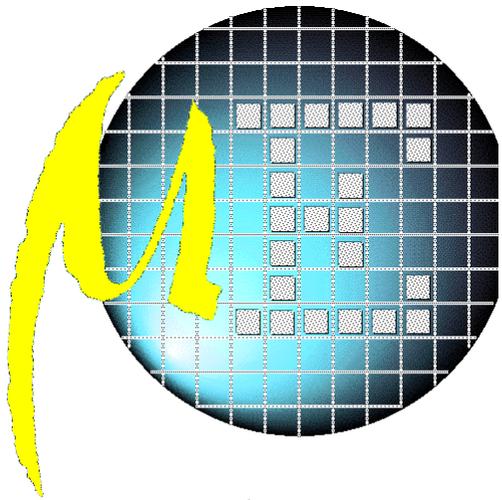


# Multiplication sur Silicium



DEA MICROÉLECTRONIQUE

Alain GUYOT

TIMA



☎ (33) 04 76 57 46 16

💻 Alain.Guyot@imag.fr

<http://tima-cmp.imag.fr/~guyot>

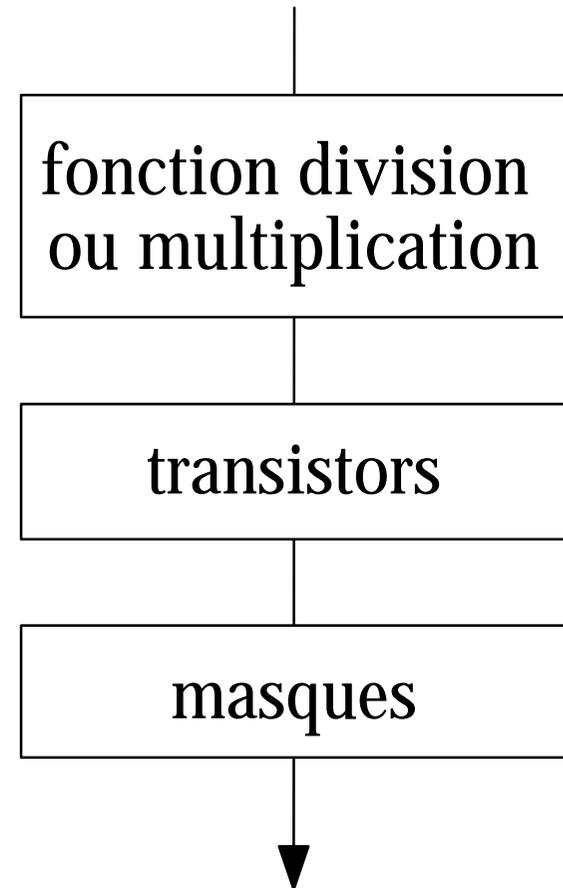
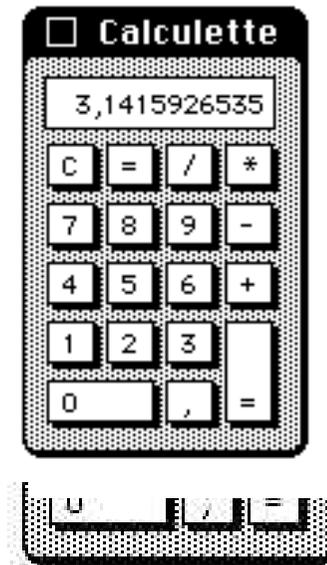
Techniques de l'Informatique et de la Microélectronique  
pour l'Architecture. Unité associée au C.N.R.S. n° B0706

But  
Réaliser des multiplieurs et des diviseurs

Optimiser la surface et/ou la vitesse

Problèmes

- Propagation de la retenue



# Généralités

Soient  $A = \sum_{i=0}^{n-1} a_i * 2^i$  ;  $B = \sum_{j=0}^{n-1} b_j * 2^j$  avec  $a_i, b_i \in \{0,1\}$ .

Le produit  $P = A * B$  s'écrit  $\left( \sum_{i=0}^{n-1} a_i * 2^i \right) * \left( \sum_{j=0}^{n-1} b_j * 2^j \right)$ ,

ou encore en appliquant la distributivité somme / produit

$$P = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} \left( a_i * b_j * 2^{i+j} \right)$$

Or  $A < 2^n$  et  $B < 2^n$  impliquent que  $P < 2^{2n}$ ,

Par conséquent  $P$  peut être écrit avec  $2n$  bits.

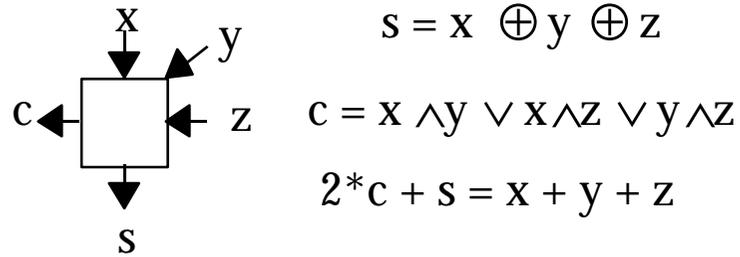
Le problème est alors de réduire la somme pondérée

de  $n^2$  bits en la somme pondérée de  $2n$  bits.

