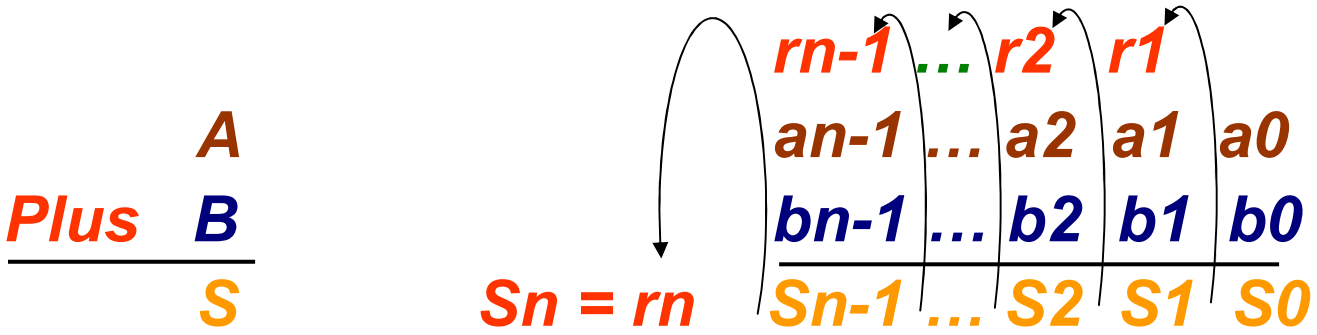
Logique combinatoire

A. M'zoughi

Etude de l'additionneur (1/2)

Addition : opération la plus fréquemment réalisée dans un calculateur ➔ l'optimiser au maximum.

❖ **Addition de 2 nombres A et B codés sur n bits chacun**



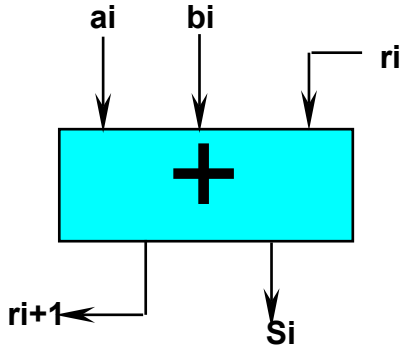
- ***Etudier directement un additionneur de 2 nombres de 32 bits est possible mais très complexe !!!***

- ***Remarque : excepté le rang 0, pour tout rang i (i=1…n-1) on a effectué à chaque fois la somme de 3 bits (ai, bi et ri report provenant du rang i-1) et on a produit 2 bits (Si et ri+1)***

Etude de l'additionneur (2/2)

❖ Autre approche :

- pour simplifier il suffit de ramener l'étude à celle d'un additionneur 3 bits.
- ensuite pour réaliser un additionneur n bits il suffit de dupliquer la logique n fois.



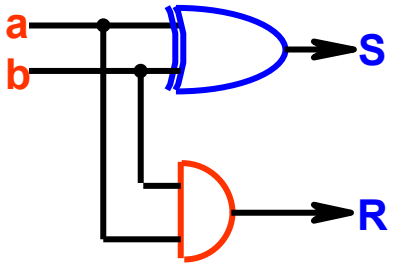
- 1er CAS : réaliser un additionneur 3 bits à partir d'un additionneur 2 bits
  - Addition de 2 bits a (plus) b
  - Résultat : 1 bit de somme S et 1 bit de report R

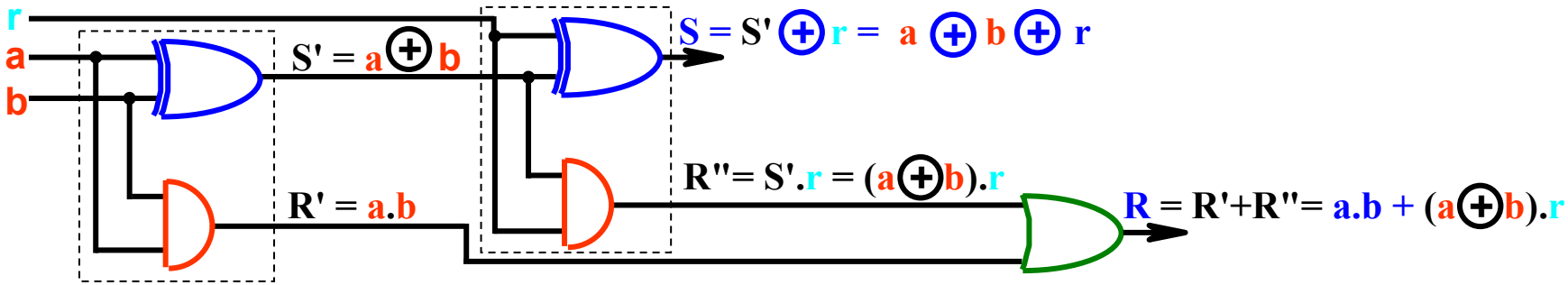
a	b	a plus b	R	S
0	0	00	0	0
0	1	01	0	1
1	0	01	0	1
1	1	10	1	0

Additionneur complet (1/4)

- **Les équations logiques :**
- $R = a.b$ $\qquad S = \bar{a}.b + a.\bar{b} = a \oplus b$

  - Additionneur 3 bits a (plus) b (plus) r
  - Résultat : 1 bit de somme S et 1 bit de report R





- **2ème CAS : réaliser directement un additionneur 3 bits**
  - Additionneur 3 bits a (plus) b (plus) r
  - Résultat : 1 bit de somme S et 1 bit de report R

Additionneur complet (2/4)

❖ Table de vérité : simplification partielle

r	a	b	r plus a plus b	R	S
0	0	0	00	0	0
0	0	1	01	0	1
0	1	0	01	0	1
0	1	1	10	1	0
1	0	0	01	0	1
1	0	1	10	1	0
1	1	0	10	1	0
1	1	1	11	1	1

R

r \ ab	00	01	11	10
0	0	0	1	0
1	0	1	1	1

S

r \ ab	00	01	11	10
0	0	1	0	1
1	1	0	1	0

$$R = a.b + r.\bar{a}.b + r.a.\bar{b} = a.b + r.(a \oplus b)$$

$$S = \bar{a}.\bar{b}.r + \bar{a}.b.\bar{r} + a.\bar{b}.\bar{r} + a.b.r$$

$$= \bar{r}.(\bar{a}.b + a.\bar{b}) + r.(\bar{a}.\bar{b} + a.b)$$

$$= \bar{r}(a \oplus b) + r.\overline{(a \oplus b)} = a \oplus b \oplus r$$

Additionneur complet (3/4)

❖ Table de vérité : simplification totale

r	a	b	r plus a plus b	R	S
0	0	0	00	0	0
0	0	1	01	0	1
0	1	0	01	0	1
0	1	1	10	1	0
1	0	0	01	0	1
1	0	1	10	1	0
1	1	0	10	1	0
1	1	1	11	1	1

R

r \ ab	00	01	11	10
0	0	0	1	0
1	0	1	1	1

S

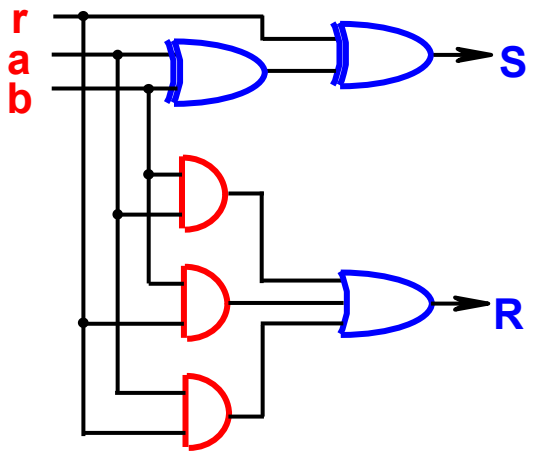
r \ ab	00	01	11	10
0	0	1	0	1
1	1	0	1	0

$$R = a.b + r.a + rb$$

$$S = \bar{a}.\bar{b}.r + \bar{a}.b.\bar{r} + a.\bar{b}.\bar{r} + a.b.r$$

$$= \bar{r}.(\bar{a}.b + a.\bar{b}) + r.(\bar{a}.\bar{b} + a.b)$$

$$= \bar{r}(a \oplus b) + r.\overline{(a \oplus b)} = a \oplus b \oplus r$$



Additionneur complet (4/4)

