一、数值数据表示

1、进位计数制及其相互转换

2、机器码表示：

- 原码、反码、补码
- 定点数与浮点数
- IEEE 754 浮点数格式　（单精度）

- 要求：
  - ✓ 各种机器数与真值的互换表示
  - ✓ 机器数的表数范围

二、非数值数据的表示

1、字符的ASCII编码：编码及排列规则

2、汉字编码

3、位图：基本概念：用点阵表达图像，用一组0-1码数据描述

4、声音数据：采样、量化方法，技术参数

5、校验码：奇偶校验

声音数据的技术参数

- 采样频率：单位时间内的采样次数
  - 声卡一般提供11.025kHz，22.05kHz和44.1kHz等不同的采样频率。
  - 采样频率定律：高于信号频率的2倍（8KHz）。
  - 采样频率越大，采样点间隔越小，声音越逼真，但数据量越大。
- 采样位数：记录每次采样值数值大小的位数
  - 通常有8bits或16bits两种。
  - 位数越大，声音的变化度细腻，相应的数据量也越大。
- 采样声道数：单声道还是立体声。
  - 单声道：在声音处理过程中只有单数据流
  - 立体声：需要左、右声道的两个数据流。
- 数据量（字节/秒）= 采样频率(Hz)×采样位数(bit)×声道数÷ 8

三、补码加/减运算

1、补码加/减运算

X= + 1001　　Y= + 0101

```
    0 1 0 0 1
+   0 0 1 0 1
-------------
    0 1 1 1 0
```

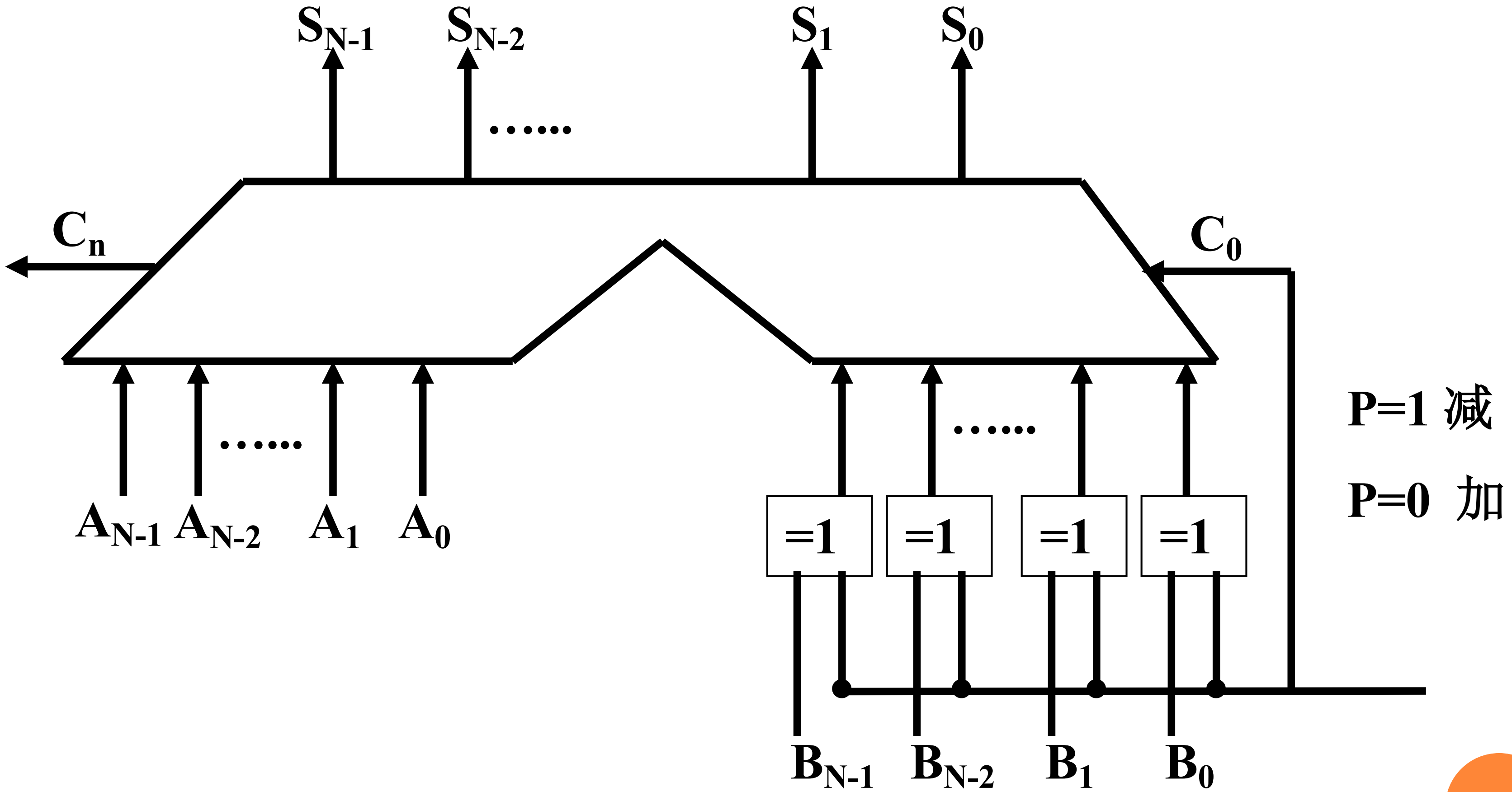
X= + 1011　　Y= - 0101

```
     0 1 0 1 1
+    1 1 0 1 1
--------------
   1 0 0 1 1 0
```