Cette réduction peut être faite

en temps  $O(n)$ ,  $O(\sqrt[2]{n})$ ,  $O(\log_{3/2} n)$ ,  $O(\log_2 n)$

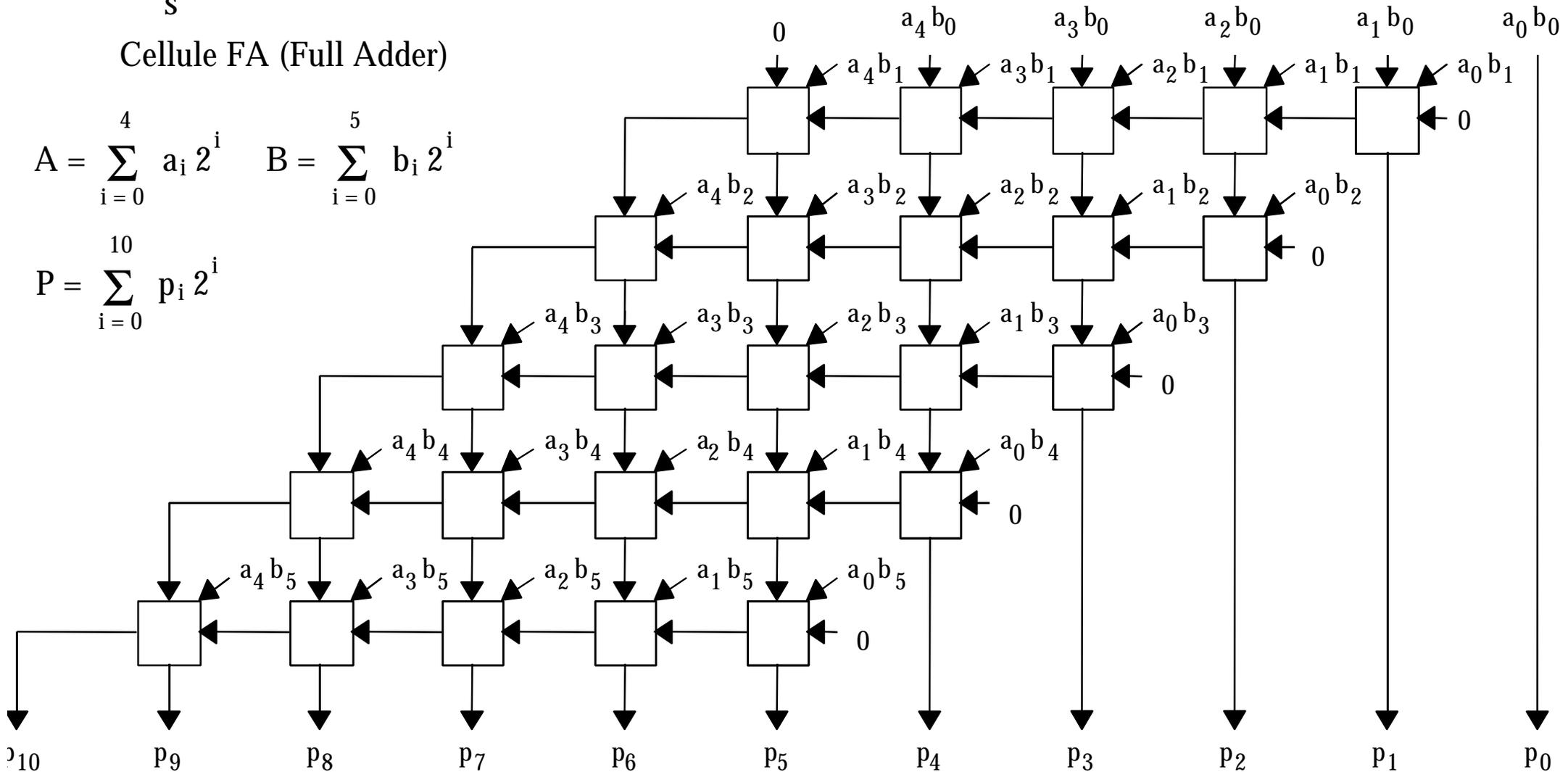
# Multiplieur naïf



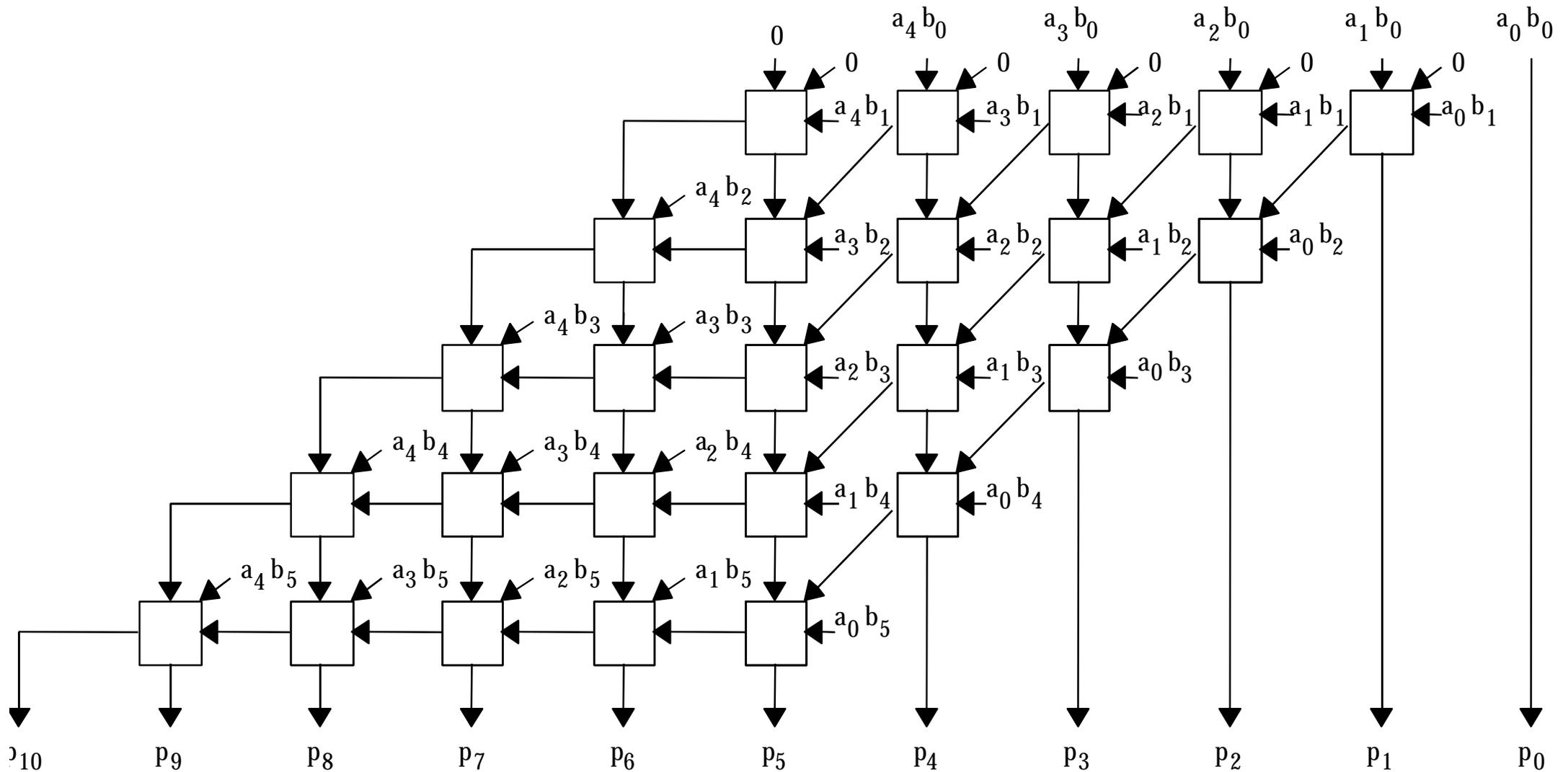
Cellule FA (Full Adder)

$$A = \sum_{i=0}^4 a_i 2^i \quad B = \sum_{i=0}^5 b_i 2^i$$

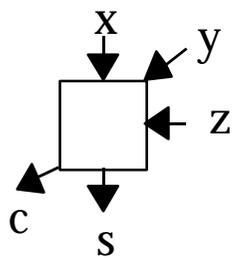
$$P = \sum_{i=0}^{10} p_i 2^i$$



# Multiplieur amélioré



# Multiplieur de Braun

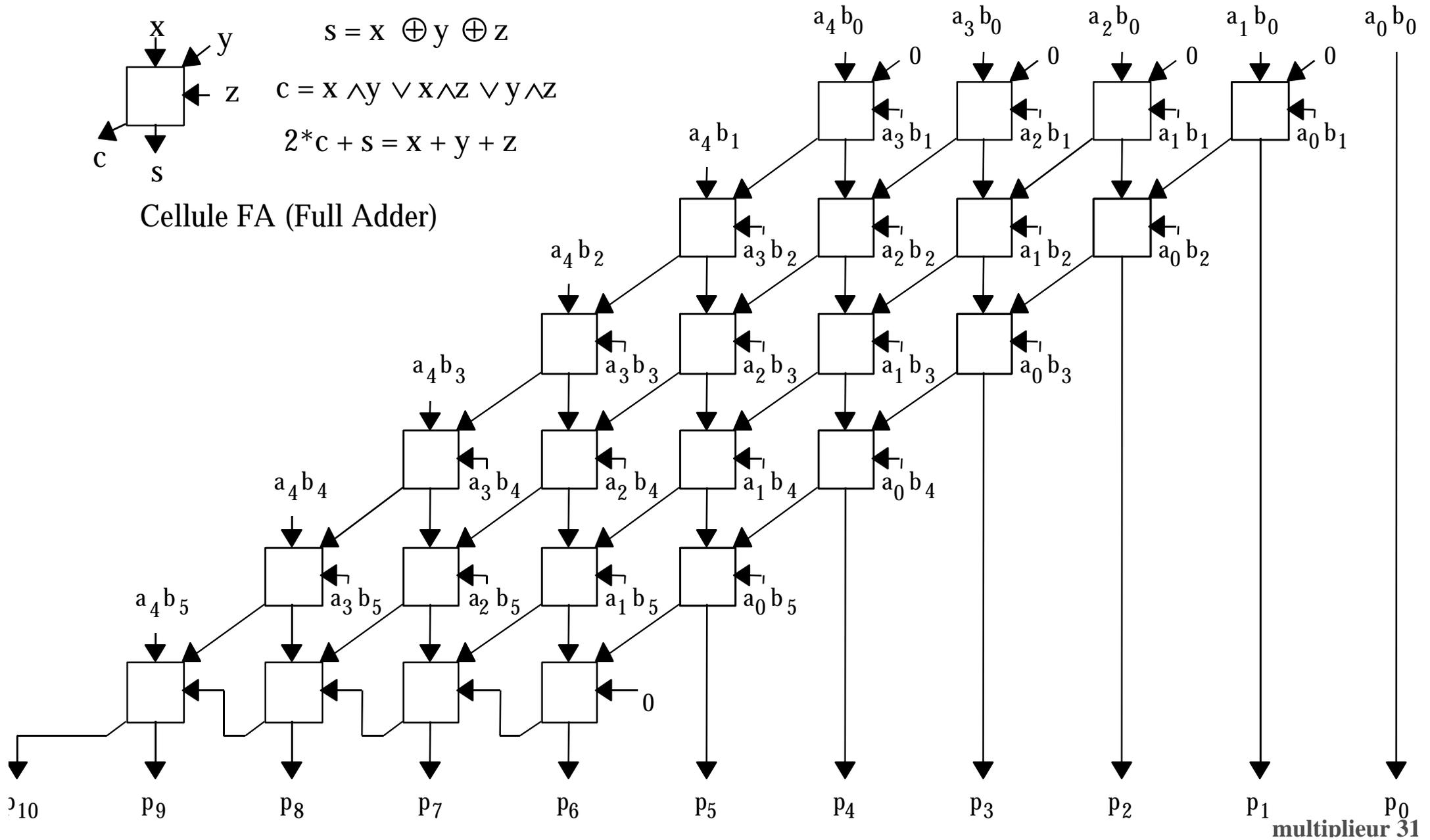


$$s = x \oplus y \oplus z$$

$$c = x \wedge y \vee x \wedge z \vee y \wedge z$$

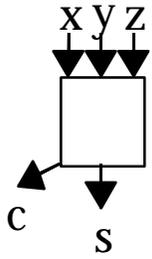
$$2 * c + s = x + y + z$$

Cellule FA (Full Adder)