❖ Réalisation avec des opérateurs NON ET (NAND)

$$\overline{S} = \overline{\overline{a}.\overline{b}.r + a.\overline{b}.\overline{r} + \overline{a}.b.\overline{r} + a.b.r}$$

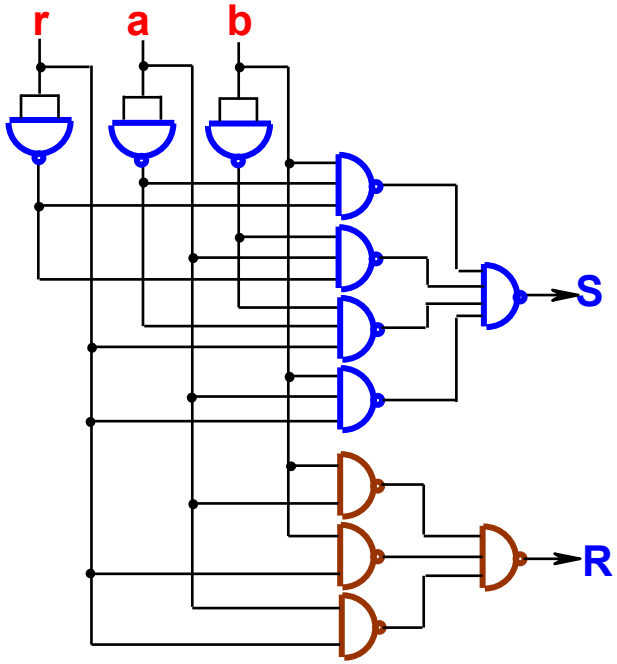
$$= \overline{(a.\overline{b}.r).(a.\overline{b}.\overline{r}).(\overline{a}.b.\overline{r}).(a.b.r)}$$

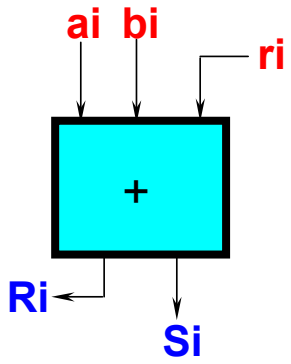
$$= ((a/a)/(b/b)/r)/((a/a)/b/(r/r))/(a/(b/b)/(r/r))/(a/b/r)$$

$$\overline{R} = \overline{a.b + r.a + r.b}$$

$$= \overline{(a.b).(r.a).(r.b)}$$

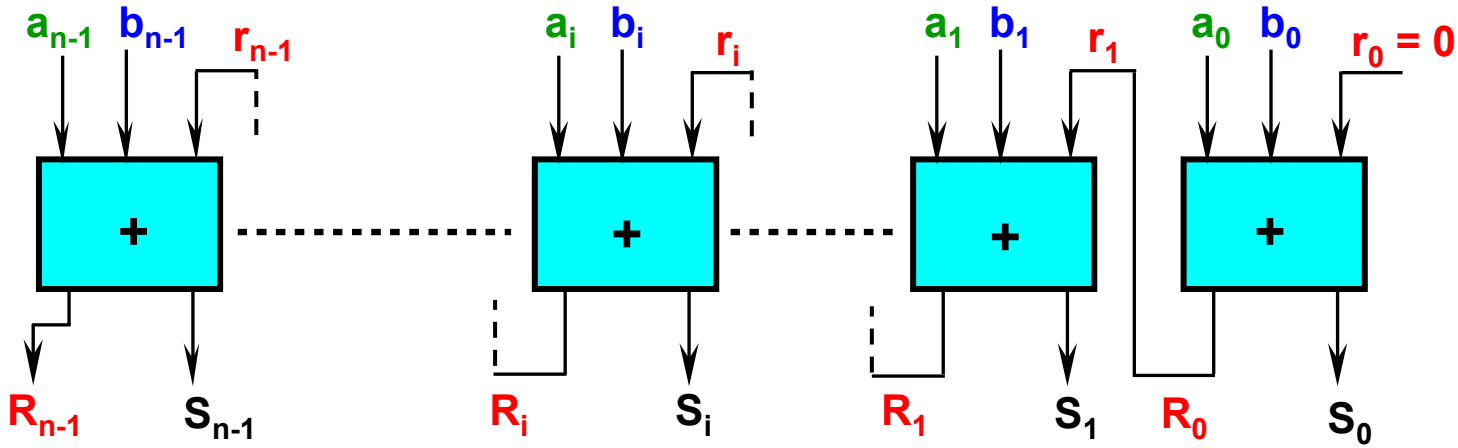
$$= (a/b)/(r/a)/(r/b)$$





Additionneur à report en cascade (Ripple carry)

❖ Additionneur n bits à partir d'un additionneur complet 1 bit



☺ Le plus simple à réaliser

☹ Le moins performant (très lent)

➢ Dans le cas où n=32, le résultat n'est obtenu qu'après la traversée :

- De 65 couches logiques (additionneur complet conçu à partir de 2 demi-additionneurs)
- 64 couches logiques (additionneur complet conçu directement)

Additionneur à report anticipé (Carry look-ahead)

- Calculer Ri en fonction des bits ai, bi et les bits de rang inférieurs jusqu'à a0, b0 et r0 (report entrant initial)
- Le report pour un additionneur complet est égal :

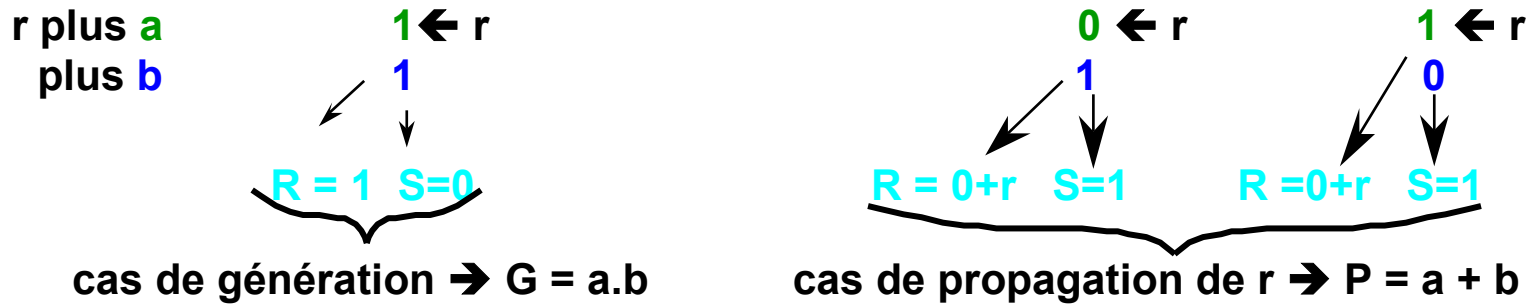
$$R = ab + ra + rb = ab + r(a + b) = ab + r(a \oplus b)$$