X= - 1001　　Y= - 0101

```
     1 0 1 1 1
+    1 1 0 1 1
--------------
   1 1 0 0 1 0
```

2、补码加/减法器



3、溢出检测

变形码检测

```
    0 0  1 1 0     +6
+   1 1  0 1 1     -5
-------------------
  1 0 0  0 0 1     +1
```

```
    1 1  1 1 0     -2
+   1 1  1 0 1     -3
-------------------
  1 1 1  0 1 1     -5
```

```
    0 0  1 1 0     +6
+   0 0  0 1 1     +3
-------------------
    0 1  0 0 1     +9
```

```
    1 1  1 0 0     -4
+   1 1  0 1 1     -5
-------------------
  1 1 0  1 1 1     -9
```

规则：两个符号参加运算，符号位为00或11，表示结果正确；

符号位为01，表示正溢；　　符号位为10，表示负溢。

实现逻辑：$V = S_{f1} \oplus S_{f2}$，

单符号位检测方法

异号相加，结果正确；

同号相加，结果的符号与操作数的不同，表示溢出；

或最高有效数字位、符号位不是同时产生进位，表示溢出。

$$\begin{array}{r} 0\ A_{n-1} \\ 0\ B_{n-1} \\ \hline 0\ S_{n-1} \end{array} \qquad \begin{array}{r} 0\ A_{n-1} \\ 0\ B_{n-1} \\ \hline 1\ S_{n-1} \end{array} \qquad \begin{array}{r} 1\ A_{n-1} \\ 1\ B_{n-1} \\ \hline \mathit{1}\ 0\ S_{n-1} \end{array} \qquad \begin{array}{r} 1\ A_{n-1} \\ 1\ B_{n-1} \\ \hline \mathit{1}\ 1\ S_{n-1} \end{array}$$

