第3章
组合逻辑电路

Video Image Processing (VIP)
Research Group @ Fudan
http://soc.fudan.edu.cn/vip/

范益波

2013.9

本章内容

如何利用单一逻辑构建复杂逻辑?

如何用电路实现加减乘除?

数字设计需要从“造轮子”开始么?

是否有多种方式实现同一逻辑?

重新认识加、减、乘、除

什么是电路的“竞争”和“冒险”?

本章要求

- 掌握组合逻辑电路的基本分析方法和一般设计过程
- 掌握常见逻辑模块的功能及其使用
- 掌握实际逻辑电路中冒险现象的形成原理及其防止方法

2.1 组合逻辑电路的分析

组合逻辑的结构：

组合逻辑电路（简称组合电路）任意时刻的输出信号仅取决于该时刻的输入信号，与信号作用前电路原来的状态无关

无记忆功能

一般分析过程

✓ 确定电路结构为组合电路

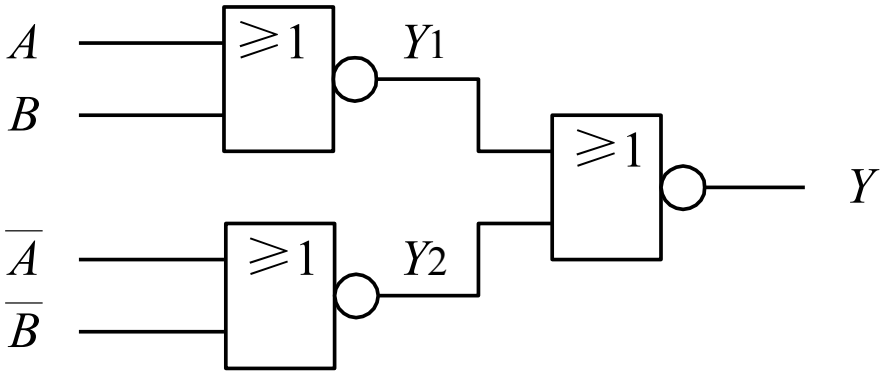
- 确定原则：输出唯一由输入确定，信号流是从输入端流向输出端，即只有信号的单向传输。

✓ 逐级分析，或划分模块后分析。得到逻辑表达式或真值表

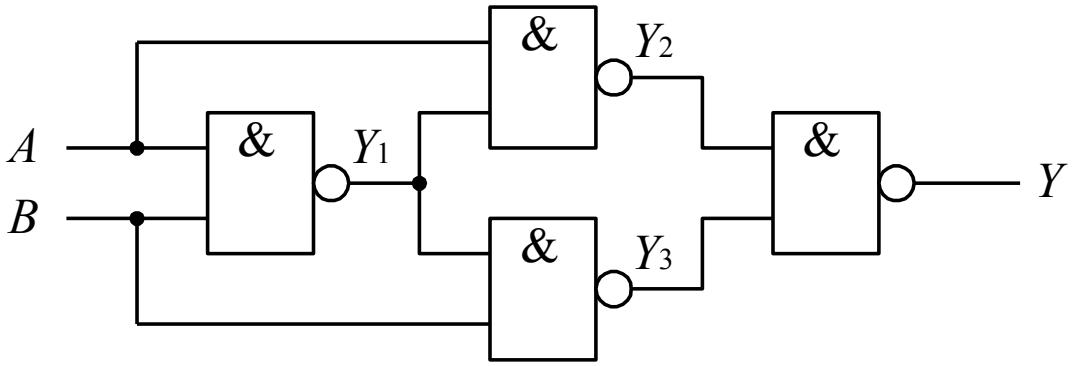
✓ 分析得到的逻辑表达式或真值表，得出电路的功能描述

组合逻辑的例：两种异或门结构

例2-1的分析
(p41)



例2-2的分析
(p41)



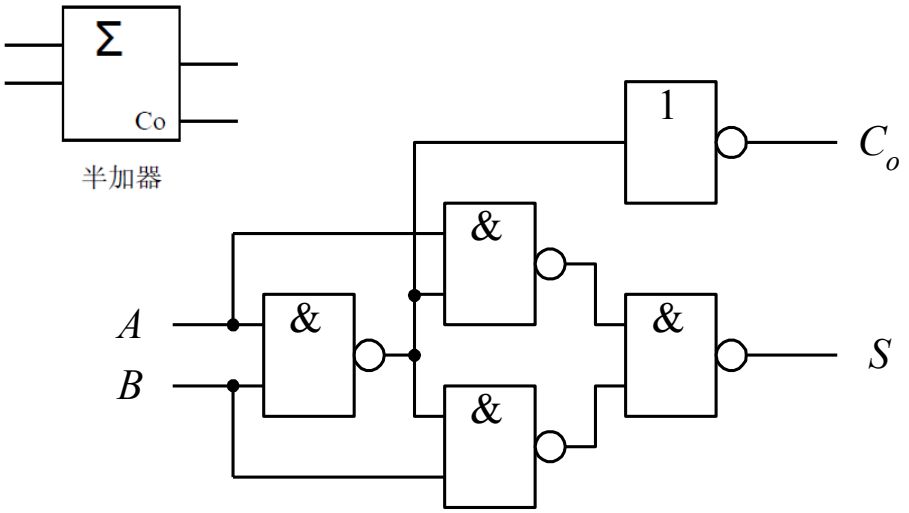
2-1采用2级或非门，例2-2采用3级与非门. 逻辑门的级数越少，电路延时就越短

半加器

本例可以看成：一个异或门(输出S)和一个与门(输出Co)的合成。

是一个带进位输出的二进制加法电路。

但是没有考虑低位来的进位。所以称为“半加器”。



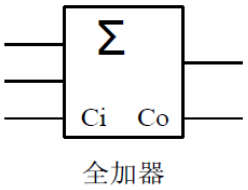
A	B	C_o	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

全加器

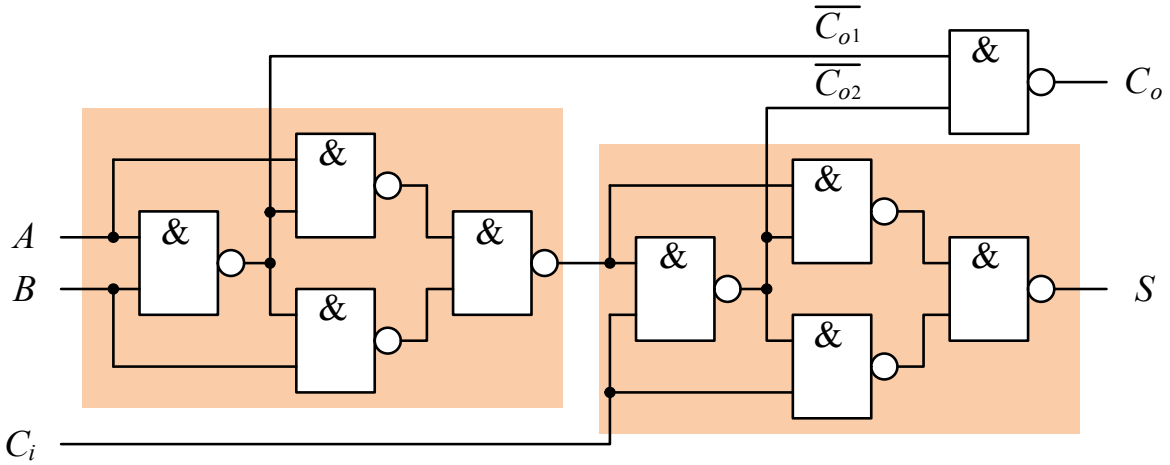
两个半加器的组合：

加数1＋加数2＋进位＝和，
进位1“或”进位2＝进位

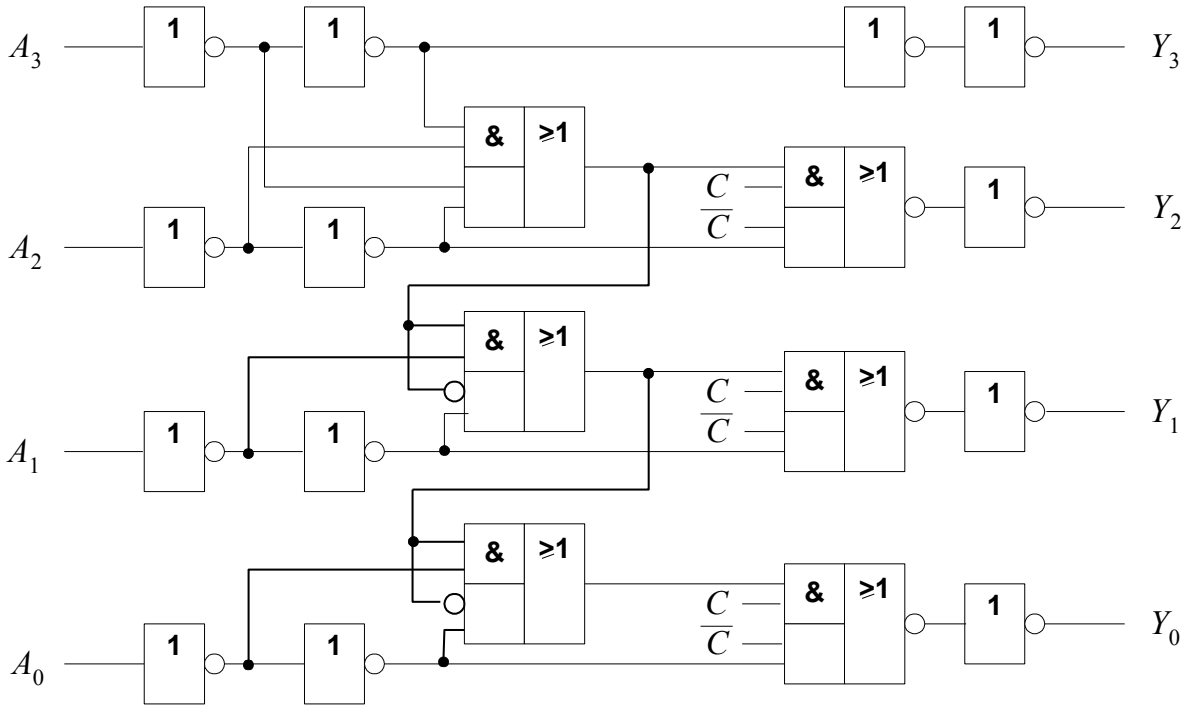
C_i	A	B	C_o	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



全加器



一个复杂的例子：代码转换



$C=0$: $Y_3=A_3$, $Y_2=A_2$, $Y_1=A_1$, $Y_0=A_0$

$C=1$: $Y_3=A_3$, $Y_2=A_3\overline{A_2}+\overline{A_3}A_2$, $Y_1=Y_2\overline{A_1}+\overline{Y_2}A_1$, $Y_0=Y_1\overline{A_0}+\overline{Y_1}A_0$

此例的真值表 $(C=1)$

格雷码:参考书p53

A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0

A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	1	0	1
1	1	1	1	1	1	0	0

结果:

C=0时输出与输入相同;C=1时输入二进制码,输出格雷码

常用组合逻辑模块

组合逻辑模块是一些基本的逻辑单元

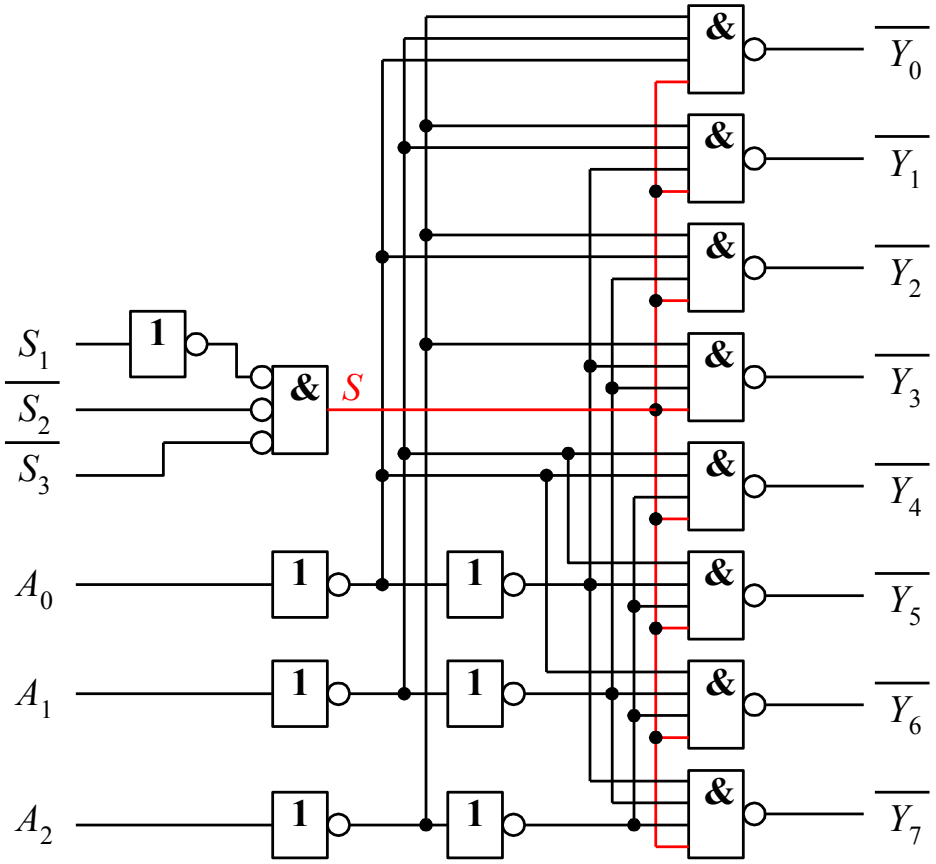
熟悉组合逻辑模块的结构与功能，可以帮助分析复杂的逻辑结构

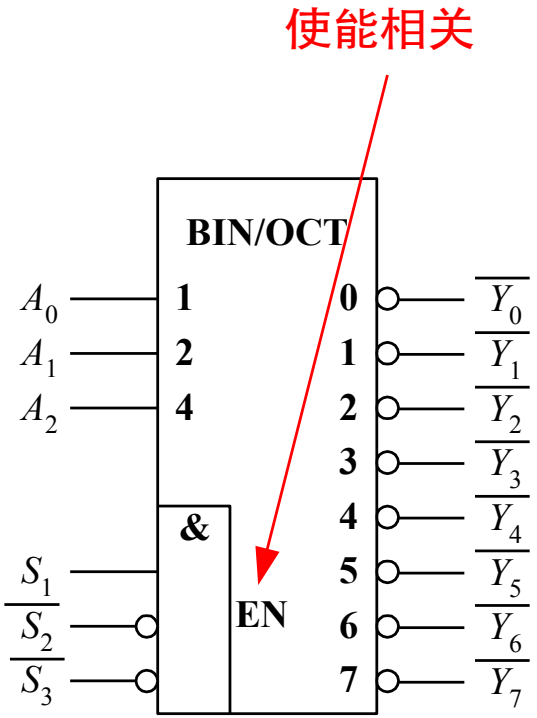
在设计逻辑电路时，可以从逻辑模块出发进行设计

译码器

- 将输入的某种代码（通常为二进制码），转换为事件或另一种代码输出的过程，称为译码。
- 转换为事件输出的译码器，是编码器的逆过程。
- 转换为另一种代码输出的译码器，根据两个代码之间的关系，可以有各种不同的译码器。
- 常见的译码器：
  - 转换为事件输出的译码器：3-8译码器、等等。
  - 转换为另一种代码输出的译码器：（LED）七段译码器、BCD译码器、等等。

3-8译码器（74LS138）





- 3个控制输入S_1、S_2、S_3:
  - 当满足 $S_1\overline{S_2}\overline{S_3}=100$, 可以进行正常译码。
  - 这三个输入称为译码器的“使能”(Enable)输入。

  3个控制端也叫片选输入端。可将多片连接起来以扩展译码器的功能。

- 输出的逻辑表达式(以Y_0为例): $\overline{Y_0}=\overline{\overline{A_2}\,\overline{A_1}\,\overline{A_0}\cdot(S_1S_2S_3)}$

- 当$S_1S_2S_3$=1时，3-8译码器的输出分别为A2A1A0这3个变量的相应编号的最小项，例如: $\overline{Y_0}=\overline{\overline{A_2}\,\overline{A_1}\,\overline{A_0}}=\overline{m_0}$