- R = 1 ssi une des deux conditions suivantes est vérifiée :
  - a = b = 1 quelque soit r ➔ alors il y a génération systématique d'un report ➔ on définit un terme dit de génération :

$$G = a.b$$

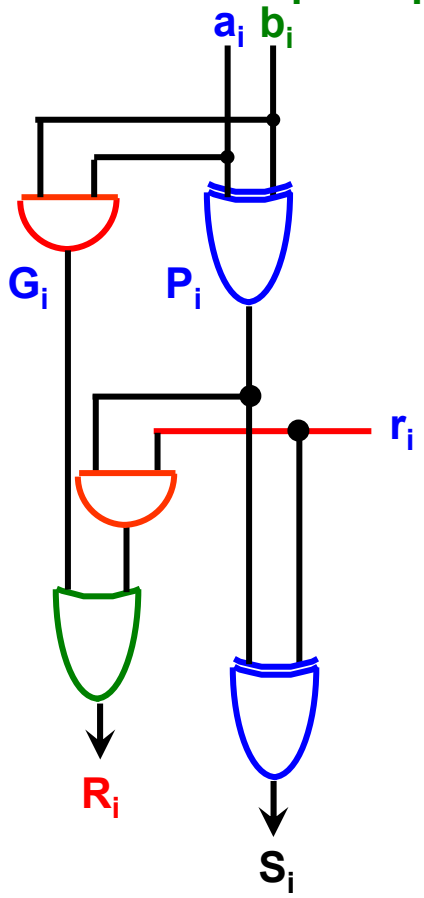
  - a ou b = 1 et r = 1 ➔ alors il y a propagation du report ➔ on définit un terme dit de propagation :

$$P = a \oplus b$$



Réalisation des termes de génération et de propagation

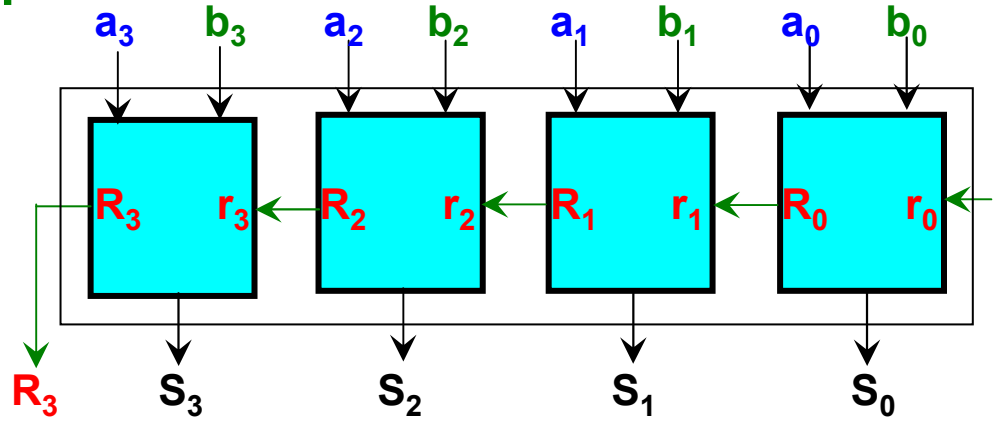
- L'équation peut se mettre sous le forme : $R_i = G_i + P_i.r_i$
- L'équation de la Somme : $S_i = a_i \oplus b_i \oplus r_i$



Equations logiques des reports

❖ Addition à report anticipé de 2 nombres de 4 bits chacun

- A ➔ a_3 a_2 a_1 a_0
- B ➔ b_3 b_2 b_1 b_0



- Le circuit de rang 0 nous donne les termes G0 et P0 ➔ $R_0 = G_0 + P_0 r_0$ avec (r_0= report initial)
- Le circuit de rang 1 nous donne les termes G1 et P1 ➔ $R_1 = G_1 + P_1 r_1$ avec ($r_1 = R_0$)
- Le circuit de rang 2 nous donne les termes G2 et P2 ➔ $R_2 = G_2 + P_2 r_2$ avec ($r_2 = R_1$)
- Le circuit de rang 3 nous donne les termes G3 et P3 ➔ $R_3 = G_3 + P_3 r_3$ avec ($r_3 = R_2$)
- En remplaçant les r_i par leur valeur :

$$R_0 = G_0 + P_0.r_0$$

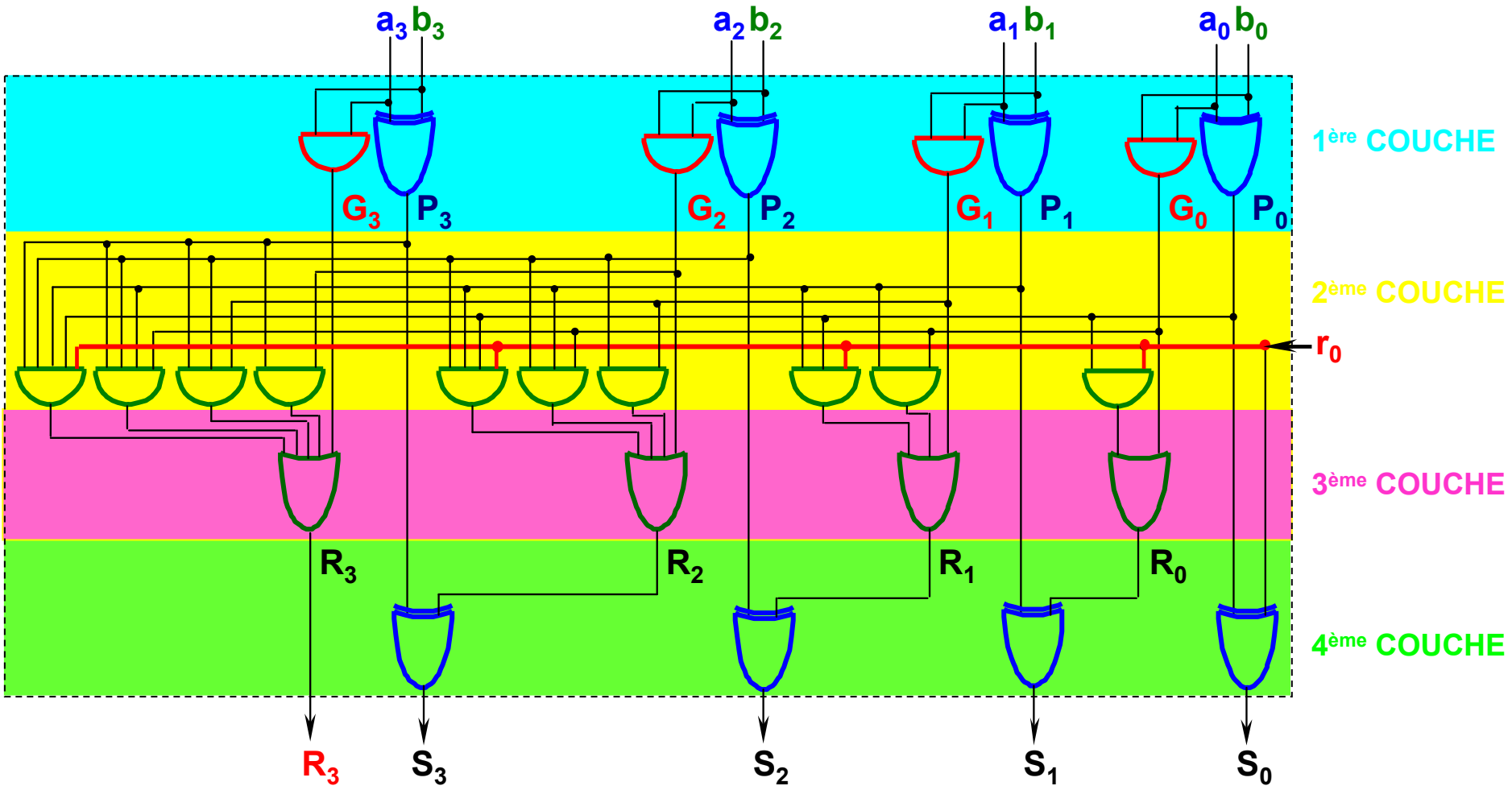
$$R_1 = G_1 + P_1.r_1 = G_1 + P_1.(G_0 + P_0.r_0) = G_1 + P_1.G_0 + P_1.P_0.r_0$$

