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

BACK

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

串行乘法算法

三 阵列乘法器

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？

```
      1011
  ×   1101
-----------
      1011
     0000
    1011
   1011
-----------
  10001111
```

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$[X]_{原}=X_S\ X_1\ X_2\ \ldots\ldots X_n$，$[Y]_{原}=Y_S\ Y_1\ Y_2\ \ldots\ldots Y_n$，$P=X\cdot Y$，$P_S$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| = |X|·|Y|

$[P]_{原}$=1，10001111

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

练习

- 已知：x=-0.1110，y=-0.1101；求：$[x \cdot y]_{原}$。
-

练习

- 已知：x=-0.1110，y=-0.1101；求：$[x\cdot y]_{原}$。
- 解：$[x]_{原}$=1.1110，x*=0.1110，x_0=1

  $[y]_{原}$=1.1101，y*=0.1101，y_0=1

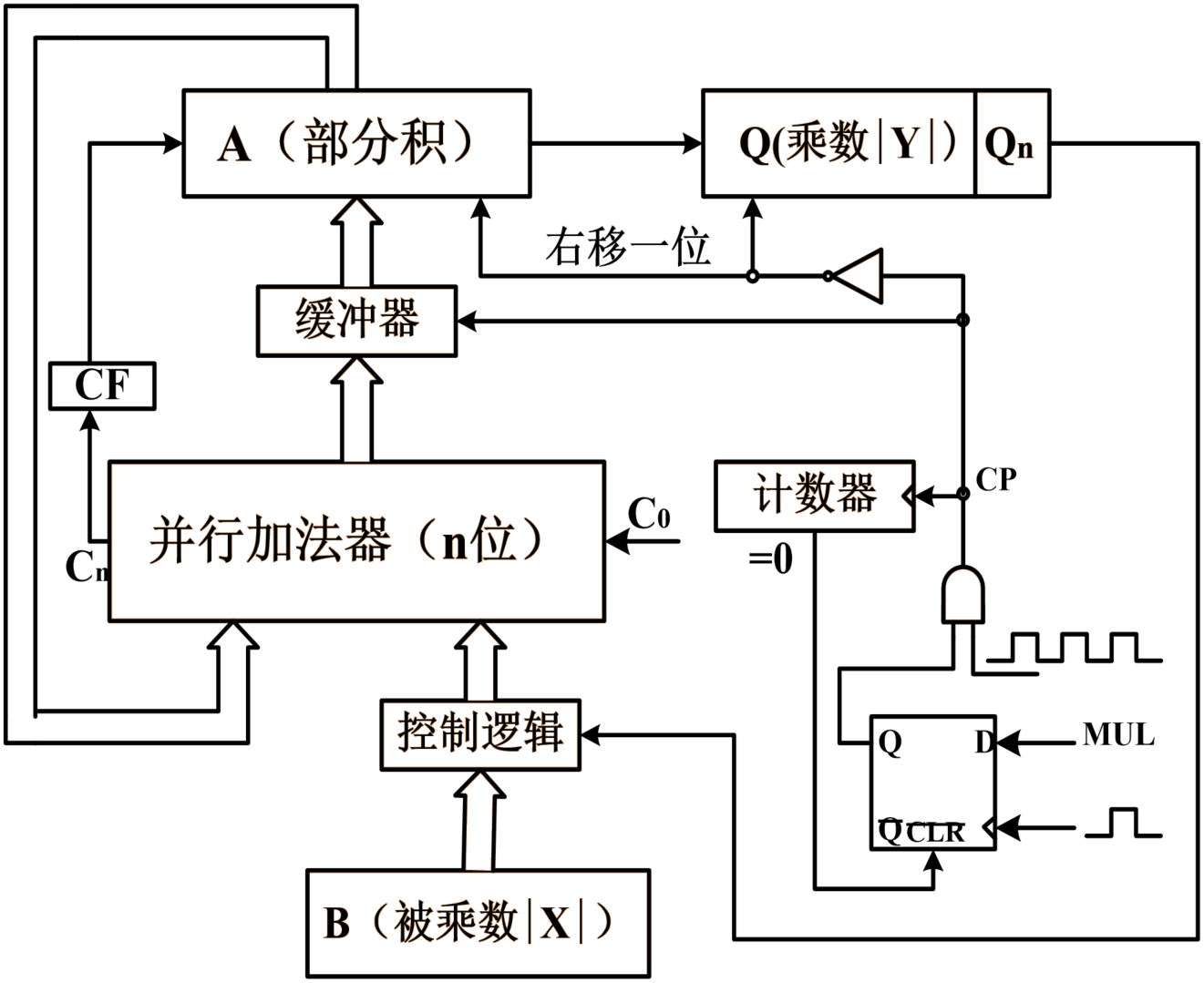
部分积	乘数	说明
00.0000 + 00.1110	1101	开始部分积z_0=0 乘数为1，加上x*
00.1110 00.0111 + 00.0000	0110	→1位得z_1， 乘数同时→1位 乘数为0，加上0
00.0111 00.0011 + 00.1110	1011	→1位得z_2， 乘数同时→1位 乘数为1，加上x*
01.0001 00.1000 + 00.1110	1101	→1位得z_3， 乘数同时→1位 乘数为1，加上x*
01.0110 00.1011	0110	→1位得z_4 乘数已全部移出