- 这种译码器也称作“最小项译码器”

$$Y_i=m_i\cdot(S_1S_2S_3)=m_i\cdot S$$

例如，8盏灯，按照输入不同，点亮。

3-8译码器的真值表

输　入					输　出							
S_1	$\overline{S_2}+\overline{S_3}$	A_2	A_1	A_0	$\overline{Y_0}$	$\overline{Y_1}$	$\overline{Y_2}$	$\overline{Y_3}$	$\overline{Y_4}$	$\overline{Y_5}$	$\overline{Y_6}$	$\overline{Y_7}$
0	x	x	x	x	1	1	1	1	1	1	1	1
x	1	x	x	x	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

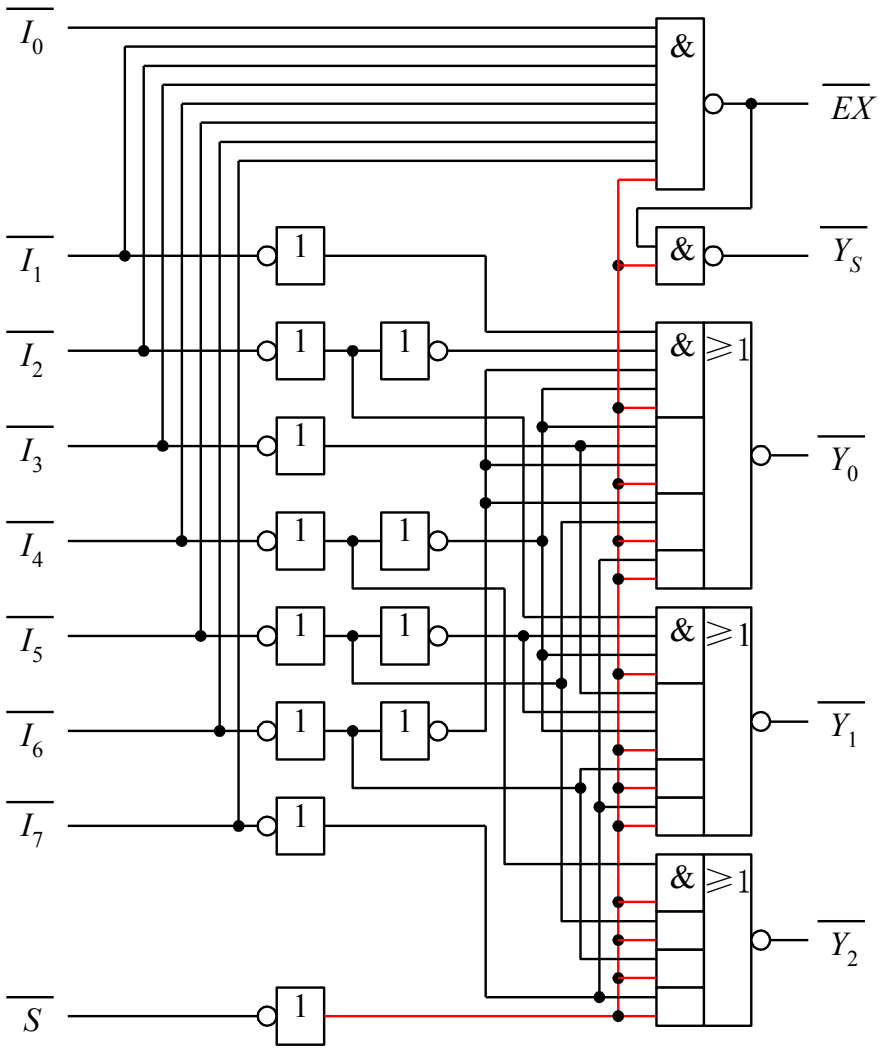
译码器的扩展

例如：用3-8译码器构成4-16译码器，5-32译码器，6-64译码等。

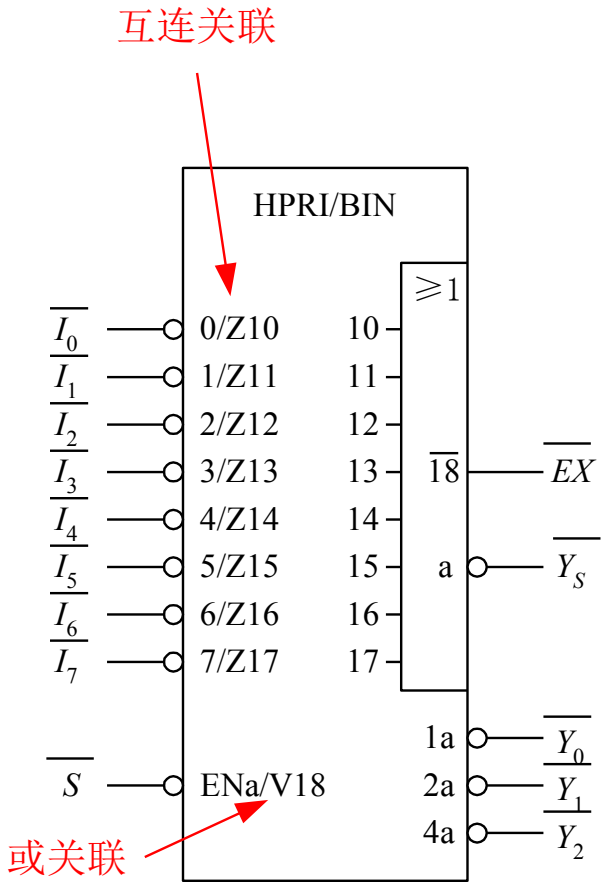
编码器

- 将输入信号（事件），用一个代码表示（输出）的过程，称为编码。
- 编码器有普通编码器和优先编码器两种。
- 普通编码器在同一个时刻只能允许有一个输入（单个事件）。
- 优先编码器允许多个事件同时发生，按照事先设定的优先级，确定输出代码。

8-3优先编码器



8个输入端，3个输出端



8-3优先编码器

- 优先编码器的逻辑表达式：

$$\overline{Y_2} = \overline{(I_4 + I_5 + I_6 + I_7) \cdot S}$$

$$\overline{Y_1} = \overline{(I_2 \overline{I_4}\,\overline{I_5} + I_3 \overline{I_4}\,\overline{I_5} + I_6 + I_7) \cdot S}$$

$$\overline{Y_0} = \overline{(I_1 \overline{I_2}\,\overline{I_4}\,\overline{I_6} + I_3 \overline{I_4}\,\overline{I_6} + I_5 \overline{I_6} + I_7) \cdot S}$$

$$\overline{E_X} = \overline{\overline{I_0}\,\overline{I_1}\,\overline{I_2}\,\overline{I_3}\,\overline{I_4}\,\overline{I_5}\,\overline{I_6}\,\overline{I_7} \cdot S}$$

$$\overline{Y_S} = \overline{\overline{E_X} \cdot S}$$

8-3优先编码器的真值表

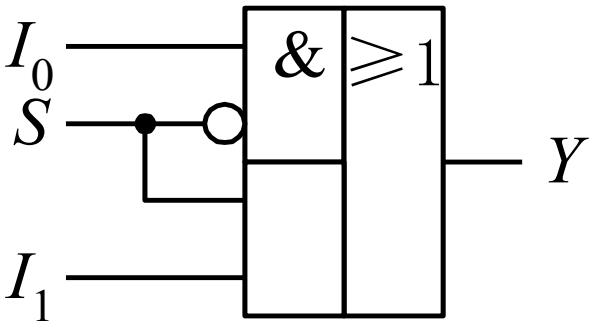
$\overline{s}$	$\overline{I_7}\ \overline{I_6}\ \overline{I_5}\ \overline{I_4}\ \overline{I_3}\ \overline{I_2}\ \overline{I_1}\ \overline{I_0}$	$Y_2\ Y_1\ Y_0$
0	0 X X X X X X X	1 1 1
0	1 0 X X X X X X	1 1 0
0	1 1 0 X X X X X	1 0 1
0	1 1 1 0 X X X X	1 0 0
0	1 1 1 1 0 X X X	0 1 1
0	1 1 1 1 1 0 X X	0 1 0
0	1 1 1 1 1 1 0 X	0 0 1
0	1 1 1 1 1 1 1 0	0 0 0
1	X X X X X X X X	1 1 1

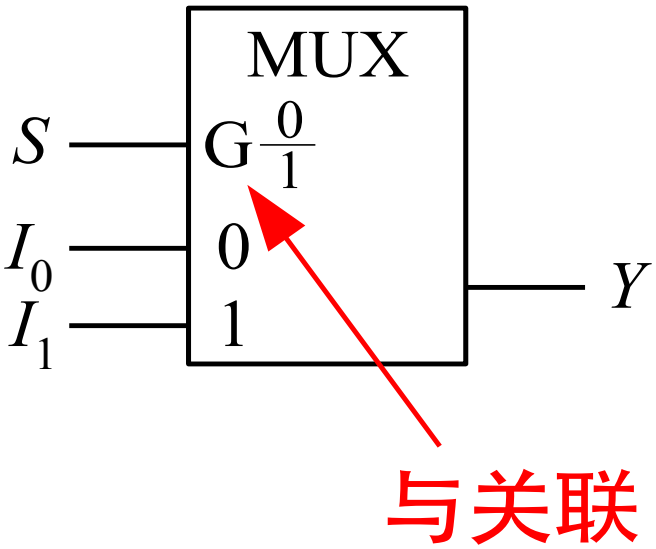
X为任意值。7号输入的优先级最高，0号输入的优先级最低。

数据选择器

- 从多个输入逻辑信号中选出一个逻辑信号送到输出端的器件，也称为多路器。
- 一个数据选择器连接m个输入，由n个选择变量决定这m个输入中的哪一个被送到输出端。这里$m = 2^n$。

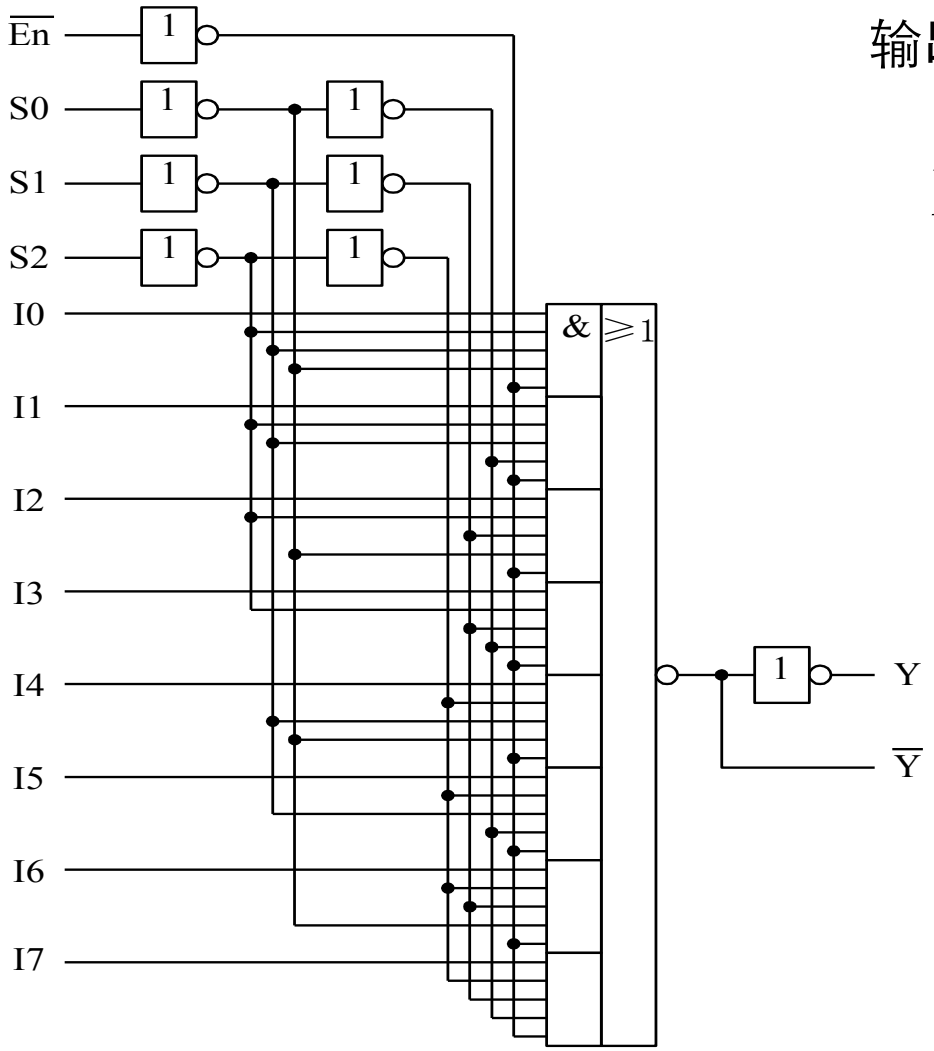
2选1数据选择器





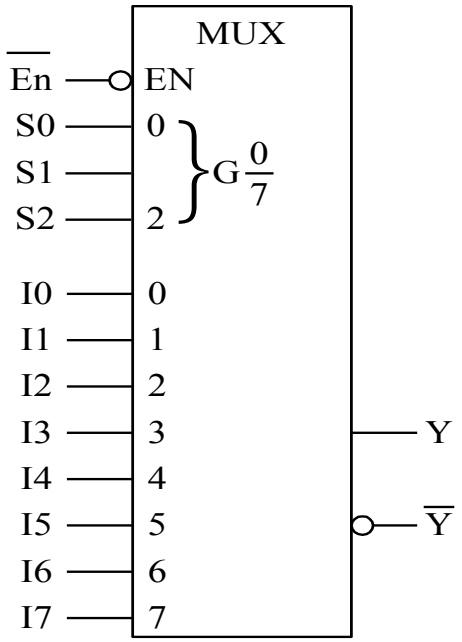
$$Y = \overline{S}I_0 + SI_1$$

8选1数据选择器



输出逻辑函数表达式：

$$Y = \sum_{i=0}^{7} EN \cdot m_i \cdot I_i$$



思考题

如何构建15选1

如何构建64选1

如何构建16选1

2.2 组合逻辑电路的设计

- 基于门电路的设计

  基本的设计方法。

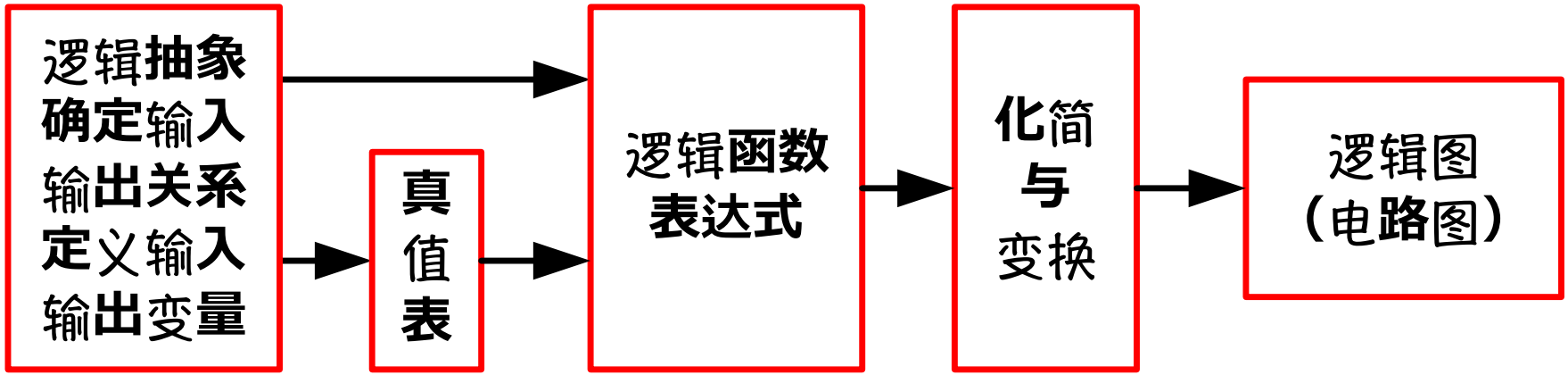
- 基于组合逻辑模块的设计

  利用组合电路模块实现主要功能，辅以门电路，结构比较简单。

- 运算电路设计

  需要熟悉二进制运算的特点，采用迭代设计。

一、基于门电路的设计方法



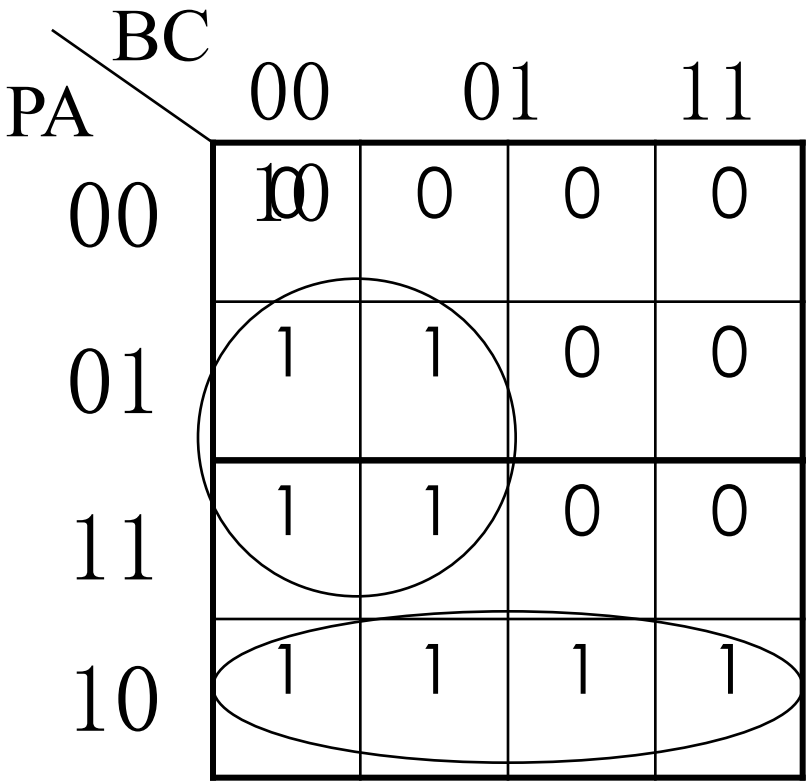
例1：完成以下设计

- 带控制端的 3 位输入代码检测电路
- 当控制端 P 为 0 时，输入 >3 并且 <6 时输出为 1
- 当控制端 P 为 1 时，输入 <6 时输出为 1
- 要求完成最简设计

