计算机组成原理与系统结构

第四章

运算方法与运算器

http://jpkc.hdu.edu.cn/computer/zcyl/dzkjdx/

第 4 章 运算方法与运算器

4.2 定点数的乘法运算及实现

一 原码乘法及实现

二 补码乘法及实现

三 阵列乘法器

串行乘法算法

一、原码乘法及实现

1

手工乘法算法

2

原码一位乘法算法

3

原码乘法的硬件实现

1、手工乘法算法

- 手工计算 1011×1101，步骤：
- 手工算法：
  - 对应每一位乘数求得 1 项位积
  - 将位积逐位左移
  - 将所有的位积一次相加，得到最后的乘积
- 在计算机上能否实现？如何实现？

$$
\begin{array}{r}
1011 \\
\times\ 1101 \\
\hline
1011 \\
0000 \\
1011 \\
1011 \\
\hline
10001111
\end{array}
$$

改造手工算法适合计算机硬件实现：

手工算法	机器算法
将所有位积一次相加	位积逐次累加
	部分积
位积左移	部分积右移

即：累加，右移，构成循环

2、原码一位乘法算法：累加、右移

- 假设 $[X]_{原}=X_S\ X_1\ X_2\ \cdots\cdots X_n$，$[Y]_{原}=Y_S\ Y_1\ Y_2\ \cdots\cdots Y_n$，$P=X\cdot Y$，$P_S$ 是积的符号：

① 符号位单独处理　$Ps=Xs \oplus Ys$

② 绝对值进行数值运算　$|P|=|X|*|Y|$

③ 初始部分积为 0，Yi=1，部分积加 |X|，Yi=0，部分积加 0，累加结果右移一位，得新部分积。

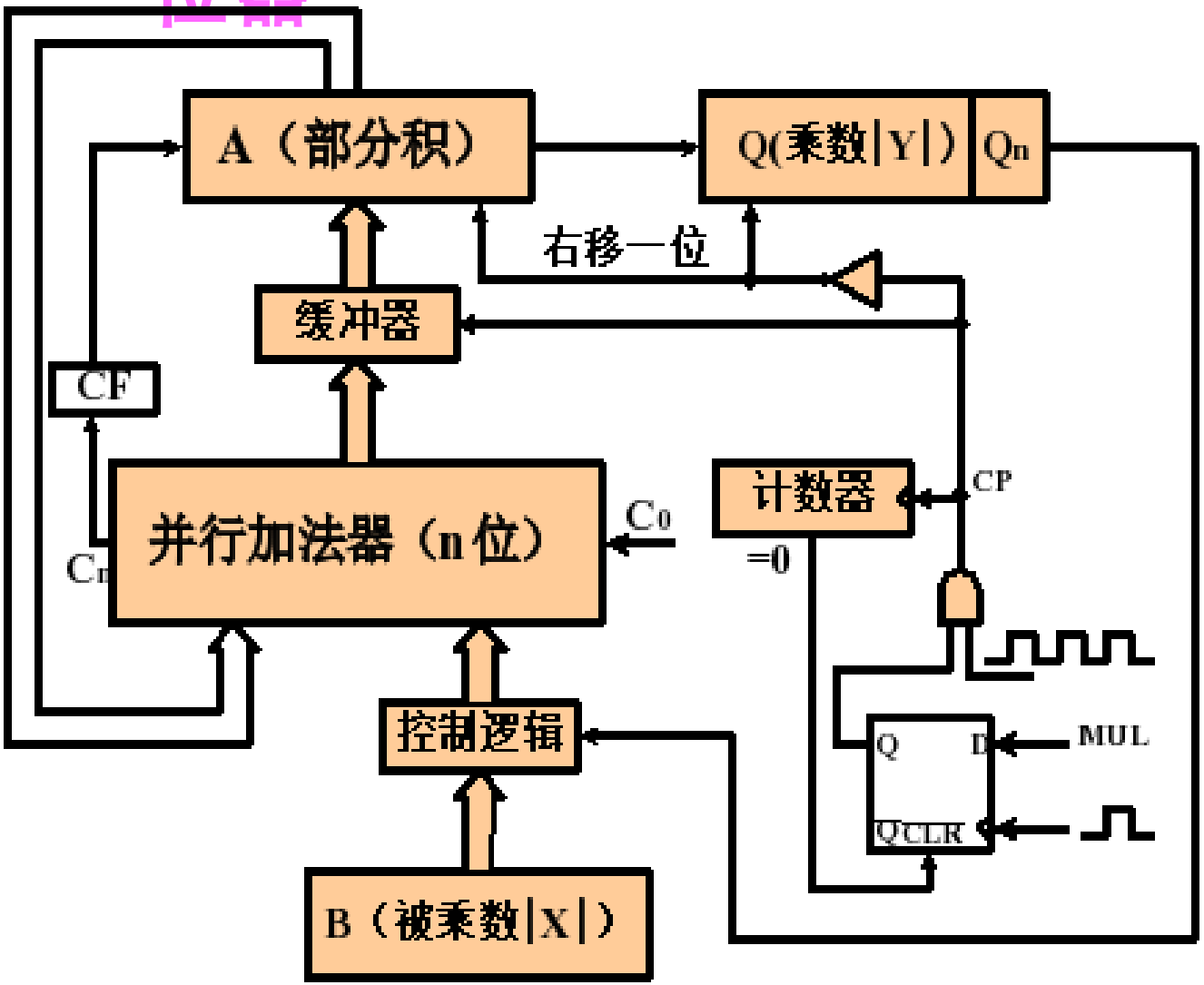
④ 累加右移 n 次，即 i=n，n−1，…，2，1

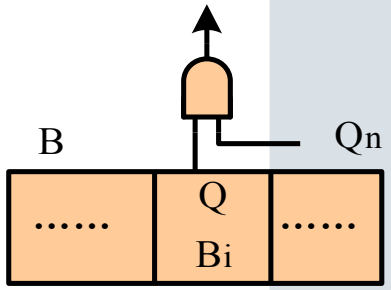
举例

- 例如： X=+1011 ， Y=-1101 ，用原码一位乘法计算 P=X · Y 。
- $[X]_{原}$ =0 ， 1011
- $[Y]_{原}$ =1 ， 1101
- Ps=Xs⊕Ys
  =0⊕1=1
  $[P]_{原}$ =1 ， 10001111
- |P| = |X| · |Y|

部分积	乘数Y	操作说明
0，0000	1 1 0 1	
+ 0，1011		Y_4=1，+|X|
0，1011		
0，0101	1 1 1 0	右移一位
+ 0，0000		Y_3=0，+0
0，0101		
0，0010	1 1 1 1	右移一位
+ 0，1011		Y_2=1，+|X|
0，1101		
0，0110	1 1 1 1	右移一位
+ 0，1011		Y_1=1，+|X|
1，0001		
0，1000	1 1 1 1	右移一位