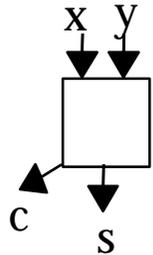
multiplieur 31

# Arbre de Wallace



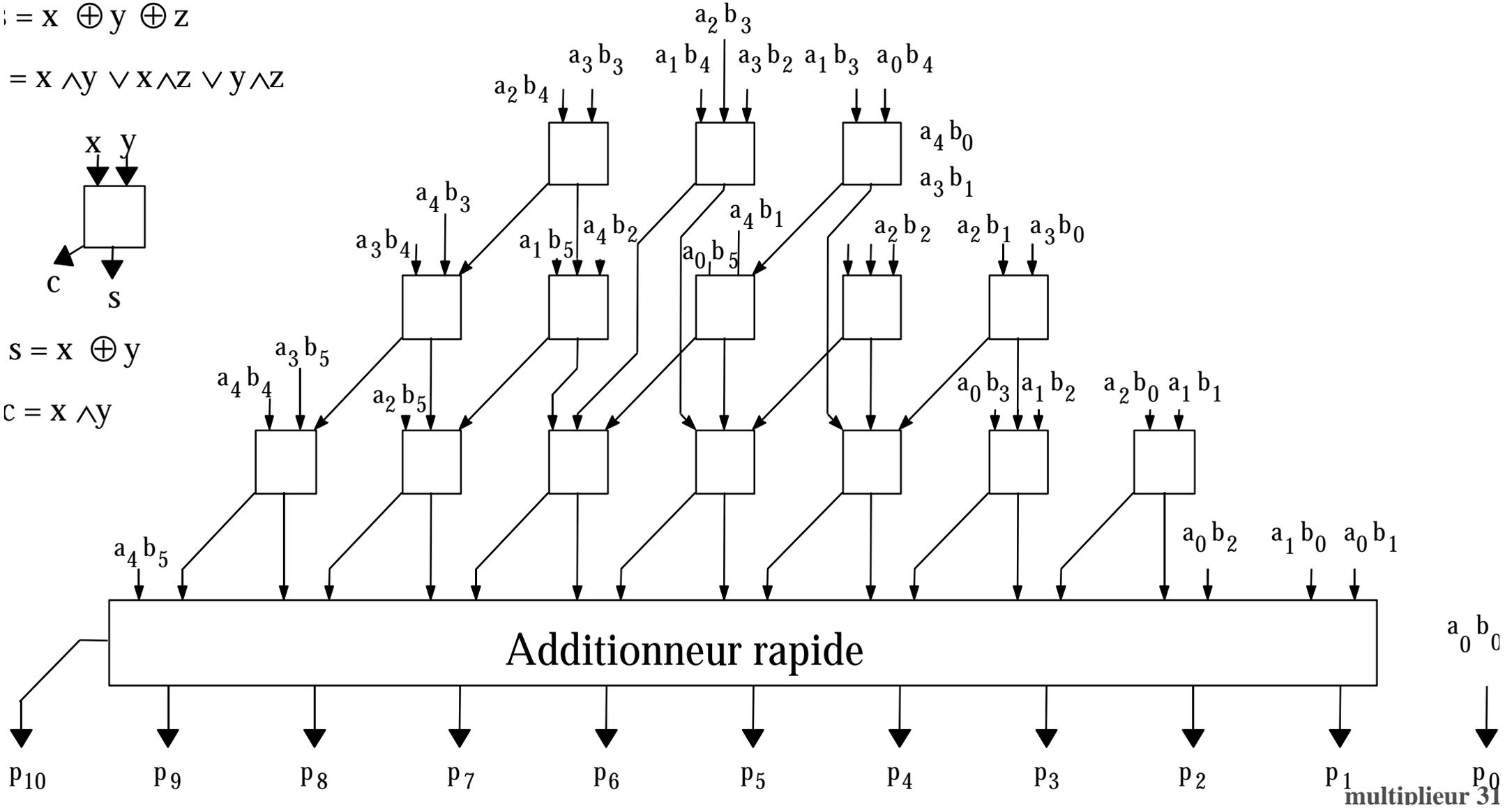
$$s = x \oplus y \oplus z$$

$$c = x \wedge y \vee x \wedge z \vee y \wedge z$$



$$s = x \oplus y$$

$$c = x \wedge y$$

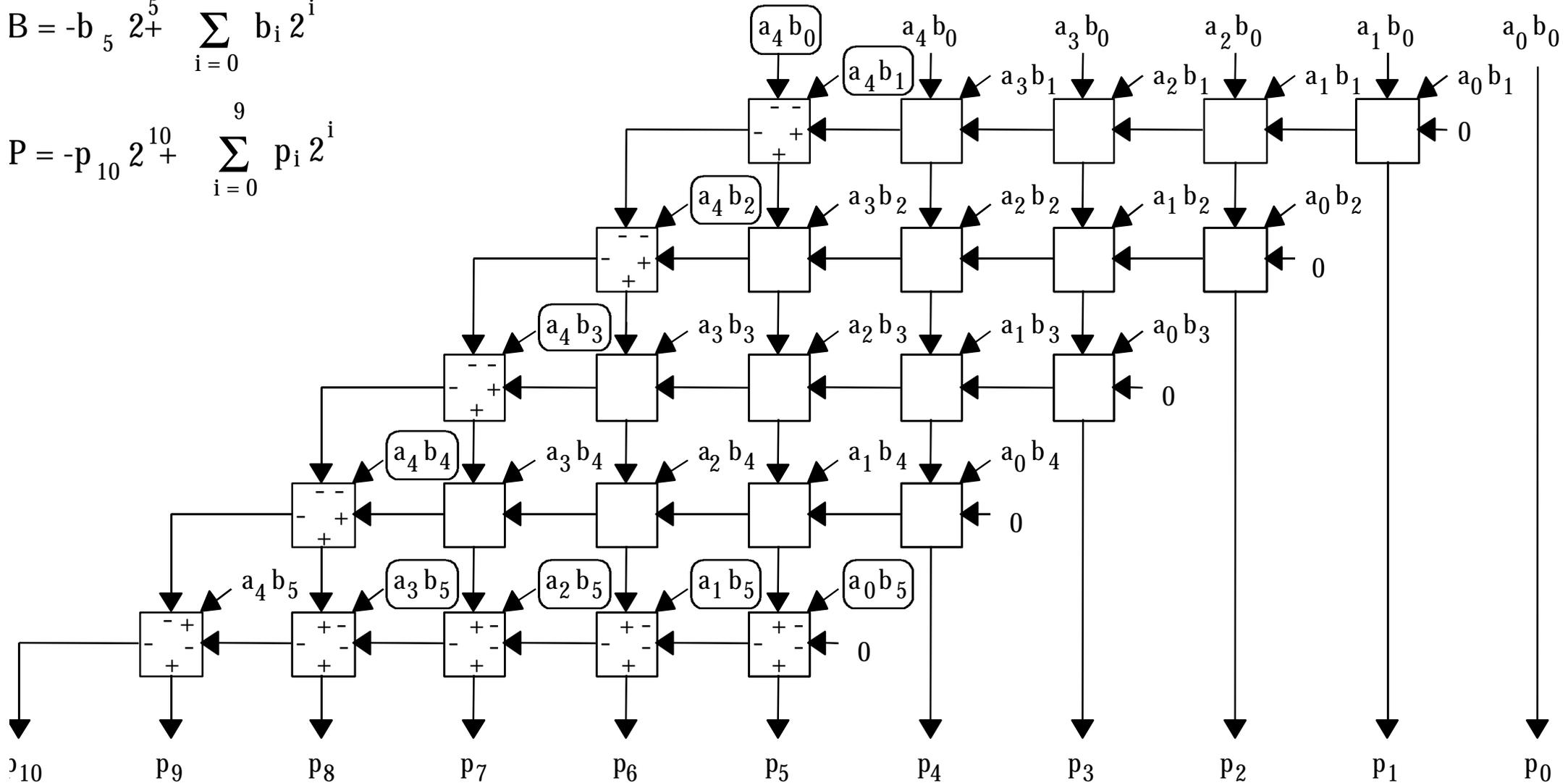


# Multiplieur signé

$$A = -a_4 2^4 + \sum_{i=0}^3 a_i 2^i$$

$$B = -b_5 2^5 + \sum_{i=0}^4 b_i 2^i$$

$$P = -p_{10} 2^{10} + \sum_{i=0}^9 p_i 2^i$$



# Multiplieur de Baugh-Wooley (1)

$$A = -a_4 2^4 + \sum_{i=0}^3 a_i 2^i$$

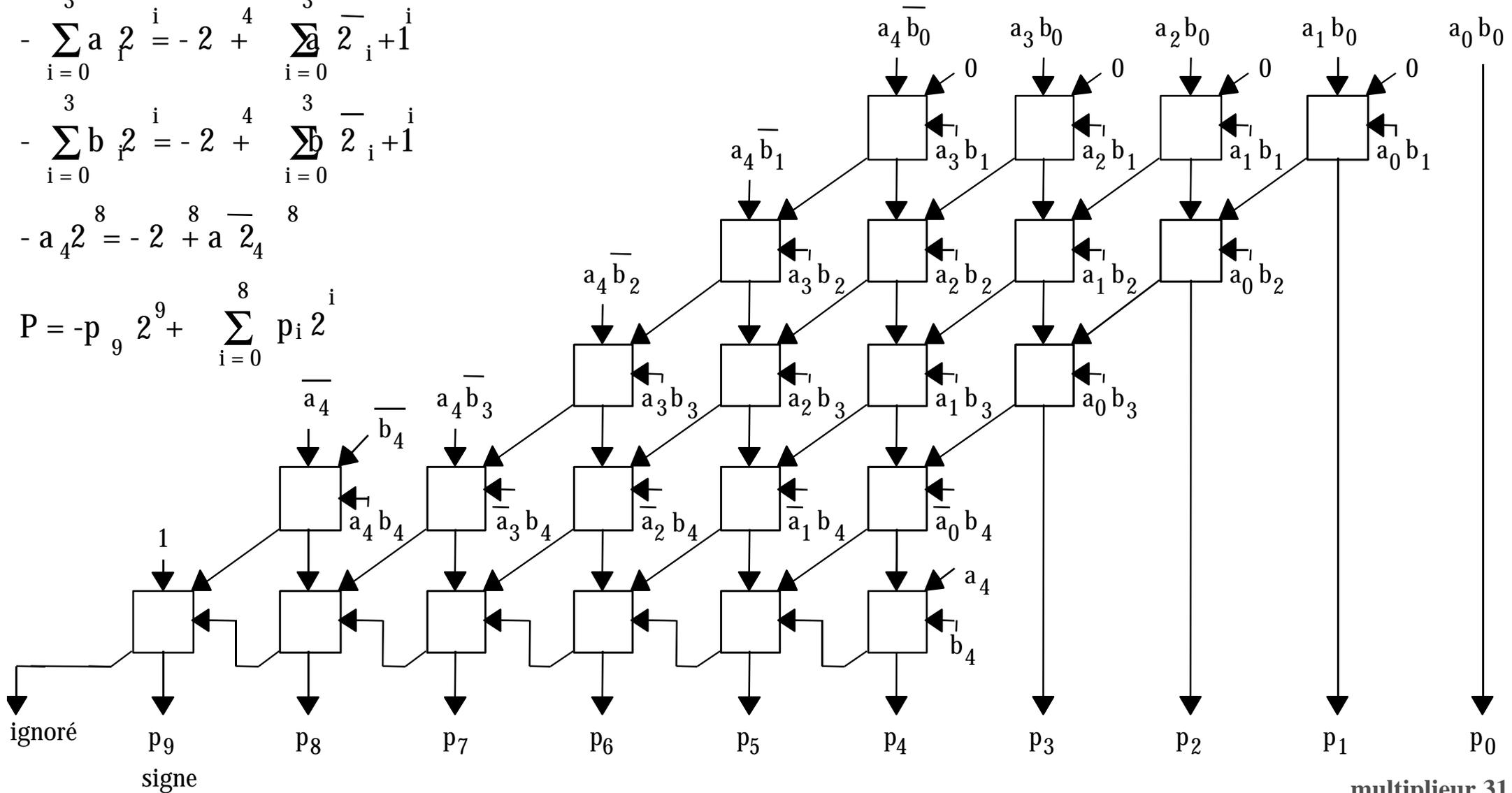
$$B = -b_4 2^4 + \sum_{i=0}^3 b_i 2^i$$

$$- \sum_{i=0}^3 a_i 2^i = -2^4 + \sum_{i=0}^3 \bar{a}_i 2^{i+1}$$

$$- \sum_{i=0}^3 b_i 2^i = -2^4 + \sum_{i=0}^3 \bar{b}_i 2^{i+1}$$

$$- a_4 2^8 = -2^8 + \bar{a}_4 2^8$$

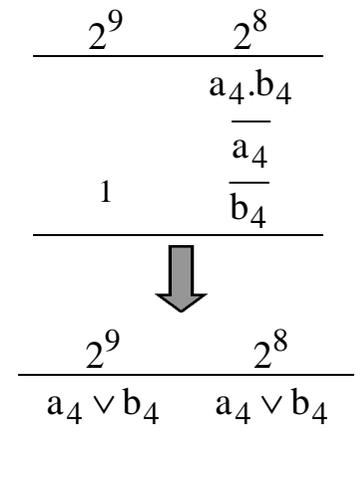
$$P = -p_9 2^9 + \sum_{i=0}^8 p_i 2^i$$



# Multiplieur de Baugh-Wooley (2)

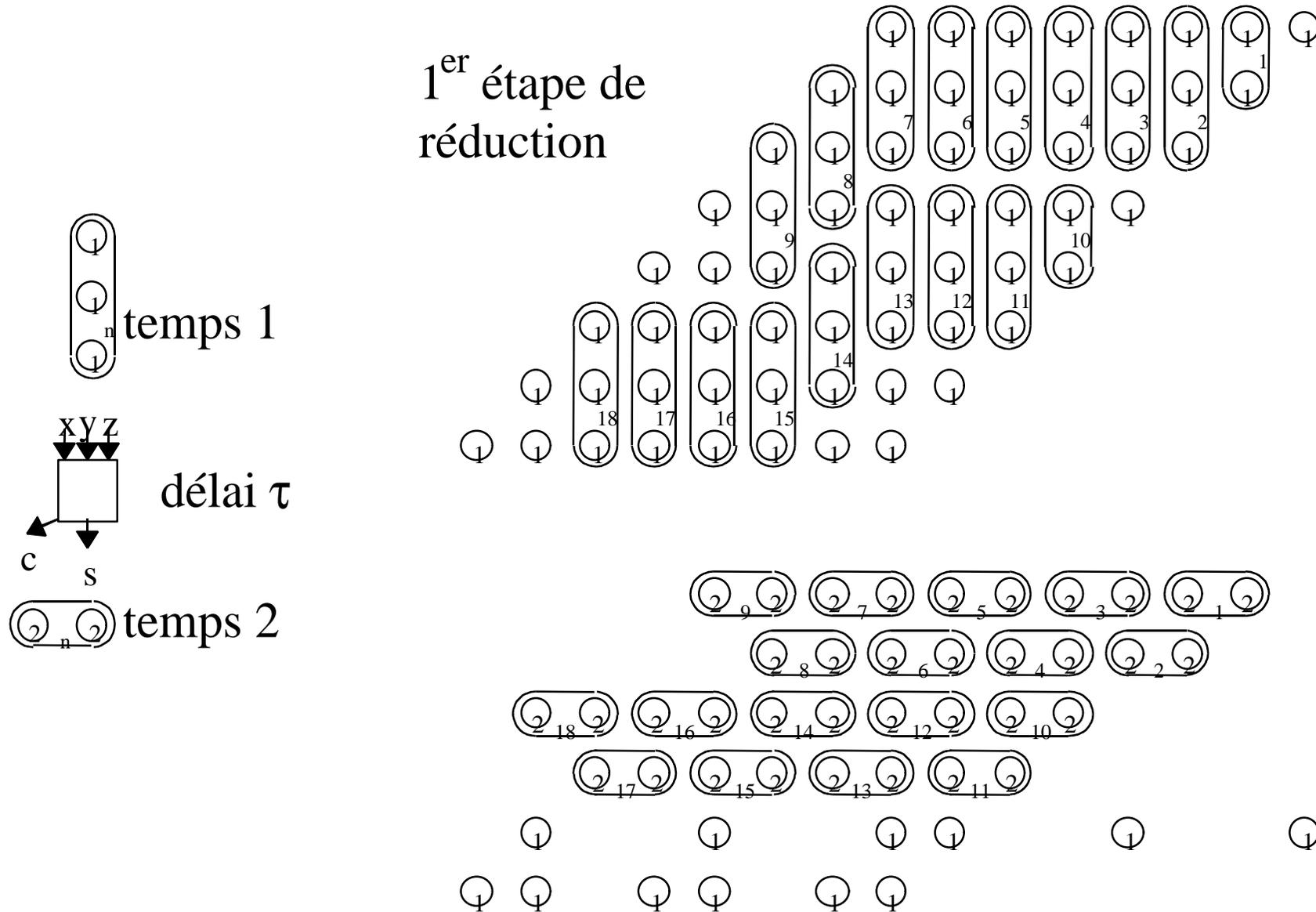
$$\begin{aligned}
 P = A * B &= \left( -a_4 2^4 + \sum_{i=0}^3 a_i 2^i \right) * \left( -b_4 2^4 + \sum_{j=0}^3 b_j 2^j \right) = a_4 \cdot b_4 2^8 + \sum_{i=0}^3 \sum_{j=0}^3 a_i \cdot b_j 2^{i+j} - a_4 2^4 \sum_{j=0}^3 b_j 2^j - b_4 2^4 \sum_{i=0}^3 a_i 2^i \\