步骤1、真值表

$PABC$	Y	$PABC$	Y
0 000	0	1 000	1
0 001	0	1 001	1
0 010	0	1 010	1
0 011	0	1 011	1
0 100	1	1 100	1
0 101	1	1 101	1
0 110	0	1 110	0
0 111	0	1 111	0

步骤2、卡诺图以及化简



$$Y = A\overline{B} + P\overline{A}$$

步骤3、完成设计（利用卡诺图运算的方案）

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

围绕1重心

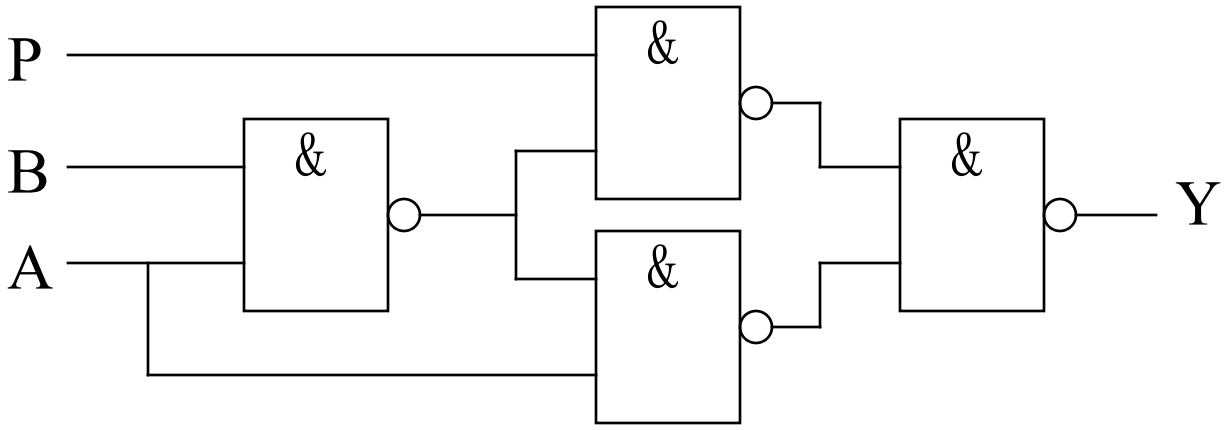
$$Y = (A+P)\overline{AB}$$

$$= P\overline{AB} + A\overline{AB}$$

$$= \overline{\overline{P\overline{AB}} \cdot \overline{A\overline{AB}}}$$

全部用与非门实现：

全部是原变量实现。



回顾：第一章中关于原变量围绕1中心的卡诺图化简

例子

参考书p51，例2-5，例2-6

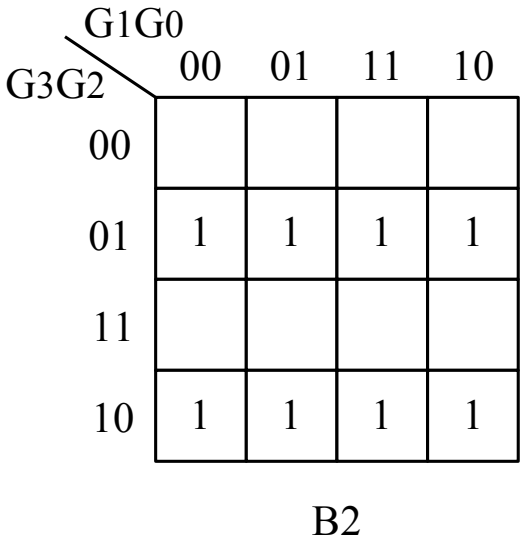
P55 2-8

例2-7 (p53) 设计一个4位格雷码和二进制码的相互转换电路

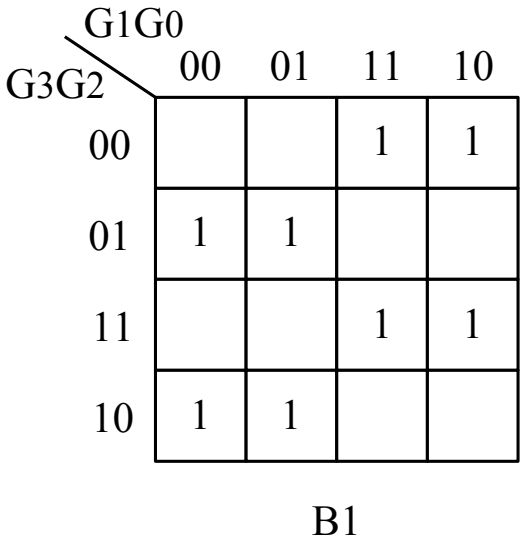
Dec	Bin	Gray	Dec	Bin	Gray
	$B_3B_2B_1B_0$	$G_3G_2G_1G_0$		$B_3B_2B_1B_0$	$G_3G_2G_1G_0$
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000

格雷码转换到二进制码：

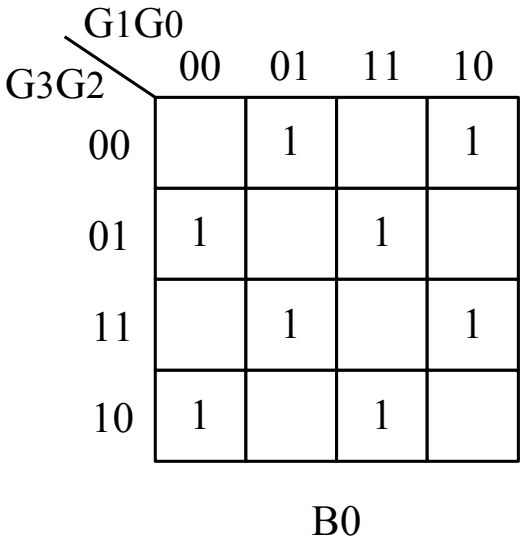
G3 = B3，G2~G0 转换到 B2~B0 的转换关系如下：



B2



B1



B0

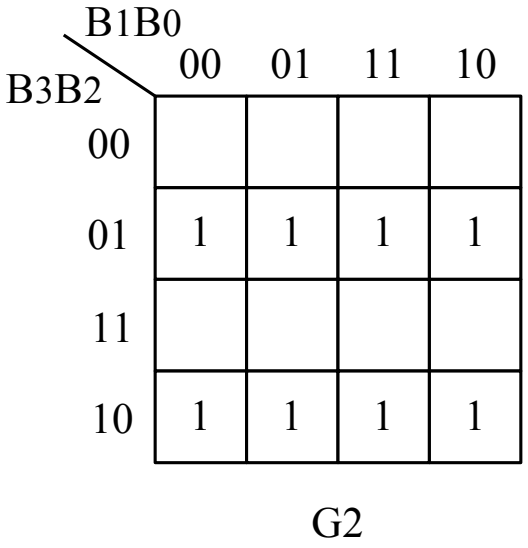
所以：

$$B_3 = G_3,\quad B_2 = G_3 \oplus G_2,\quad B_1 = B_2 \oplus G_1,\quad B_0 = B_1 \oplus G_0$$

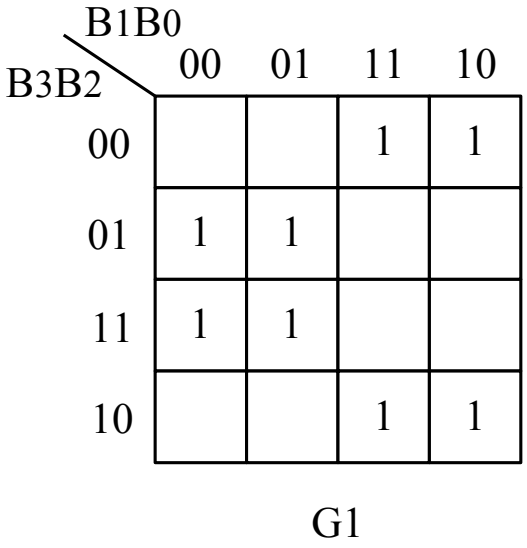
利用异或门化简卡诺图

二进制码转换到格雷码

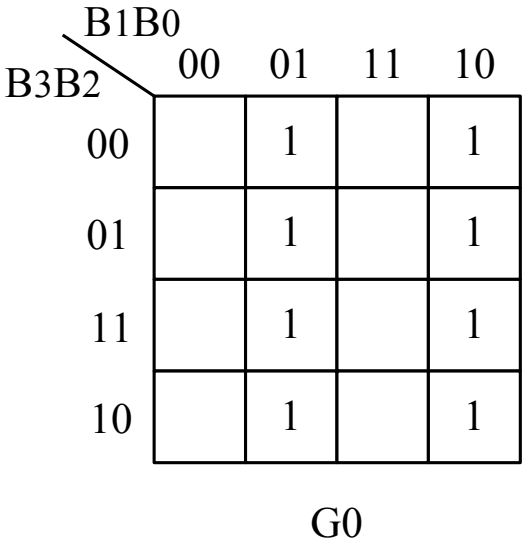
G3 = B3，B2~B0 转换到 G2~G0 的转换关系如下:



G2



G1



G0

所以：

$$G_3 = B_3,\quad G_2 = B_3 \oplus B_2,\quad G_1 = B_2 \oplus B_1,\quad G_0 = B_1 \oplus B_0$$

结果：以 S 作为选择端，S=0，$G \rightarrow B$；S=1，$B \rightarrow G$

S=0

$$Y_3 = X_3, \quad Y_2 = X_3 \oplus X_2$$

$$Y_1 = Y_2 \oplus X_1, \quad Y_0 = Y_1 \oplus X_0$$

S=1

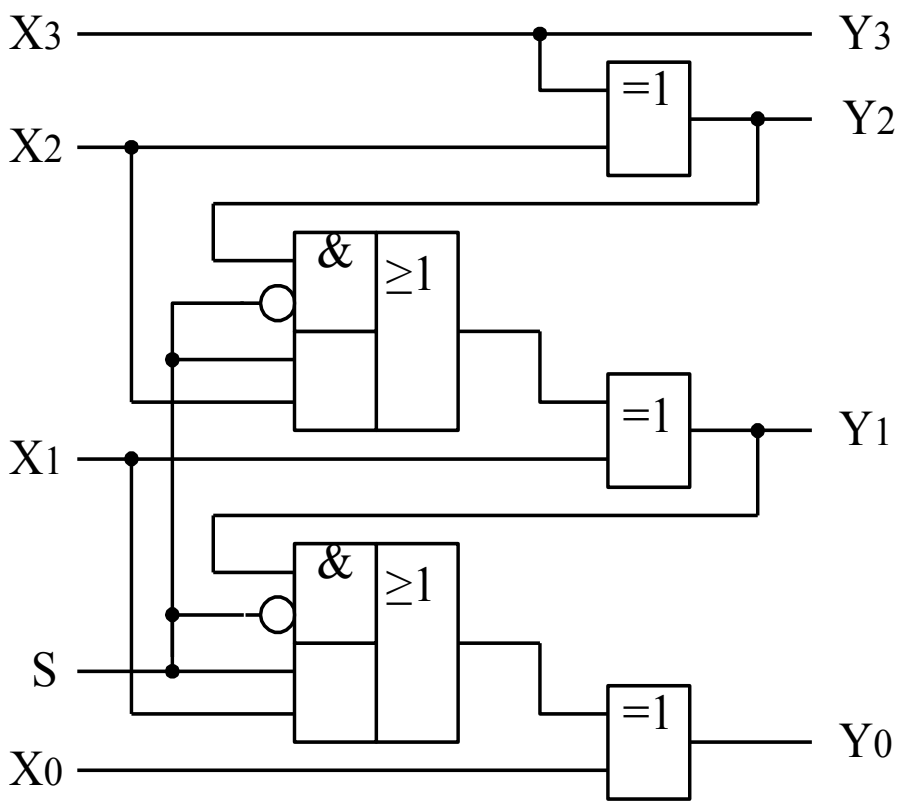
$$Y_3 = X_3, \quad Y_2 = X_3 \oplus X_2$$

$$Y_1 = X_2 \oplus X_1, \quad Y_0 = X_1 \oplus X_0$$

合成后的Y_1和Y_2

$$Y_1 = (\overline{S}Y_2 + SX_2) \oplus X_1$$

$$Y_0 = (\overline{S}Y_1 + SX_1) \oplus X_0$$



例3 一个应用例子

某特种录音机，具有下列功能：

按下A轨键，磁带正转；按下B轨键，磁带反转

按下高速键，磁带高速转，方向由A、B轨键确定

按下快退键，磁带高速反转，方向由A、B轨键确定

试设计控制电路

解：此问题的逻辑抽象为：

输入：A＝1、0表示 A 轨运行、停止

B=1、0表示 B 轨运行、停止

F＝1、0表示高速、常速

R＝1、0表示反转、正转

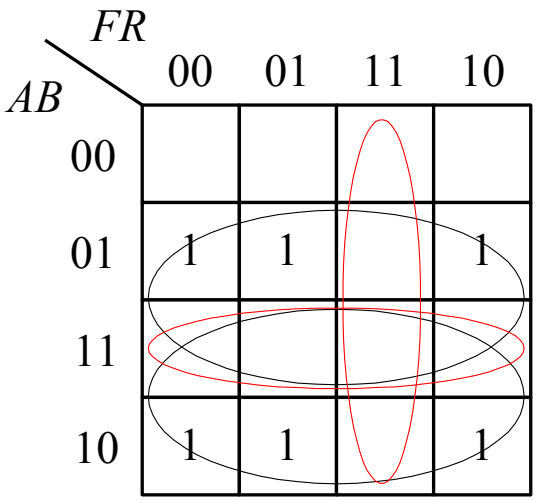
输出：M=1、0表示电机运转、停止

$RL1$=1、0表示电机反转、正转

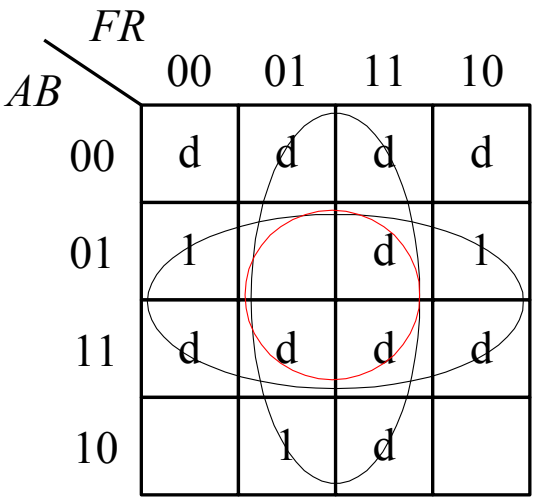
$RL2$=1、0表示电机高速、常速

真值表

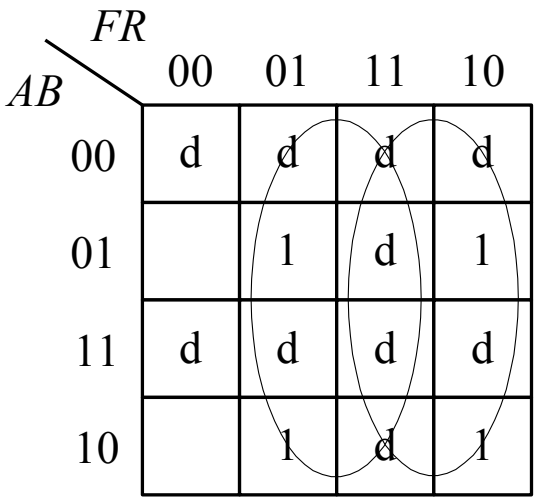
A A轨	*B* B轨	*F* 高速前进	*R* 高速后退	*M* 1/0 = 转/停	*RL*1 1/0 = 反/正	*RL*2 1/0 = 高/常
0	0	x	x	0	d	d
1	0	0	0	1	0	0
1	0	0	1	1	1	1
1	0	1	0	1	0	1
1	0	1	1	0	d	d
0	1	0	0	1	1	0
0	1	0	1	1	0	1
0	1	1	0	1	1	1
0	1	1	1	0	d	d
1	1	x	x	0	d	d