$$R_2 = G_2 + P_2.r_2 = G_2 + P_2.(G_1 + P_1.G_0 + P_1.P_0.r_0) = G_2 + P_2.G_1 + P_2.P_1.G_0 + P_2.P_1.P_0.r_0$$

$$R_3 = G_3 + P_3.r_3 = G_3 + P_3.(G_2 + P_2.G_1 + P_2.P_1.G_0 + P_2.P_1.P_0.r_0)$$

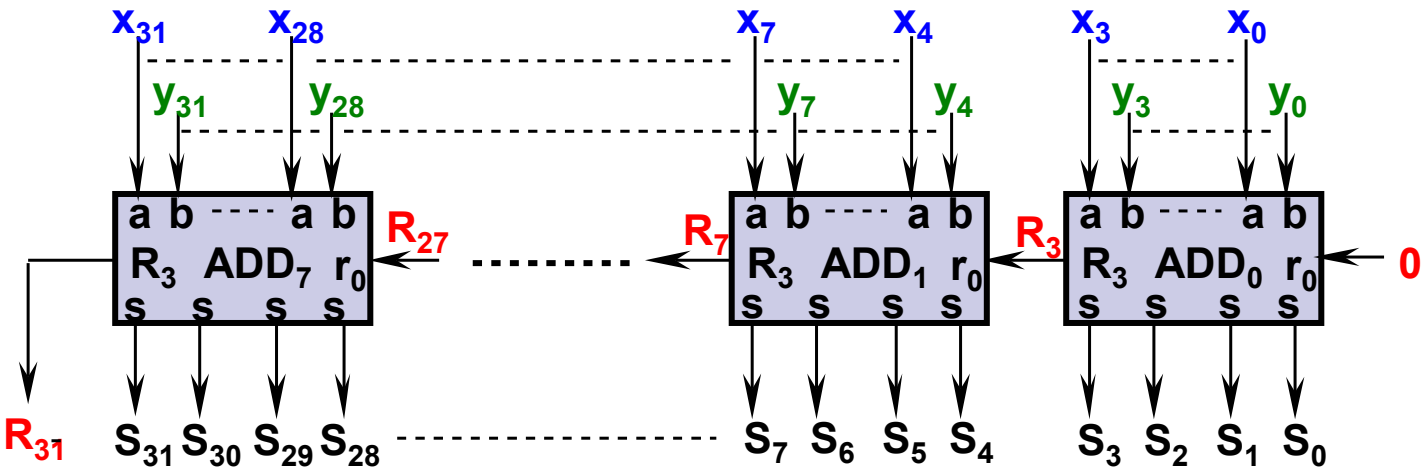
$$= G_3 + P_3.G_2 + P_3.P_2.G_1 + P_3.P_2.P_1.G_0 + P_3.P_2.P_1.P_0.r_0$$

Additionneur rapide à report anticipé de 4 bits



Additionneur 32 bits à report anticipé

❖ **Exemple addition de 2 nombres X et Y de 32 bits chacun**



Le report R_0 issu des bits de poids faibles se propage à travers 17 couches d'opérateurs

Addition à report anticipé à plusieurs niveaux (1/8)

- **L'équation du report sortant de l'additionneur 4 bits à report anticipé est la suivante :**

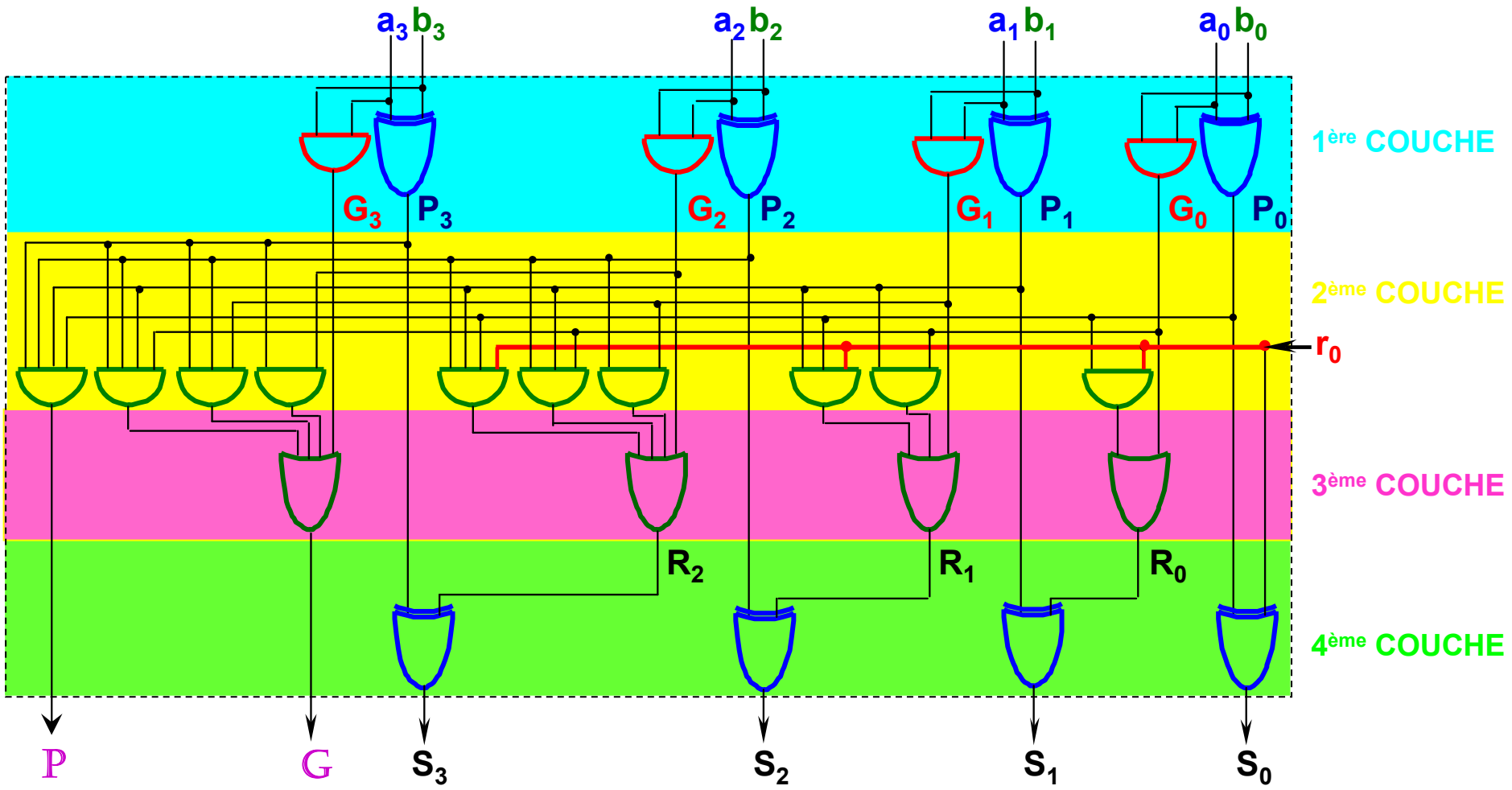
$$R3 = G3 + G2.P3 + G1.P3.P2 + G0.P3.P2.P1 + P3.P2.P1.P0.r0$$