3、原码乘法的硬件实现：加法器、移位器





控制逻辑电路

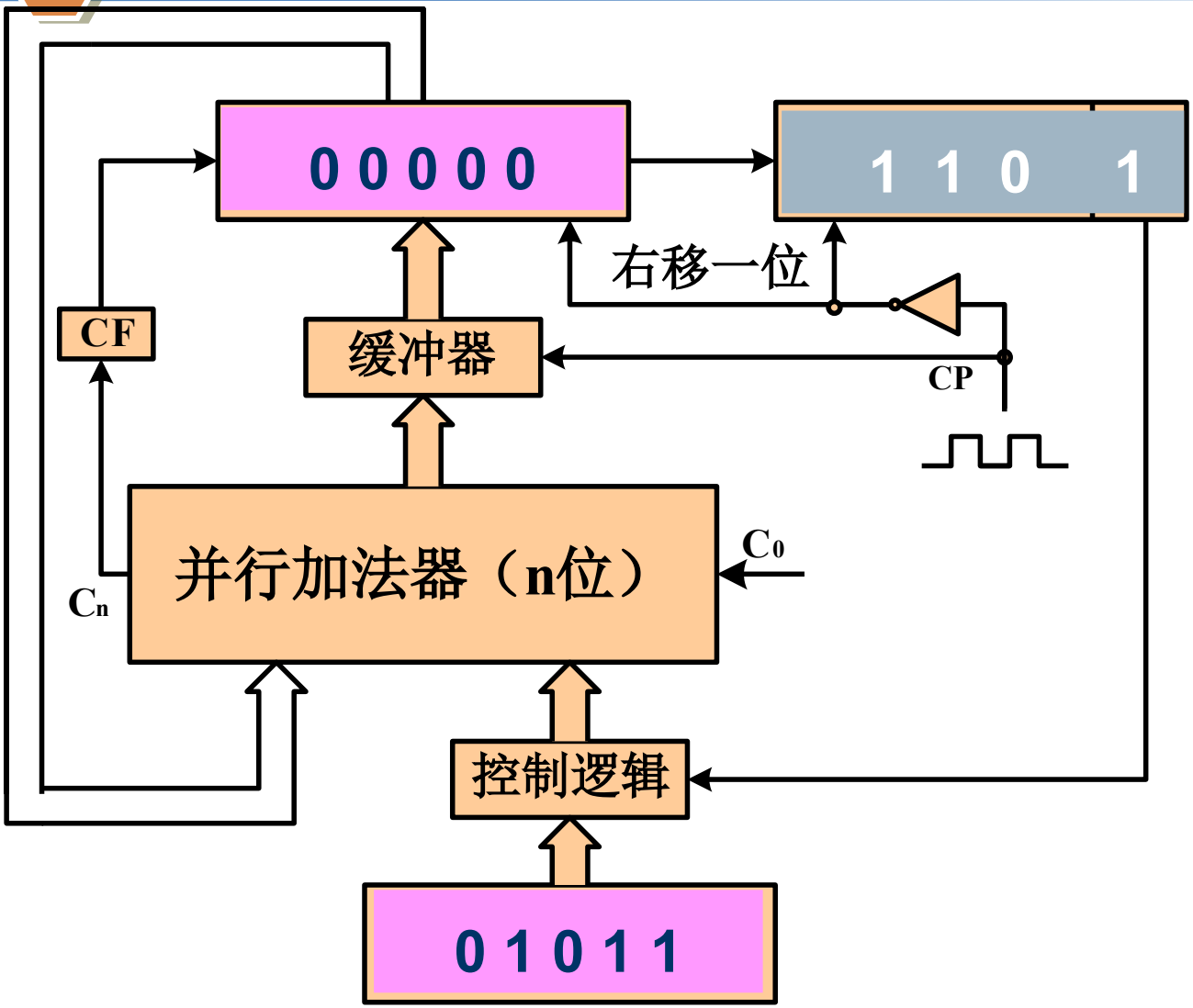
3、原码乘法的硬件实现

- A：累加寄存器
- B：被乘数寄存器
- Q：乘数寄存器
- A、Q：右移寄存器

初始：
- A=0
- B=|X|
- Q=|Y|
- 计数器 =n

累加、右移 n 次

结果：
- A= 乘积高位
- B =|X|
- Q= 乘积低位
- 计数器 =0

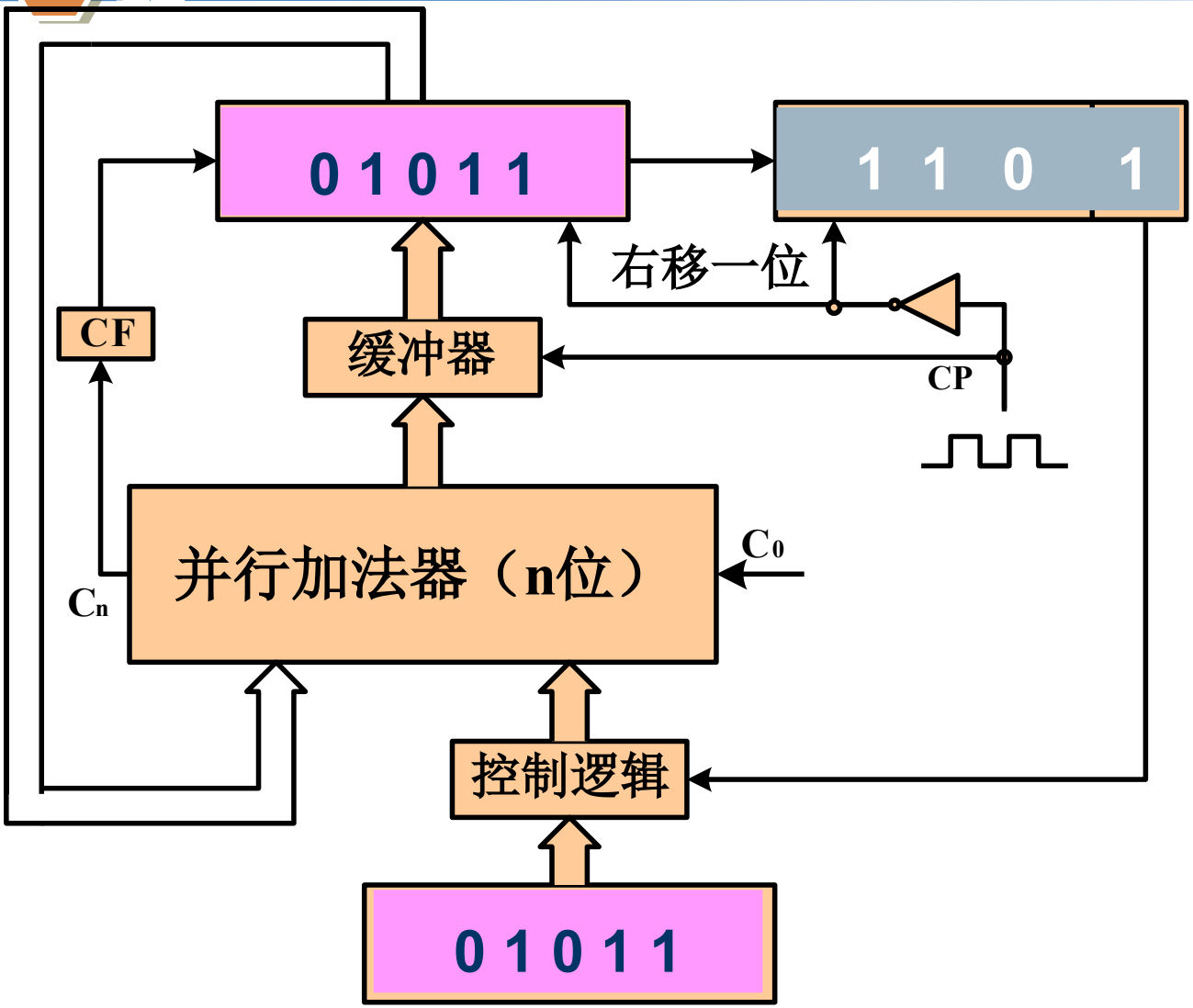

0 0 0 0 0
1 1 0 1
右移一位
CF
缓冲器
CP
并行加法器（n位）