M



$RL1$



$RL2$

$$M = (A + B)\overline{(AB + FR)}$$

$$RL1 = (B + R)\overline{BR}$$

$$RL2 = F + R$$

以上只是一种方案，可能有其他方案

二、基于组合逻辑模块的设计方法

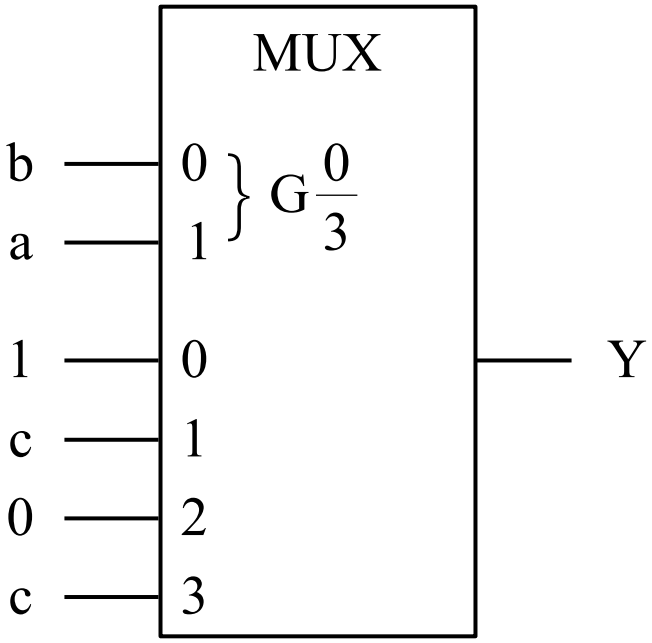
1、用数据选择器构成组合电路 (例2-9，p59)

$$Y(S_1,S_0,I_i)=\overline{S}_1\,\overline{S}_0\,I_0+\overline{S}_1\,S_0\,I_1+S_1\,\overline{S}_0\,I_2+S_1\,S_0\,I_3$$

$$=(\overline{a}\,\overline{b})1+(\overline{a}\,b)\,c+(a\,\overline{b})0+(a\,b)\,c$$

$$=\overline{a}\,\overline{b}+\overline{a}\,b\,c+a\,b\,c$$

给定逻辑函数，用数据选择器实现电路。



一般而言，用 2^n 选 1 数据选择器实现 n+1 个输入变量的逻辑函数需要且仅需要一个非门。例如 (2-10, p60)：

$$Z = \overline{C}_1\overline{C}_2(A \oplus B) + \overline{C}_1 C_2(\overline{A+B}) + C_1\overline{C}_2(\overline{AB}) + C_1 C_2(\overline{A \oplus B})$$

$$= \overline{C}_1\overline{C}_2\,\overline{A}B + \overline{C}_1\overline{C}_2 A\overline{B} + \overline{C}_1 C_2\,\overline{A}\,\overline{B} +$$

$$+ C_1\overline{C}_2\,\overline{A} + C_1\overline{C}_2\,\overline{B} + C_1 C_2 AB +$$

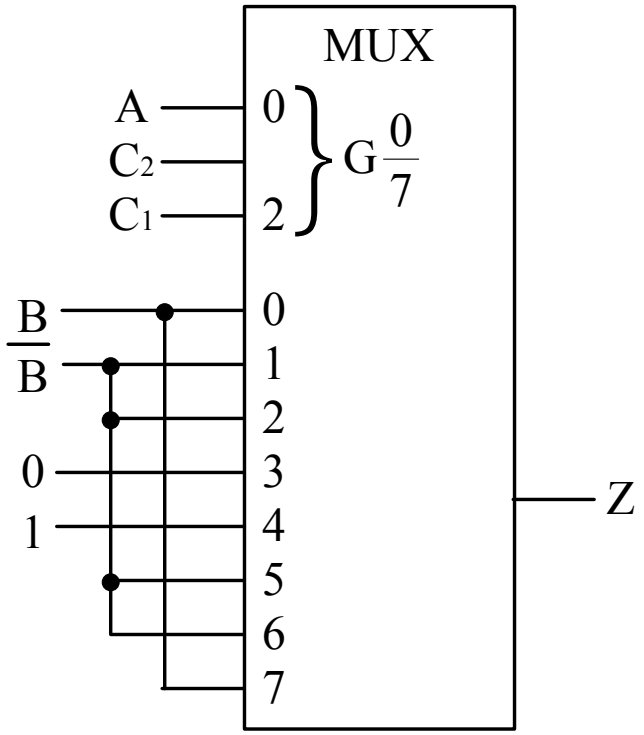
$$+ C_1 C_2\,\overline{A}\,\overline{B}$$

$$= (\overline{C}_1\overline{C}_2\,\overline{A})\cdot B + (\overline{C}_1\overline{C}_2 A)\cdot\overline{B} +$$

$$+ (\overline{C}_1 C_2\,\overline{A})\cdot\overline{B} + (\overline{C}_1 C_2 A)\cdot 0 +$$

$$+ (C_1\overline{C}_2\,\overline{A})\cdot 1 + (C_1\overline{C}_2 A)\cdot\overline{B} +$$

$$+ (C_1 C_2\,\overline{A})\cdot\overline{B} + (C_1 C_2 A)\cdot B$$



选B做数据端，则还需要一个非门。

特定条件下，用 2^n 选 1 数据选择器实现 $n+1$ 个输入变量的逻辑函数可以不需要非门。仍以前例说明如下：

$$Z = \overline{C}_1\overline{C}_2(A \oplus B) + \overline{C}_1 C_2(\overline{A+B}) + C_1\overline{C}_2(\overline{AB}) + C_1 C_2(\overline{A \oplus B})$$

$$= \overline{C}_1\overline{C}_2\,\overline{A}B + \overline{C}_1\overline{C}_2 A\overline{B} + \overline{C}_1 C_2\,\overline{A}\,\overline{B} +$$

$$+ C_1\overline{C}_2\,\overline{A} + C_1\overline{C}_2\,\overline{B} + C_1 C_2 AB +$$

$$+ C_1 C_2\,\overline{A}\,\overline{B}$$

$$= C_2\,\overline{A}\,\overline{B} + C_2\,\overline{A}\,\overline{B}\cdot C_1 + \overline{C}_2\,\overline{A}\,\overline{B}\cdot C_1 +$$

$$+ \overline{C}_2\,\overline{A}B + C_2 AB\cdot C_1 + \overline{C}_2 A\overline{B}$$

$$= \overline{C}_2\,\overline{A}\,\overline{B}\cdot C_1 + \overline{C}_2\,\overline{A}B\cdot 1 +$$

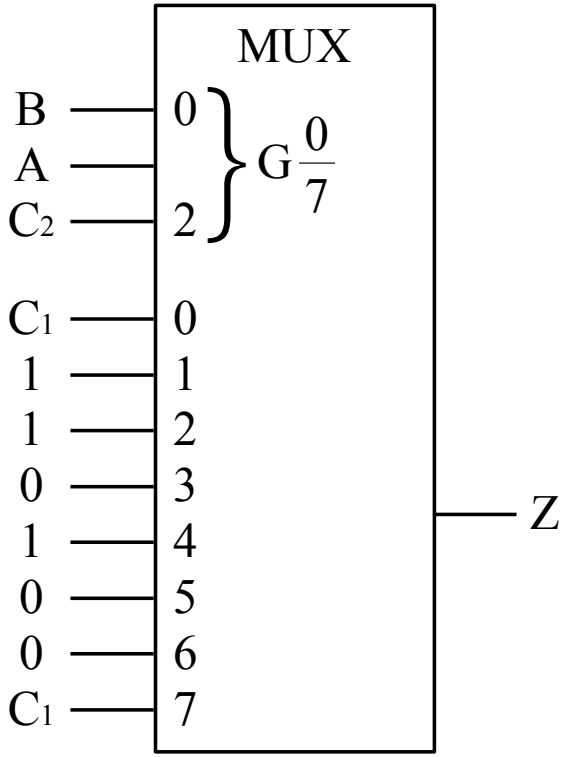
$$+ \overline{C}_2 A\overline{B}\cdot 1 + \overline{C}_2 AB\cdot 0 +$$

$$+ C_2\,\overline{A}\,\overline{B}\cdot 1 + C_2\,\overline{A}\,B\cdot 0 +$$

$$+ C_2 A\,\overline{B}\cdot 0 + C_2 AB\cdot C_1$$

化简

无C_1的“非”项，所以以C_1作为数据端

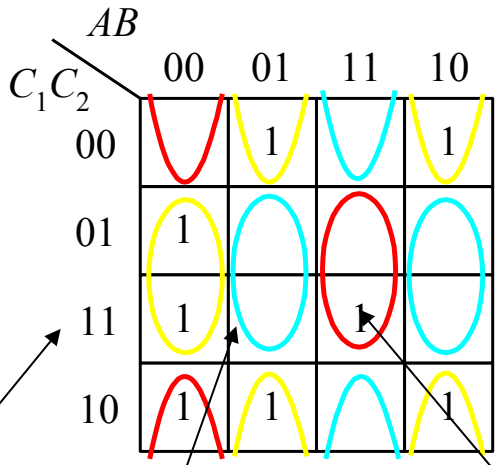


影射变量卡诺图

原来问题的卡诺图

C_1C_2 \ AB	00	01	11	10
00		1		1
01	1			
11	1		1	
10	1	1		1

设法合并一个变量的卡诺图



影射变量卡诺图

C_2 \ AB	00	01	11	10
0	C_1	1	0	1
1	1	0	C_1	0

卡诺圈内都是1

卡诺圈内的函数值与要合并的变量值相同

卡诺圈内都是0

结构对称

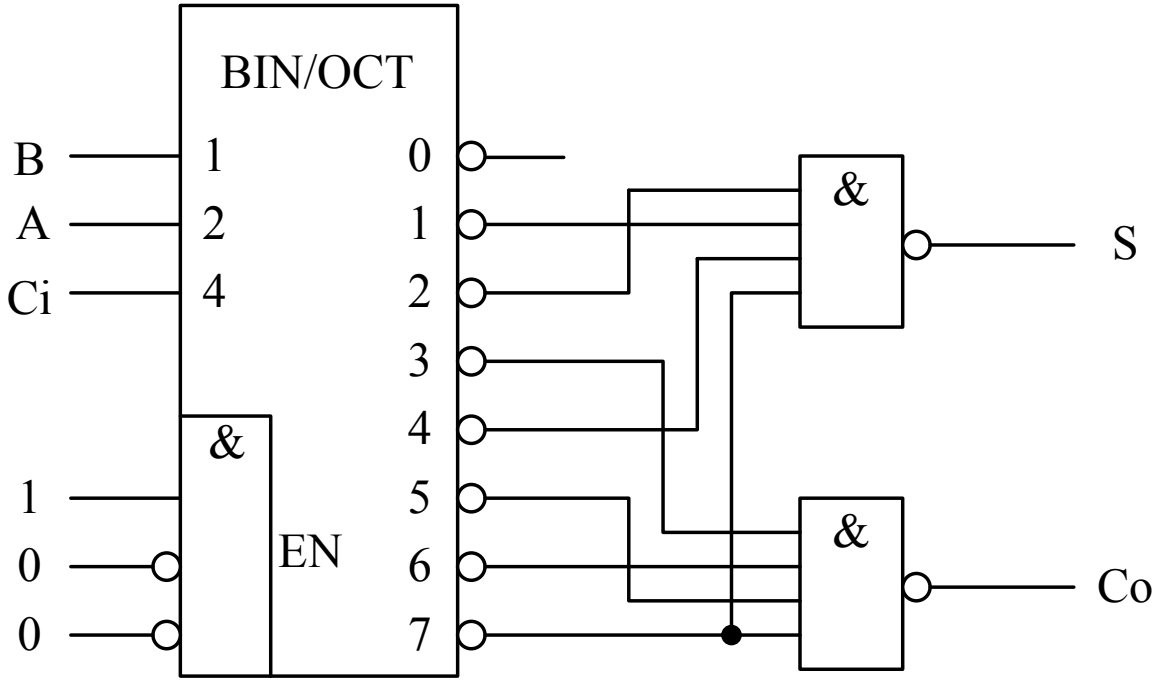
复习前面的影射变量。此处如果有C1反，则又达不到要求了。

2、用译码器构成组合电路

例：全加器电路 （2-11， p61）

$$S = m_1 + m_2 + m_4 + m_7 = \overline{\overline{Y}_1 \cdot \overline{Y}_2 \cdot \overline{Y}_4 \cdot \overline{Y}_7}$$

$$C_o = m_3 + m_5 + m_6 + m_7 = \overline{\overline{Y}_3 \cdot \overline{Y}_5 \cdot \overline{Y}_6 \cdot \overline{Y}_7}$$

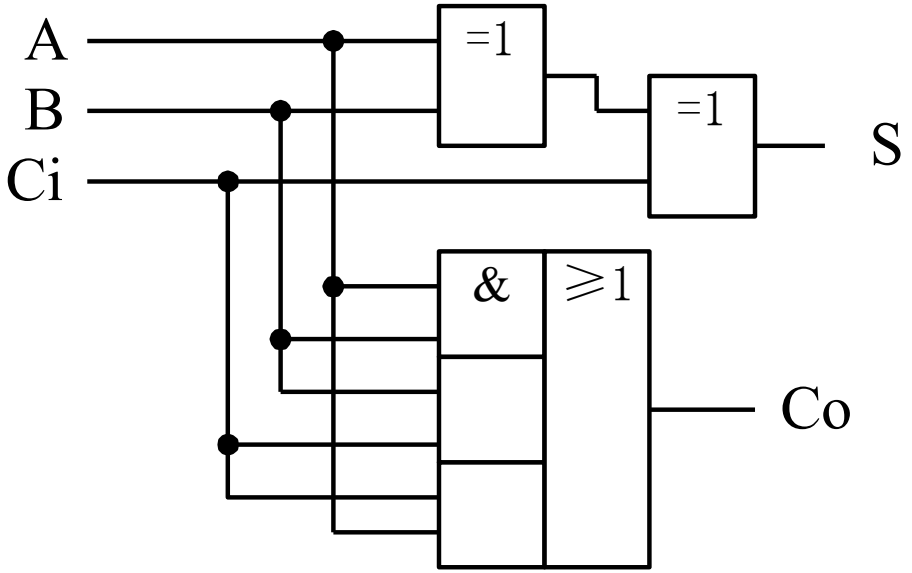


三、运算电路设计

1、加法器

具有最短延时的全加器电路(1位)