- **R3 dépend :**
  - D'un terme de génération : $\mathbb{G}$ = **G3 + G2.P3 + G1.P3.P2 + G0.P3.P2.P1**
  - D'un terme de propagation : $\mathbb{P}$ = **P3.P2.P1.P0**

- **On peut donc écrire R3 sous la forme : $R3 = \mathbb{G} + \mathbb{P}.r0$**

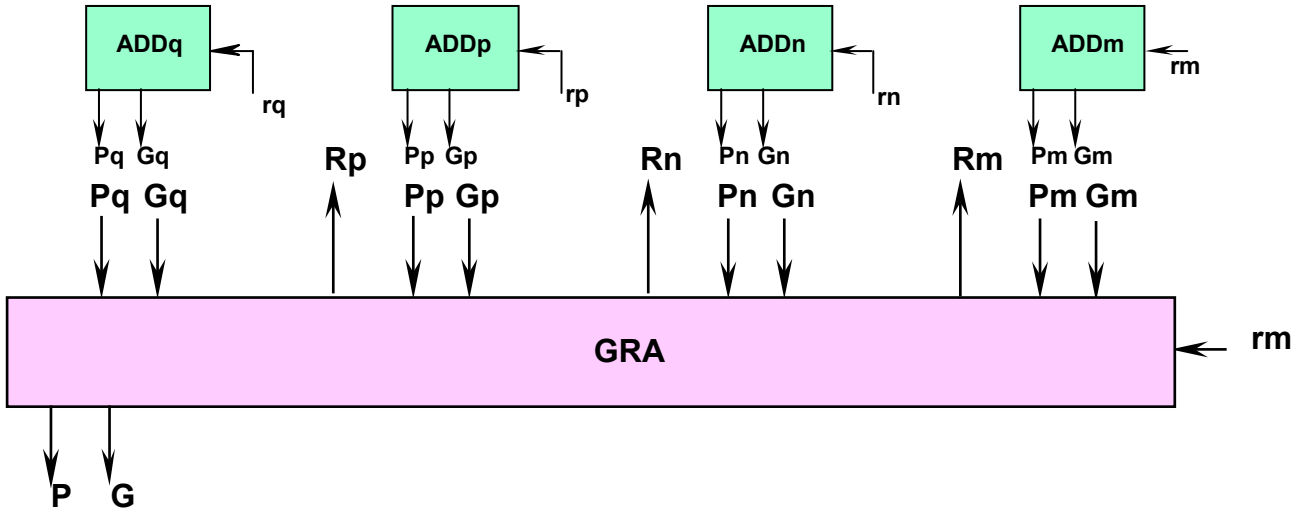
Addition à report anticipé à plusieurs niveaux (2/8)

Schéma de l'additionneur 4 bits à report anticipé modifié



Addition à report anticipé à plusieurs niveaux (3/8)

- On peut avec des circuits dits générateurs de report anticipé monter des additionneurs modifiés 4 bits pour améliorer la performance



- On a :
  - Rm = Gm + Pm.rm
  - Rn = Gn + Pn.rn avec rn = Rm
  - Rp = Gp + Pp.rp avec rp = Rn
  - Rq = Gq + Pq.rq avec rq = Rp

Addition à report anticipé à plusieurs niveaux (4/8)

- **On a**
  - Rm = Gm + Pm.rm
  - Rn = Gn + Pn.rn avec rn = Rm
  - Rp = Gp + Pp.rp avec rp = Rn
  - Rq = Gq + Pq.rq avec rq = Rp

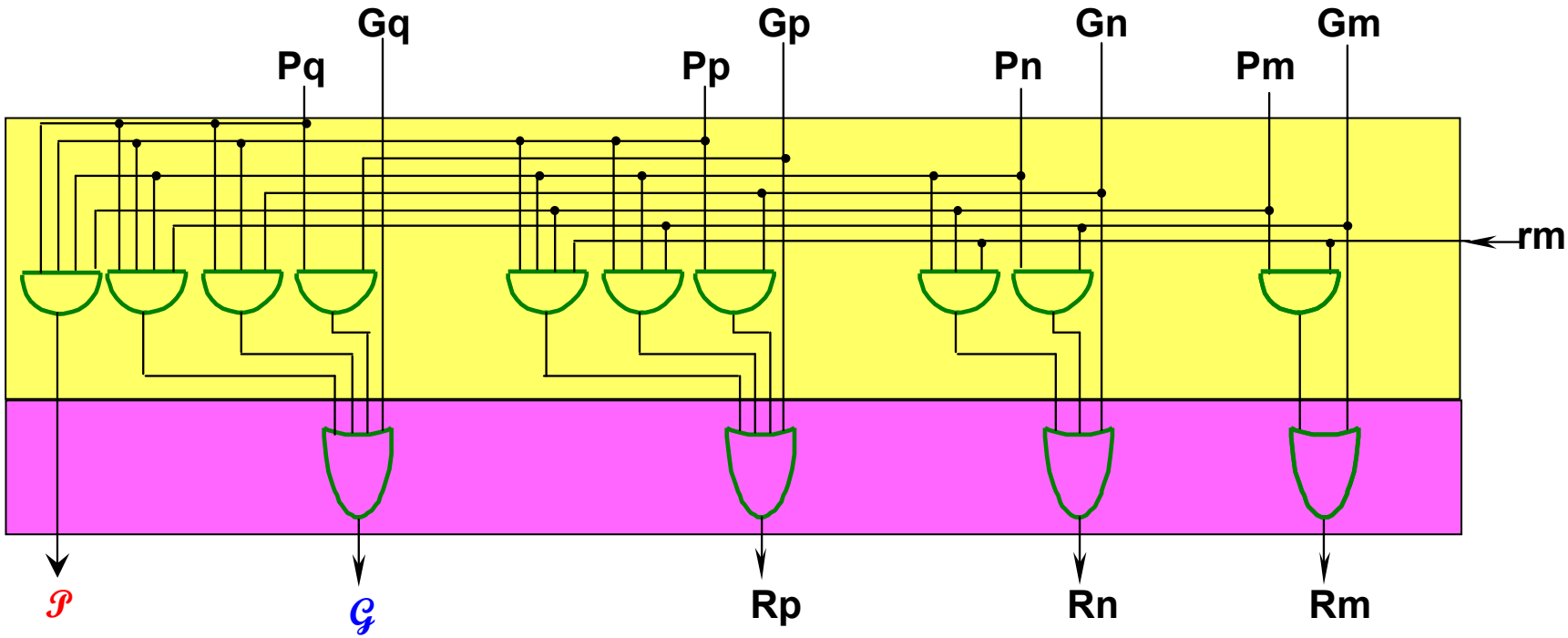
- **En ramplaçant le report entrant de rang i+1 par les reports sortant de rang i**
  - Rm = Gm + Pm.rm
  - Rn = Gn + Pn.(Gm + Pm.rm) = Gn + Gm.Pn + Pm.Pn.rm
  - Rp = Gp + Pp.(Gn + Gm.Pn + Pm.Pn.rm) = Gp + Gn.Pp + Gm.Pn.Pp + Pm.Pn.Pp.rm
  - Rq = Gq + Pq.(Gp + Gn.Pp + Gm.Pn.Pp + Pm.Pn.Pp.rm)
    = Gq + Gp.Pq + Gn.Pp.Pq + Gm.Pn.Pp.Pq + Pm.Pn.Pp.Pq.rm