C0
Cn
控制逻辑
0 1 0 1 1

第一次求部分

加运算：＋ |X|

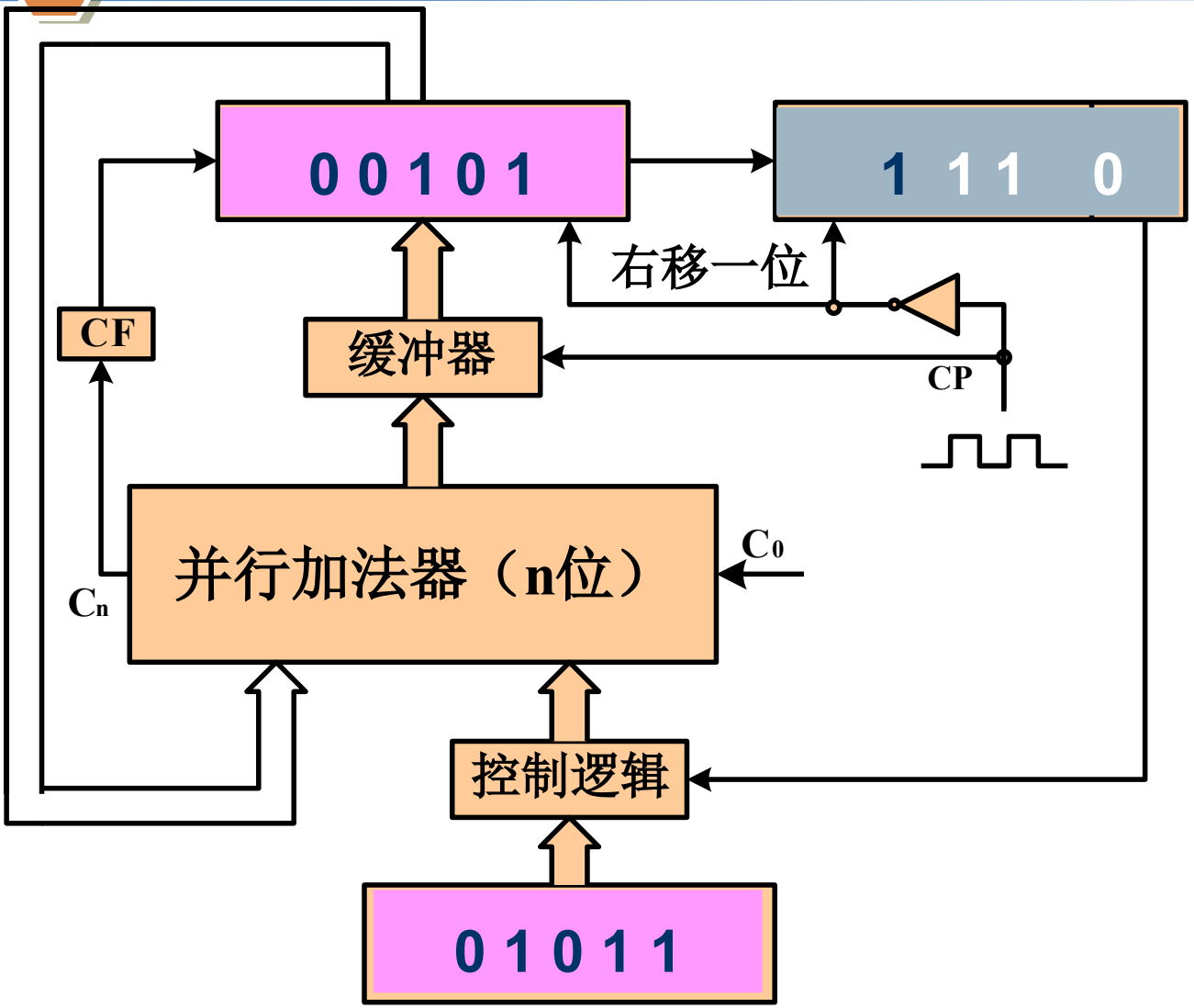
00000 1101

01011 1101



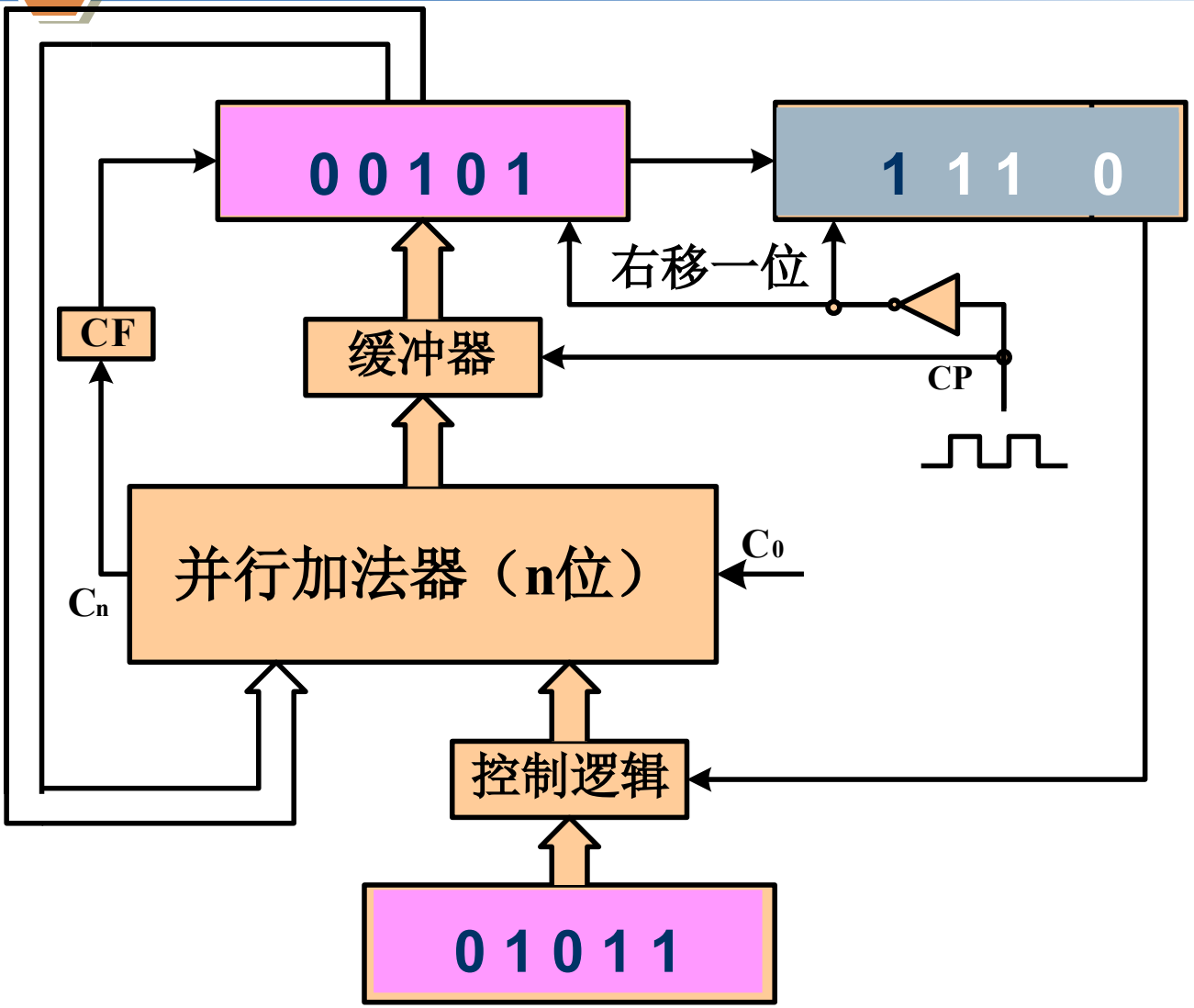
第一次求部分积　右移 1 位

00000	1101
01011	1101
00101	1110



第二次求部分积 加运算：＋0

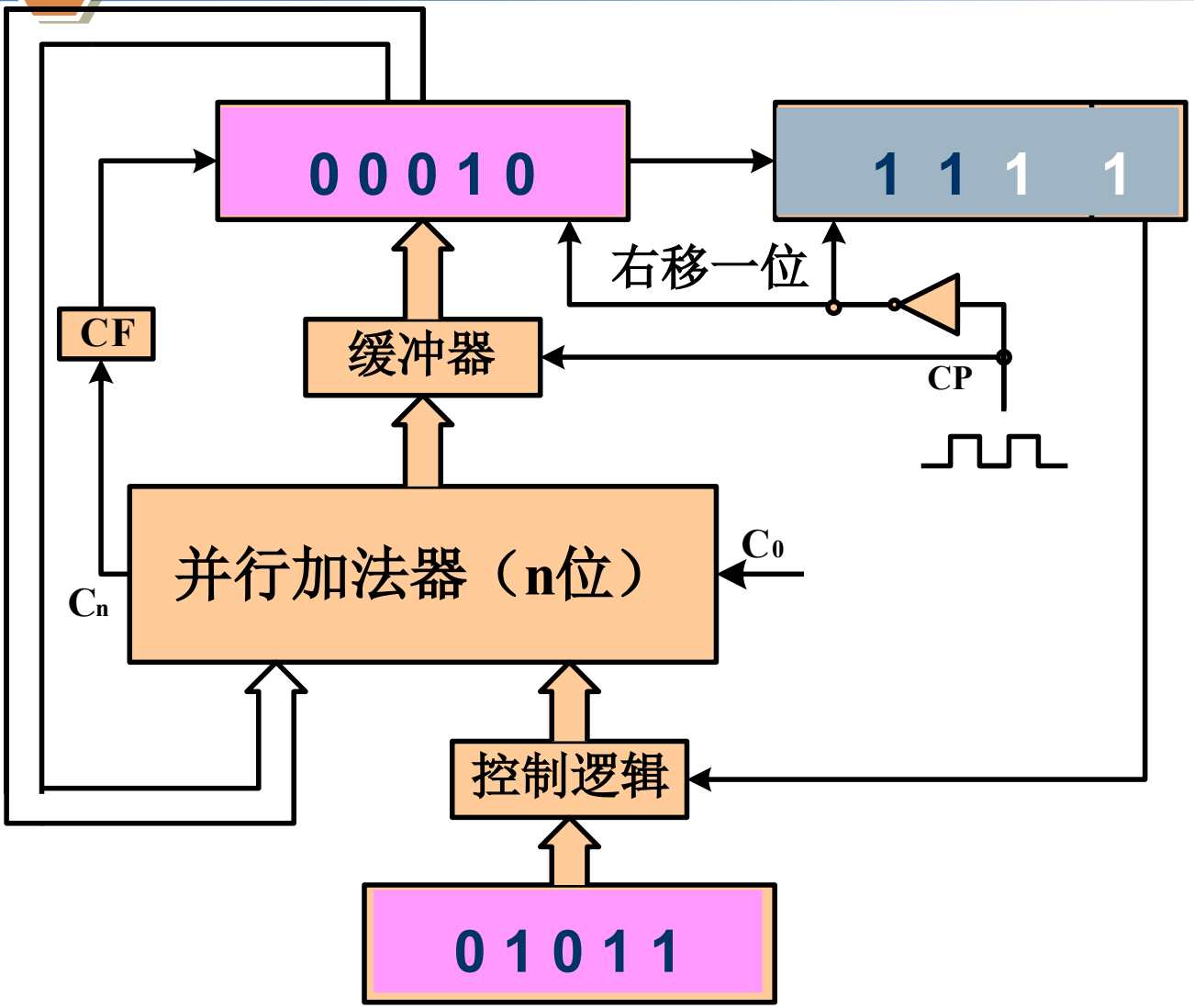
00000	1101
01011	1101
00101	1110
00101	1110