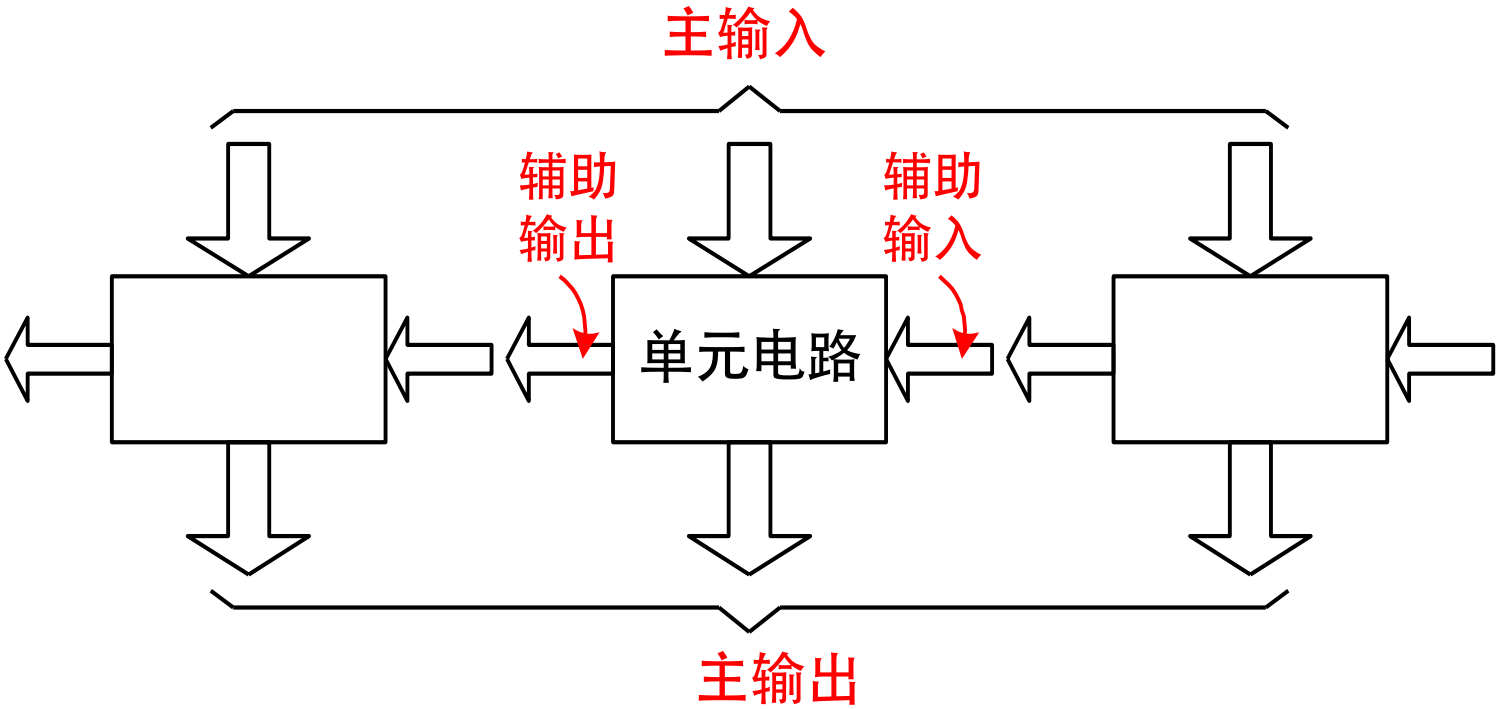
$$
\begin{array}{cccccc}
 & & A_3 & A_2 & A_1 & A_0 \\
 & & B_3 & B_2 & B_1 & B_0 \\
+ & C_3 & C_2 & C_1 & C_0 & \\
\hline
 & C_{o3} & S_3 & S_2 & S_1 & S_0
\end{array}
$$



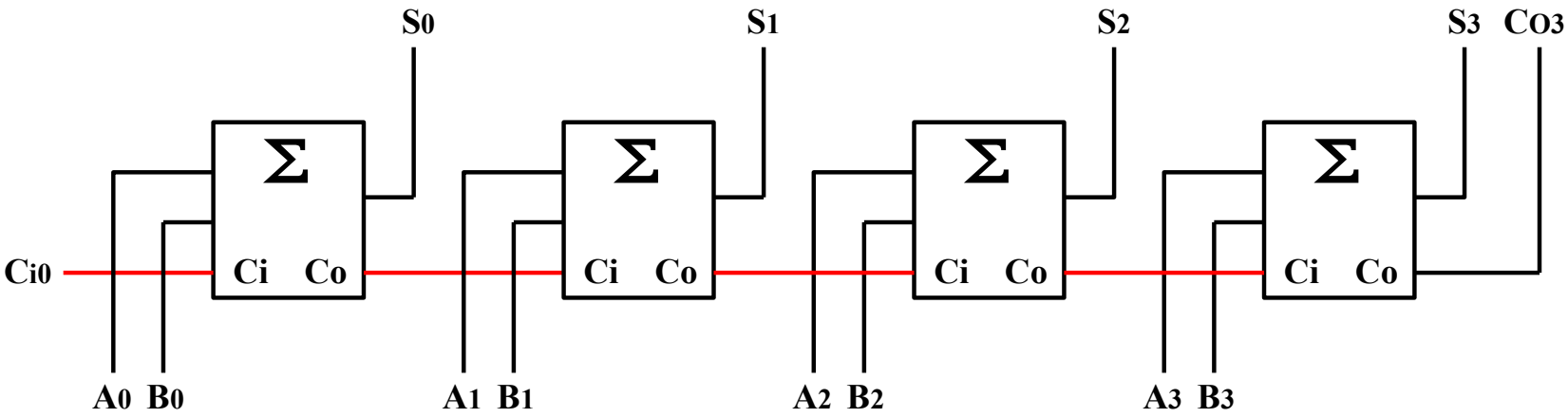
迭代设计原理

多个单元电路组合成一个系统时，为简化设计过程，可采用迭代设计

迭代设计法的实质是：若在一个设计中有许多重复的单元，设法找出它们串联的规律，然后设计其中一个单元，将这些单元按照需要进行串联，就构成了最终结果。



具有串行进位的4位二进制加法器



串行进位的加法器是采用迭代设计的具体例子

特点：1、只要设计一个单元，可以一直迭代下去，设计方便

2、由于后级运算要等待前级结果，运算速度慢

加法器的超前进位

$$C_i = A_iB_i + A_iC_{i-1} + B_iC_{i-1} = A_iB_i + (A_i + B_i)C_{i-1} = G_i + P_iC_{i-1}$$

其中$G_i = A_iB_i$，称为进位产生信号

$P_i = A_i + B_i$，称为进位传播信号

$$C_0 = G_0 + P_0C_{-1}$$

$$C_1 = G_1 + P_1C_0$$

$$= G_1 + P_1G_0 + P_1P_0C_{-1}$$

$$C_2 = G_2 + P_2C_1$$

$$= G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_{-1}$$

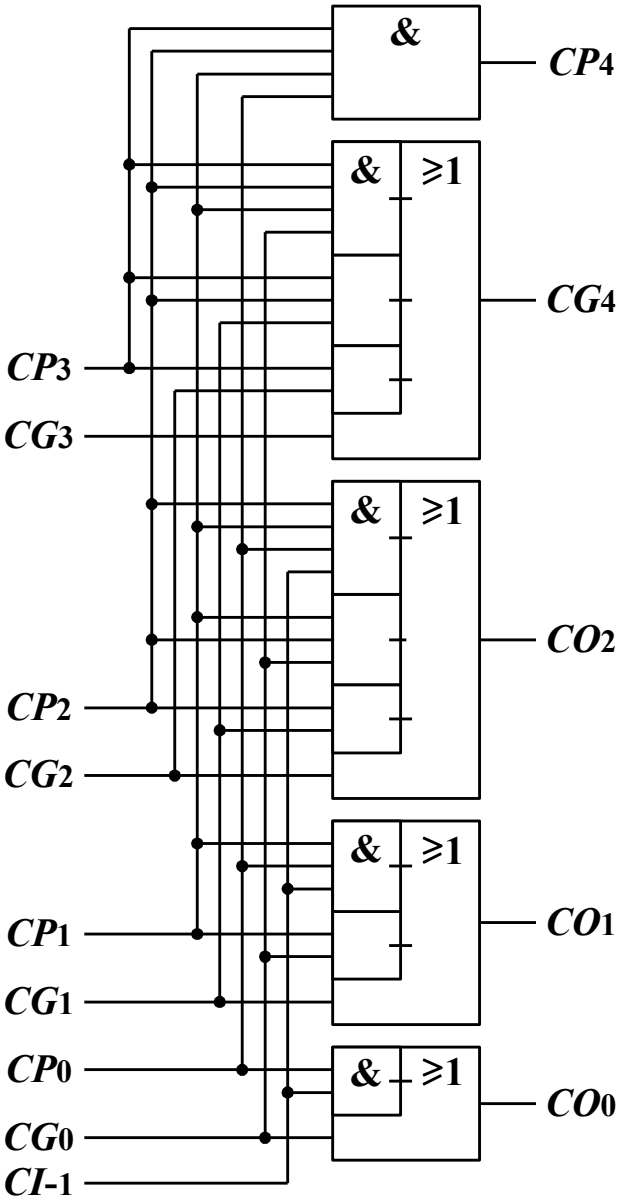
$$C_3 = G_3 + P_3C_2$$

$$= G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0 + P_3P_2P_1P_0C_{-1}$$

……

超前进位主要要解决的问题就是迭代设计中的速度问题

4位超前进位电路

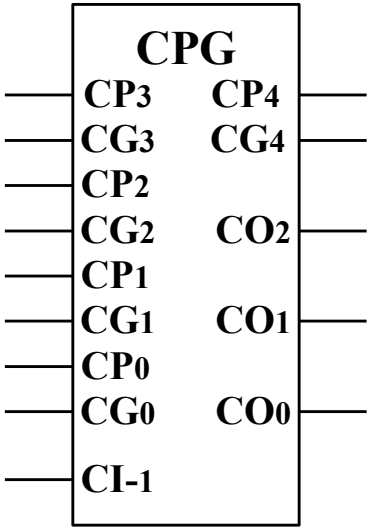


$$C_0 = G_0 + P_0 C_{-1}$$

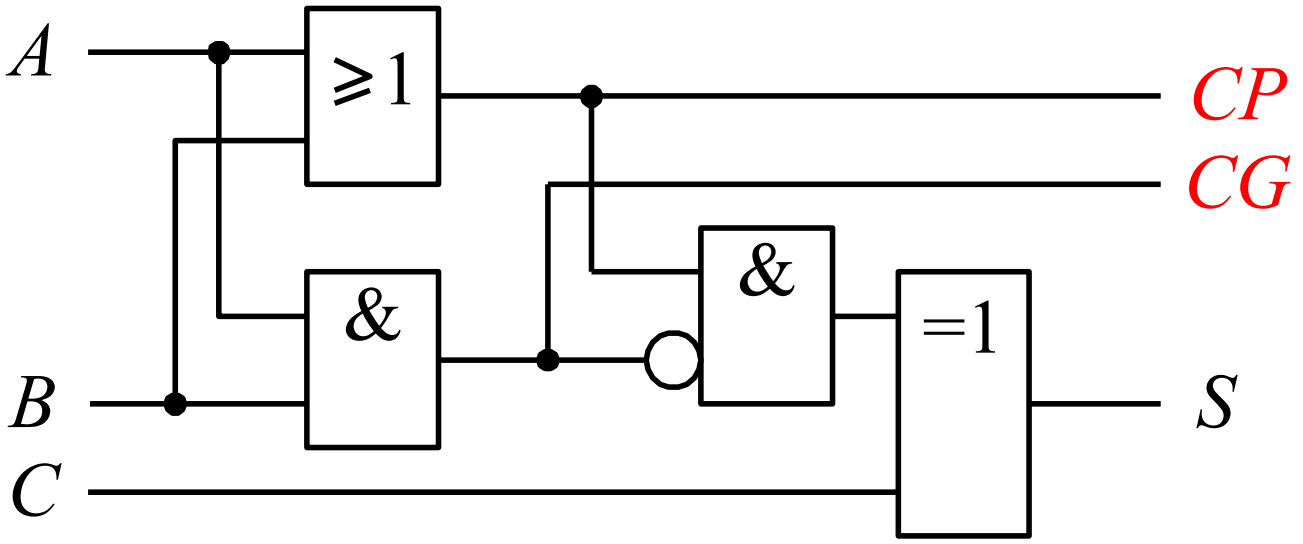
$$C_1 = G_1 + P_1 C_0$$
$$= G_1 + P_1 G_0 + P_1 P_0 C_{-1}$$

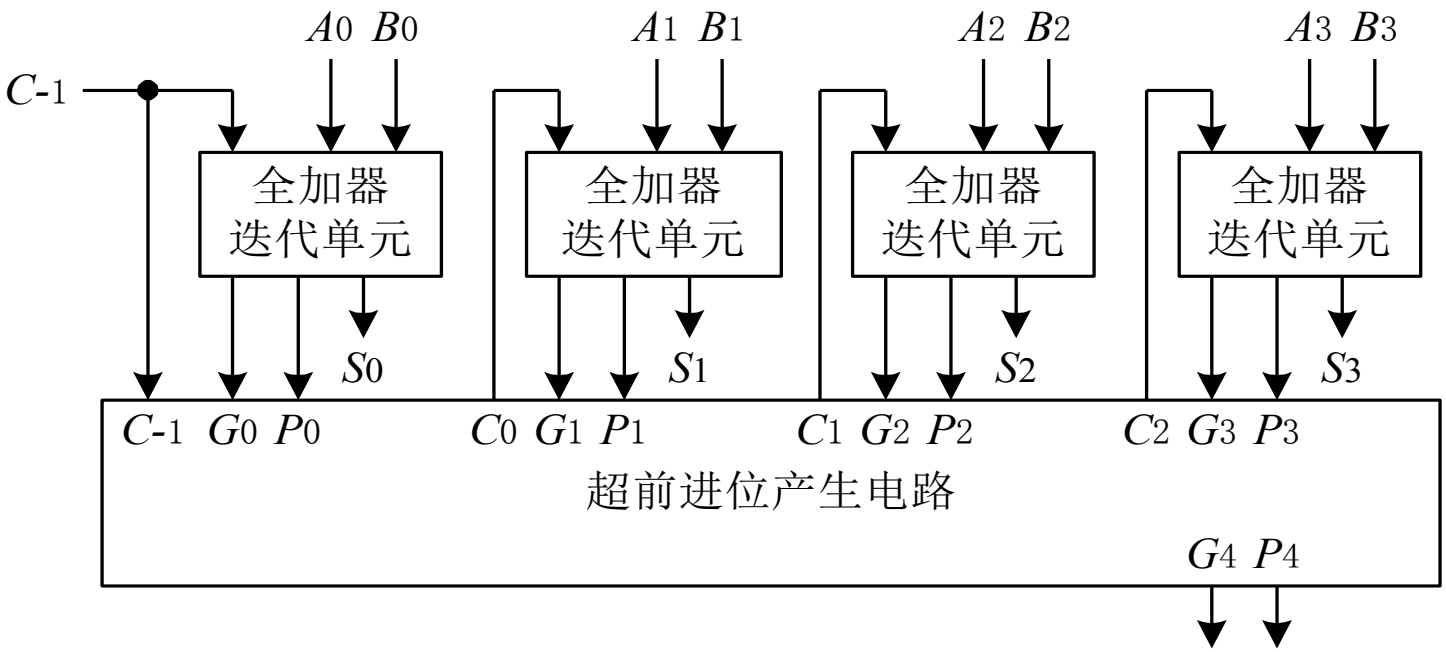
$$C_2 = G_2 + P_2 C_1$$
$$= G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_{-1}$$

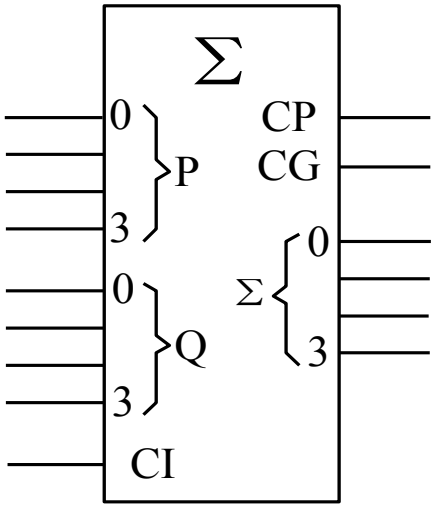
$$C_3 = G_3 + P_3 C_2$$
$$= G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_{-1}$$