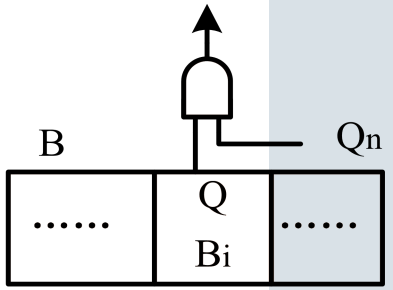
即x*·y*=0.10110110

乘积的符号位为x0和y0的异或，即0。

故$[x\cdot y]_{原}$=0.10110110。

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：

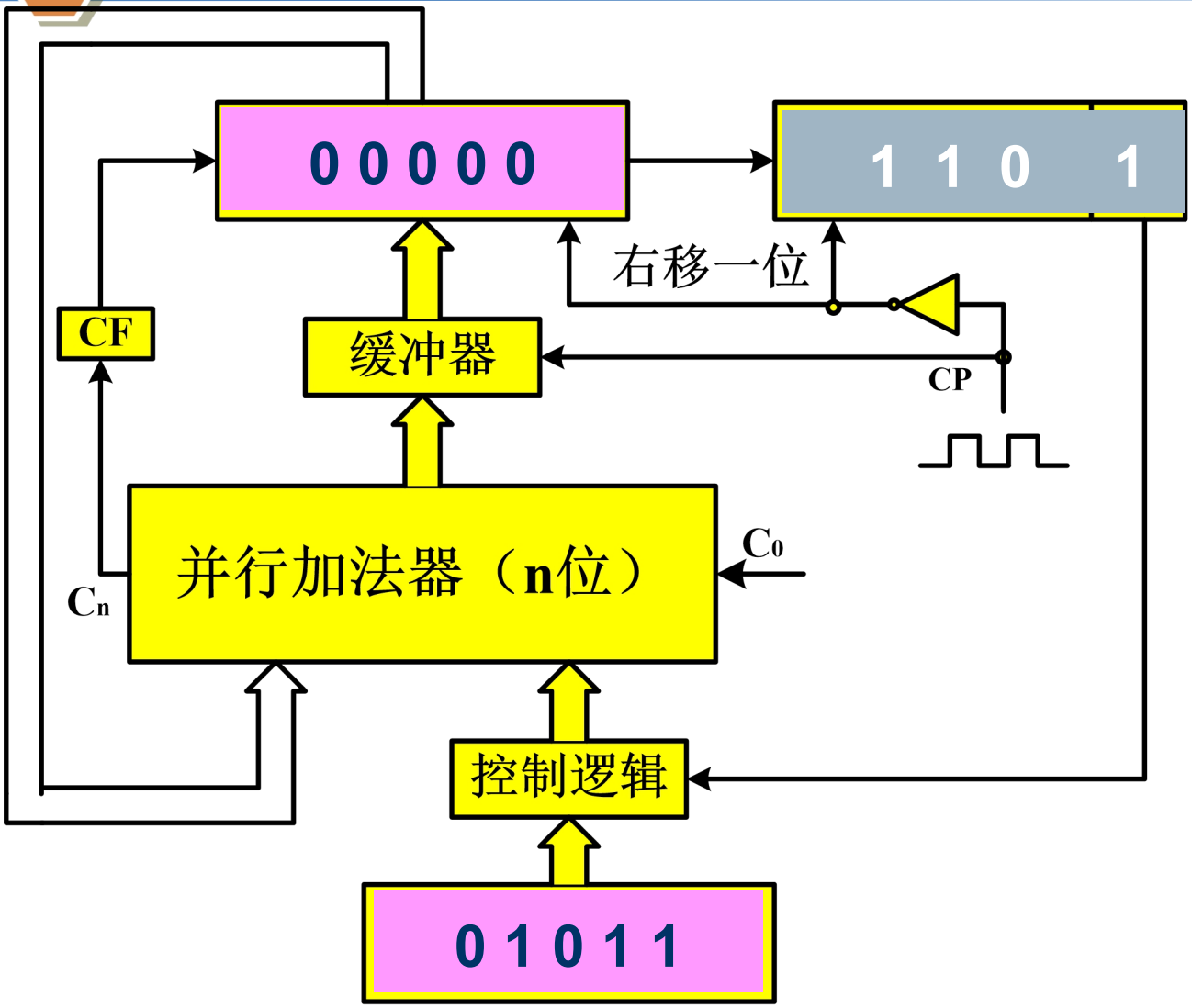
- A=0
