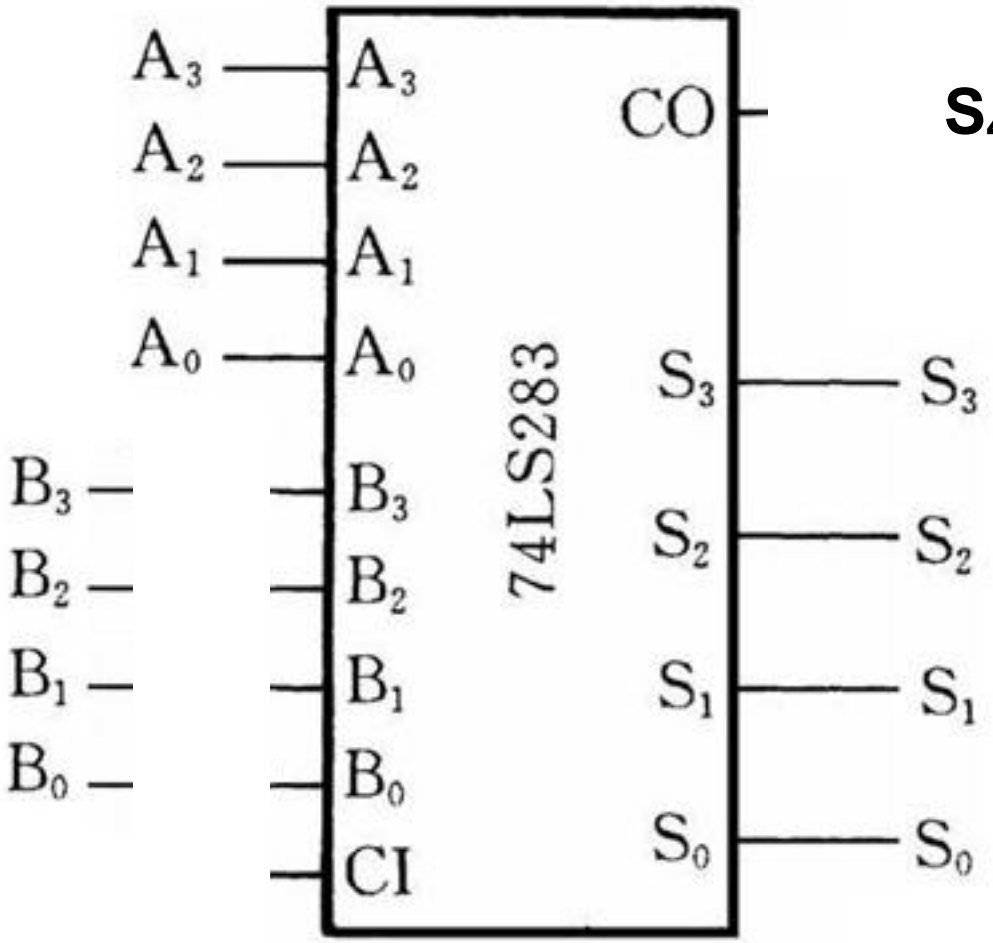
$$Y_3Y_2Y_1Y_0 = DCBA + 0011$$



输入				输出			
D	C	B	A	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

例：用74LS283设计四位减法器。

$A - B = C \rightarrow A+(-B)=C \rightarrow A_{补}+(-B)_{补}=C_{补}$



5位

$A_3A_2A_1A_0$

14 补→ 0,1110

- 11 补→ 1,0101

3 补→ (1)0,0011

$S_4S_3S_2S_1S_0$

2 补→ 0,0010

- 11 补→ 1,0101

- 9 补→ (1)1,0111

讨论：分析电路的逻辑功能。