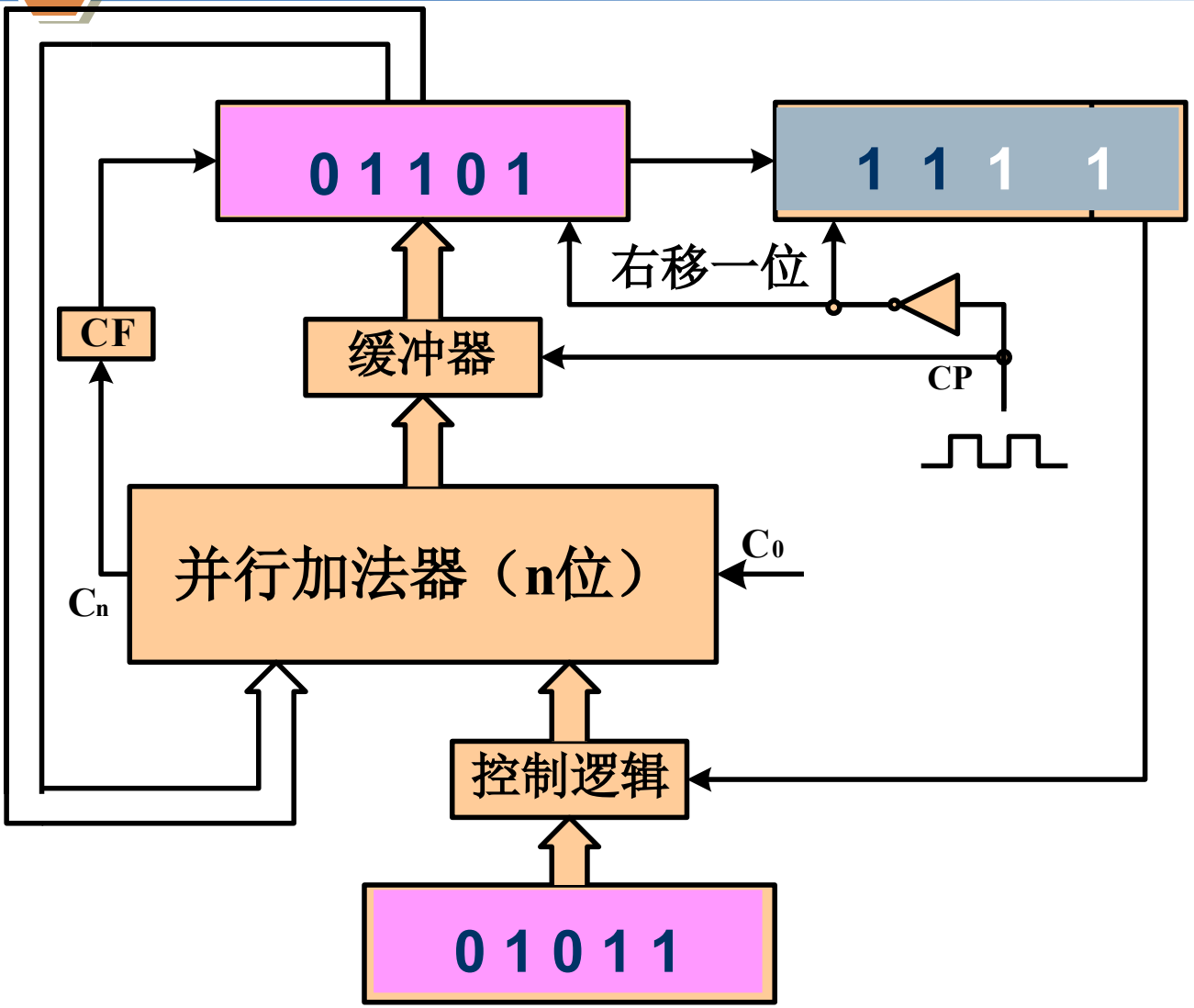
第二次求部分积

右移 1 位

00000	1101
01011	1101
00101	1110
00101	1110
00010	1111



第三次求部分积 加运算：＋ |X|

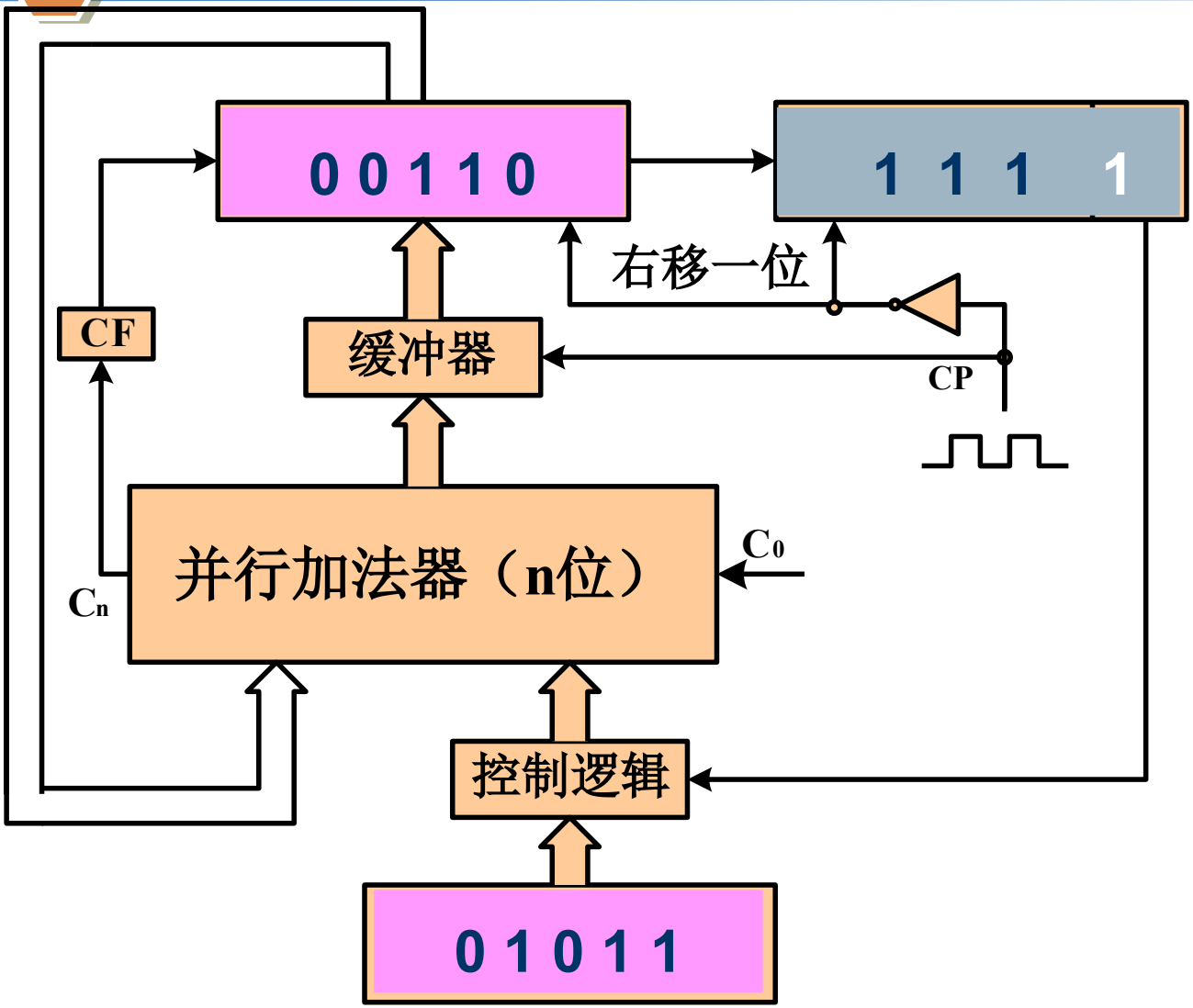


00000	1101
01011	1101
00101	1110
00101	1110
00010	1111
01101	1111

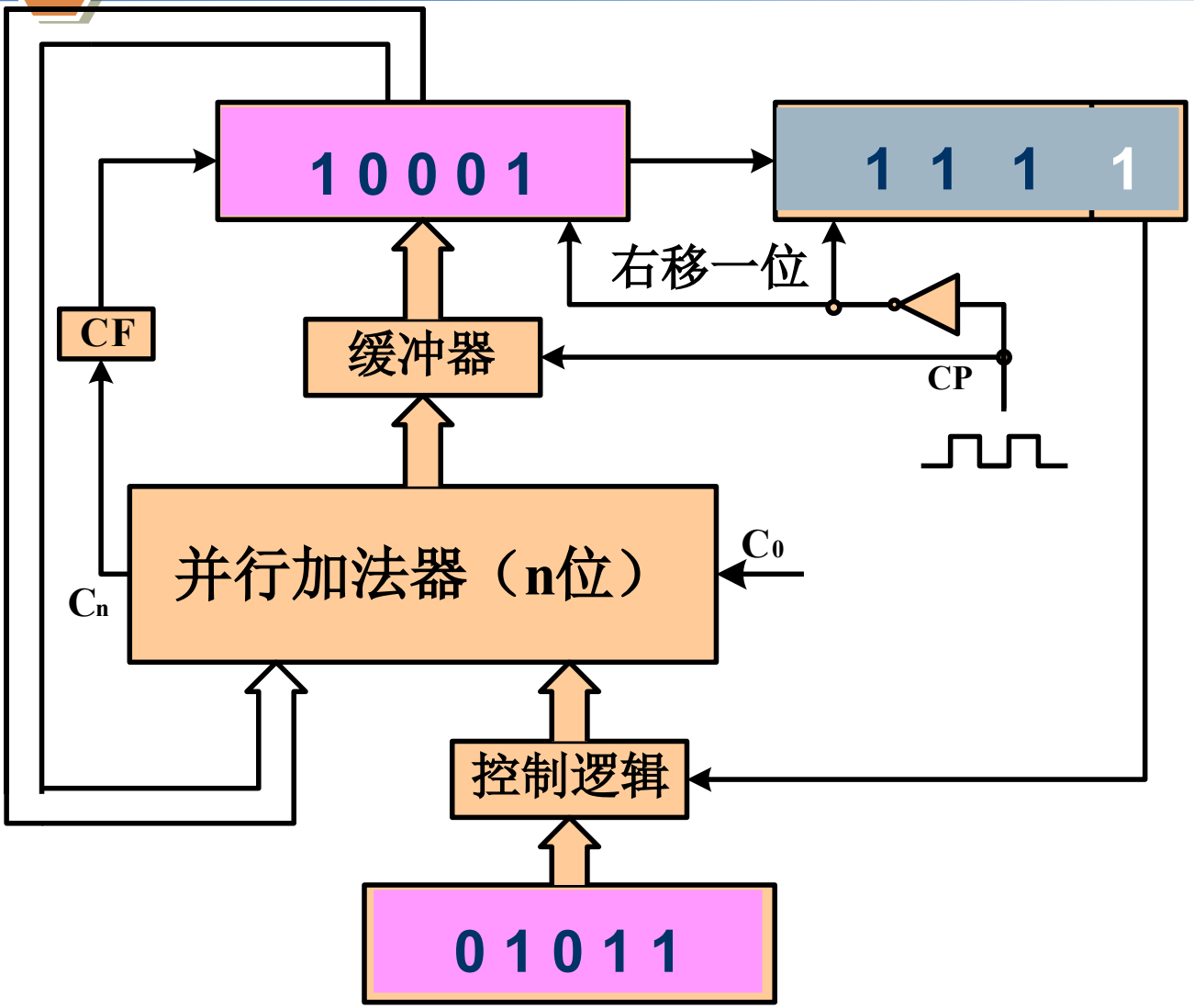
第三次求部分积

右移 1 位

00000	1101
01011	1101
00101	1110
00101	1110
00010	1111
01101	1111
00110	1111

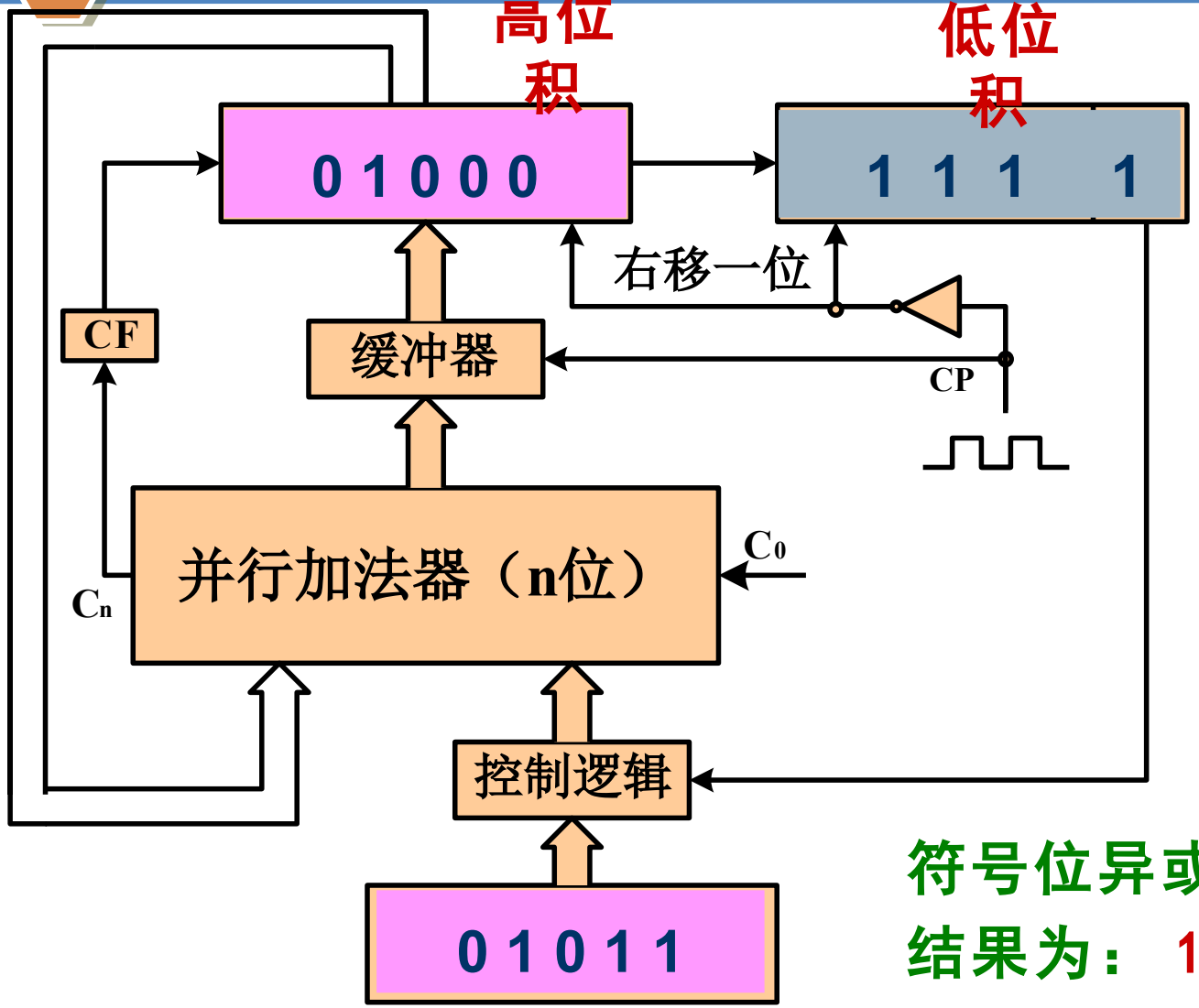


第四次求部分积　加运算：＋|X|



00000	1101
01011	1101
00101	1110
00101	1110
00010	1111
01101	1111
00110	1111
10001	1111