实现逻辑：

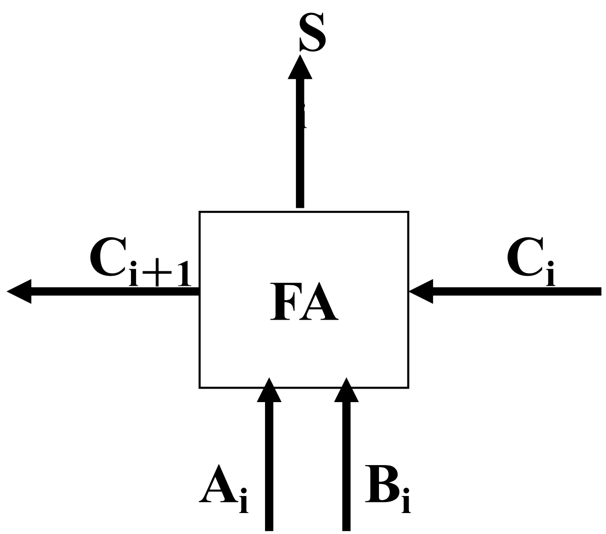
$$V = C_{N-1} \oplus C_N$$

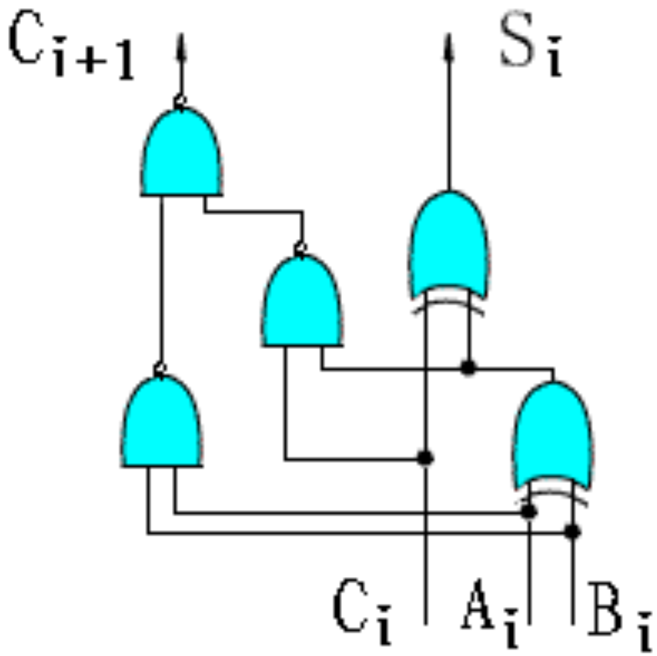
4、一位全加器FA

$$S_i = A_i \oplus B_i \oplus C_i$$

$$C_{i+1} = A_iB_i + A_iC_i + B_iC_i$$

$$= A_iB_i + (A_i \oplus B_i)C_i$$

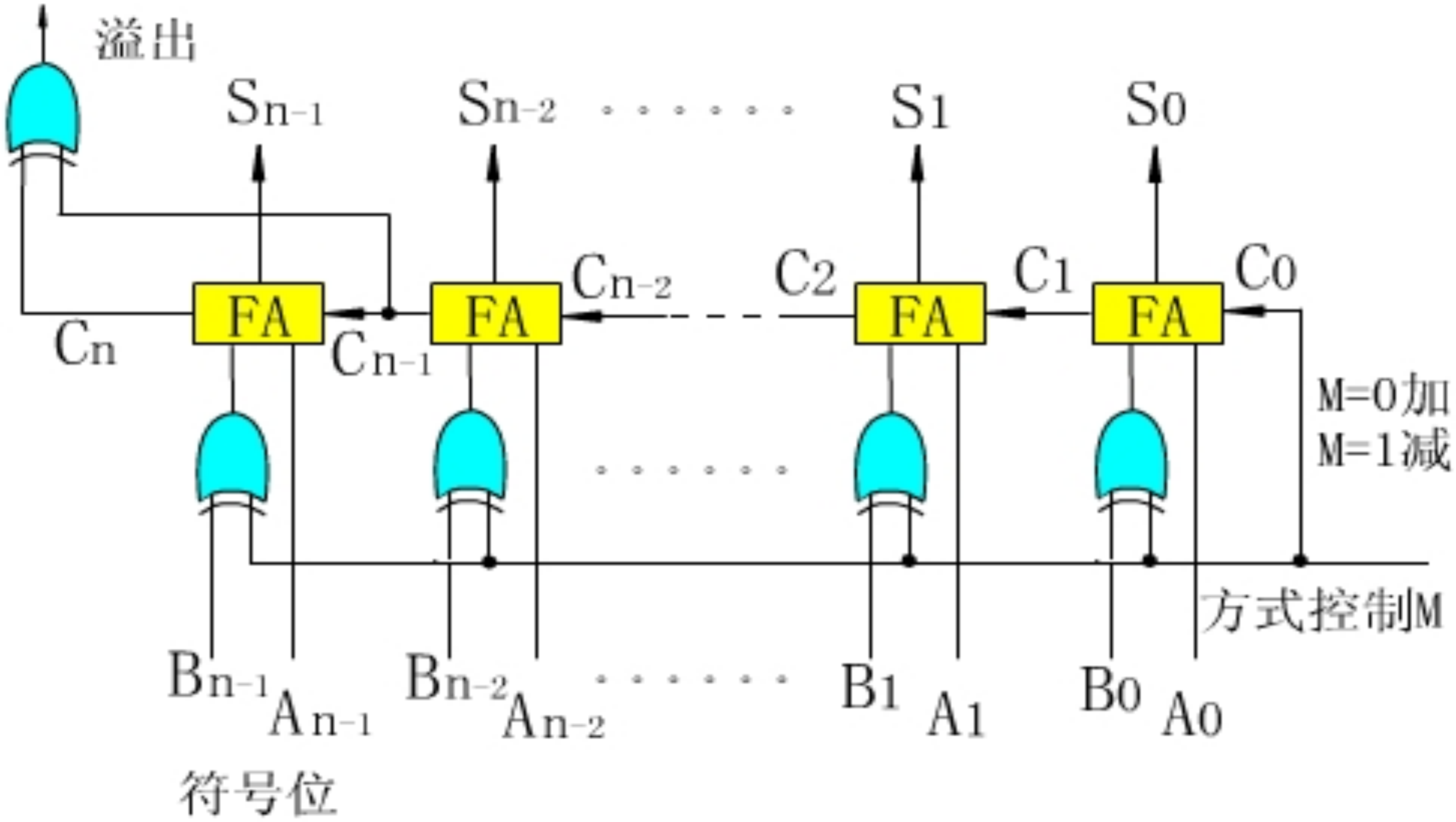




(a)

FA（全加器）逻辑电路图

5、N位行波进位加法器



行波进位的补码加法/减法器

四、定点乘法运算

1、串行原码1位乘算法

设 x ＝ 1101，y ＝ 1011　求x*y

	部分积	乘数	
			部分积初始化为0
	0 0 0 0 0	0 1 0 1 1	
+X	0 1 1 0 1		乘数最低位为1，加上被乘数
	0 1 1 0 1	0 1 0 1 1	部分积右移，前面补0
	0 0 1 1 0	1 0 1 0 1	乘数最低位为1，加上被乘数
+X	0 1 1 0 1		
	1 0 0 1 1	1 0 1 0 1	部分积右移，前面补0
	0 1 0 0 1	1 1 0 1 0	乘数最低位为0，加上0
+0	0 0 0 0 0		
	0 1 0 0 1	1 1 0 1 0	部分积右移，前面补0
	0 0 1 0 0	1 1 1 0 1	乘数最低位为1，加上被乘数
+X	0 1 1 0 1		
	1 0 0 0 1	1 1 1 0 1	部分积右移，前面补0
	0 1 0 0 0	1 1 1 1 0	运算四次结束，数值部分运算