- B=|X|
- Q=|Y|
- 计数器=n

累加、右移n次

结果：

- A=乘积高位
- B =|X|
- Q=乘积低位
- 计数器=0

1011×1101　为各寄存器赋初值　00000　1101

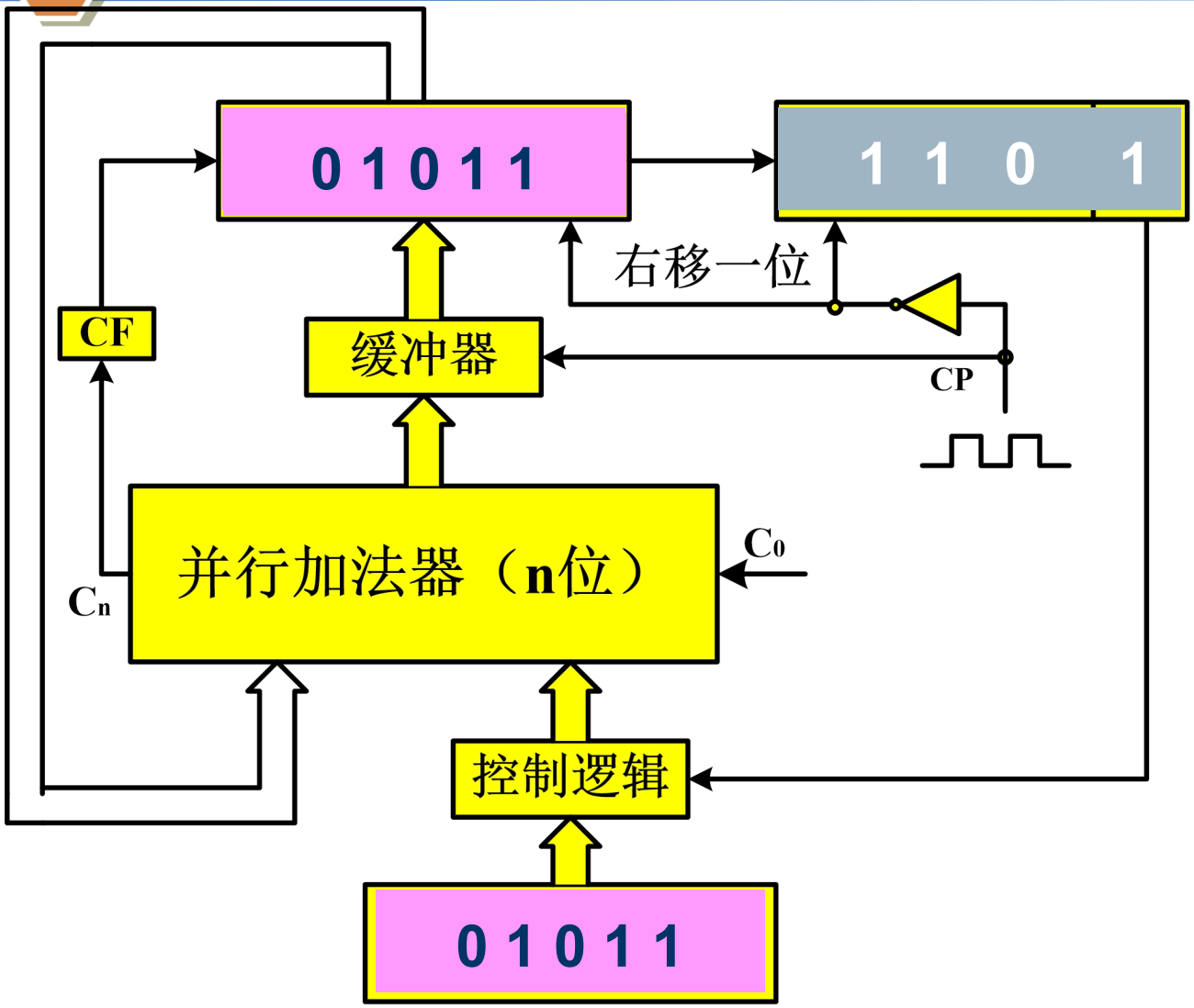


第一次求部分积

加运算：＋|X|

00000 1101

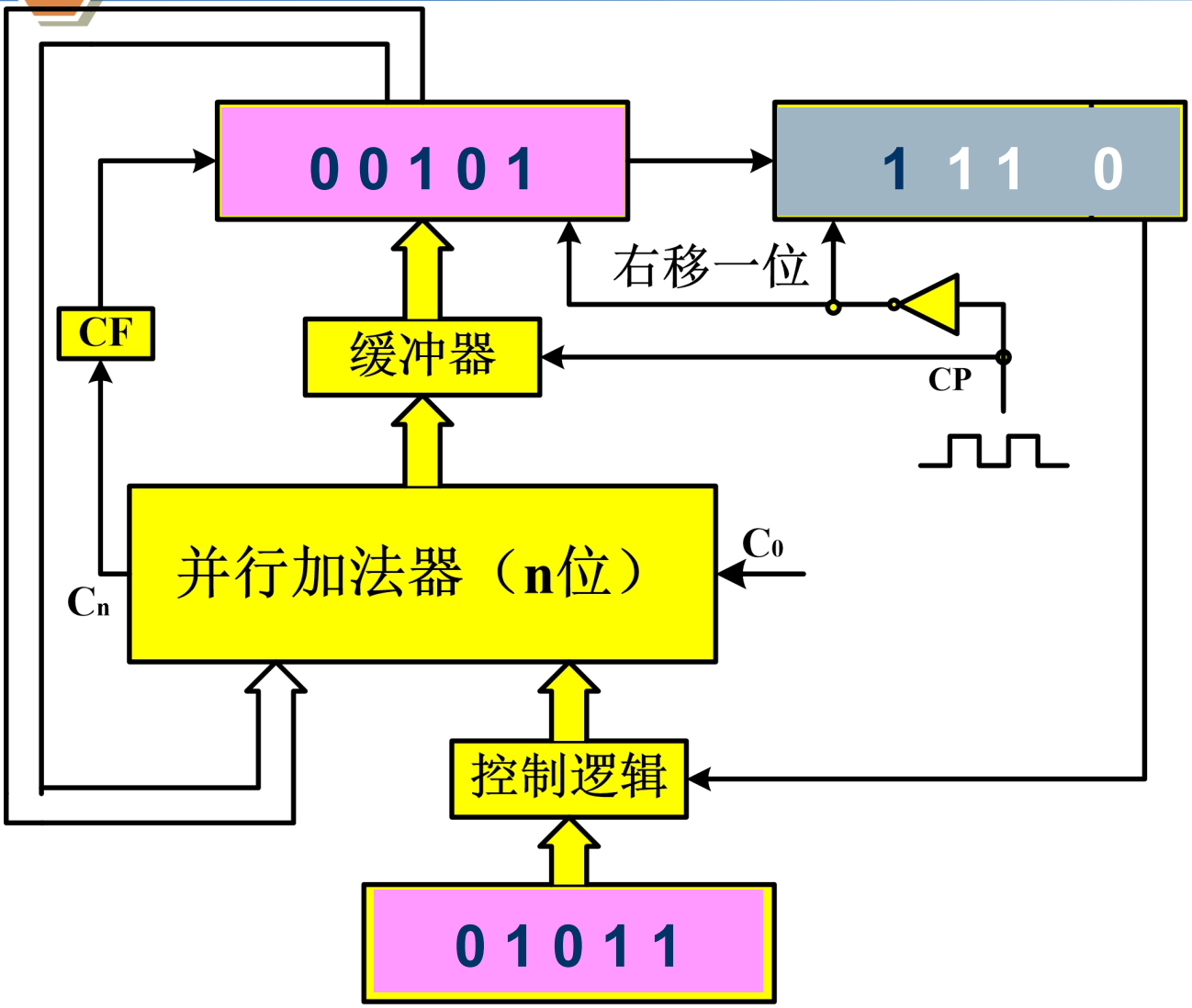