- **On peut encore Rq sous la forme : Rq = $\mathcal{G}$ + $\mathcal{P}$.rm**
  - avec $\mathcal{G}$ = **Gq + Gp.Pq + Gn.Pp.Pq + Gm.Pn.Pp.Pq**
  - et $\mathcal{P}$ = **Pm.Pn.Pp.Pq**

Les circuits générateurs de report anticipé permettent à partir de Gm, Pm, Gn, Pn, Gp, Pp, Gq, Pq et rm de fournir en sortie les fonctions de report anticipé Rm, Rn, Rp et Rq

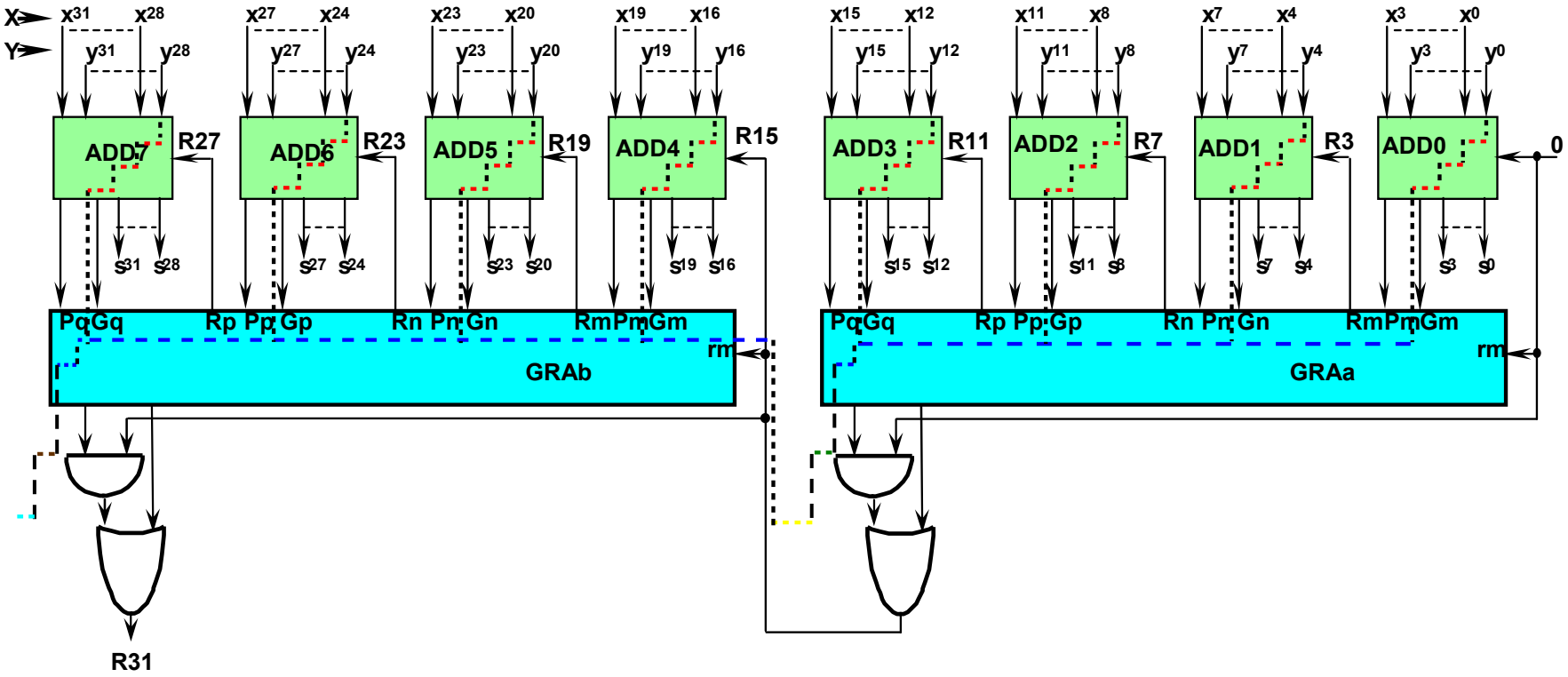
Addition à report anticipé à plusieurs niveaux (5/8)

❖ Schéma logique du générateur de report anticipé



Addition à report anticipé à plusieurs niveaux (6/8)

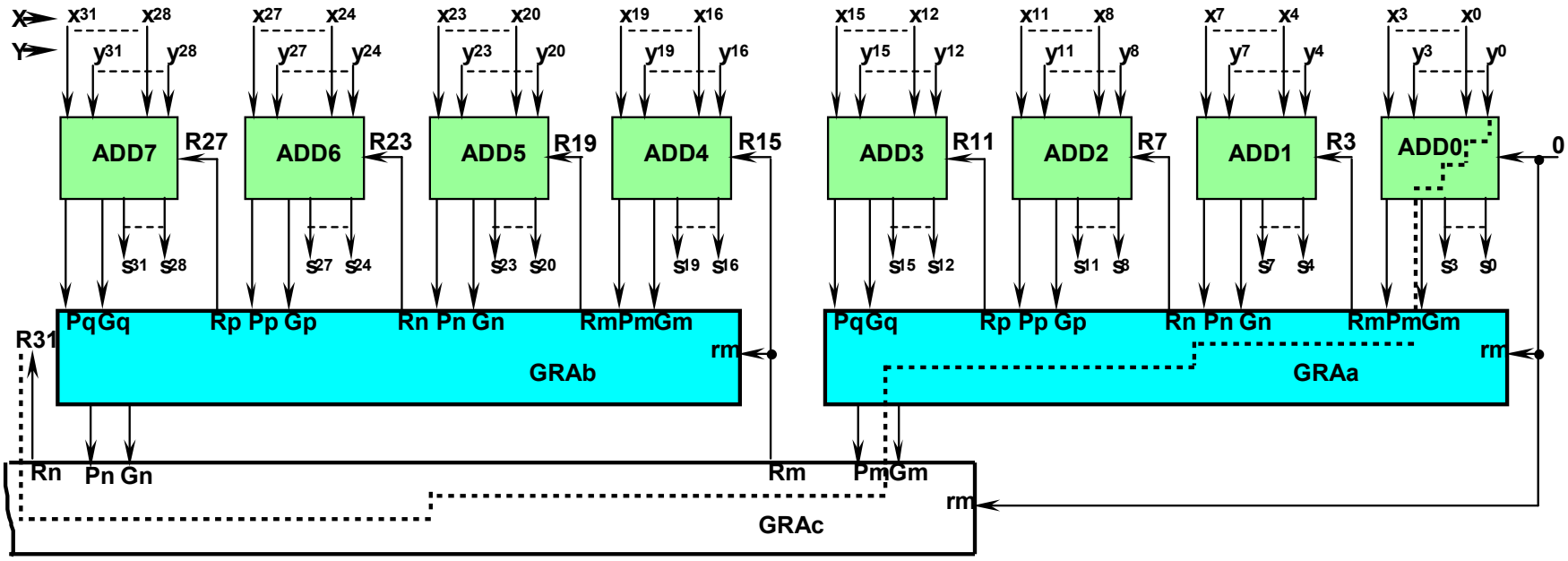
❖ ***Deux niveaux :*** **additionneur 32 bits à 2 niveaux de calcul de report anticipé**



9 couches au lieu de 17 avec des additionneurs anticipés monté en cascade

Addition à report anticipé à plusieurs niveaux (7/8)

- ***trois niveaux :*** **additionneur 32 bits à 3 niveaux de calcul de report anticipé**



7 couches au lieu de 9 avec des additionneurs anticipés montés en 2 niveaux

Addition à report anticipé à plusieurs niveaux (8/8)

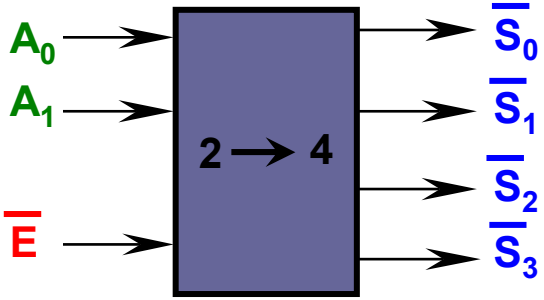
- **Tableau récapitulatif de la comparaison portant sur le report R0**

	Nombres de couches d'opérateurs			
Addition de 2 nombres	**8 bits**	**16 bits**	**32 bits**	**64 bits**
additionneurs complet avec report en	16(17)	32(33)	64(65)	128(129)
additionneurs à report anticipé (1 niveau)	7	9	17	33
additionneurs à report anticipé (2 niveaux)	-	7	9	19
additionneurs à report anticipé (3 niveaux)	-	-	7	9

Les valeurs entre parenthèses représentent le cas d'un additionneur complet réalisé à partir de deux demi-additionneurs.