x*y= 10001111

2、不带符号位的并行阵列乘法器

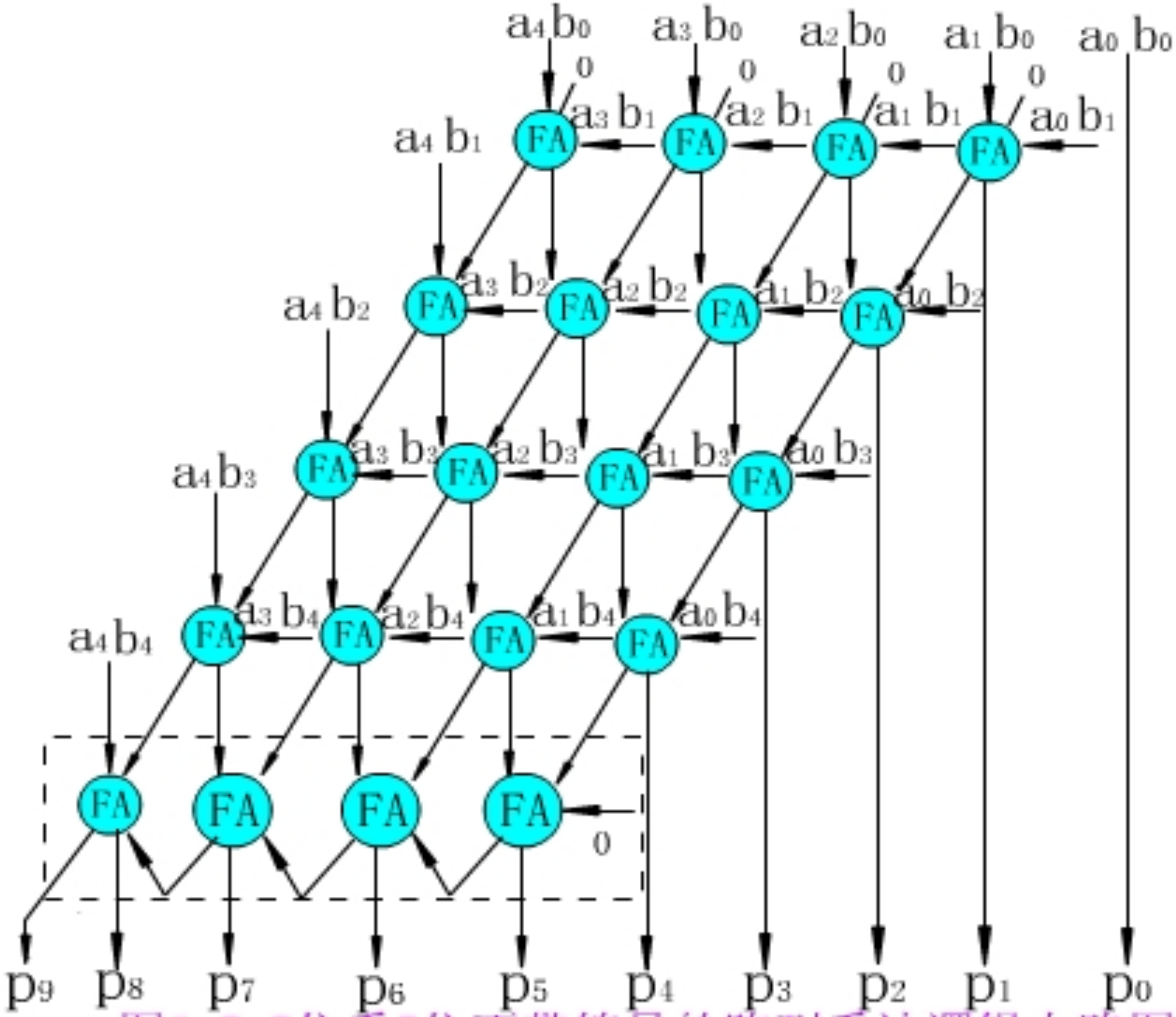


图2.5 5位乘5位不带符号的阵列乘法逻辑电路图

■纵向传递本位和

■斜向 转递进位

$$\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(2^{i+j})a_i b_j$$

五、定点除法运算

1、恢复余数法

X=00.1001　　　Y=00.1011　　　[-Y]补＝11.0101

例：

	被除数/余数 X　R	商数 Q
	00.1001	
+[-Y]补	11.0101	
	11.1110	
+[Y]补	00.1011	
	00.1001	0
	01.0010	
+[-Y]补	11.0101	
	00.0111	0.1
	00.1110	
+[-Y]补	11.0101	
	00.0011	0.11

例：

	被除数/余数	商数
	X　R	Q
	00.0110	
$+[-Y]_{补}$	11.0101	
	11.1011	
$+[Y]_{补}$	00.1011	
	00.0110	0.110
	00.1100	
$+[-Y]_{补}$	11.0101	
	00.0001	0.1101