01011 1101



第一次求部分积

右移1位

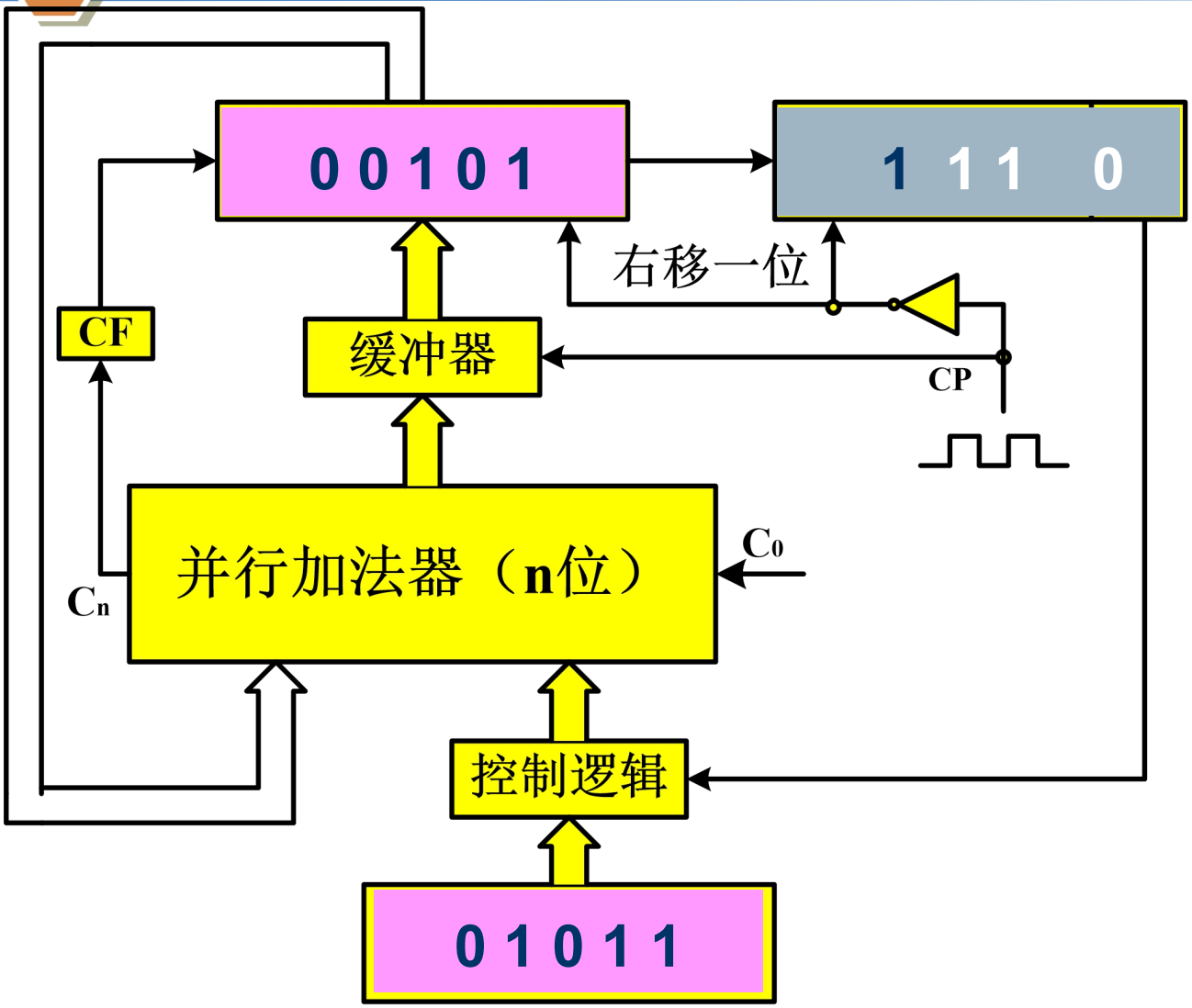
00000	1101
01011	1101
00101	1110



第二次求部分积

加运算：+0

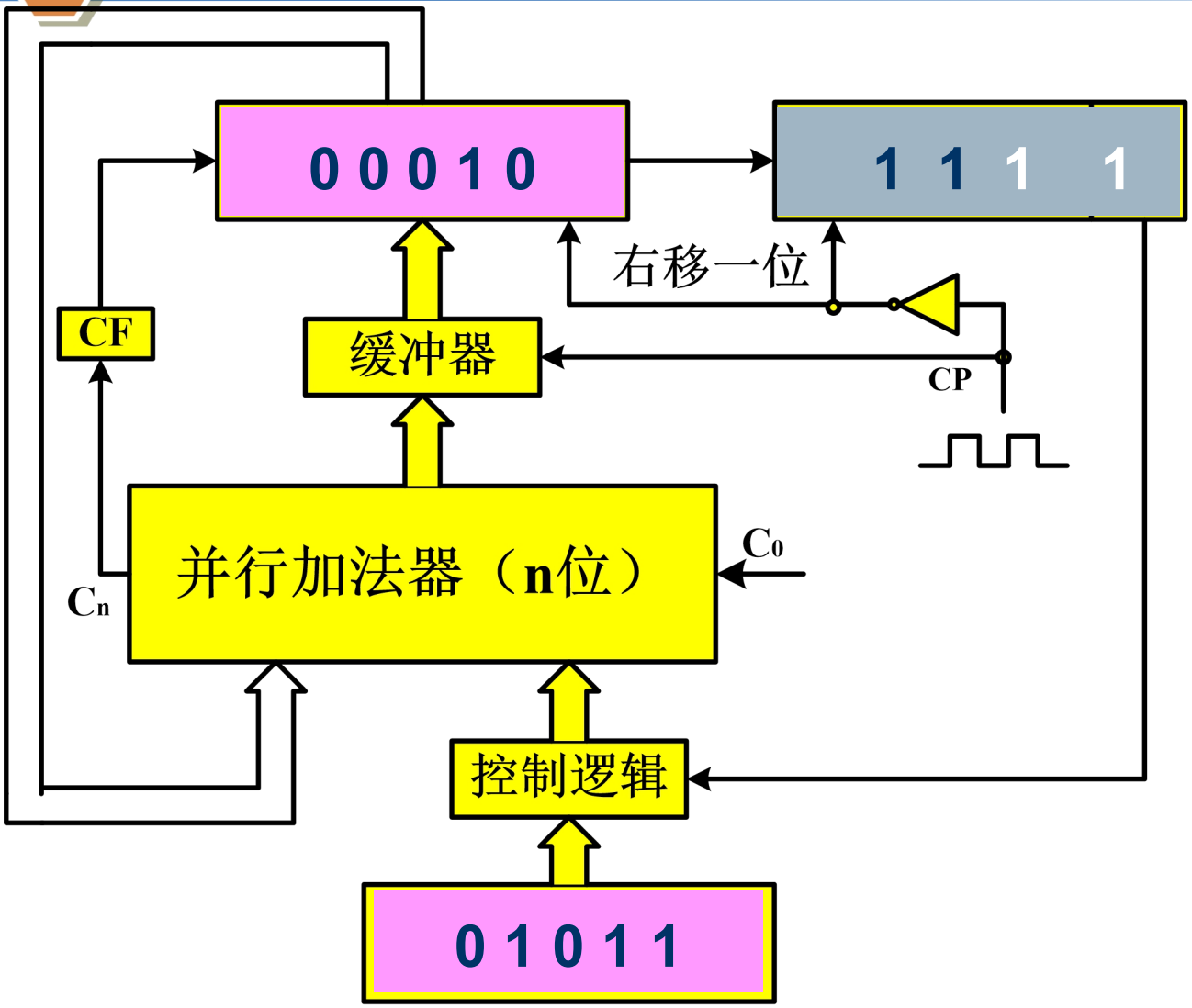
00000	1101
01011	1101
00101	1110
00101	1110