配合超前进位电路的全加器迭代单元

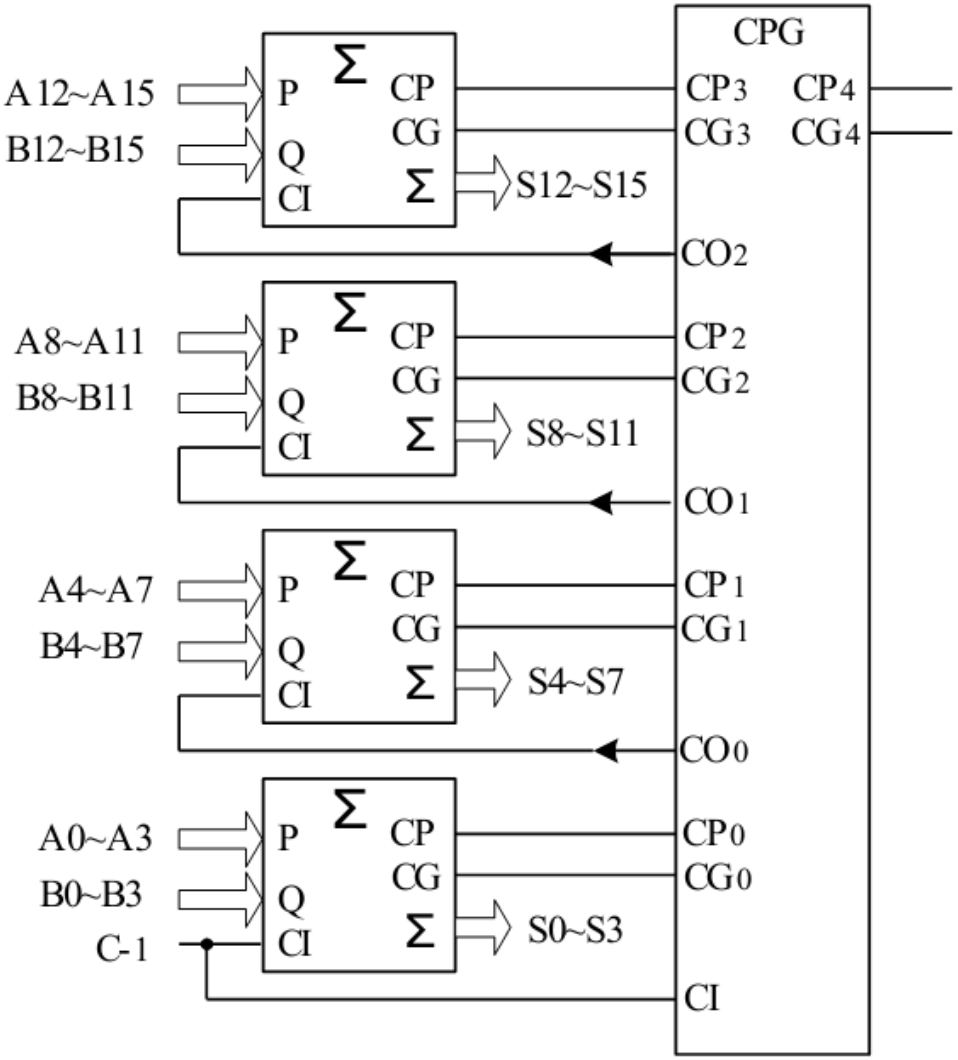






带超前进位的4位加法器

具有超前进位的16位加法器的迭代结构



利用加法器实现组合逻辑

例：设计一个能将BCD码转换为余3码的代码转换器

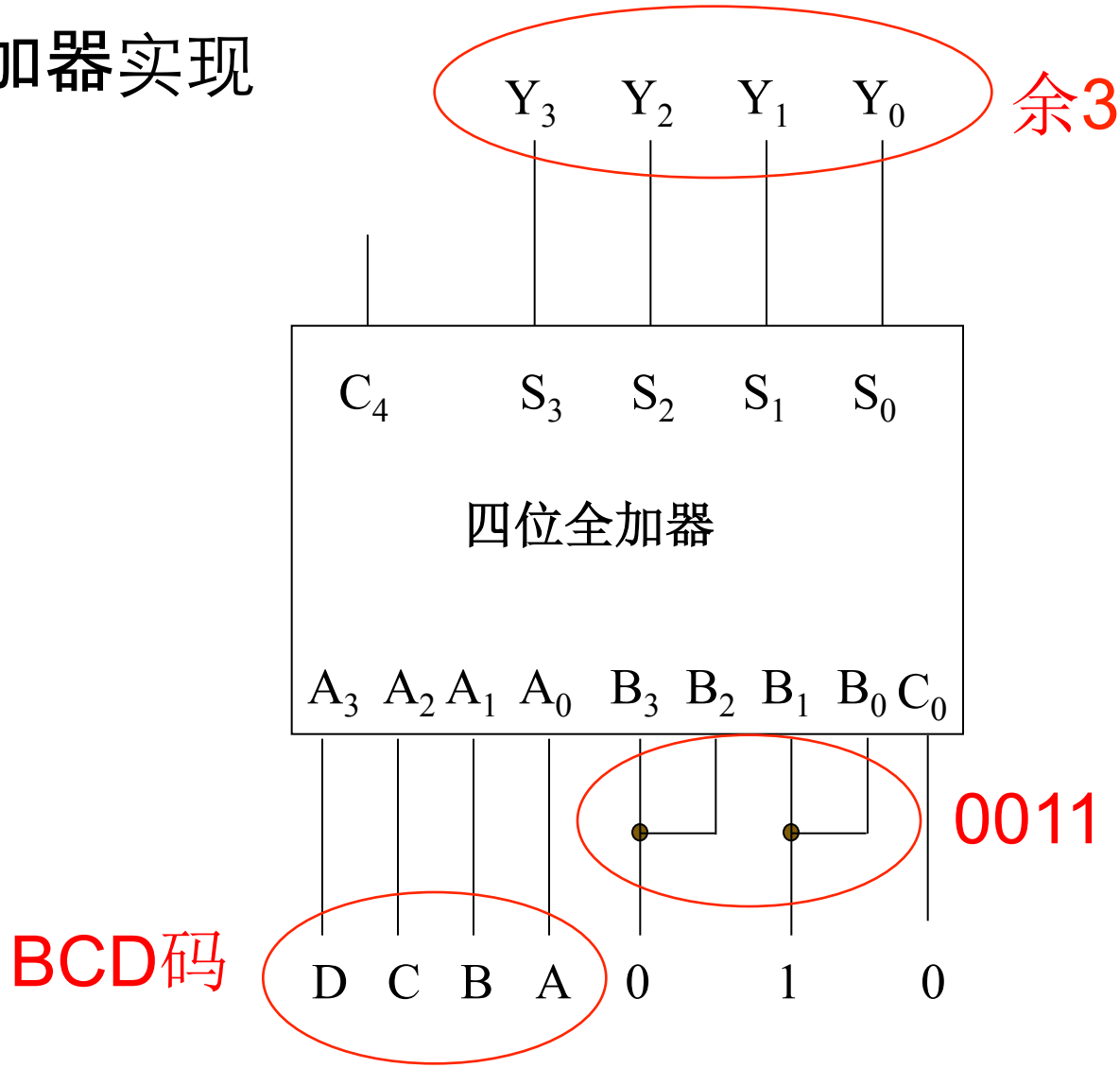
分析：

由余3码与BCD码的代码表可知，余3码的函数表达式为：

$$Y_3Y_2Y_1Y_0 = DCBA + 0011$$

8421BCD 码 $DCBA$	余 3 码 $Y_3Y_2Y_1Y_0$
0000	0011
0001	0100
0010	0101
0011	0110
0100	0111
0101	1000
0110	1001
0111	1010
1000	1011
1001	1100

利用4位全加器实现


Y_3 Y_2 Y_1 Y_0
余3码
C_4 S_3 S_2 S_1 S_0
四位全加器
A_3 A_2 A_1 A_0 B_3 B_2 B_1 B_0 C_0
0011
BCD码
D C B A 0 1 0

2、减法器

1位半减器 $(B_0, D) = X - Y$

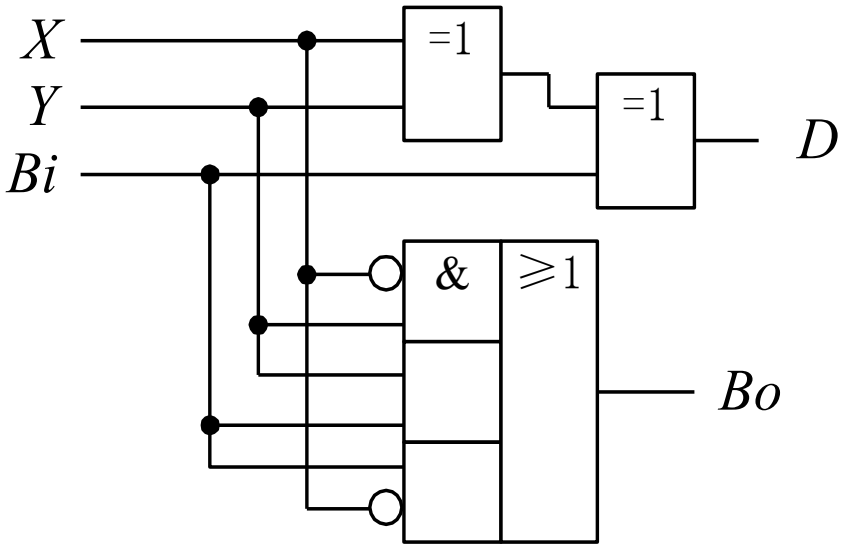
1位全减器 $(B_O, D) = X - Y - B_i$

Bi \ XY	00	01	11	10
0	0	1	0	1
1	1	0	1	0

D

Bi \ XY	00	01	11	10
0	0	1	0	0
1	1	1	1	0

Bo



列真值表，得卡诺图，化简后给出电路。

二进制补码

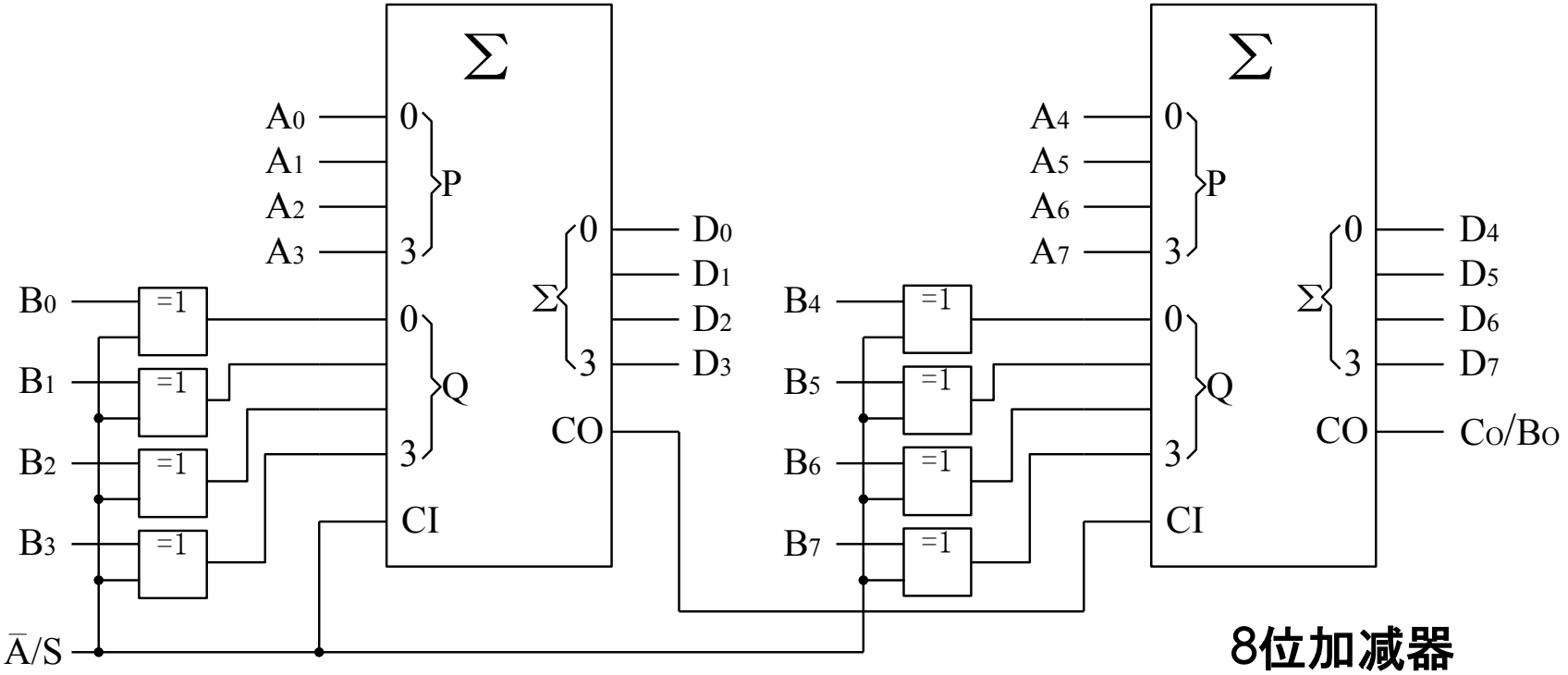
一个包含符号位在内为 n 位的有符号二进制数，正数用原码表示，负数用补码表示。

正数和零：$x = a$；负数：$x = 2^n - a$。其中a是该有符号数的绝对值。

补码的求法：绝对值按位取反再加 1

多位减法器

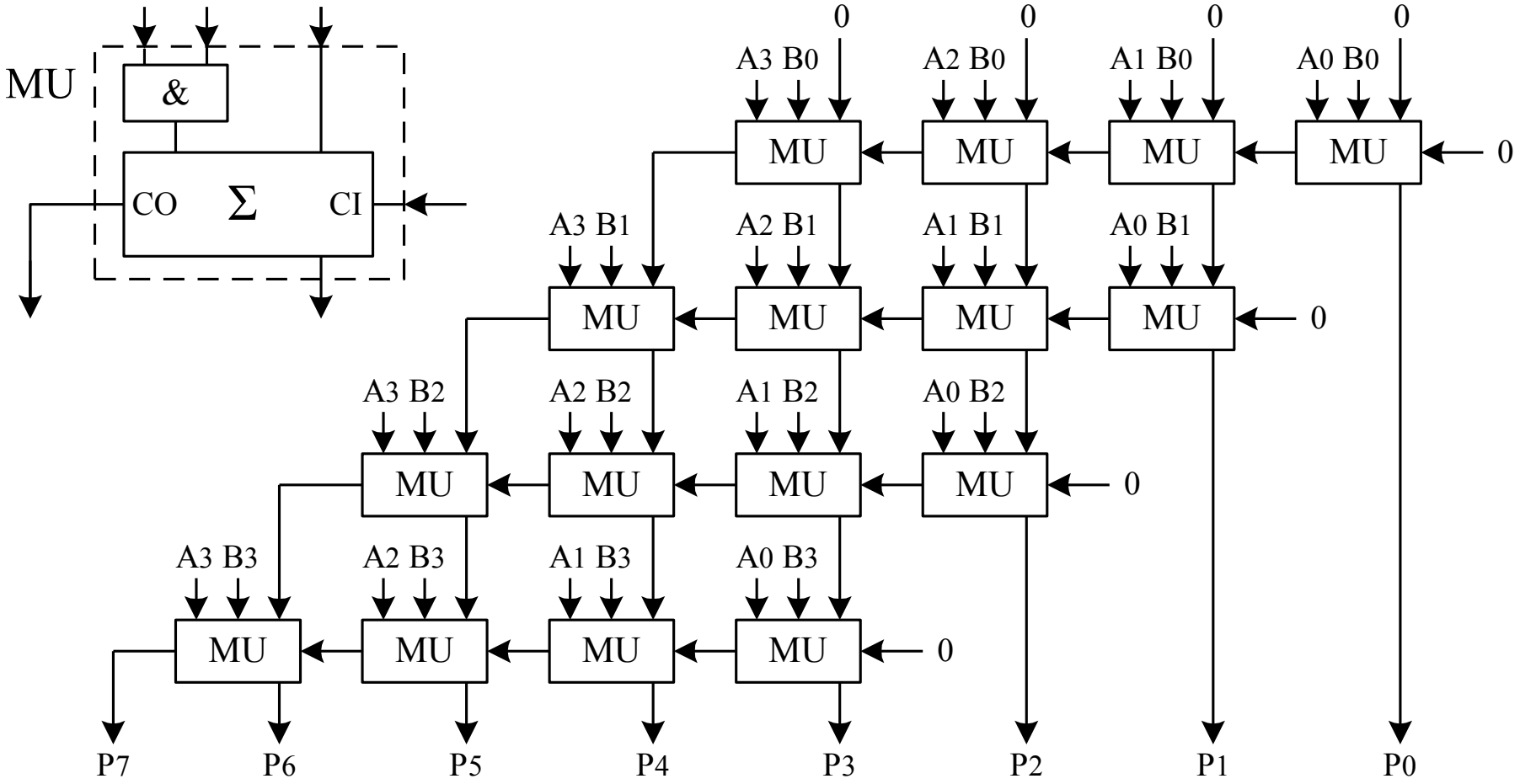
将被减数作为加法器的一个加数，将减数以补码形式相加（按位取反再加1：按位取“非”，同时将最低位的进位置成逻辑1），就实现了二进制减法。