Etude du décodeur (1/2)

- **Un décodeur est un circuit identificateur de minterms, il permet de réaliser la fonction de sélection**
  - Il possède n entrées et N sorties tel que $N=2^n$ (en général)
  - Il possède une entrée de validation ("Enable")



- **Table de vérité d'un décodeur de 2 vers 4**

E	A1	A0	S0	S1	S2	S3	Remarque
1	X	X	1	1	1	1	Inactif
0	0	0	0	1	1	1	Actif
0	0	1	1	0	1	1	
0	1	0	1	1	0	1	
0	1	1	1	1	1	0	

Etude du décodeur (2/2)

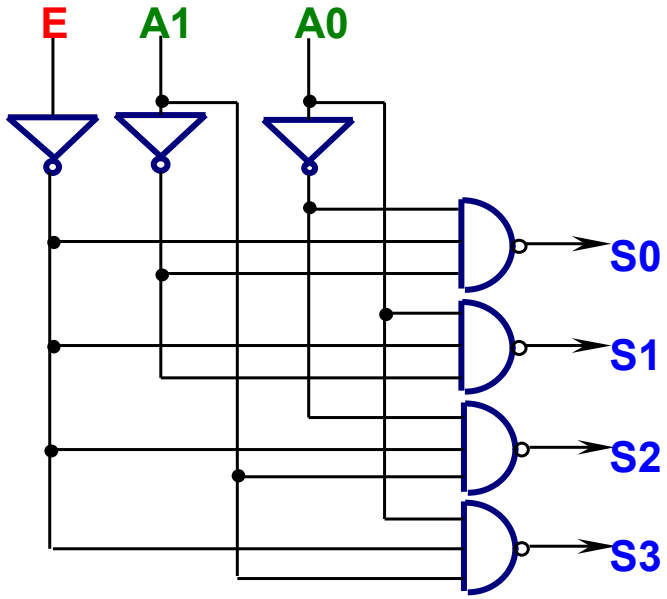
❖ **Détermination des fonctions S_0, S_1, S_2, S_3**

$$S_0 = E + A_1 + A_0 = \overline{\overline{E + A_1 + A_0}} = \overline{\overline{E}.\overline{A_1}.\overline{A_0}} = (E/E)/(A_1/A_1)/(A_0/A_0)$$

$$S_1 = E + A_1 + \overline{A_0} = \overline{\overline{E + A_1 + \overline{A_0}}} = \overline{\overline{E}.\overline{A_1}.A_0} = (E/E)/(A_1/A_1)/A_0$$

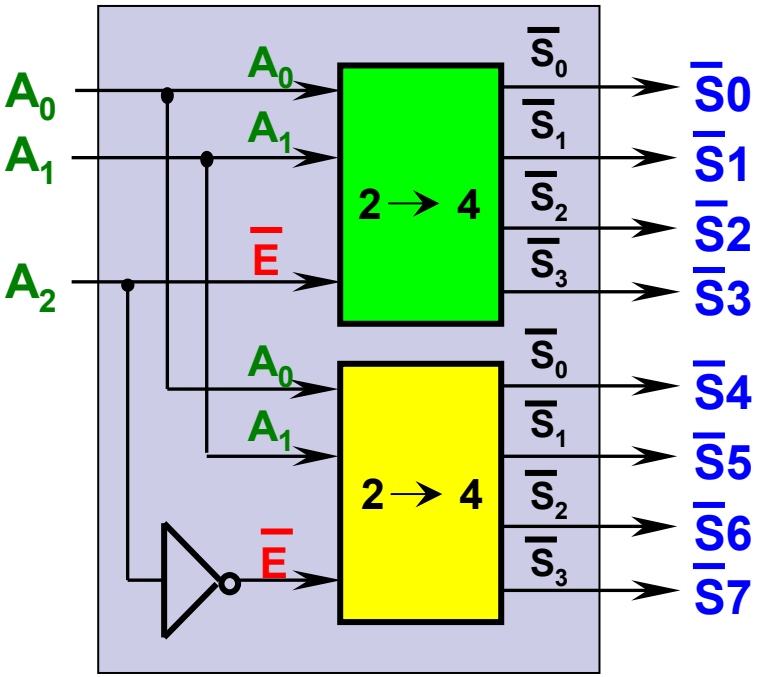
$$S_2 = E + \overline{A_1} + A_0 = \overline{\overline{E + \overline{A_1} + A_0}} = \overline{\overline{E}.A_1.\overline{A_0}} = (E/E)/A_1/(A_0/A_0)$$

$$S_3 = E + \overline{A_1} + \overline{A_0} = \overline{\overline{E + \overline{A_1} + \overline{A_0}}} = \overline{\overline{E}.A_1.A_0} = (E/E)/A_1/A_0$$



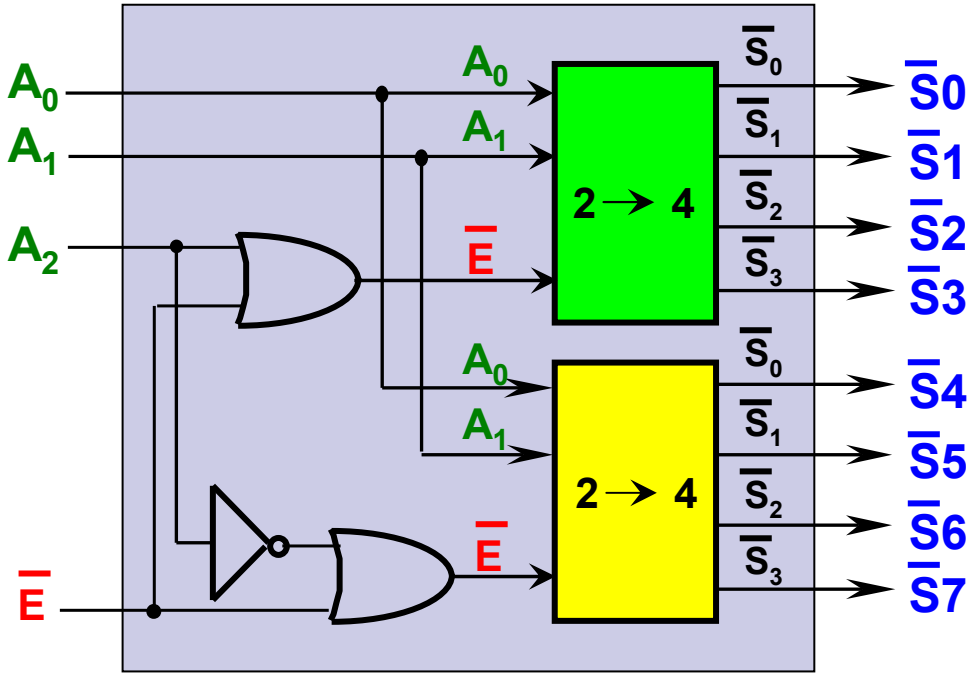
Association de décodeurs (1/2)