第四次求部分积

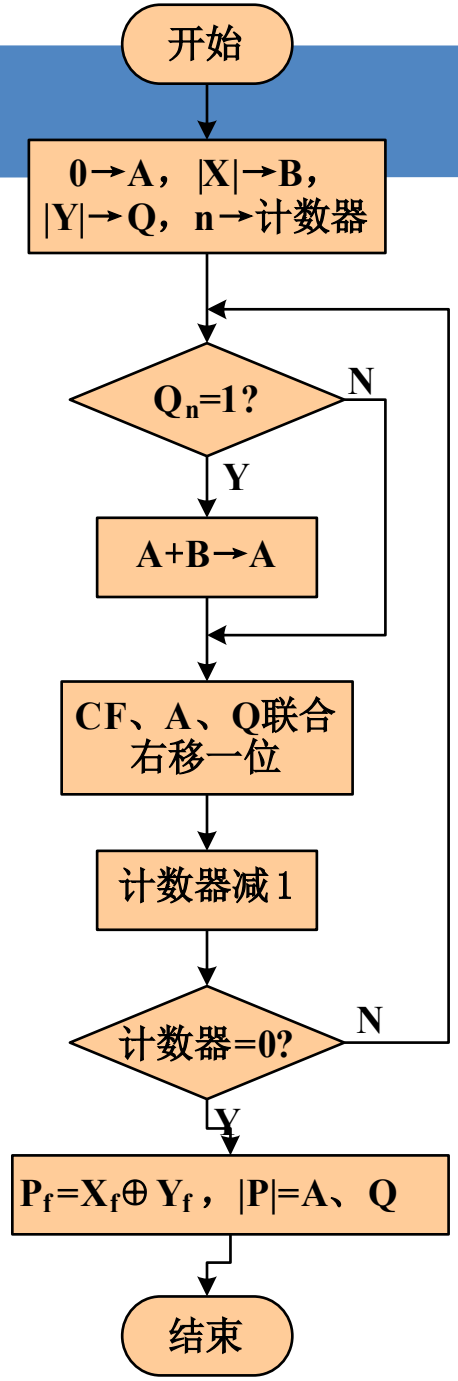


右移 1 位		
	00000	1101
	01011	1101
	00101	1110
	00101	1110
	00010	1111
	01101	1111
	00110	1111
	01000	1111

符号位异或

结果为： 1, 10001111

原码一位乘法流程


开始
0→A，|X|→B，
|Y|→Q，n→计数器
Qn=1?
N
Y
A+B→A
CF、A、Q联合
右移一位
计数器减1
计数器=0?
N
Y
Pf=Xf⊕Yf，|P|=A、Q
结束

二、补码乘法及实现

- 1、补码乘法算法
- （1）补码一位乘法——校正法
- （2）补码一位乘法——Booth算法
- 2、补码乘法的硬件实现

（1）补码一位乘法——校正法

- 假设 $[X]_{补} = X_0 . X_1 \cdots\cdots X_n$ ，
- $[Y]_{补} = Y_0 . Y_1 \cdots\cdots Y_n$ ，
- 则有：
- $[X \cdot Y]_{补} = [X]_{补} \cdot （0.Y_1 \cdots\cdots Y_n）+ Y_0 \cdot [-X]_{补}$
- 证明如下：

（1）补码一位乘法——校正法

- 当被乘数 X 的符号任意，Y 为正数时：根据补码定义有：

$[X]_{补} = 2 + X = 2^{n+1} + X \quad (\mathrm{mod}\ 2)$

$[Y]_{补} = Y$　　则：

$[X]_{补} \cdot [Y]_{补} = (2^{n+1} + X) \cdot Y = 2^{n+1} \cdot Y + X \cdot Y$

$= 2^{n+1} \cdot (0.Y_1 \cdots\cdots Y_n) + X \cdot Y$

$= 2 \cdot (Y_1 \cdots\cdots Y_n) + X \cdot Y = 2 + X \cdot Y \quad (\mathrm{mod}\ 2)$

$= [X \cdot Y]_{补}$

- 即：Y >0 时，
  - $[X \cdot Y]_{补} = [X]_{补} \cdot [Y]_{补} = [X]_{补} \cdot$

（1）补码一位乘法——校正法

- 当被乘数 X 的符号任意，Y 为负数时：

$[Y]_{补} = 2 + Y = 1.Y_1\cdots\cdots Y_n$ 则：

$Y = [Y]_{补} - 2 = 0.Y_1\cdots\cdots Y_n - 1$

$[X \cdot Y]_{补} = [X \cdot 0.Y_1\cdots\cdots Y_n - X]_{补}$

$= [X \cdot 0.Y_1\cdots\cdots Y_n]_{补} + [-X]_{补}$

因为 $0.Y_1\cdots\cdots Y_n > 0$，所以：

$[X \cdot 0.Y_1\cdots\cdots Y_n]_{补} = [X]_{补} \cdot (0.Y_1\cdots\cdots Y_n)$

- 所以：Y<0 时，
  - $[X \cdot Y]_{补} = [X]_{补} \cdot (0.Y_1\cdots\cdots Y_n) + [-X]_{补}$

校正法举例 1

- X=+0.1011，Y=−0.1101，用补码一位乘法的校正法计算 P=X·Y。

$[X]_{补}$=00.1011

$[Y]_{补}$=11.0011

$[-X]_{补}$=11.0101

$[X \cdot Y]_{补}$ = 1.0111 0001

X·Y = -0.1000 1111

部分积	乘数Y	操作说明
00.0000	0 0 1 1	
+ 00.1011		Y_4=1，+$[X]_{补}$
00.1011		
00.0101	1 0 0 1	右移一位
+ 00.1011		Y_3=1，+$[X]_{补}$
01.0000		
00.1000	0 1 0 0	右移一位
+ 00.0000		Y_2=0，+0
00.1000		
00.0100	0 0 1 0	右移一位
+ 00.0000		Y_1=0，+0
00.0100		
00.0010	0 0 0 1	右移一位
+ 11.0101		Y_0=1，+$[-X]_{补}$校正
11.0111	0 0 0 1	

校正法举例 2

- 设 X=-0.1101 Y=-0.1011，即：
- $[X]_{补}=11.0011$
- $[Y]_{补}=11.0101$
- $[-X]_{补}=0.1101$
- 求 $[X*Y]_{补}$

计算结果：

$[X*Y]_{补}=0.10001111$

	部分积	乘数Y
	00.0000	0 1 0 1
$+[X]_{补}$	11.0011	
	11.0011	
右移一位	11.1001	1 0 1 0
+0	00.0000	
	11.1001	
右移一位	11.1100	1 1 0 1
$+[X]_{补}$	11.0011	
	10.1111	
右移一位	11.0111	1 1 1 0
+0	00.0000	
	11.0111	
右移一位	11.1011	1 1 1 1
$+[-X]_{补}$校正	00.1101	
	00.1000	1 1 1 1

（2）补码一位乘法——Booth

❖ 做出如下推导 ：

$$[X \cdot Y]_{补} = [X]_{补} \cdot (0.Y_1 \cdots\cdots Y_n) + Y_0 \cdot [-X]_{补}$$

$$= [X]_{补} \cdot (Y_1 \cdot 2^{-1} + Y_2 \cdot 2^{-2} + \cdots\cdots + Y_n \cdot 2^{-n} - Y_0)$$

$$= [X]_{补} \cdot [Y_1 \cdot (2^0 - 2^{-1}) + Y_2 \cdot (2^{-1} - 2^{-2}) + \cdots\cdots + Y_n \cdot (2^{-n+1} - 2^{-n}) - Y_0 \cdot 2^0]$$

$$= [X]_{补} \cdot [Y_1 \cdot 2^0 - Y_1 \cdot 2^{-1} + Y_2 \cdot 2^{-1} - Y_2 \cdot 2^{-2} + \cdots\cdots + Y_n \cdot 2^{-n+1} - Y_n \cdot 2^{-n} - Y_0 \cdot 2^0]$$

$$= [X]_{补} \cdot [(Y_1 - Y_0) \cdot 2^0 + (Y_2 - Y_1) \cdot 2^{-1} + (Y_3 - Y_2) \cdot 2^{-2} + \cdots\cdots + (Y_n - Y_{n-1}) \cdot 2^{-n+1} - Y_n \cdot 2^{-n}]$$

$$= [X]_{补} \cdot [(Y_1 - Y_0) \cdot 2^0 + (Y_2 - Y_1) \cdot 2^{-1} + (Y_3 - Y_2) \cdot 2^{-2} + \cdots\cdots + (Y_n - Y_{n-1}) \cdot 2^{-n+1} + (Y_{n+1} - Y_n) \cdot 2^{-n}]$$

$$= [X]_{补} \cdot (a_0 \cdot 2^0 + a_1 \cdot 2^{-1} + a_2 \cdot 2^{-2} + \cdots\cdots + a_{n-1} \cdot 2^{-n+1} + a_n \cdot 2^{-n})$$

Booth 算法的运算规则

假设 $[Y]_{补}$ = Y0 . Y1……Yn

① 被乘数 X 和乘数 Y 均以补码的形式参加乘法运算，运算的结果是积的补码。

② 部分积和被乘数 X 采用双符号位，乘数 Y 采用单符号位。

③ 初始部分积为 0；运算前，在乘数 Y 的补码末位后添加一位附加位 Yn+1，初始为 0。

④ 根据 YnYn+1 的值，按照表 4.3 进行累加右移操作，右移时遵循补码的移位规

⑤ 累加 n+1 次，右移 n 次，

Y_n	Y_{n+1}	操作
0	0	+0 ，右移一位
0	1	$+[X]_{补}$，右移一位
1	0	$+[-X]_{补}$，右移一位
1	1	+0 ，右移一位