第二次求部分积

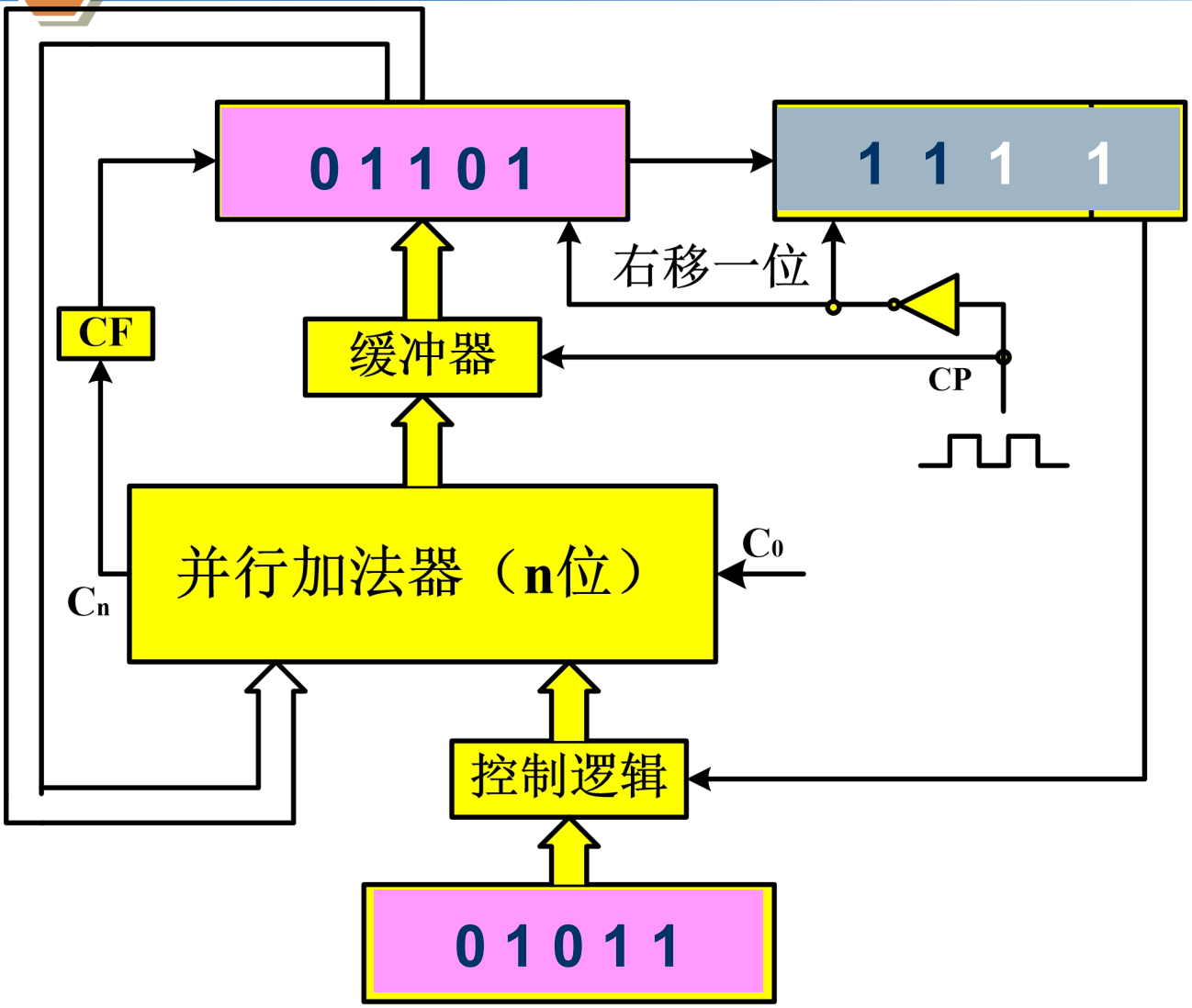
右移1位

00000	1101
01011	1101
00101	1110
00101	1110
00010	1111



第三次求部分积

加运算：＋|X|

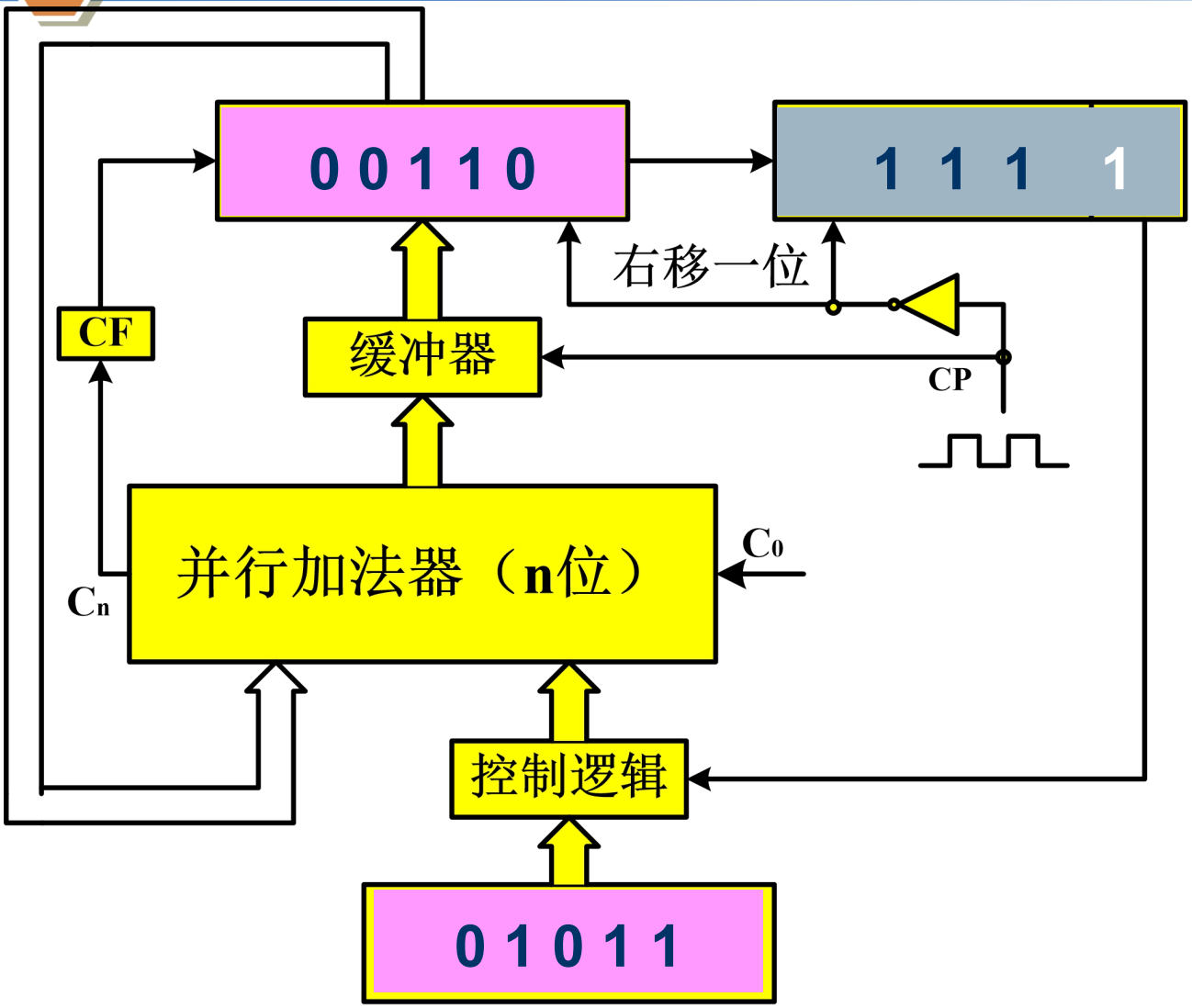


00000	1101
01011	1101