算法缺点：计算步骤不确定，难于控制。

2、不恢复余数法

- 试商： $R_i = 2R_{i-1} - Y$

  若 $R_i > 0$ 正确 $R_{i+1} = 2R_i - Y$

  若 $R_i < 0$ 失败 $R_i' = R_i + Y$

  $R_{i+1} = 2R_i' - Y = 2(R_i + Y) - Y$

  $= 2R_i + Y$

- 规则：$R_i = 2R_{i-1} - Y$

  若 $R_i > 0$ 商1 $R_{i+1} = 2R_i - Y$

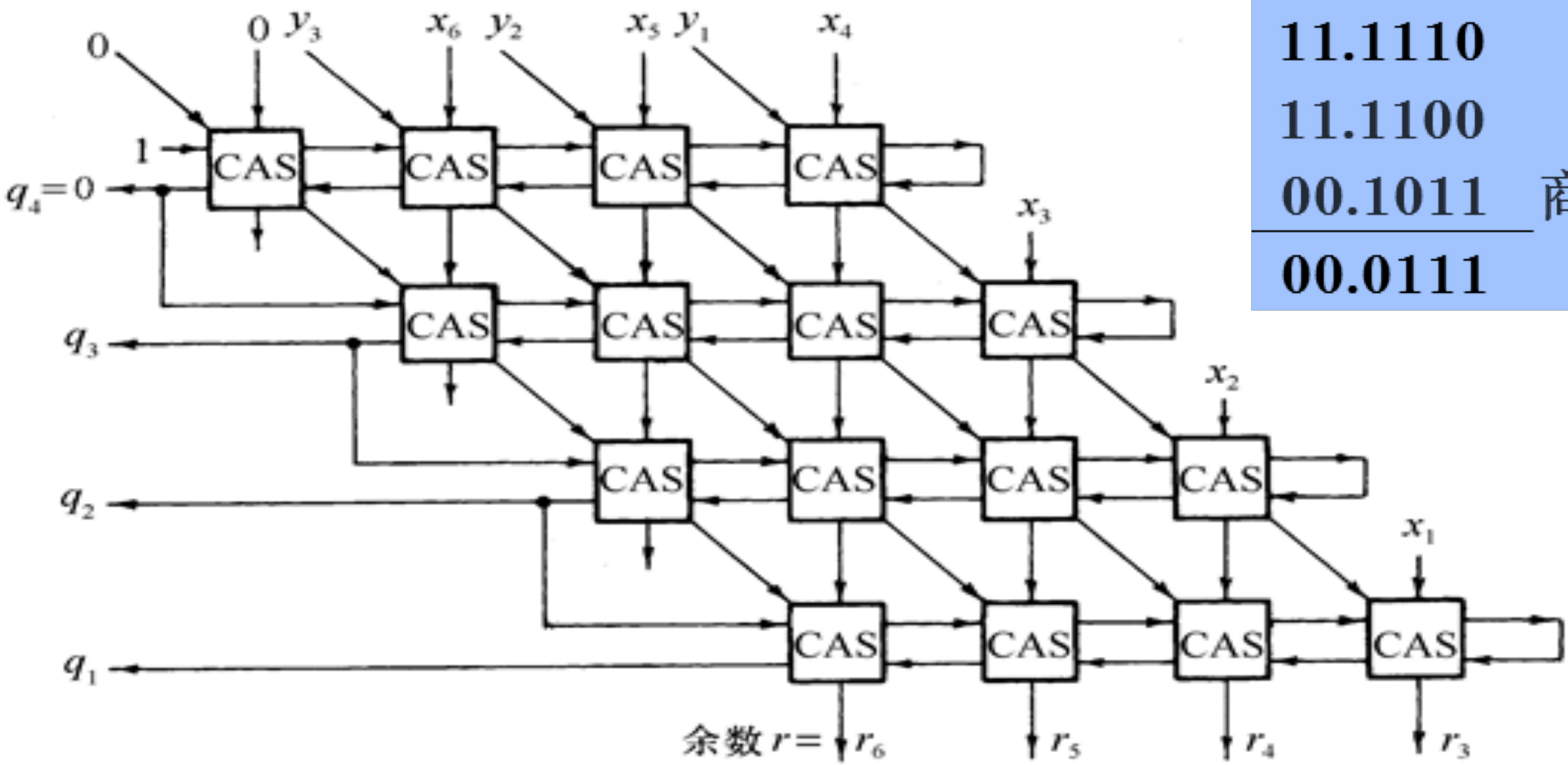
  若 $R_i < 0$ 商0 $R_{i+1} = 2R_i + Y$

例： 被除数/余数 商数

$X=00.1001$
$Y=00.1011$
$[-Y]_{补}=11.0101$

X / R	Q
00.1001	
11.0101	(起步做减法)
11.1110	
11.1100	0(左移做加法)
00.1011	
00.0111	
00.1110	0.1 (左移做减法)
11.0101	
00.0011	
00.0110	0.11 (左移做减法)
11.0101	
11.1011	
11.0110	0.110 (左移做加法)
00.1011	
00.0001	0.1101(停止运算)

3、不恢复余数法的阵列除法器



00.1001
11.0101　商0

11.1110
11.1100
00.1011　商1

00.0111

- 垂直线：被除数（中间余数）；　斜线：除数；　水平线：加减控制线P
- 首行：P=1，做减法【 $0.x_6x_5x_4x_3x_2x_1 - 0.y_3y_2y_1$ 】；
- 后续，由符号位进位决定q值（商）和下一行的p值（控制参数）：
  - 有进位：p=q=1 【减法】；符号位不进位：p=q=0 【加法】