Booth 算法举例

- X=+0.1011，Y=-0.1101，用补码一位乘法的 Booth 算法计算 P=X·Y。
- $[X]_{补}$=00.1011
- $[Y]_{补}$=11.0011
- $[-X]_{补}$=11.0101

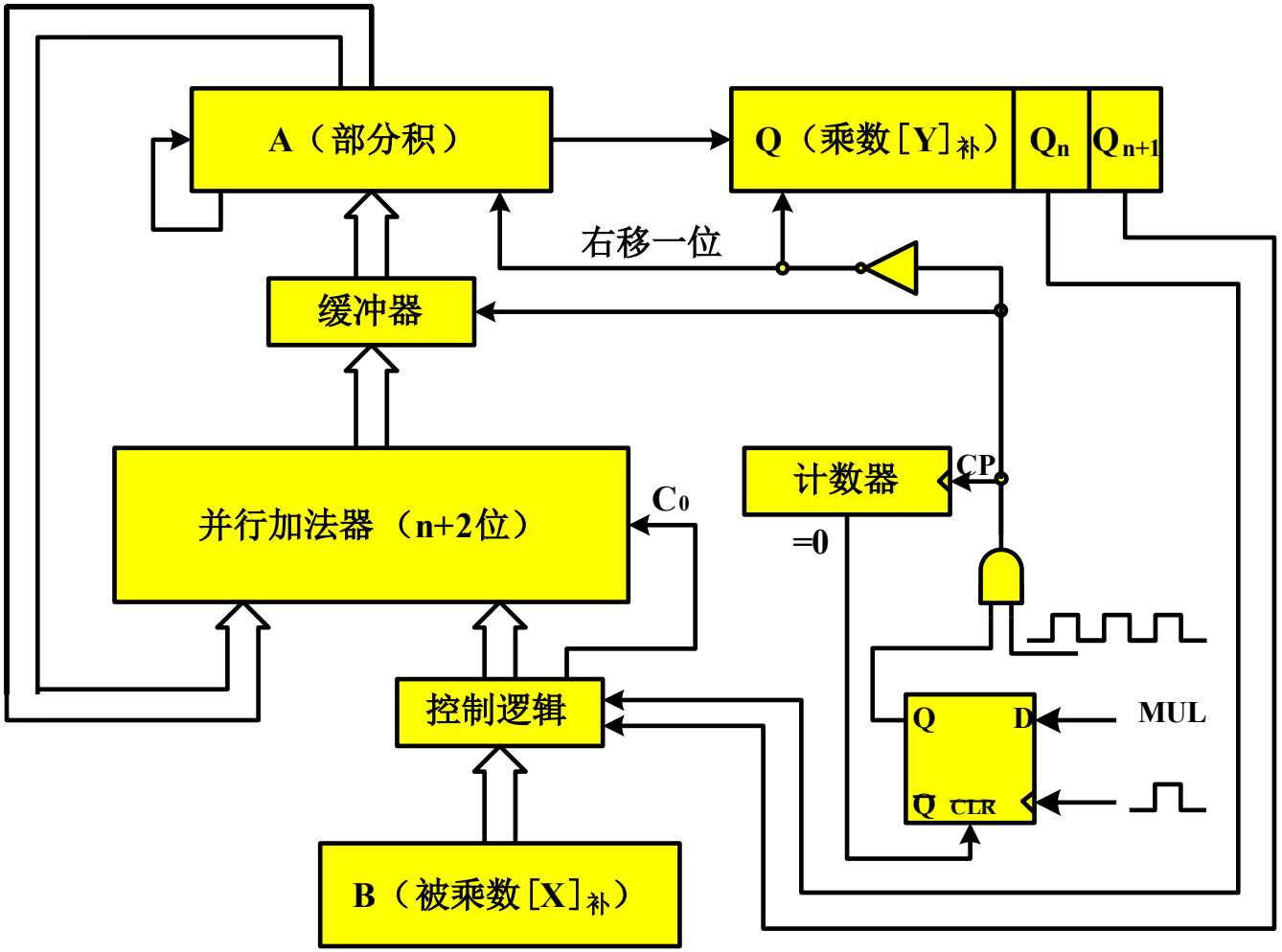
	部分积	乘数Y (Y_n Y_{n+1})	操作说明
	00.0000	1.0 0 1 1 0	
+	11.0101		Y_4Y_5=10，+$[-X]_{补}$
	11.0101		
	11.1010	1 1.0 0 1 1	右移一位
+	00.0000		Y_3Y_4=11，+0
	11.1010		
	11.1101	0 1 1.0 0 1	右移一位
+	00.1011		Y_2Y_3=01，+$[X]_{补}$
	00.1000		
	00.0100	0 0 1 1.0 0	右移一位
+	00.0000		Y_1Y_2=00，+0
	00.0100		
	00.0010	0 0 0 1 1.0	右移一位
+	11.0101		Y_0Y_1=10，+$[-X]_{补}$
	11.0111	0 0 0 1	