 &= a_4 \cdot b_4 2^8 + \sum_{i=0}^3 \sum_{j=0}^3 a_i \cdot b_j 2^{i+j} + a_4 2^4 \left( -2^4 + \sum_{j=0}^3 \overline{b_j} 2^j + 1 \right) + b_4 2^4 \left( -2^4 + \sum_{i=0}^3 \overline{a_i} 2^i + 1 \right) \\
 &= a_4 \cdot b_4 2^8 + \sum_{i=0}^3 \sum_{j=0}^3 a_i \cdot b_j 2^{i+j} + (\overline{a_4} - 1 + \overline{b_4} - 1) 2^8 + (a_4 + b_4) 2^4 + \sum_{i=0}^3 (a_4 \cdot \overline{b_i} + \overline{a_i} \cdot b_4) 2^{i+4}
 \end{aligned}$$

$2^9$	$2^8$	$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
					$a_4 \cdot \overline{b_0}$	$a_3 \cdot b_0$	$a_2 \cdot b_0$	$a_1 \cdot b_0$	$a_0 \cdot b_0$
				$a_4 \cdot \overline{b_1}$	$a_3 \cdot b_1$	$a_2 \cdot b_1$	$a_1 \cdot b_1$	$a_0 \cdot b_1$	
			$a_4 \cdot \overline{b_2}$	$a_3 \cdot b_2$	$a_2 \cdot b_2$	$a_1 \cdot b_2$	$a_0 \cdot b_2$		
		$a_4 \cdot \overline{b_3}$	$a_3 \cdot b_3$	$a_2 \cdot b_3$	$a_1 \cdot b_3$	$a_0 \cdot b_3$			
	$a_4 \cdot \overline{b_4}$	$\overline{a_3} \cdot b_4$	$\overline{a_2} \cdot b_4$	$\overline{a_1} \cdot b_4$	$\overline{a_0} \cdot b_4$				
	$\overline{a_4}$				$a_4$				
1	$\overline{b_4}$				$b_4$				