00101	1110
00101	1110
00010	1111
01101	1111

第三次求部分积

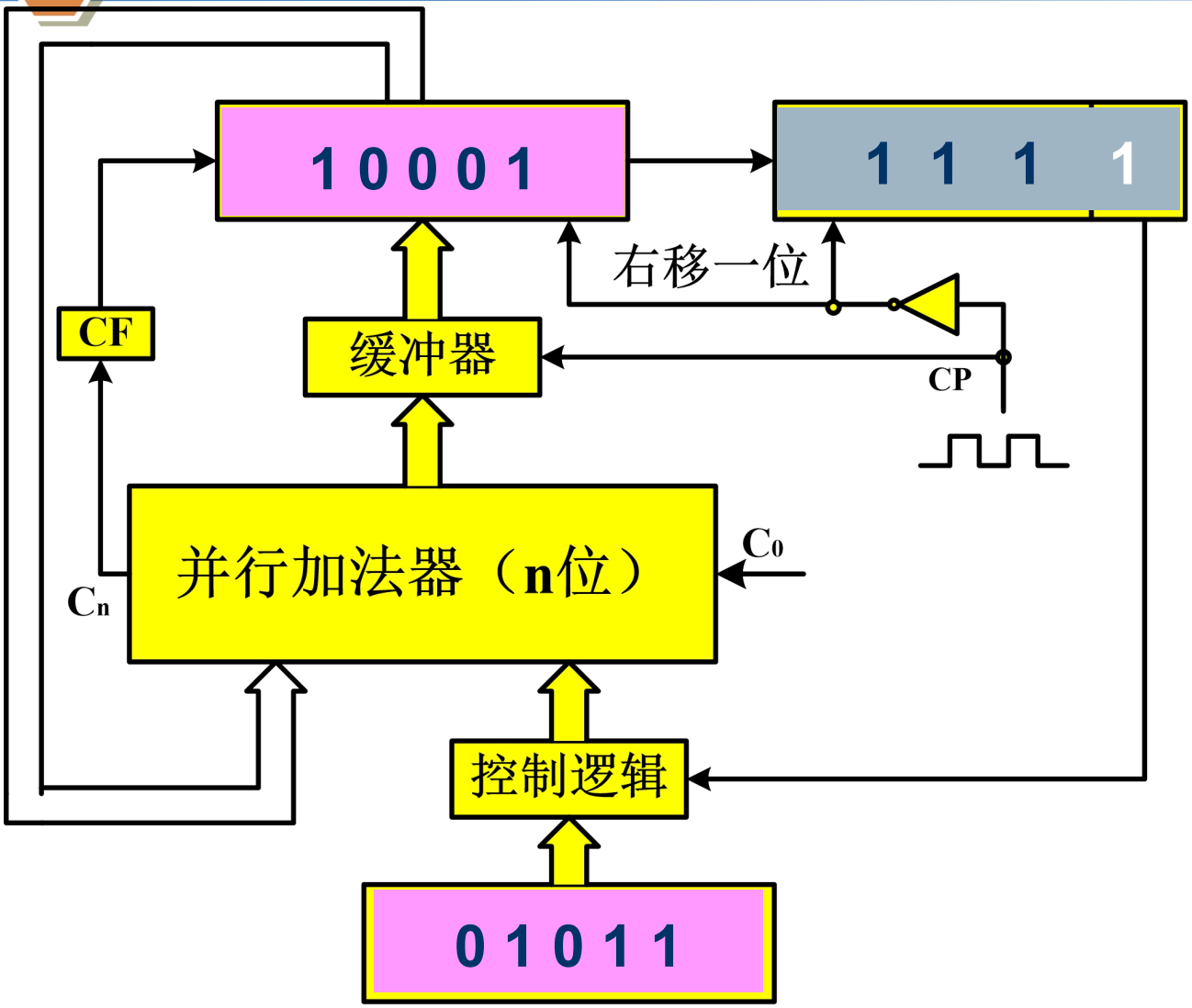
右移1位

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



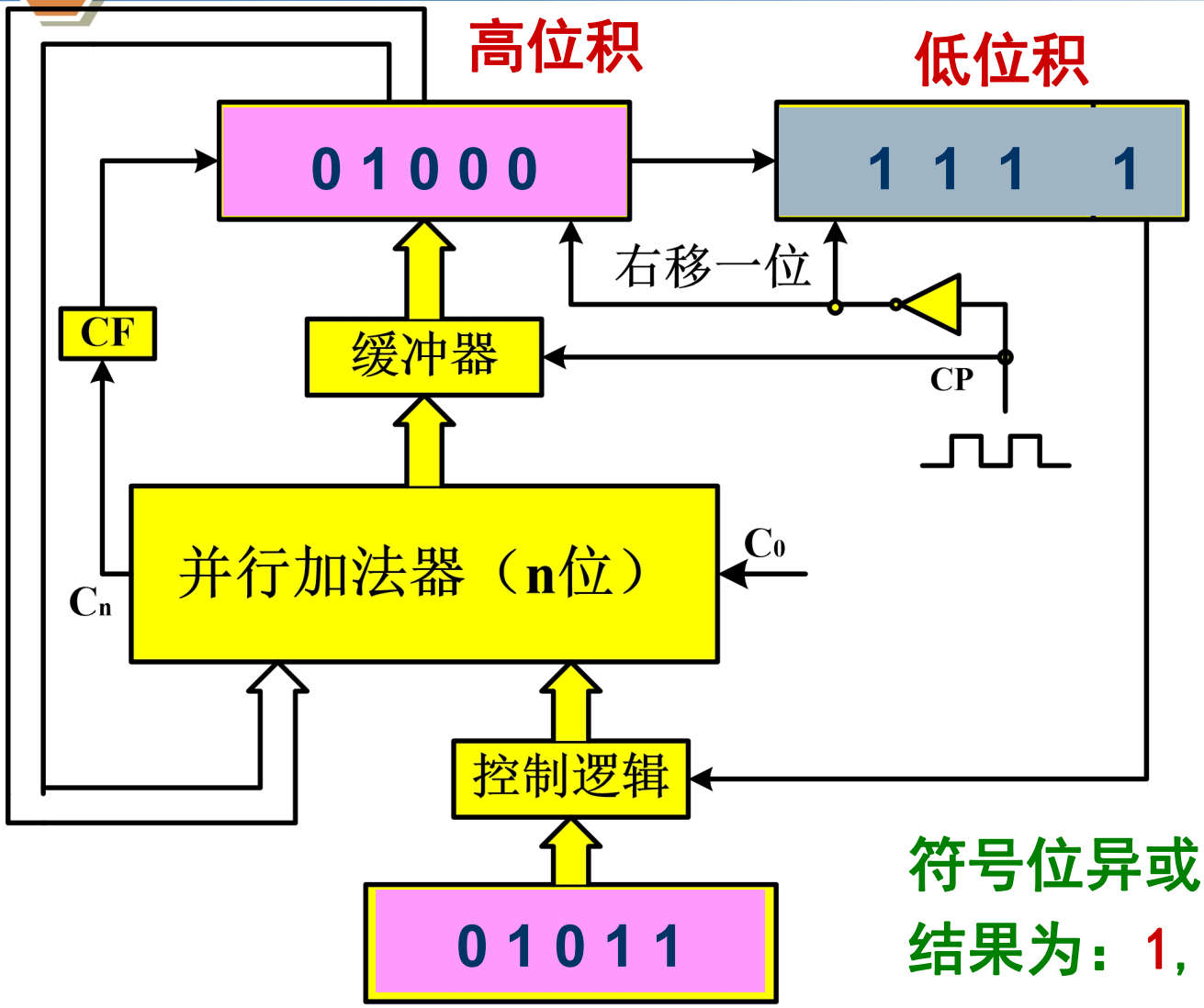
第四次求部分积