$[X \cdot Y]_{补}$ = 1.0111 0001

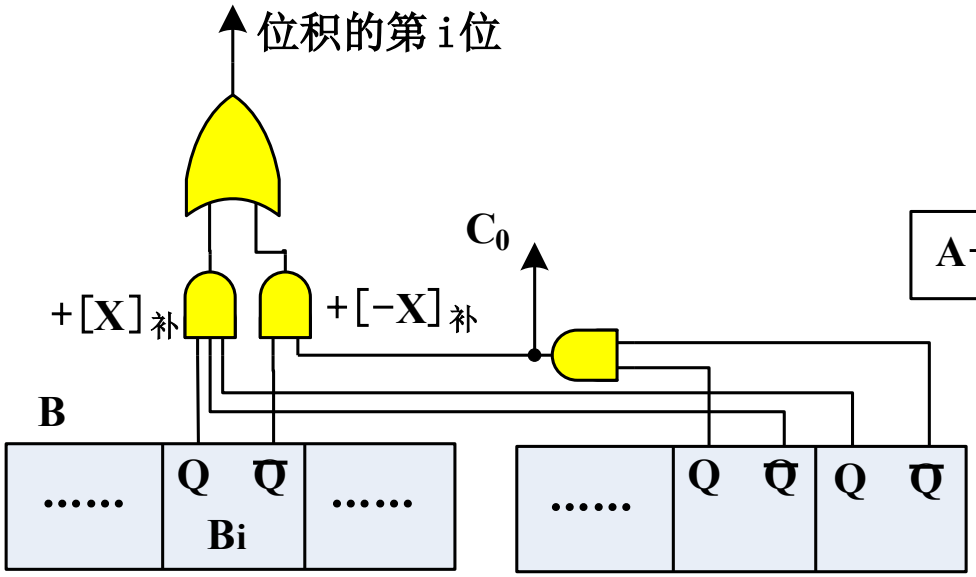
X·Y = −0.1000 1111

2、补码乘法的硬件实现

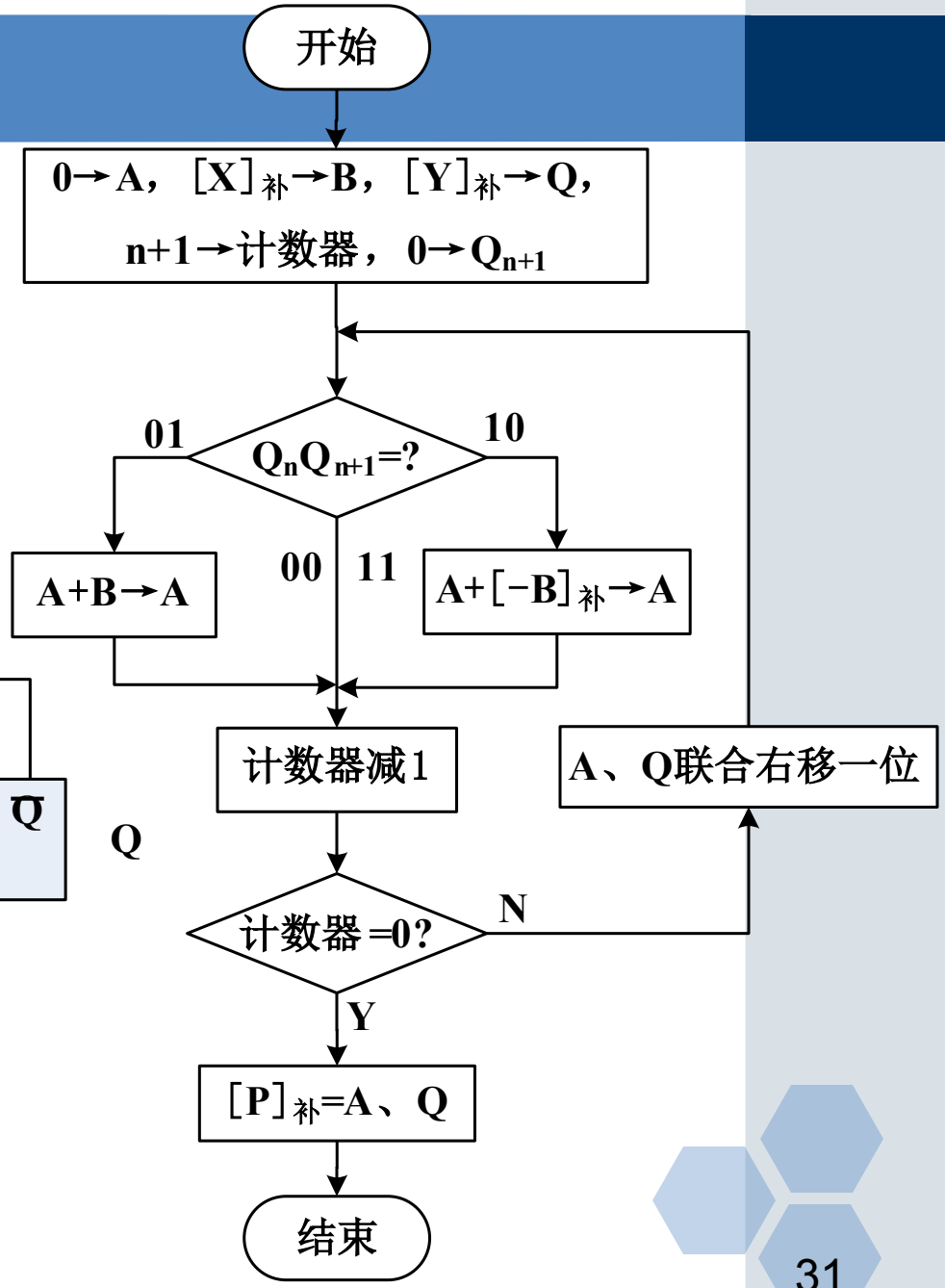
❖ Booth 乘法的硬件实现



补码乘法的 Booth 算法流程



控制逻辑电路

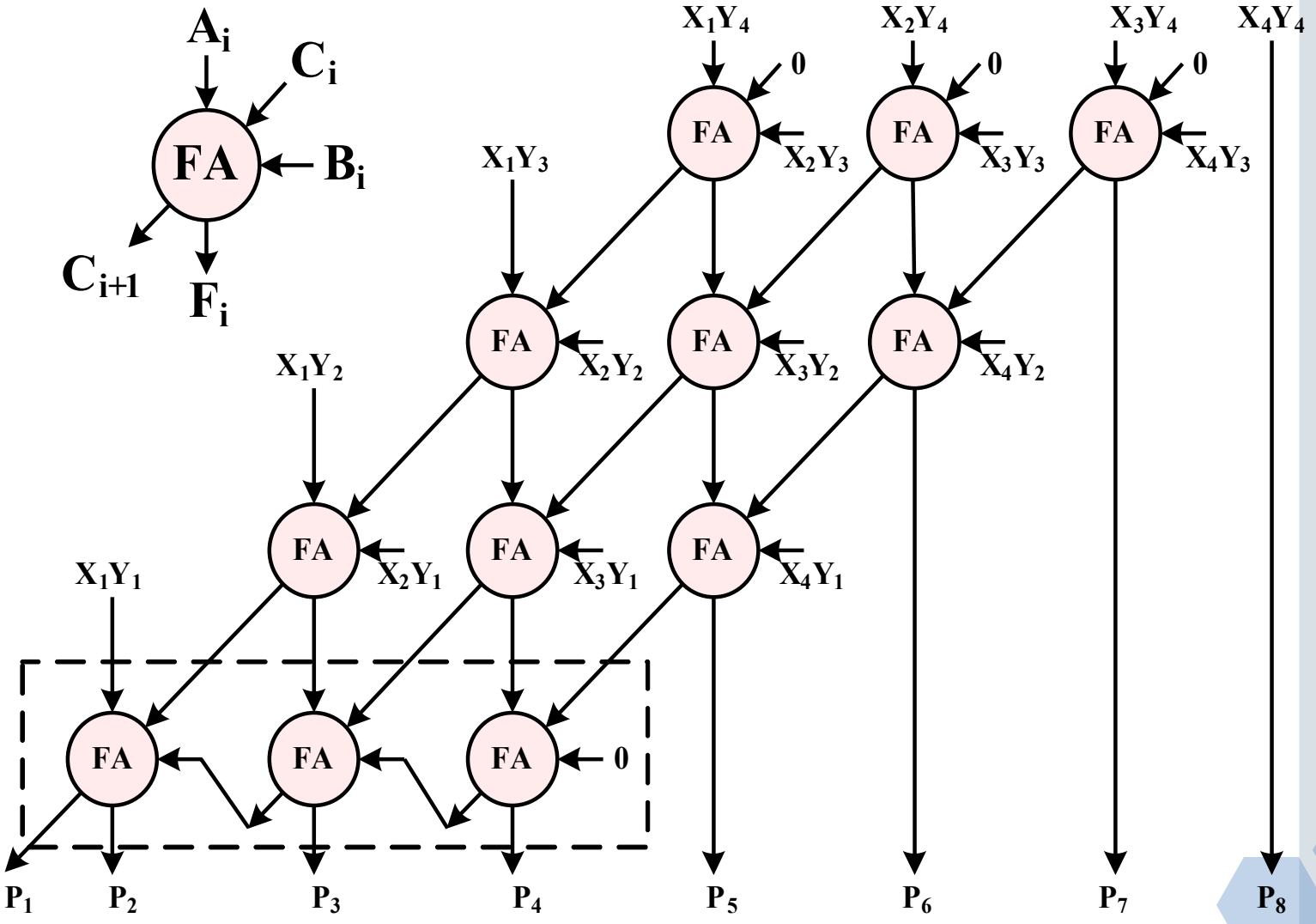


三、阵列乘法器

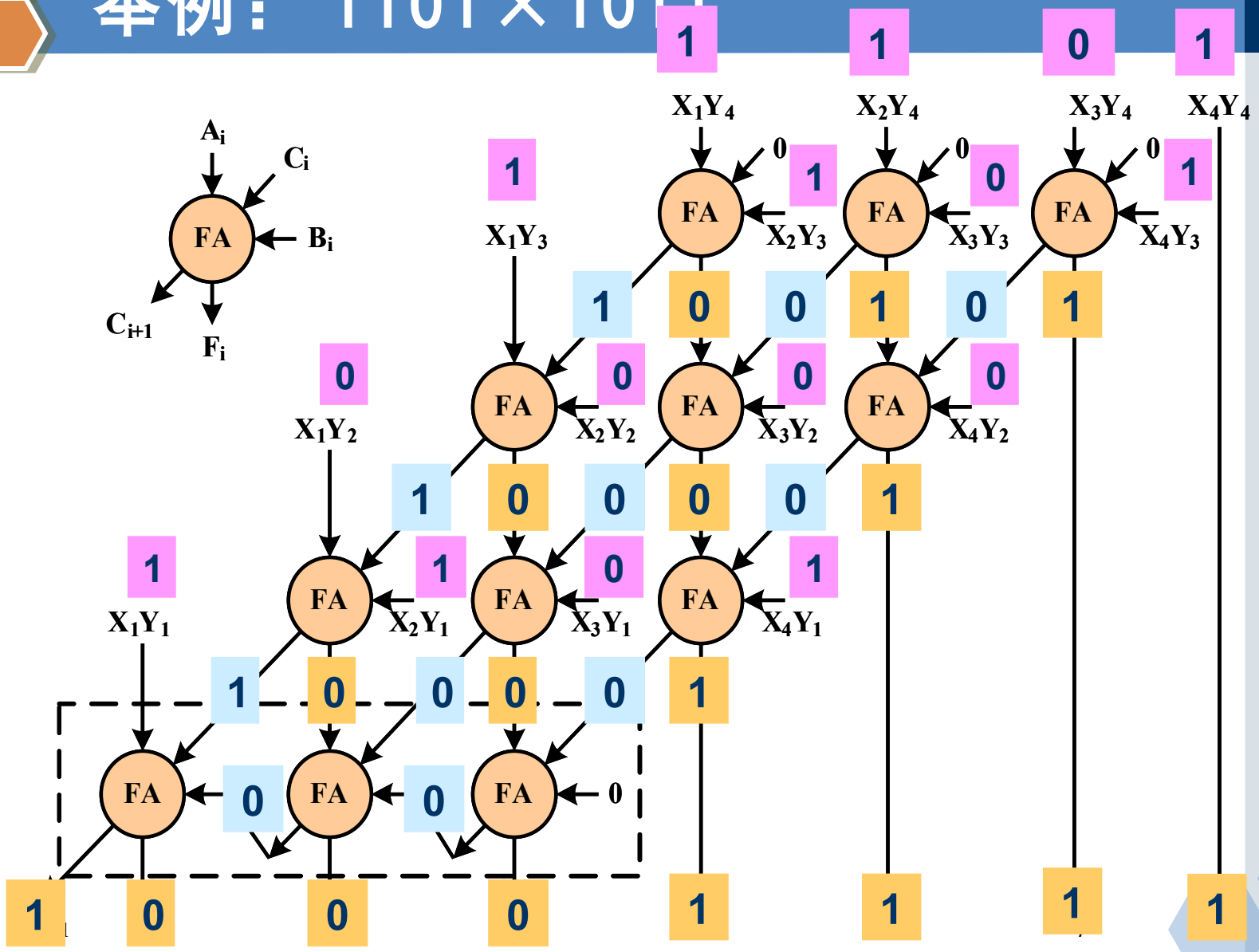
- 原理类似于二进制手工算法
  - 位积的每一位 XiYj 用一个与门产生。
  - 每一次的累加都用单独一组 FA 实现。
  - 本次累加的 FA 之间的进位，送至下一次再累加。

$$
\begin{array}{cccccccc}
 & & & X_1 & X_2 & X_3 & X_4 & \\
 & & \times & Y_1 & Y_2 & Y_3 & Y_4 & \\
\hline
 & & & X_1Y_4 & X_2Y_4 & X_3Y_4 & X_4Y_4 & \\
 & & X_1Y_3 & X_2Y_3 & X_3Y_3 & X_4Y_3 & & \\
 & X_1Y_2 & X_2Y_2 & X_3Y_2 & X_4Y_2 & & & \\
+ & X_1Y_1 & X_2Y_1 & X_3Y_1 & X_4Y_1 & & & \\
\hline
P_1 & P_2 & P_3 & P_4 & P_5 & P_6 & P_7 & P_8
\end{array}
$$