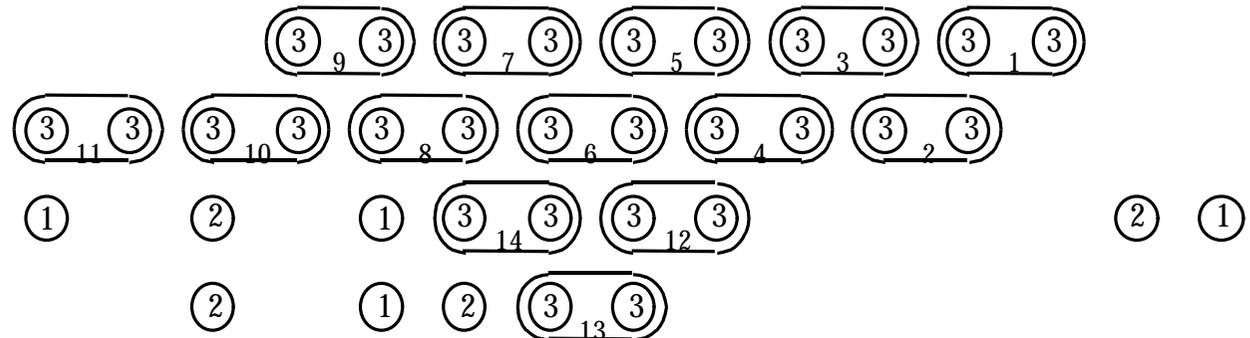
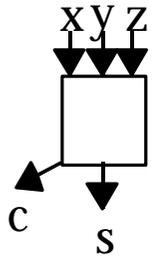
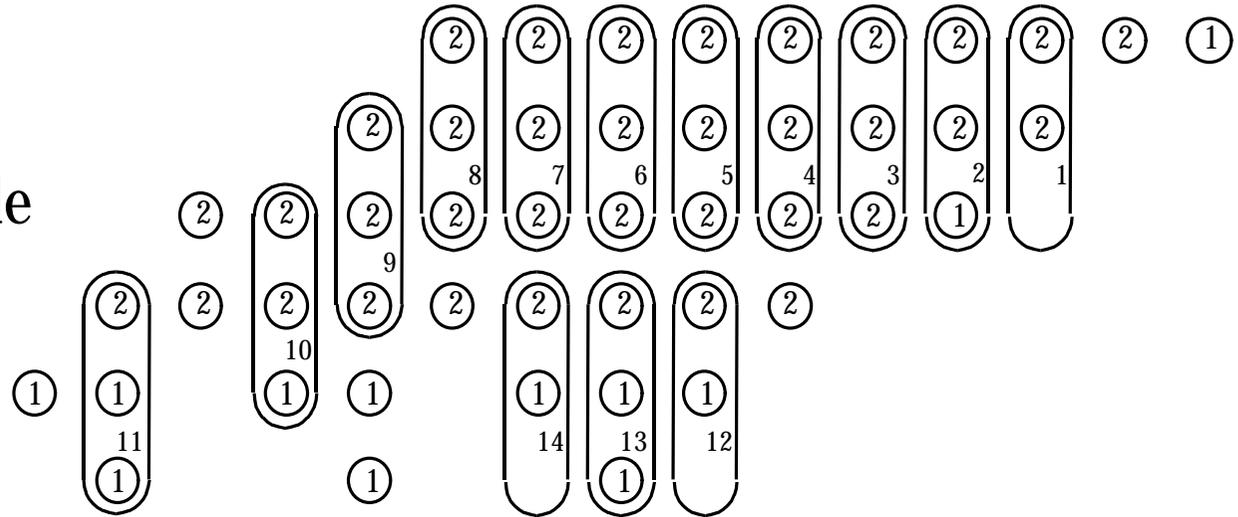
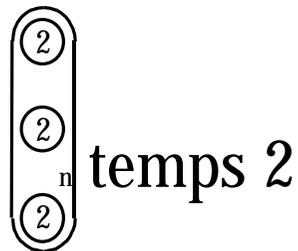
Amélioration  
de Blankenship

# Arbres de Wallace (1)

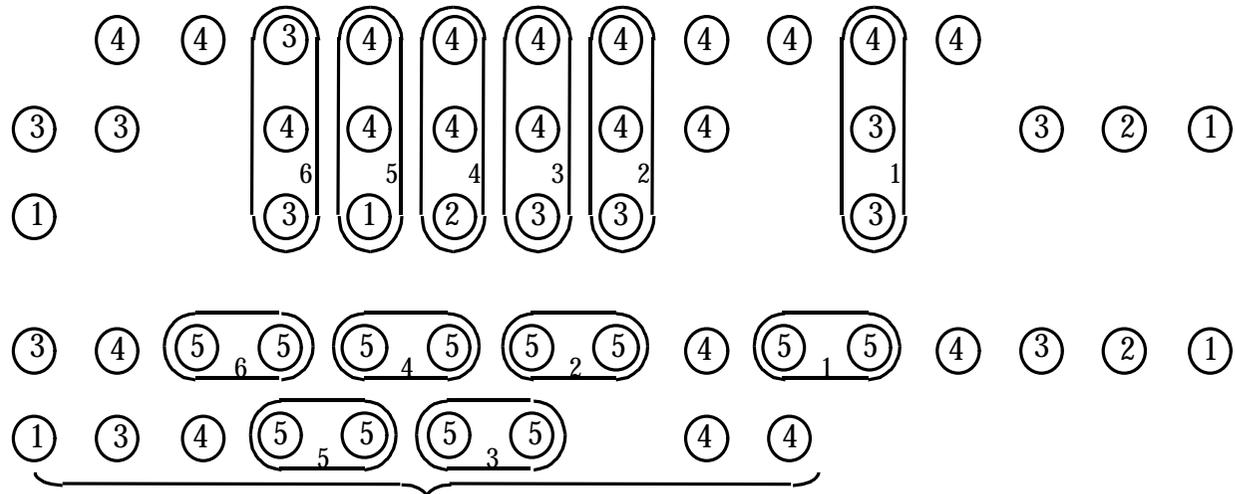
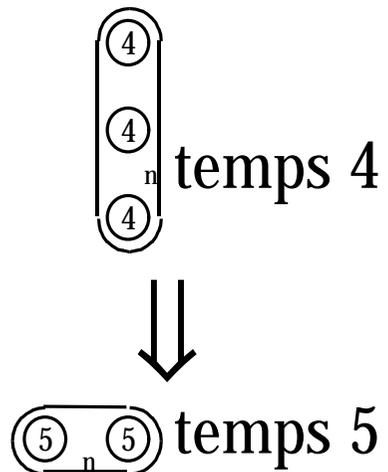
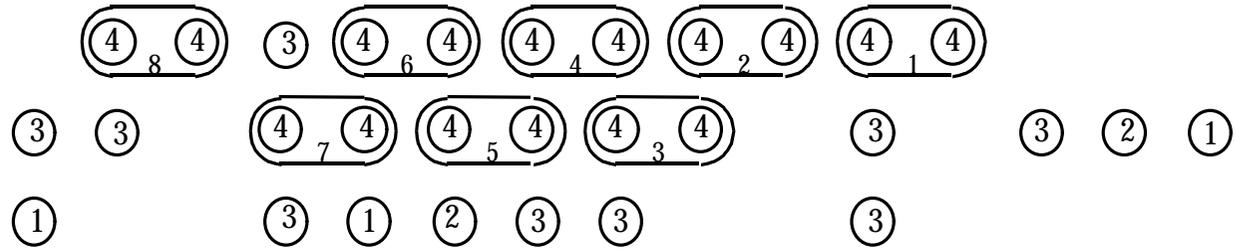
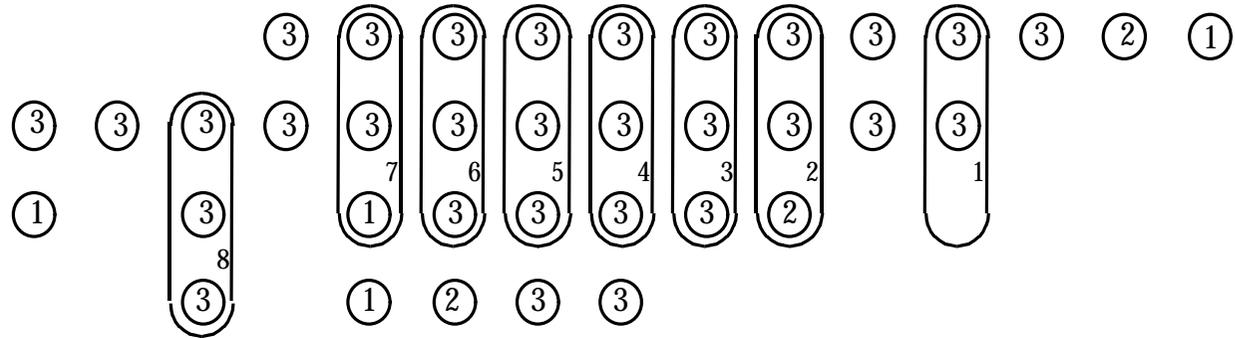
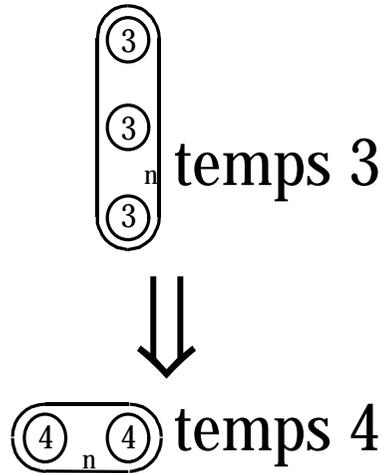


# Arbre de Wallace (2)

2<sup>eme</sup> étape de réduction



# Arbre de Wallace (3)



# Généralité sur la multiplication en arbre

## 1<sup>ere</sup> Étape

Calculer les  $c_{ij} = a_i * b_j$

## 2<sup>eme</sup> Étape

Réduire  $\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} (c_{ij} * 2^{i+j})$  en  $\sum_{i=0}^{2n-2} (r_i + s_i) * 2^i$

avec un réseau d'additionneurs (arbre) sans propagation de retenue  
À cause de l'associativité de l'opération, il y a beaucoup de solutions

## 3<sup>eme</sup> Étape

Réduire  $\sum_{i=0}^{2n-2} (r_i + s_i) * 2^i$  en  $P = \sum_{i=0}^{2n-1} p_i * 2^i$

avec un additionneur à propagation de retenue.