[例23] $x=0.101001$, $y=0.111$, 求 $x\div y$。

解:

$[-y]_补=1.001$　　除数右移

被除数 x	0.1 0 1 0 0 1		
减 y	1.0 0 1		
余数为负	1.1 1 0 0 0 1	<0	无进位 $q0=0$
加 y	0.0 1 1 1		
余数为正	0.0 0 1 1 0 1	>0	有进位 $q1=1$
减 y	1.1 1 0 0 1		
余数为负	1.1 1 1 1 1 1	<0	无进位 $q2=0$
加 y	0.0 0 0 1 1 1		
余数为正	0.0 0 0 1 1 0	>0	无进位 $q3=1$

故得　　商 $q=q0.q1q2q3=0.101$

余数 $r=(0.00r3r4r5r6)=0.000110$

六、先行进位ALU原理

1、先行进位（4位）基本原理

$$C_{n+1} = Y_0 + X_0 C_n$$

$$C_{n+2} = Y_1 + X_1 C_{n+1} = Y_1 + X_1 (Y_0 + X_0 C_n)$$

$$= Y_1 + Y_0 X_1 + X_0 X_1 C_n$$

$$C_{n+3} = Y_2 + X_2 C_{n+2} = Y_2 + X_2 (Y_1 + Y_0 X_1 + X_0 X_1 C_n)$$

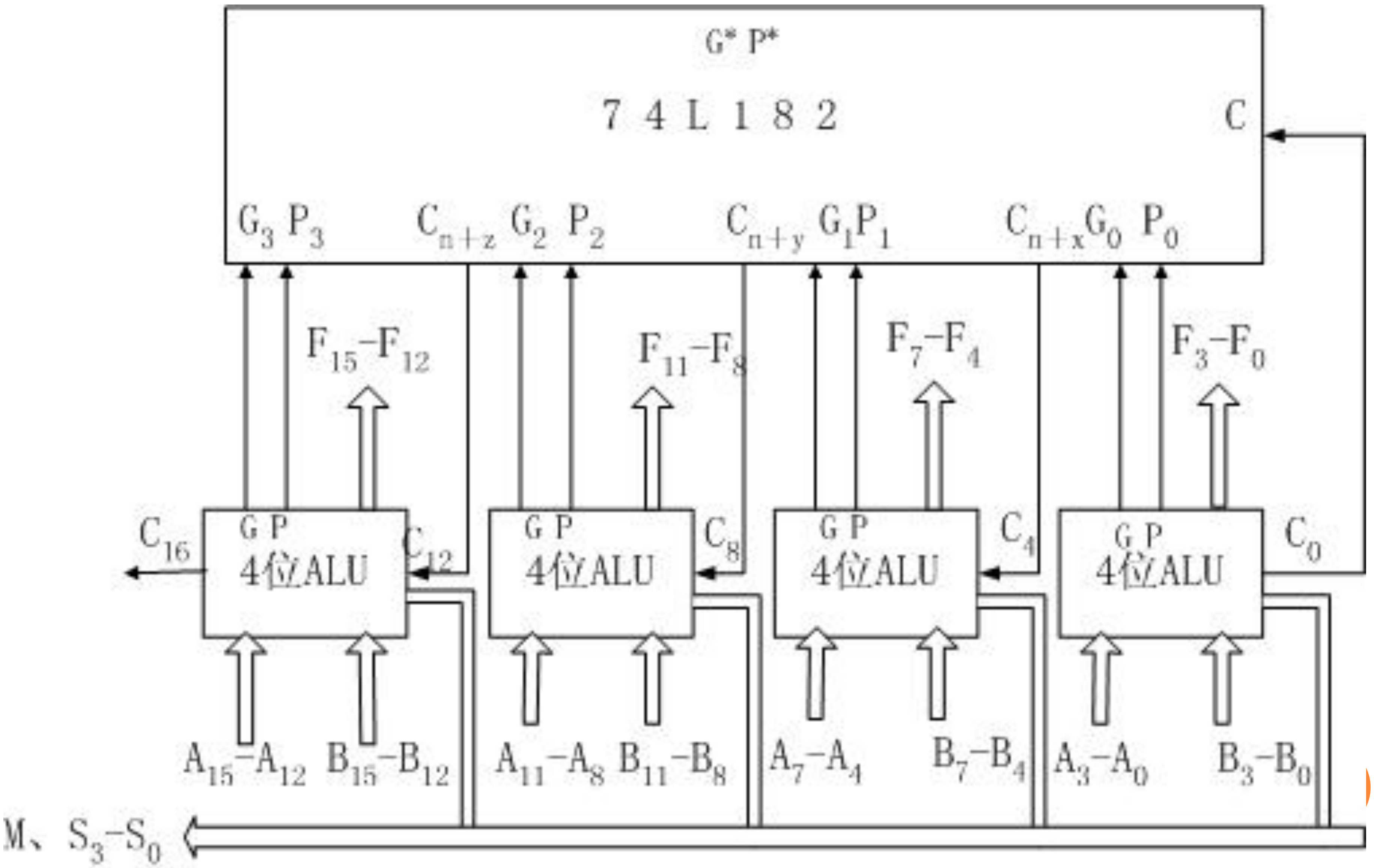
$$= Y_2 + Y_1 X_2 + Y_0 X_1 X_2 + X_0 X_1 X_2 C_n$$