8位加减器

3、乘法器

$$
\begin{array}{llllllll}
\text{乘数}A & & & & A3 & A2 & A1 & A0 \\
\text{乘数}B & & & & B3 & B2 & B1 & B0 \\
\hline
\text{部分积} & & & & & P30 & P20 & P10 \quad P00 \\
\text{部分积} & & & & P31 & P21 & P11 & P01 \\
\text{部分积} & & & P32 & P22 & P12 & P02 & \\
\text{部分积} & & P33 & P23 & P13 & P03 & & \\
\hline
\text{最后积} & P7 & P6 & P5 & P4 & P3 & P2 & P1 \quad P0
\end{array}
$$



此处：思考一下其它的乘法器组织方法？斜方向进位？

4、除法器

10101　商

除数B 0101 ）01101011　被除数A

- 0101

0011　够减，商=1，余数$R0 = A - B'$

- 0000

0110　不够减，商=0，余数$R1 = R0$

- 0101

0011　够减，商=1，余数$R2 = R0 - B'/4$

- 0000

0111　不够减，商=0，余数$R3 = R2$

- 0101

010　够减，商=1，余数$R4 = R2 - B'/16$

加减交替法：

第一步：试商 $$R_i = R_{i-1} - \frac{1}{2^i} B'$$

当$R_i \geq 0$时，够减，商等于1，将此余数保留到下一次。下一个余数为

$$R_{i+1} = R_i - \frac{1}{2^{i+1}} B'$$

当$R_i < 0$时，不够减，商等于0，应该将余数恢复为原来的余数。下一个余数为

$$R_{i+1} = R_{i-1} - \frac{1}{2^{i+1}} B'$$

由于$R_{i-1} = (R_i + \frac{1}{2^i} B')$，$\frac{1}{2^i} B' = 2 \times \frac{1}{2^{i+1}} B'$，所以上式就是

$$R_{i+1} = (R_i + \frac{1}{2^i} B') - \frac{1}{2^{i+1}} B' = R_i + \frac{1}{2^{i+1}} B'$$

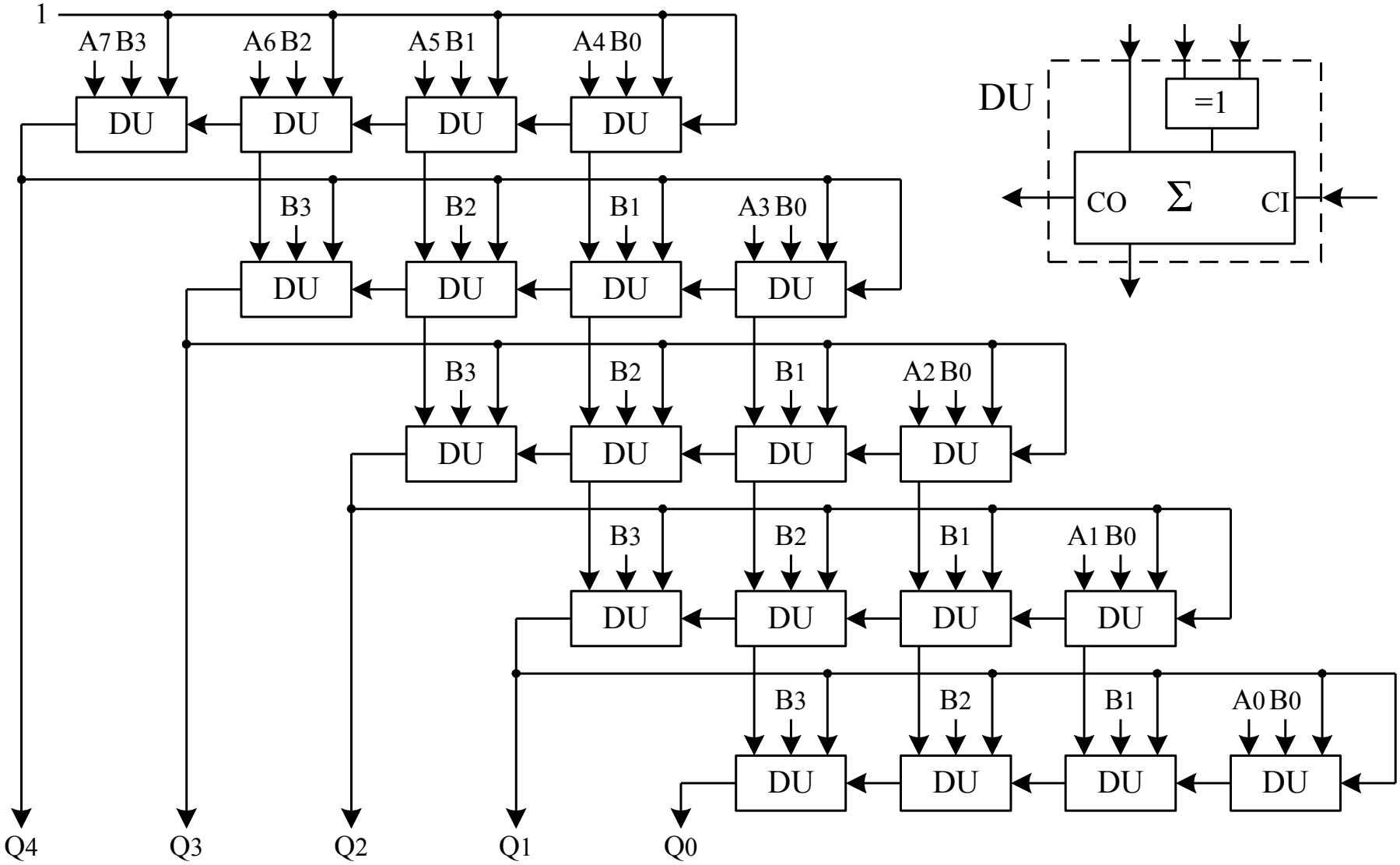
加减交替法的运算步骤

1. 第一次运算时从被除数的最高位开始减去除数，得到余数。
2. 若某次余数为正数（符号位为0），则对应的商为1，下一步运算时减去右移一位的除数得到新的余数；若某次余数为负数（符号位为1），则对应的商为0，下一步运算时加上右移一位的除数得到新的余数。
3. 重复第二步运算，直到余数小于除数为止。

运算规律：

用二进制补码运算来做减法。若结果是正数，则符号位为0，并且产生符号位的进位（进位为1）；若结果是负数，则符号位为1，并且不产生符号位的进位（进位为0）。所以，符号位的进位就是所求的商。

	10101	商
0101 ）	01101011	
	＋ 1011	加*B*的补码（减*B*）
	100011	符号位=0，符号位进位=1（商=1）
	＋ 1011	够减，加*B*/2的补码（减*B*/2）
	011100	符号位=1，符号位进位=0（商=0）
	＋ 0101	不够减，加*B*/4
	100011	符号位=0，符号位进位=1（商=1）
	＋ 1011	加*B*/8的补码（减*B*/8）
	011101	符号位=1，符号位进位=0（商=0）
	＋ 0101	不够减，加*B*/16
	10010	符号位=0，符号位进位=1（商=1）


1
A7 B3
A6 B2
A5 B1
A4 B0
DU
B3
B2
B1
A3 B0
A2 B0
A1 B0
A0 B0
Q4
Q3
Q2
Q1
Q0
=1
CO
Σ
CI

5、数字比较器

数据比较器有两组输入变量，它将输入的两组逻辑变量看成是两个二进制数A与B，然后对这两个二进制数进行数值比较。比较的结果有三种情况：$A>B$、$A<B$和$A=B$。

输 入		输 出		
A	B	$A>B$	$A=B$	$A<B$
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

1位数字比较器的真值表

比较器的迭代单元的真值表

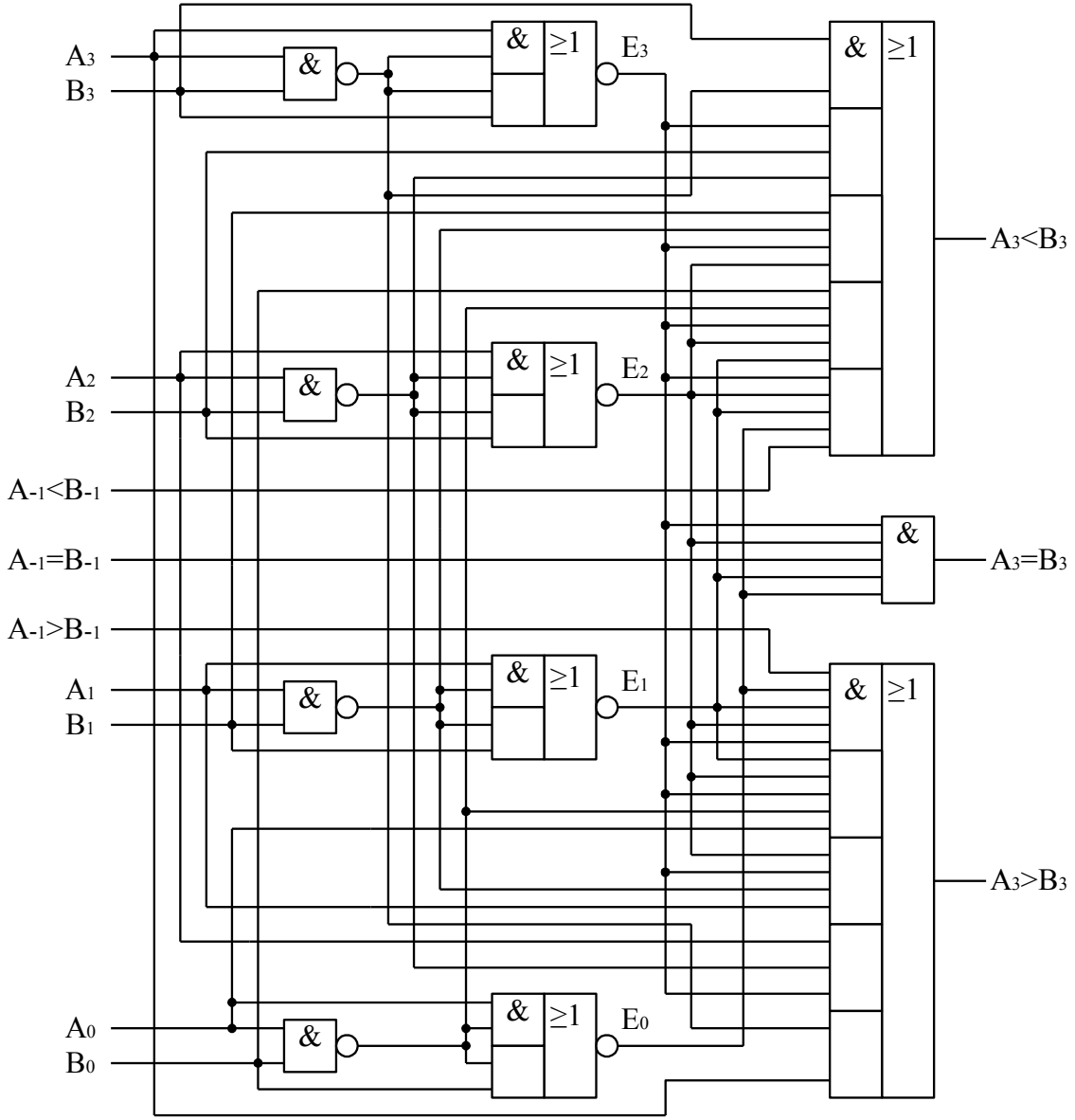
辅 助 输 入			输 入		辅 助 输 出		
$A_{i-1}>B_{i-1}$	$A_{i-1}=B_{i-1}$	$A_{i-1}<B_{i-1}$	A_i	B_i	$A_i>B_i$	$A_i=B_i$	$A_i<B_i$
0	1	0	0	0	0	1	0
1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	1
x	x	x	0	1	0	0	1
x	x	x	1	0	1	0	0
0	1	0	1	1	0	1	0
1	0	0	1	1	1	0	0
0	0	1	1	1	0	0	1

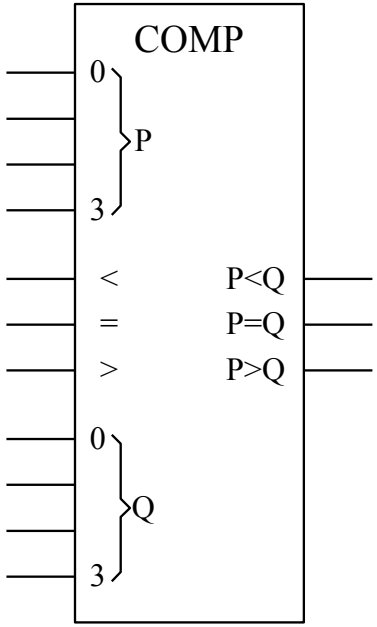
$$[A_i > B_i] = A_i \overline{B}_i + (\overline{A_i \oplus B_i}) \cdot [A_{i-1} > B_{i-1}]$$

$$[A_i = B_i] = (\overline{A_i \oplus B_i}) \cdot [A_{i-1} = B_{i-1}]$$

$$[A_i < B_i] = \overline{A}_i B_i + (\overline{A_i \oplus B_i}) \cdot [A_{i-1} < B_{i-1}]$$

1. 每个输出由两部分组成：本位比较结果和低位比较结果的进位。
2. 本位比较相等的条件为A、B的“同或”再“与”低位比较相等的结果。
3. 输出 $[A_i > B_i]$ 的条件有两个：第一个条件是本位结果满足$A_i > B_i$，另一个条件是本位的比较结果相等时，低位比较结果$A_i\text{-}1 > B_i\text{-}1$。这两个条件任意满足一个即可，所以是“或”关系。
4. 输出 $[A_i < B_i]$ 的结构与 $[A_i > B_i]$ 类似。


A3
B3
&
&
≥1
E3
A2
B2
&
&
≥1
E2
A-1<B-1
A-1=B-1
A-1>B-1
A1
B1
&
&
≥1
E1
A0
B0
&
&
≥1
E0
&
≥1
A3<B3
&
A3=B3
&
≥1
A3>B3


COMP
0
P
3
<
=
>
P<Q
P=Q
P>Q
0
Q
3

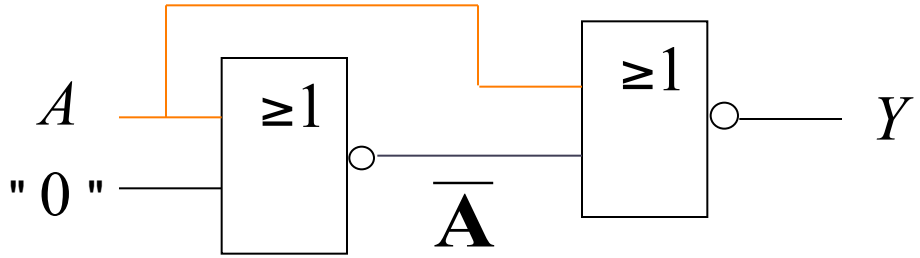
6、算术逻辑单元

算术逻辑单元（Arithmetic Logic Unit，简称ALU）是数字计算机中的一个核心运算部件。通常这个单元的输入被称为操作数，操作数可以是二进制数、十进制数或逻辑变量。进入ALU的操作数可以执行算术和逻辑运算。可执行的算术运算有两个操作数的加法（有进位和没有进位）、减法（有借位和没有借位）、单个操作数的加1、减1、以及数值比较等等；某些ALU还可以执行两个操作数的乘法、除法。可执行的逻辑运算一般均按位进行，有两个操作数的“与”、“或”、“与非”、“或非”、“异或”、“异或非”和单个操作数的“非”等等。