- Le signal de validation permet d'associer 2 décodeurs de 2 vers 4 pour obtenir un décodeur de 3 vers 8



Association de décodeurs (2/2)

❖ **Ajout d'un signal de validation global**

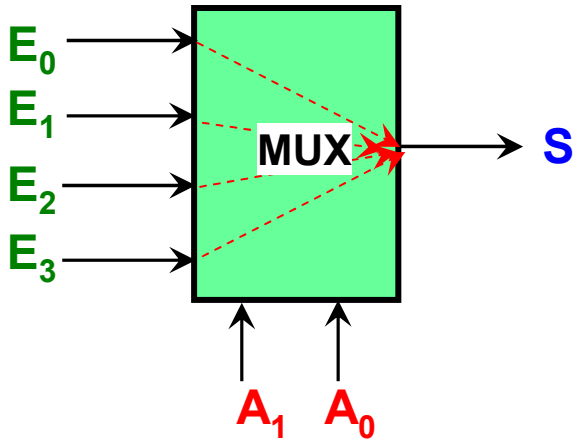


En procédant de façon récursive il est possible d'obtenir un décodeur de taille quelconque ! ! !

Etude du multiplexeur (1/2)

- **Un multiplexeur réalise la fonction d'aiguillage il est muni de :**
  - N entrées (voies) d'informations et n entrées (lignes) de contrôle, tel que $N = 2^n$
  - Une sortie (voie) d'information
  - Les lignes de contrôles permettent d'indiquer le numéro de l'entrée (voie) à aiguiller vers la sortie

  - Exemple : Multiplexeur de 4 vers 1

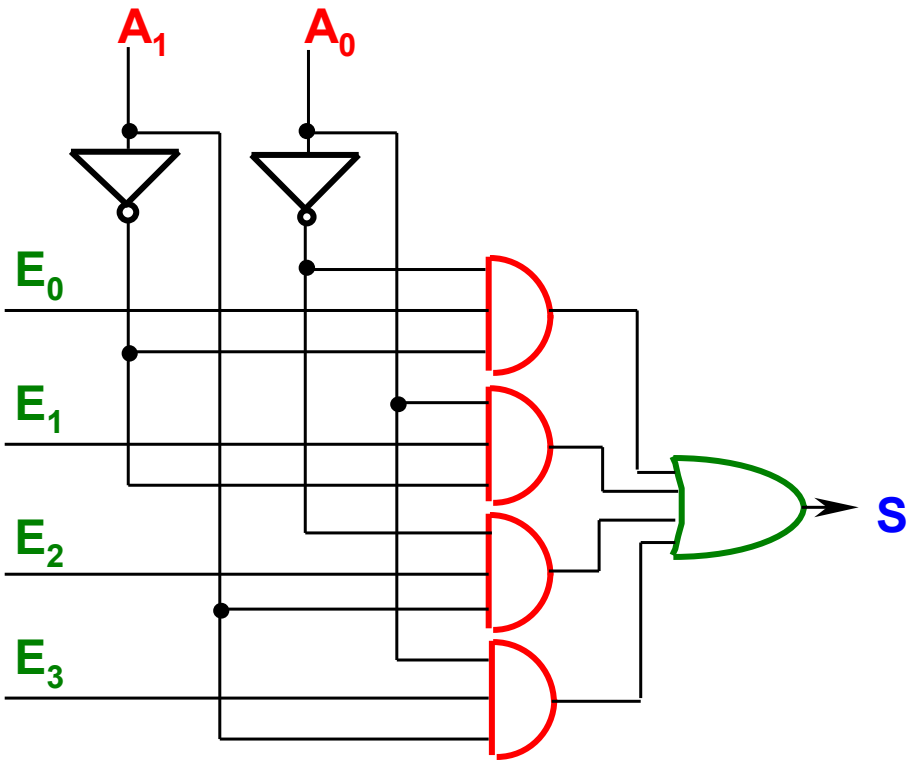


A_1	A_0	S
0	0	E_0
0	1	E_1
1	0	E_2
1	1	E_3

$$S = \overline{A_1}.\overline{A_0}.E_0 + \overline{A_1}.A_0.E_1 + A_1.\overline{A_0}.E_2 + A_1.A_0.E_3$$

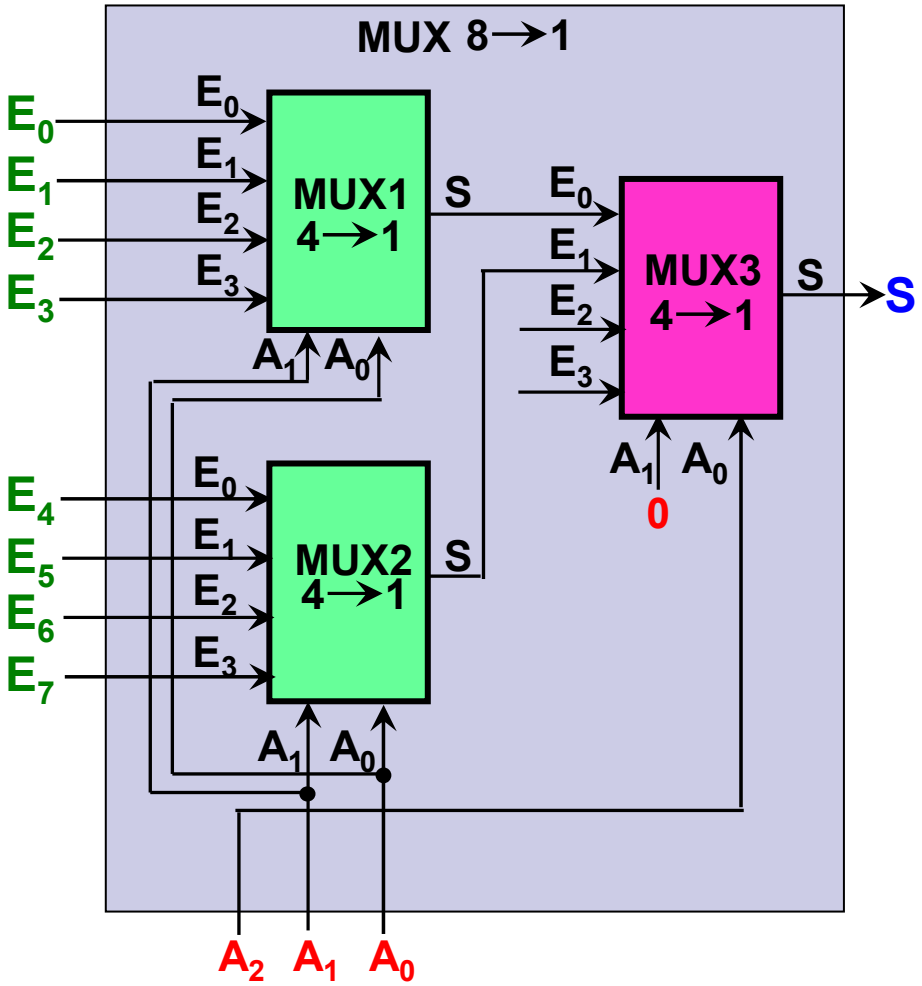
Etude du multiplexeur (2/2)

❖ **Schéma du multiplexeur de 4 vers 1**



Association de multiplexeurs

- Réalisation d'un multiplexeur de 8 vers 1 à partir de 3 multiplexeurs de 4 vers 1



FIN