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

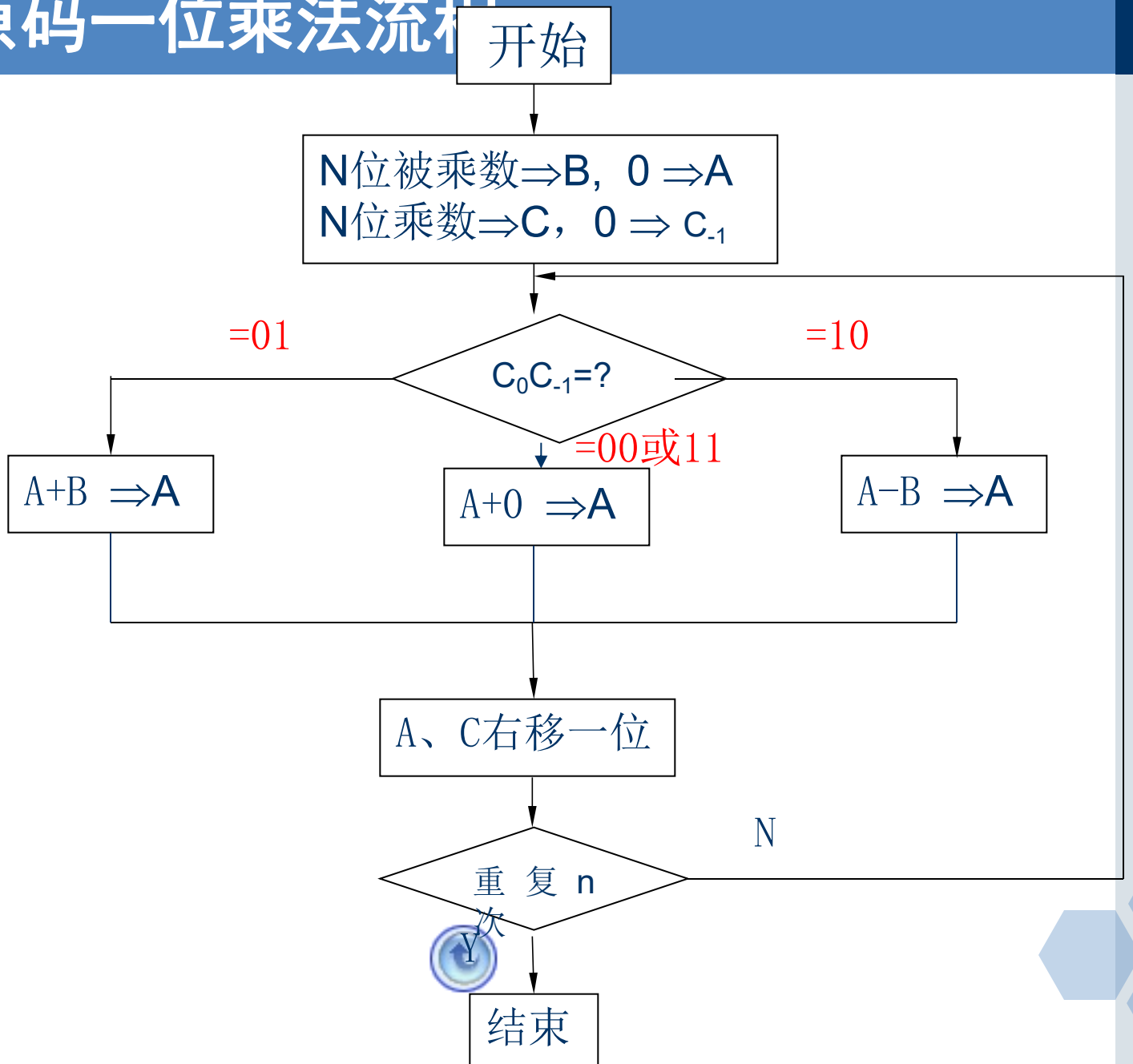


右移1位		
	00000	1101
	01011	1101
	00101	1110
	00101	1110
	00010	1111
	01101	1111
	00110	1111
	01000	1111