Utiliser l'additionneur le plus approprié

$$O(n), O(\sqrt[2]{n}), O(\log_2 n)$$

# Arbre linéaire $O(n)$

$$A*B = \sum_{i=0}^7 A*b_i*2^i$$

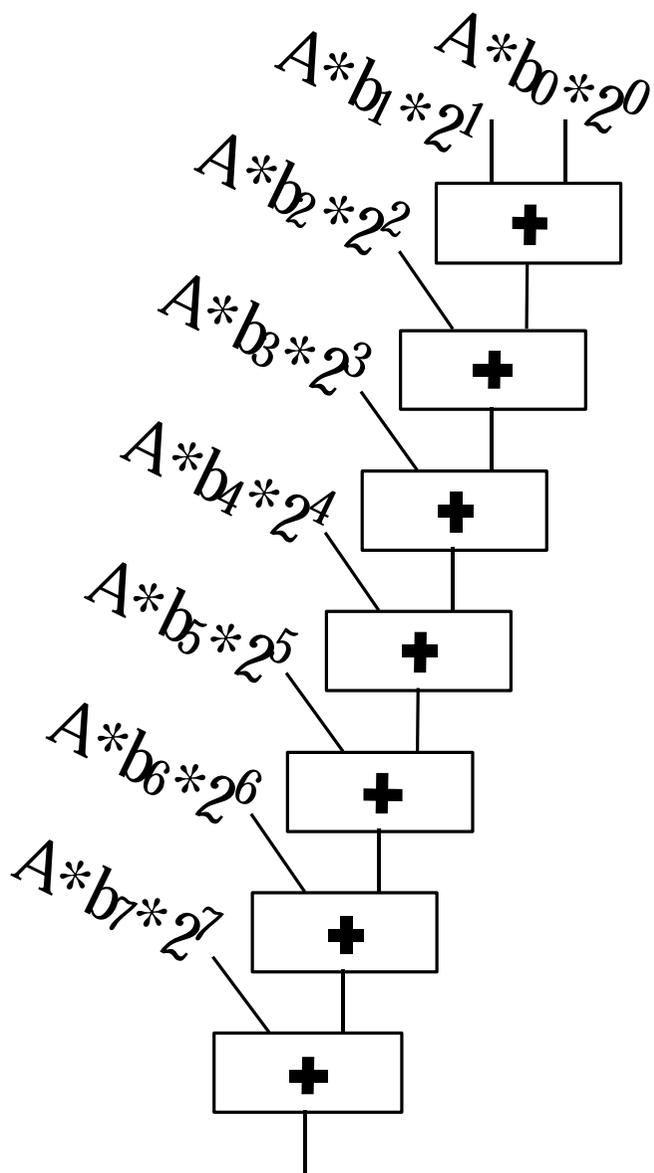


Figure 1

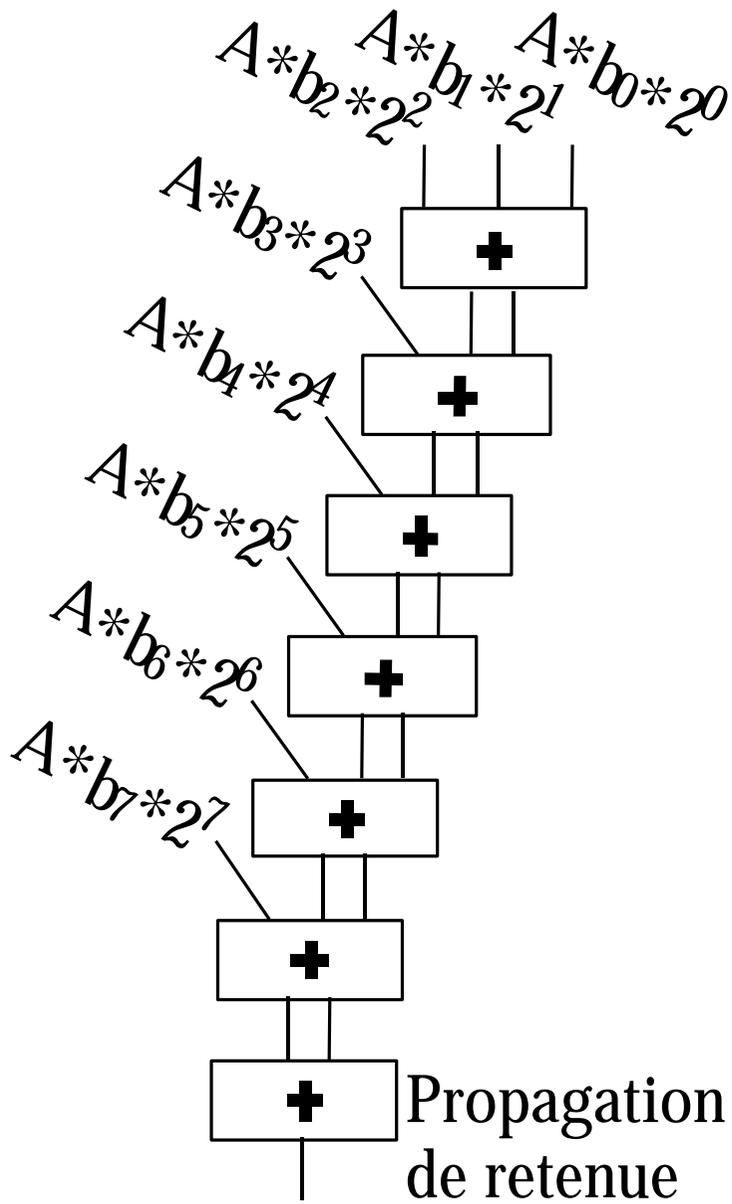


Figure 2

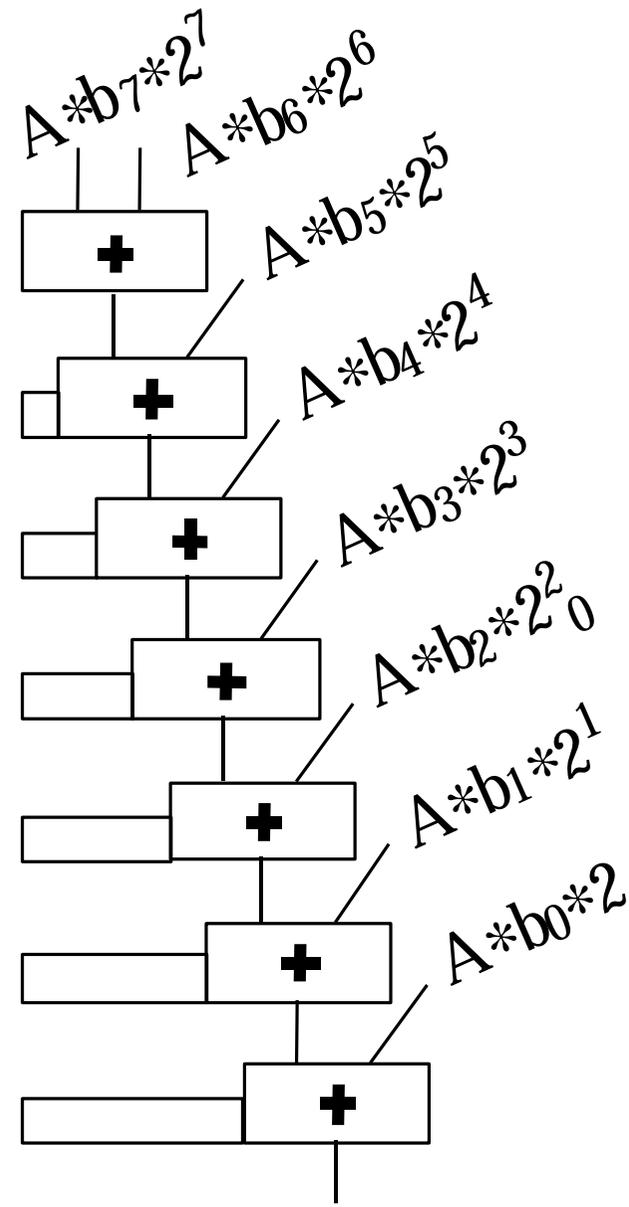
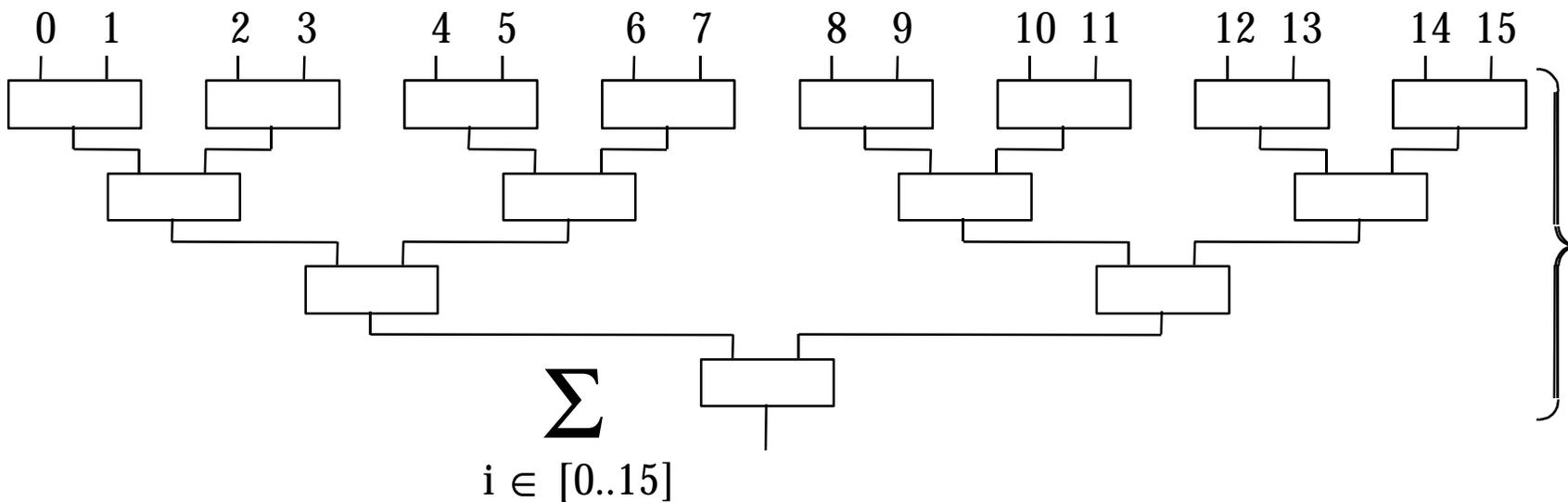