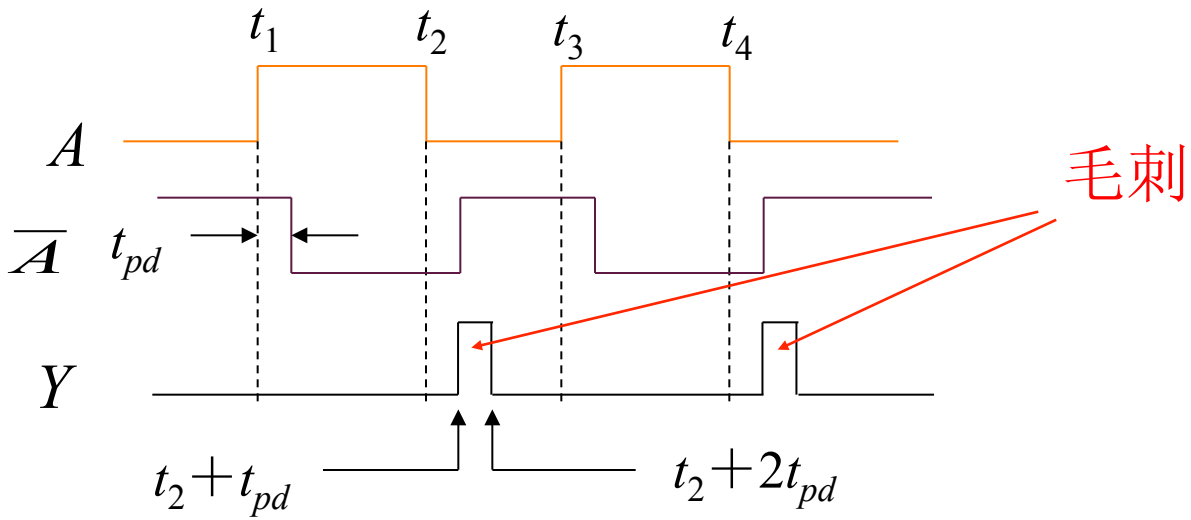
p75-77, 图2-35, 表2-10

2.4 组合逻辑电路中的竞争 – 冒险

两级或非门电路



波形图



竞争，冒险

当一个门的输入有两个或两个以上变量发生改变时，由于这些变量（信号）是经过不同路径产生的，使得它们状态改变的时刻有先有后，这种时差引起的现象称为竞争。

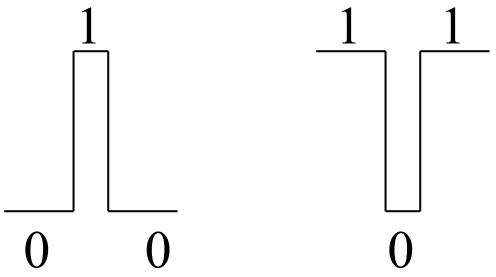
竞争的结果若导致冒险（险象）发生（如上例中的毛刺），并造成错误的后果，则称这种竞争为临界竞争；竞争的结果不导致冒险发生，或虽有冒险发生，但不影响系统的工作，则称这种竞争为非临界竞争。

冒险的类型

从冒险的波形上，可分为静态和动态冒险。

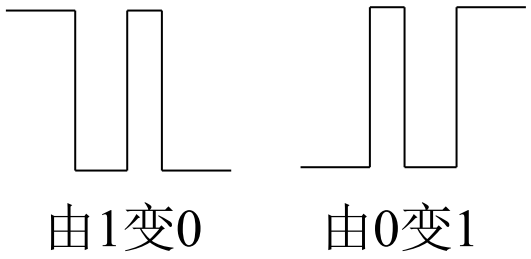
输入信号变化前后，输出的稳态值是一样的，但在输入信号变化时，输出产生了毛刺，这种冒险称为静态冒险。若输出的稳态值为0，出现了正的尖脉冲毛刺，则称为静态0冒险；若输出稳态值为1，出现了负的尖脉冲毛刺，则称为静态1冒险。

输入信号变化前后，输出的稳态值不同，并在边沿处出现了毛刺，称为动态冒险。



静态0冒险

静态1冒险



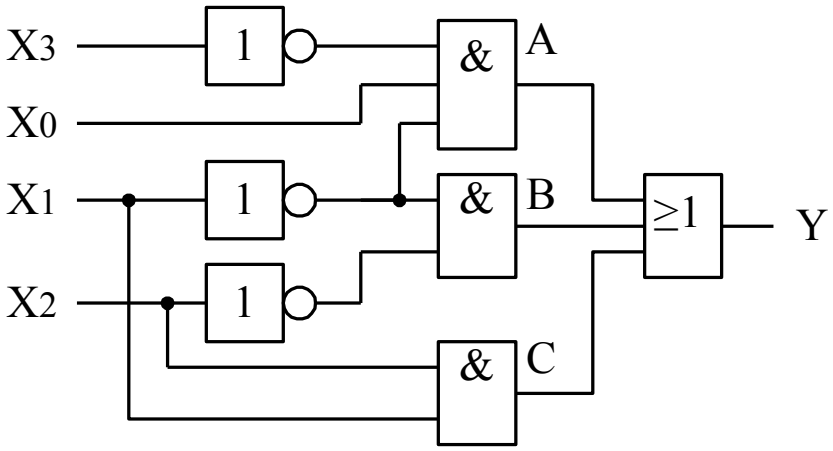
动态冒险

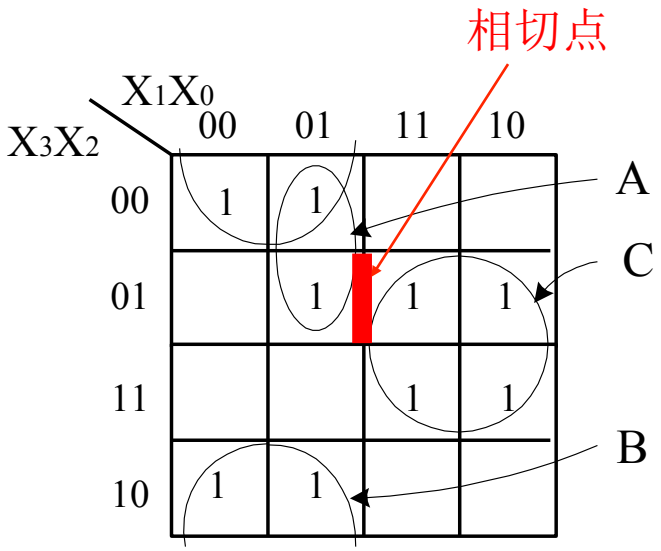
检查竞争－冒险的方法

1、输入可以转换成 $Y = A + \overline{A}$ 或 $Y = A \cdot \overline{A}$ 的形式

2、在卡诺图上可以观察到相切的卡诺圈

以上方法只有在每个瞬时只有一个输入发生状态改变的条件下才适用。





消除竞争－冒险的方法

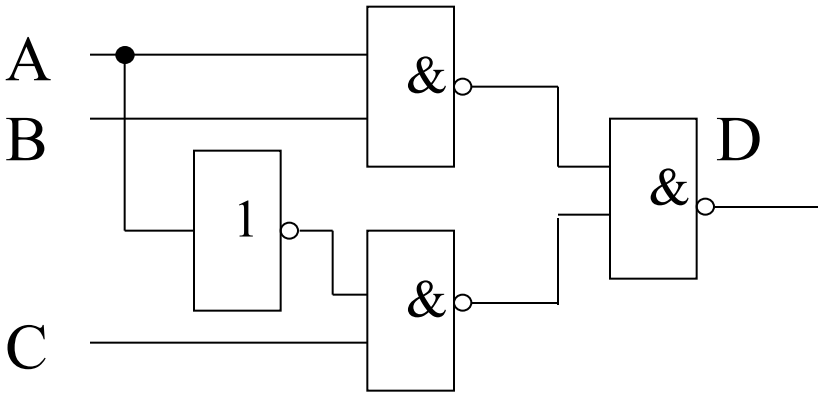
1、在输出端对地接入一个小电容。优点是简单易行，而缺点是增加了输出电压波形的上升时间和下降时间，使波形变坏，并且完全无法在集成电路内部实现。

2、修改逻辑设计。增加冗余项可以消除竞争-冒险，但是适用范围仍然很有限，只能消除由于单个输入发生变化引起的竞争－冒险。

3、在电路中引入选通脉冲 。可以消除所有的冒险（包括静态冒险和动态冒险），并且容易实现，但需注意：这时正常的输出信号也将变成脉冲信号，而且它们的宽度与选通脉冲相同。

增加冗余项消除冒险

有险象的电路

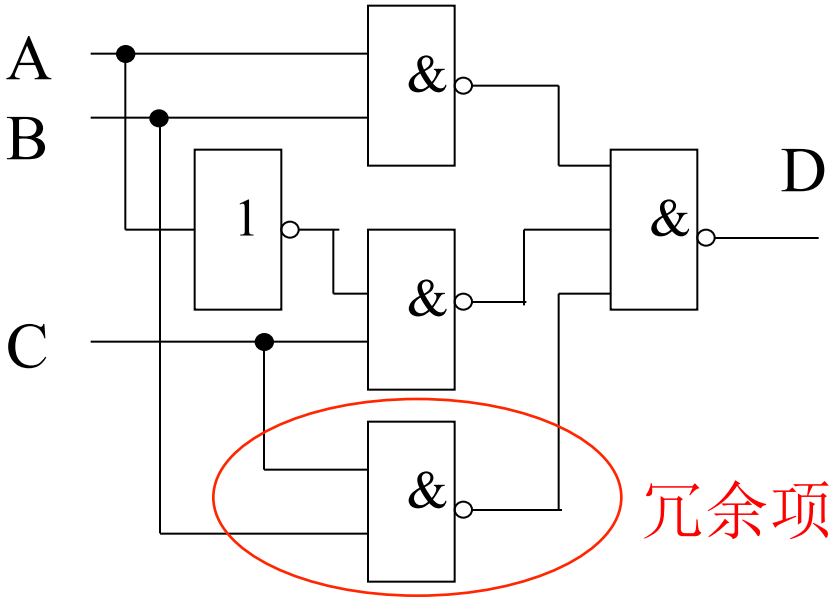


$$D=AB+\overline{A}C$$

当B=C=1时

$$D=A+\overline{A}$$

无险象的电路



$$D=AB+\overline{A}C+BC$$

当B=C=1时

$$D=A+\overline{A}+1=1$$

卡诺图法增加冗余项消除冒险

有相切的卡诺图

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

$$D = AB + \overline{A}C$$

相切点

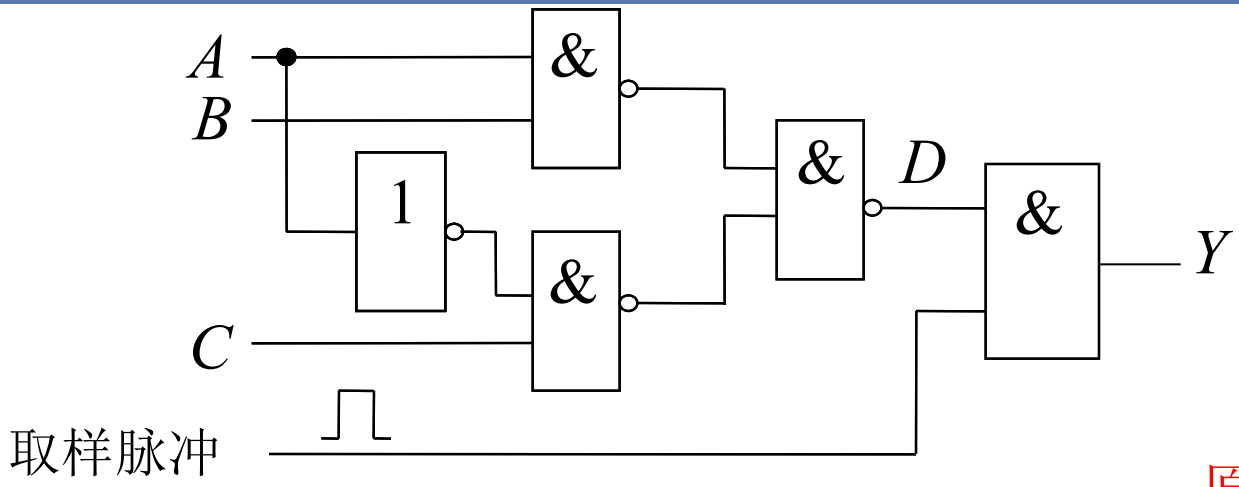
加冗余项的卡诺图

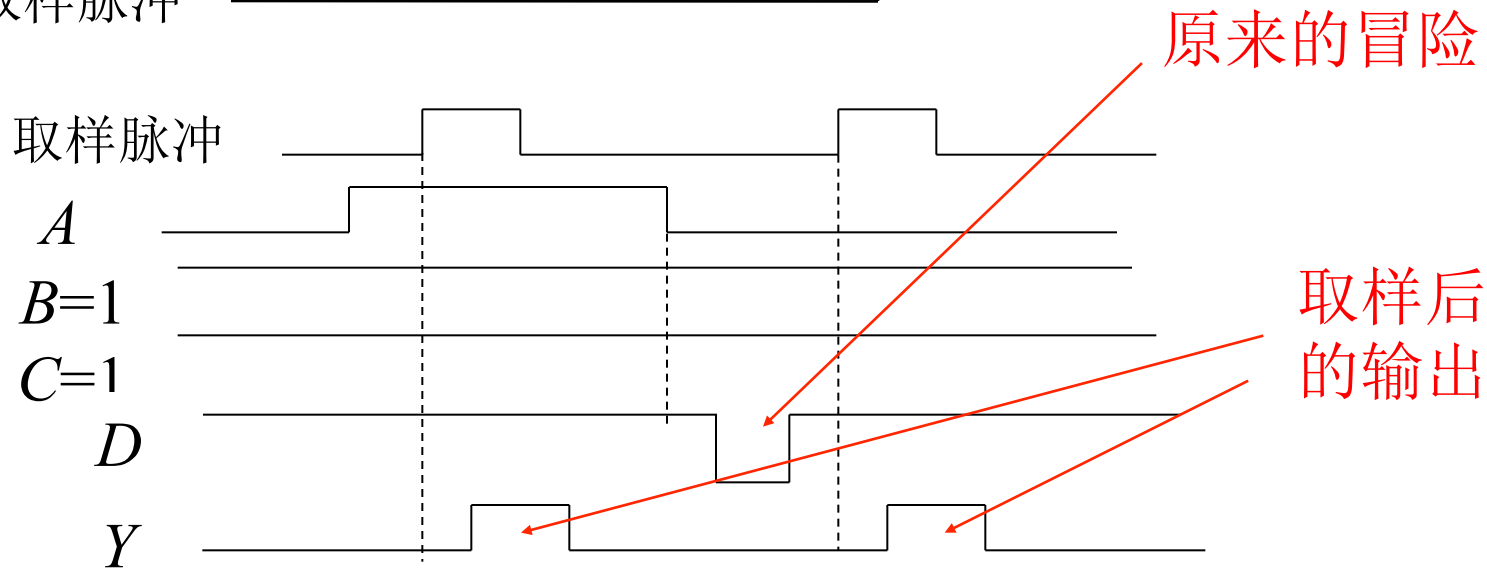
BC / A	00	01	11	10
0	0	1	1	0
1	0	0	1	1

$$D = AB + \overline{A}C + BC$$

相切点被消除

利用取样脉冲克服冒险





VIP

Video Image Processing

Research Group @ Fudan

http://soc.fudan.edu.cn/vip/

Thank you !

Acknowledgement:

Part of contents are referenced from Yufeng Xie, Ting Yi PPT