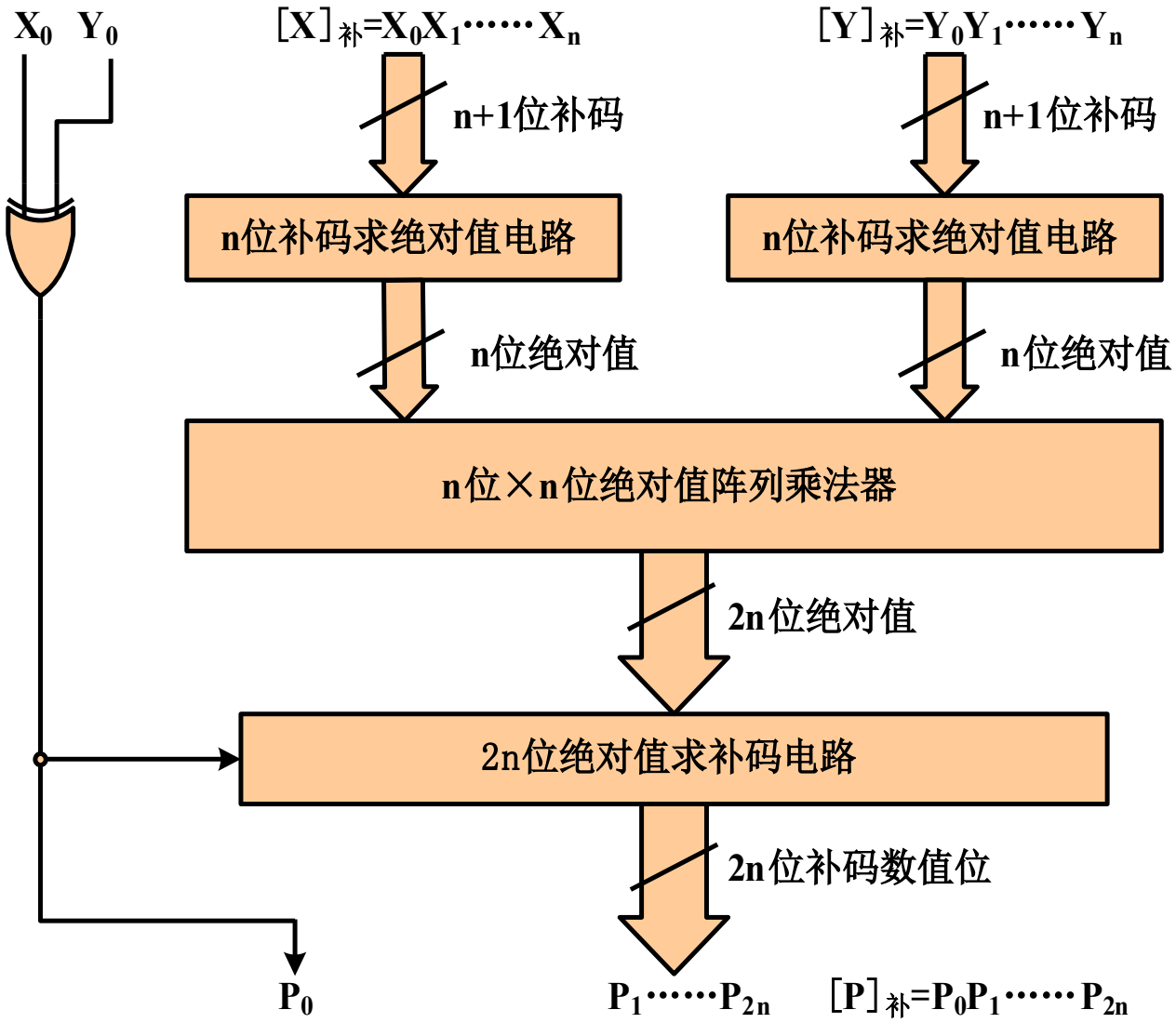
绝对值阵列乘法器

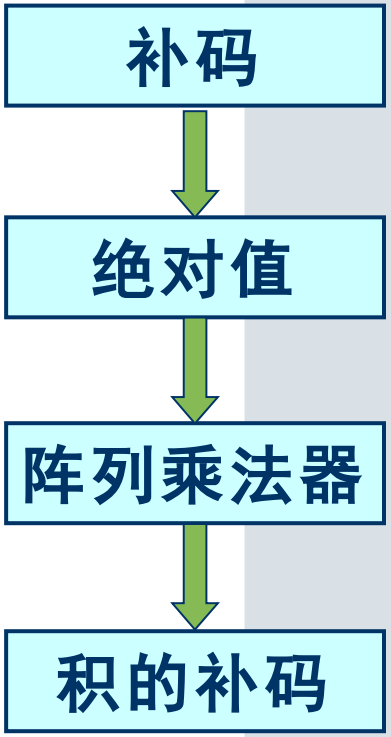

Ai
Ci
FA
Bi
Ci+1
Fi
X1Y4
X2Y4
X3Y4
X4Y4
0
X1Y3
X2Y3
X3Y3
X4Y3
X1Y2
X2Y2
X3Y2
X4Y2
X1Y1
X2Y1
X3Y1
X4Y1
P1
P2
P3
P4
P5
P6
P7
P8

举例： 1101×1011

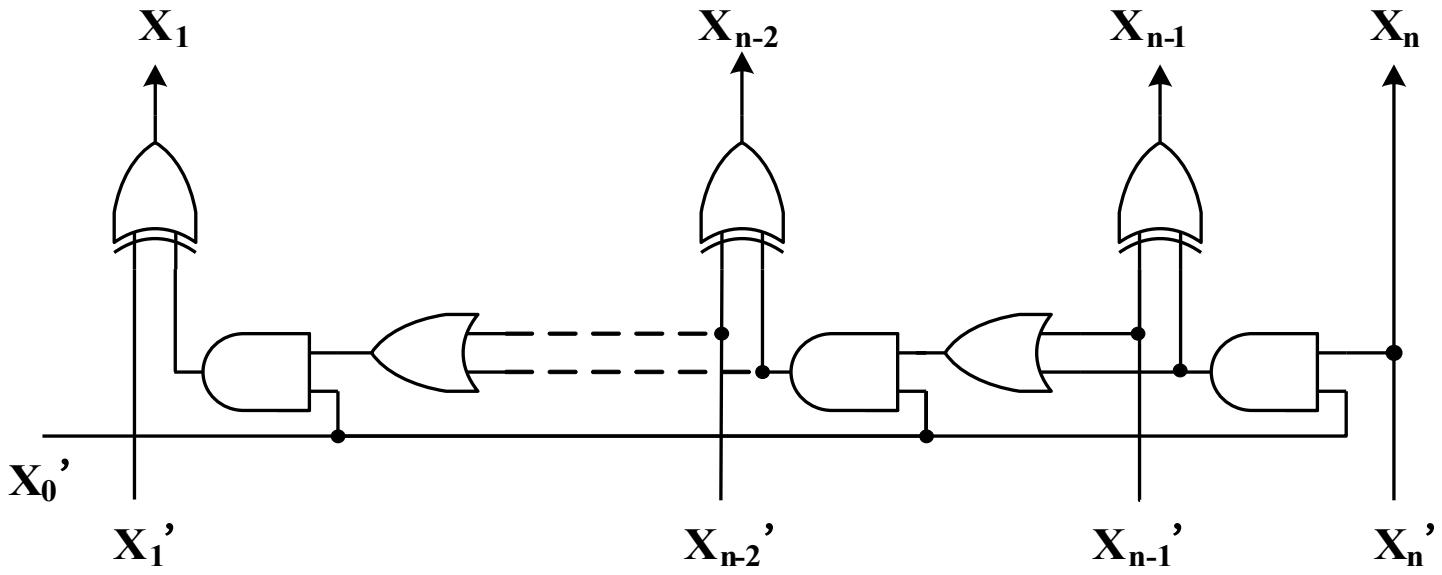

Ai
Ci
FA
Bi
Ci+1
Fi
X1Y4
X2Y4
X3Y4
X4Y4
X1Y3
X2Y3
X3Y3
X4Y3
X1Y2
X2Y2
X3Y2
X4Y2
X1Y1
X2Y1
X3Y1
X4Y1

补码阵列乘法器


X_0 Y_0
$[X]_{补}=X_0X_1\cdots\cdots X_n$
$[Y]_{补}=Y_0Y_1\cdots\cdots Y_n$
n+1位补码
n+1位补码
n位补码求绝对值电路
n位补码求绝对值电路
n位绝对值
n位绝对值
n位×n位绝对值阵列乘法器
2n位绝对值
2n位绝对值求补码电路
2n位补码数值位
P_0
$P_1\cdots\cdots P_{2n}$
$[P]_{补}=P_0P_1\cdots\cdots P_{2n}$

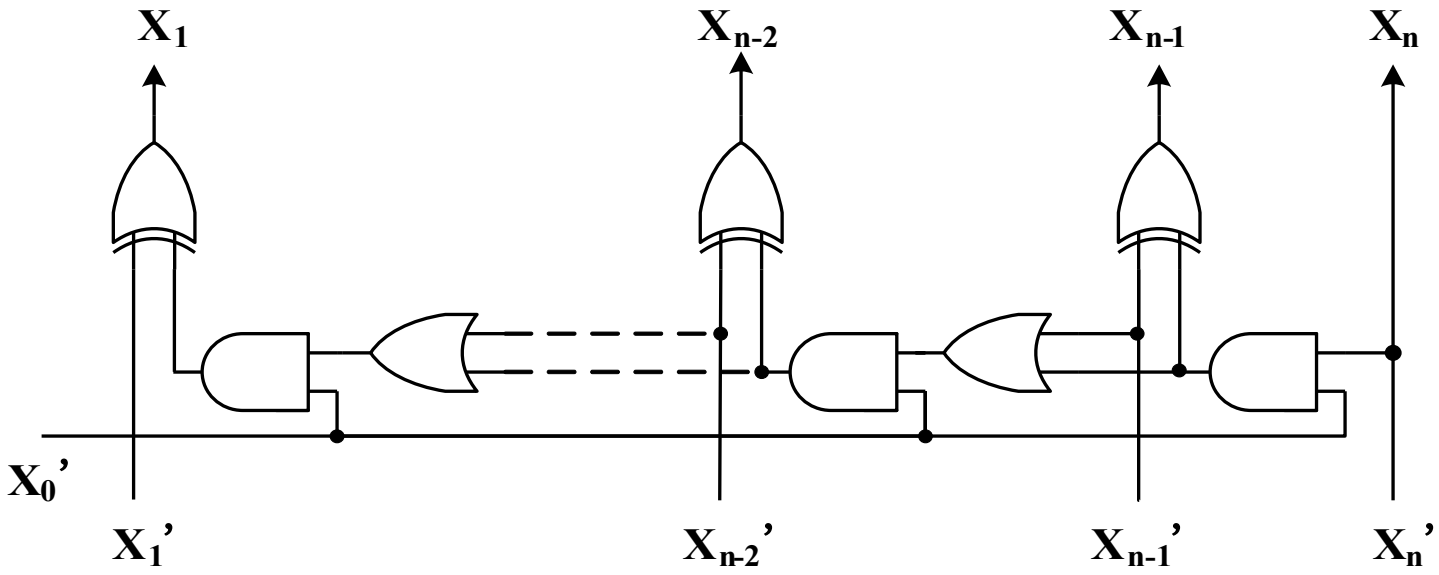

补码
绝对值
阵列乘法器
积的补码

补码→绝对值电路



- 输入：$[X]_{补}=X_0'\ X_1'\ X_2'\ \cdots\cdots\ X_n'$
- 得到：$|X|=X_1\ X_2\cdots\cdots\ X_n$

绝对值→补码电路



- 输入：$|X|=X_1'\ X_2'\ \cdots\cdots\ X_n'$，符号位 X_0'

  $X<0$：$X_0'=1$，$X\geqslant 0$：$X_0'=0$

The End !