Figure 3 multiplieur 32



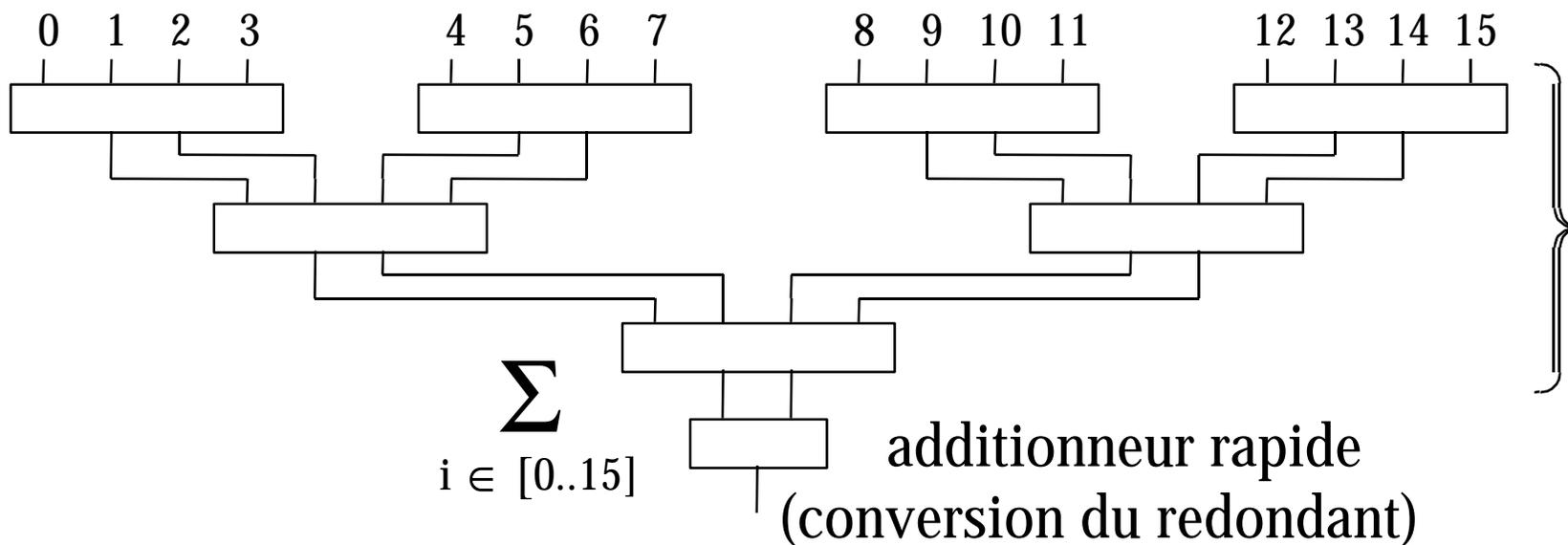
# Arbres Binaires $O(\log_2 n)$



additionneurs  
rapides

$$S = \sum_{i=0}^m s_i * 2^i$$

$s \in \{0,1\}$



additionneurs  
redondants

$$S = \sum_{i=0}^m s_i * 2^i$$

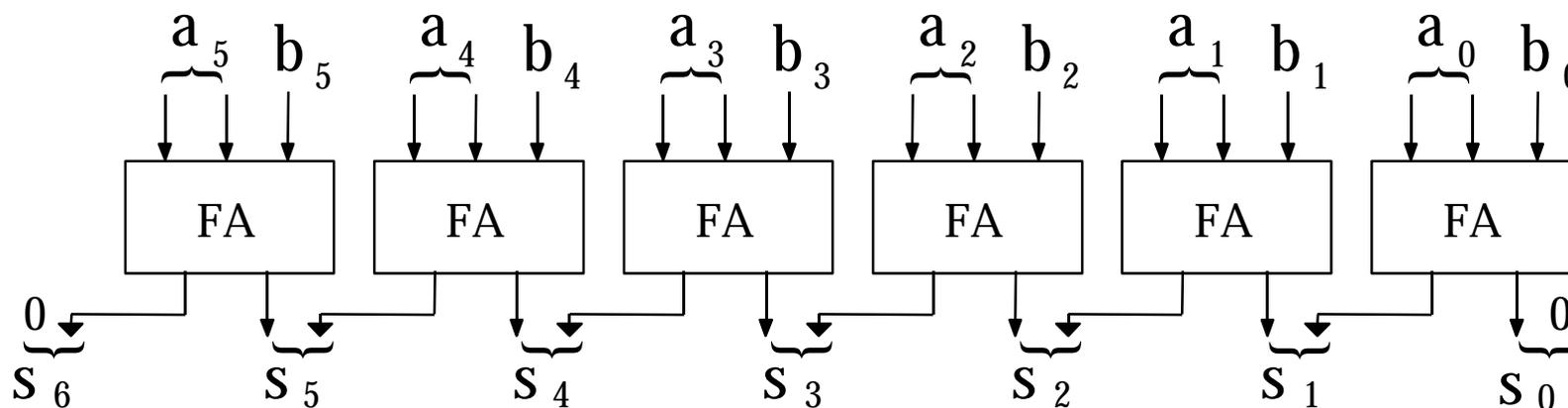
$s \in \{0,1,2\}$

additionneur rapide  
(conversion du redondant)

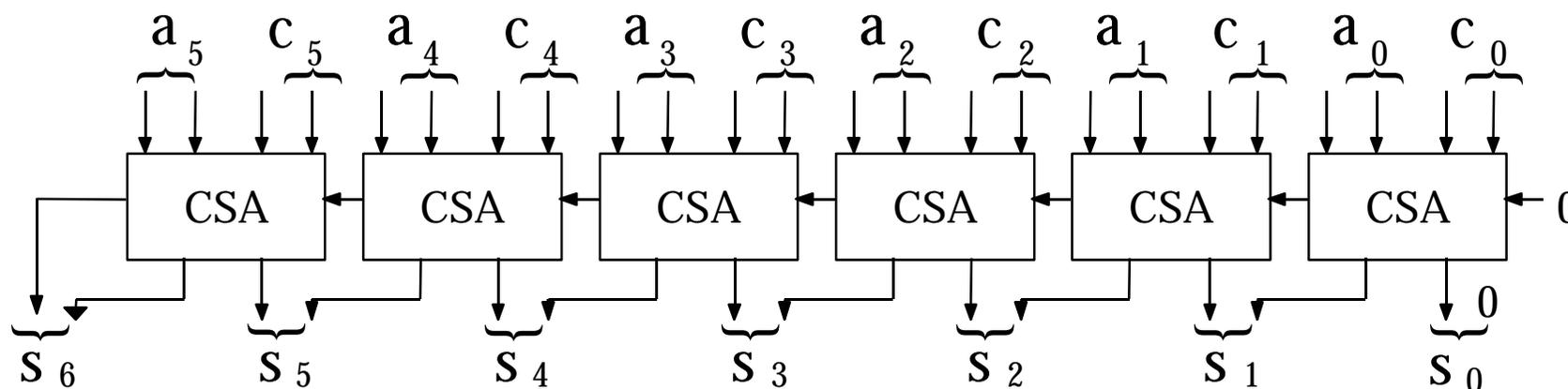
# Rappel sur les additions sans propagation

$$A = \sum_{i=0}^{n-1} a_i 2^i \quad C = \sum_{i=0}^{n-1} c_i 2^i \quad S = \sum_{i=0}^n s_i 2^i \quad a_i, c_i, s_i \in \{0,1,2\}$$

## 1- Hybride: Carry-save + Conventionnel → Carry-save



## 2- Binaire: Carry-save + Carry-save → Carry-save





# Codage de Booth

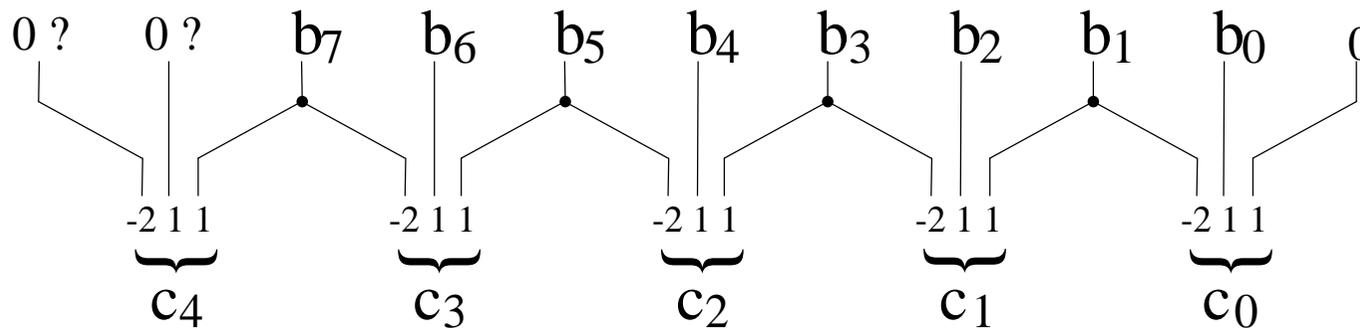
code de B	$B = \sum_{i=0}^{n-1} b_i * 2^i$	$B = \sum_{i=0}^{(n-2)/2} c_i * 4^i$
produit A * B	$A * B = \sum_{i=0}^{n-1} A * b_i * 2^i$	$A * B = \sum_{i=0}^{(n-2)/2} A * c_i * 4^i$
valeurs des chiffres	$b_i \in \{0,1\}$	$c_i \in \{-2, -1, 0, 1, 2\}$
nombre d'opérations	n-1	(n-2) / 2
type d'opération	addition	addition/ soustraction