符号位异或

结果为：1, 10001111

原码一位乘法流程

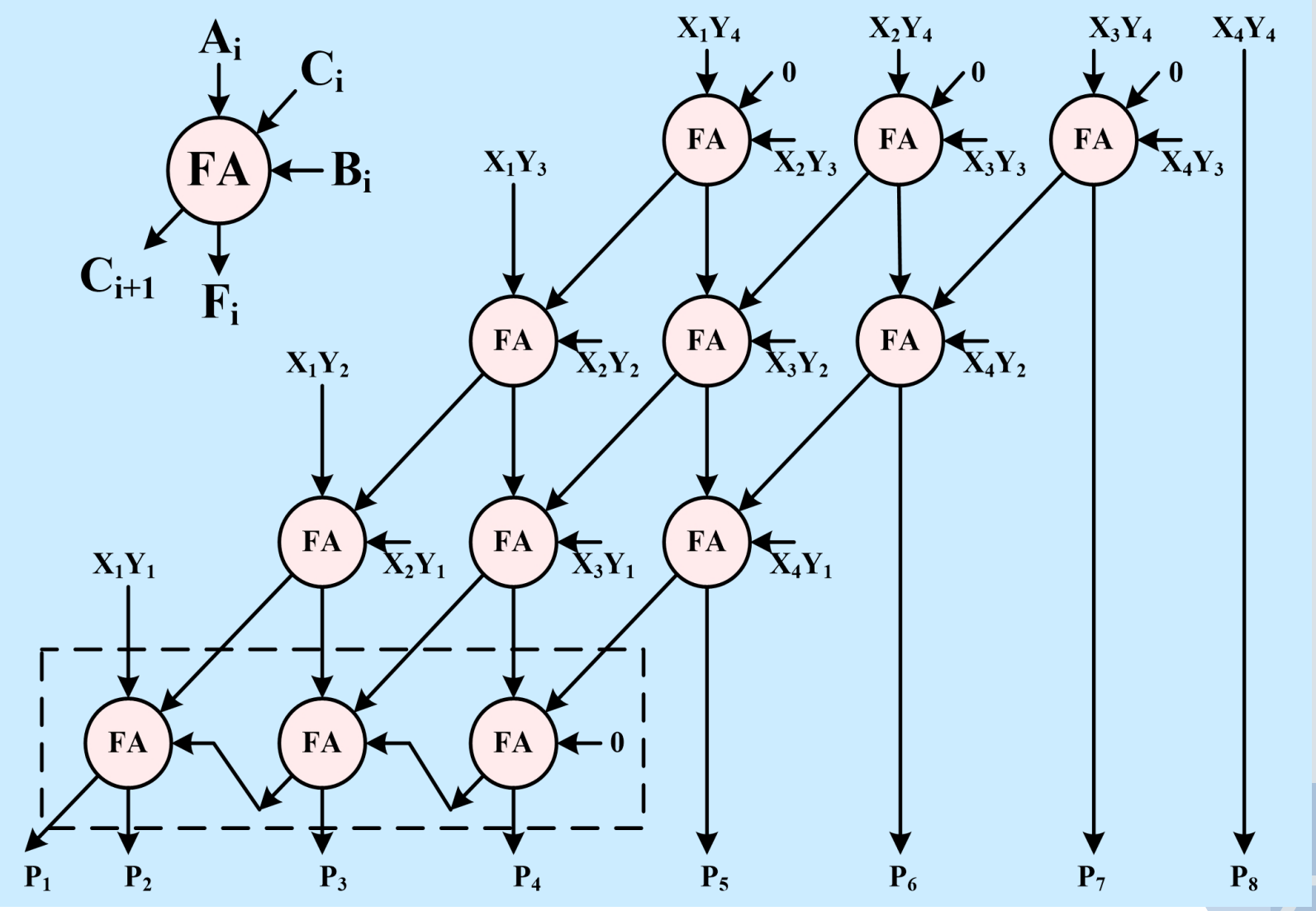

开始
N位被乘数⇒B，0 ⇒A
N位乘数⇒C，0 ⇒ C-1
C0C-1=?
=01
=10
=00或11
A+B ⇒A
A+0 ⇒A
A-B ⇒A
A、C右移一位
重 复 n 次
N
Y
结束

三、阵列乘法器

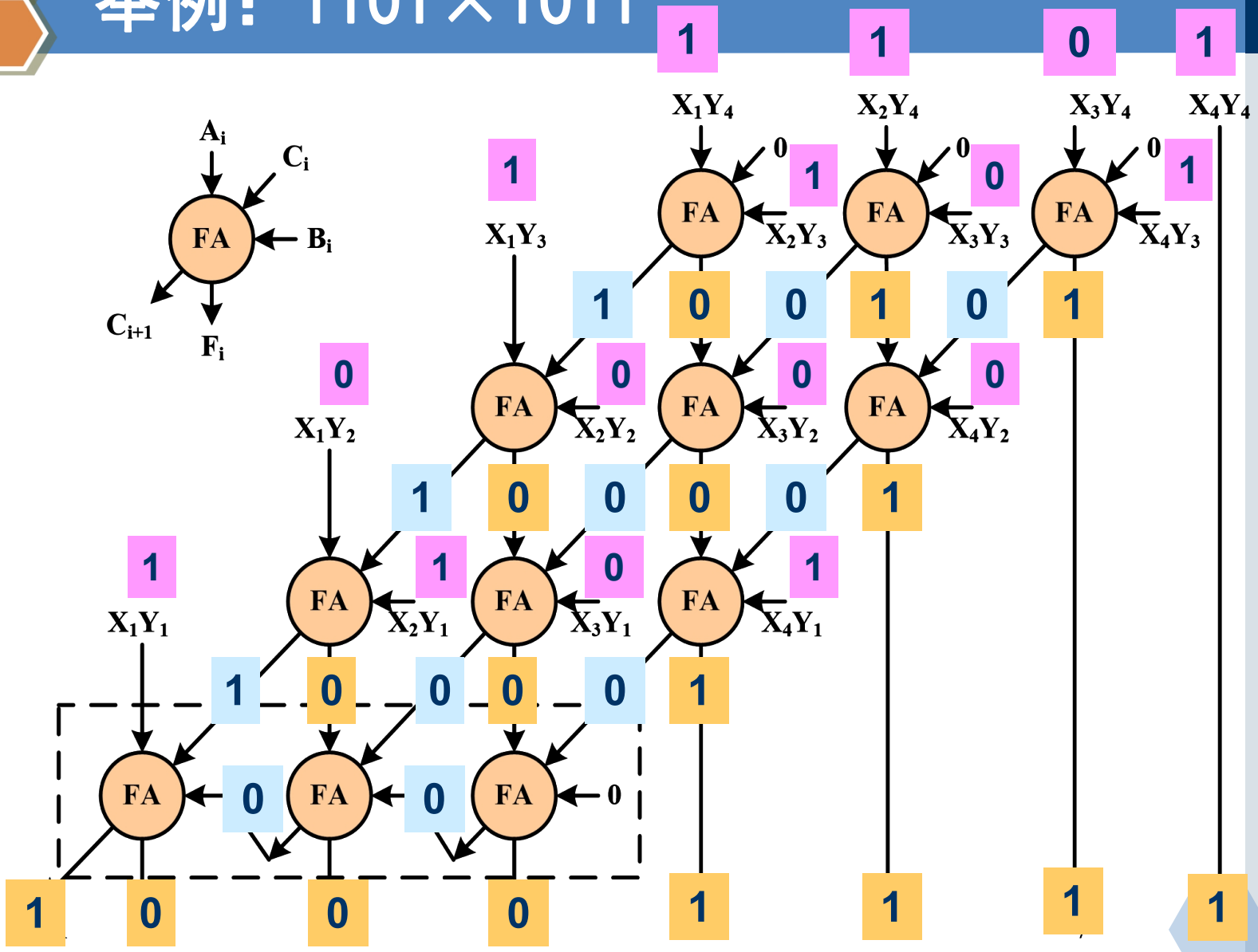
- 原理类似于二进制手工算法
  - 位积的每一位XiYj用一个与门产生。
  - 每一次的累加都用单独一组FA实现。
  - 本次累加的FA之间的进位，送至下一次再累加。

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

绝对值阵列乘法器


A_i
C_i
FA
B_i
C_{i+1}
F_i
X_1Y_4
X_2Y_4
X_3Y_4
X_4Y_4
0
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
P_1
P_2
P_3
P_4
P_5
P_6
P_7
P_8

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
FA
0
1

The End!