$$C_{n+4} = Y_3 + X_3 C_{n+3} = Y_3 + X_3 (Y_2 + Y_1 X_2 + Y_0 X_1 X_2 + X_0 X_1 X_2 C_n)$$

$$= Y_3 + Y_2 X_3 + Y_1 X_2 X_3 + Y_0 X_1 X_2 X_3 + X_0 X_1 X_2 X_3 C_n$$

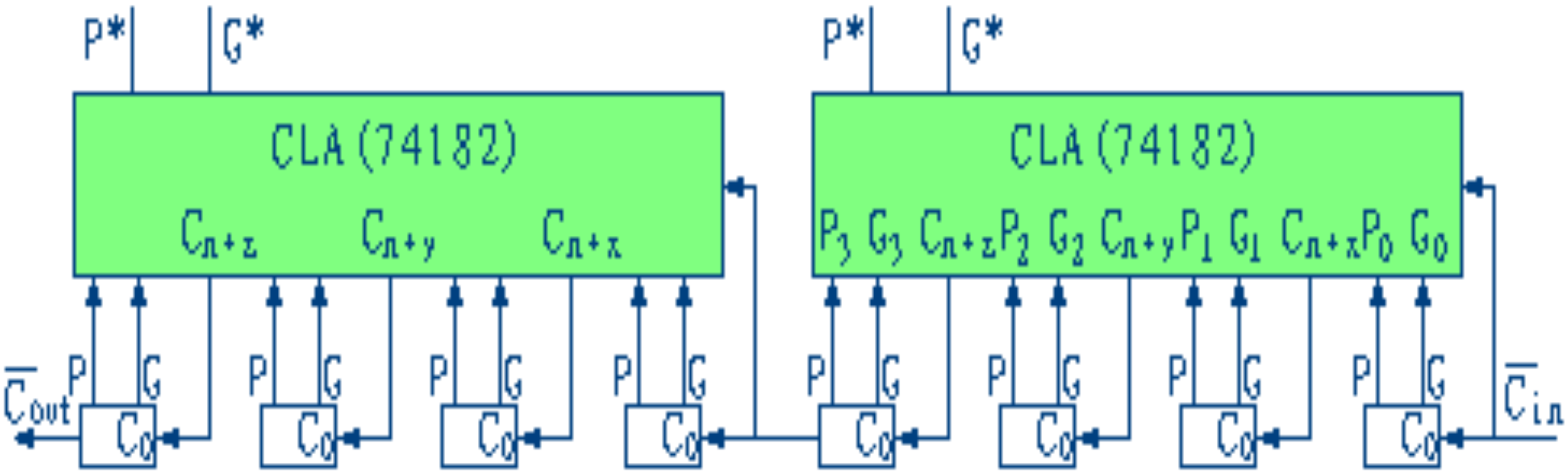
可以在相同的时间延迟内，同时获得 C_{n+1}、 C_{n+2}、 C_{n+3}、 C_{n+4}

2、16位二级先行进位ALU逻辑框图



3、32位ALU逻辑方框图

- 二个“74182” CLA器件
- 八个“74181” 4位ALU



4、64位组间先行进位ALU

- 5个“74182”CLA器件
- 16个“74181”4位ALU