# Conversion en code de Booth (1)

$$B = \sum_{i=0}^{n-1} b_i * 2^i \quad \longrightarrow \quad B = \sum_{i=0}^{\frac{n-2}{2}} c_i * 4^i$$

Soient  $B'' = \sum_{i=0}^{\frac{n-2}{2}} b_{2i} * 2^{2i}$      $B' = \sum_{i=0}^{\frac{n-2}{2}} b_{2i+1} * 2^{2i+1}$      $B = B'' + B' = B'' + (2 * B') - 2 * (1/2 * B')$

termes de rang pair
termes de rang impair
décalé à gauche
décalé à droite



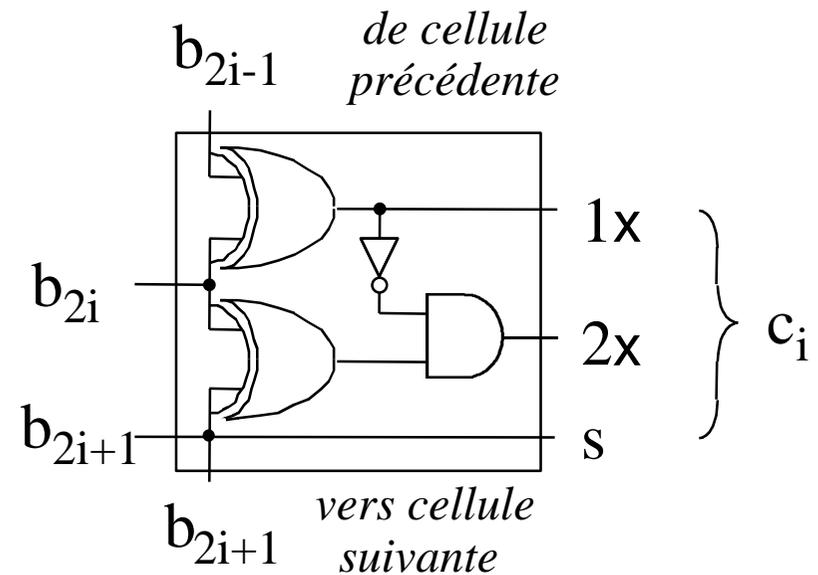
la valeur d'un chiffre  $C_i$  est la somme pondérée de ses bits

$$c_i = -2 b_{2i+1} + b_{2i} + b_{2i-1} \quad c_i \in \{-2, -1, 0, 1, 2\}$$

La somme pondérée des chiffres qui entrent est égale à la somme pondérée des chiffres qui sortent

# Conversion en code de Booth (2)

$b_{2i+1}$	$b_{2i}$	$b_{2i-1}$	$c_i$	1x	2x	s
0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	1	0	1	1	0	0
0	1	1	2	0	1	0
1	0	0	-2	0	1	1
1	0	1	-1	1	0	1
1	1	0	-1	1	0	1
1	1	1	-0	0	0	1



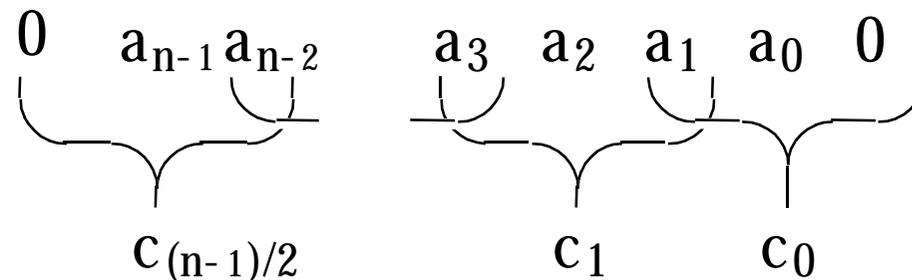
$$c_i * A = (s \oplus (A \wedge 1x \vee (2 * A) \wedge 2x)) + s$$

La multiplication d'un nombre par un chiffre est plus aisée mais la valeur d'un chiffre  $c_i$  n'est plus la somme pondérée de ses bits (code signe + valeur absolue).

# Remarques sur le code de Booth

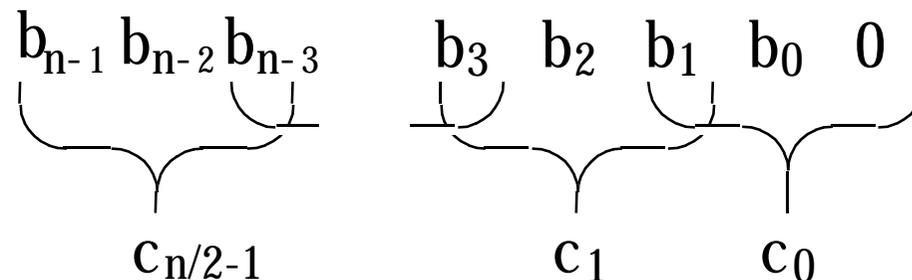
Entiers positifs

$$A = \sum_{i=0}^{n-1} a_i 2^i$$



Entiers signés

$$B = -b_{n-1} 2^{n-1} + \sum_{i=0}^{n-2} b_i 2^i$$



La notation de Booth est redondante:  $(0 \ 2) = (1 \ 2)$

# Conversion en code de Booth (3)

$b_{j+1}$	$b_j$	$c_{j+1}$	
0	0	0	extérieur de chaîne
0	1	+1	tête de chaîne
1	0	-1	queue de chaîne
1	1	0	intérieur de chaîne

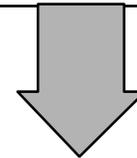
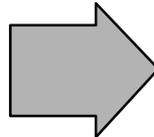
# Multiplieur rapide

multiplicande  $A = \sum_{i=0}^n a_i * 2^i$

multiplieur

$$B = \sum_{i=0}^n b_i * 2^i$$

conversion  
de standard à  
"code de Booth"



conversion de  
"carry save" à  
standard

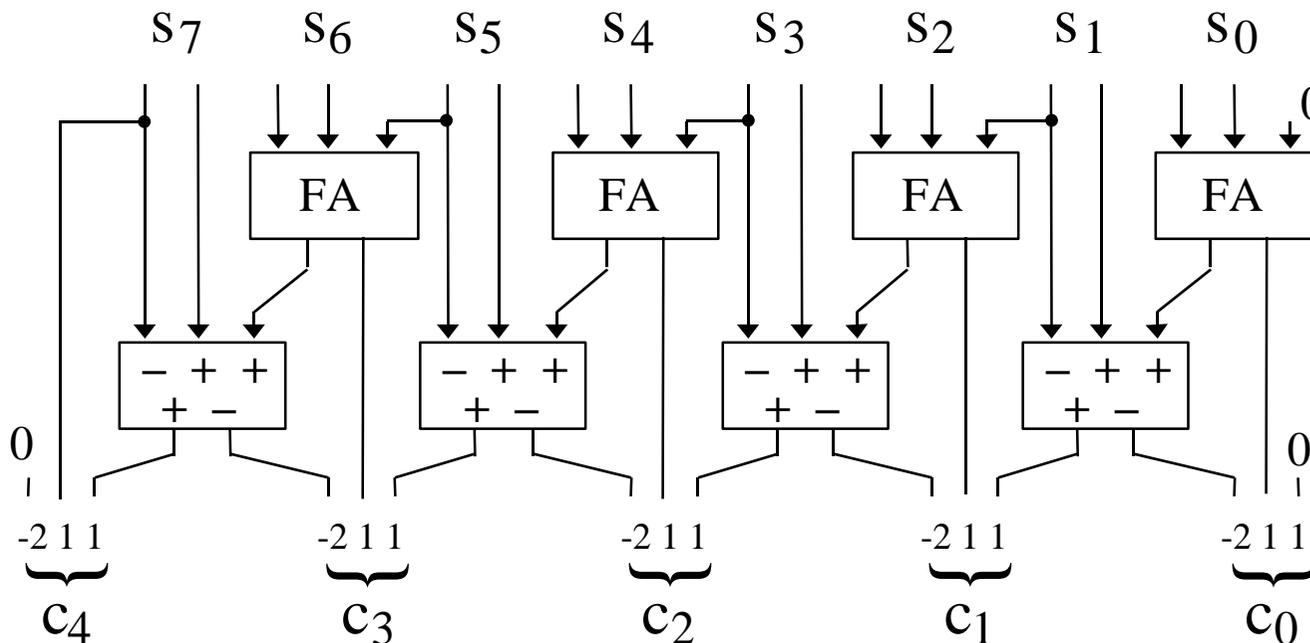
produit  
 $A * B$

# Conversion de CS à code de Booth

Si le produit est un addende, un multiplieur ou un dividende, il n'est pas nécessaire de le convertir le « carry save » en notation conventionnelle.

$$P = \sum_{i=0}^{n-1} s_i * 2^i \quad \longrightarrow \quad P = \sum_{i=0}^{\frac{n-2}{2}} c_i * 4^i$$

$s_i \in \{0,1,2\}$   
 $c_i \in \{-2,-1,0,1,2\}$



La somme pondérée des chiffres qui entrent est égale à la somme pondérée des chiffres qui sortent. multiplieur 33

# Conclusion sur la multiplication combinatoire

Comme un multiplieur n'est rien qu'un ensemble d'additionneurs, tout ce que nous savons sur l'addition s'applique

Le problème qui reste est celui de la topologie (plan de masse)

Si les opérandes sont arrondis, alors seule la partie poids forts est significative

Si un multiplieur et un additionneur sont dessinés avec la même stratégie, pour la même taille un multiplieur est environ deux fois plus lent .