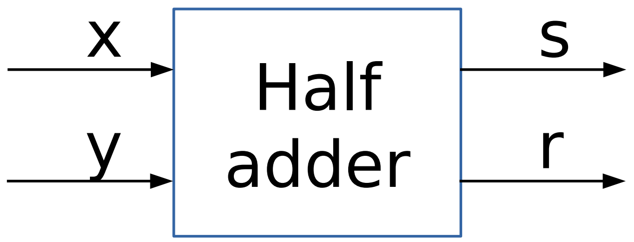
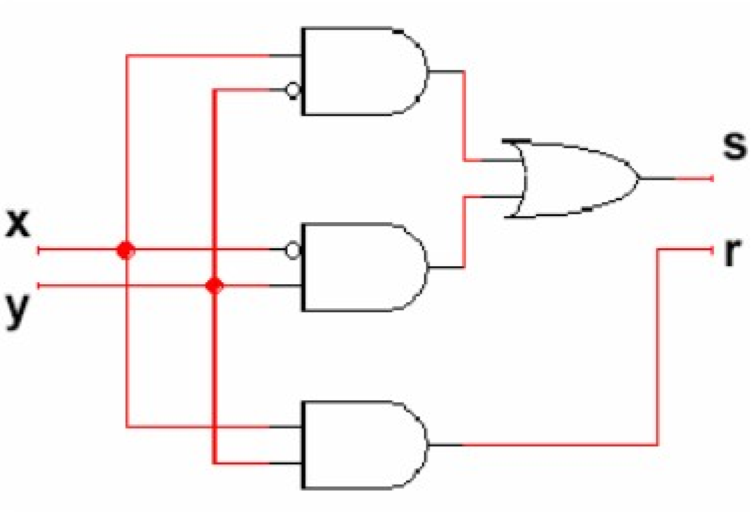
Università di Napoli Federico II – Scuola Politecnica e delle Scienze di Base
Corso di Laurea in Ingegneria Informatica

Corso di Calcolatori Elettronici I

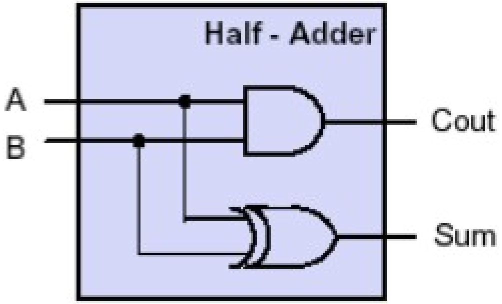
Macchine combinatorie: addizionatori binari

Half adder





x	y	s	r
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



$s = \overline{x}y + x\overline{y} = x \oplus y$

$r = xy$

Full adder (1/2)

x	y	r	S	R
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

S

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

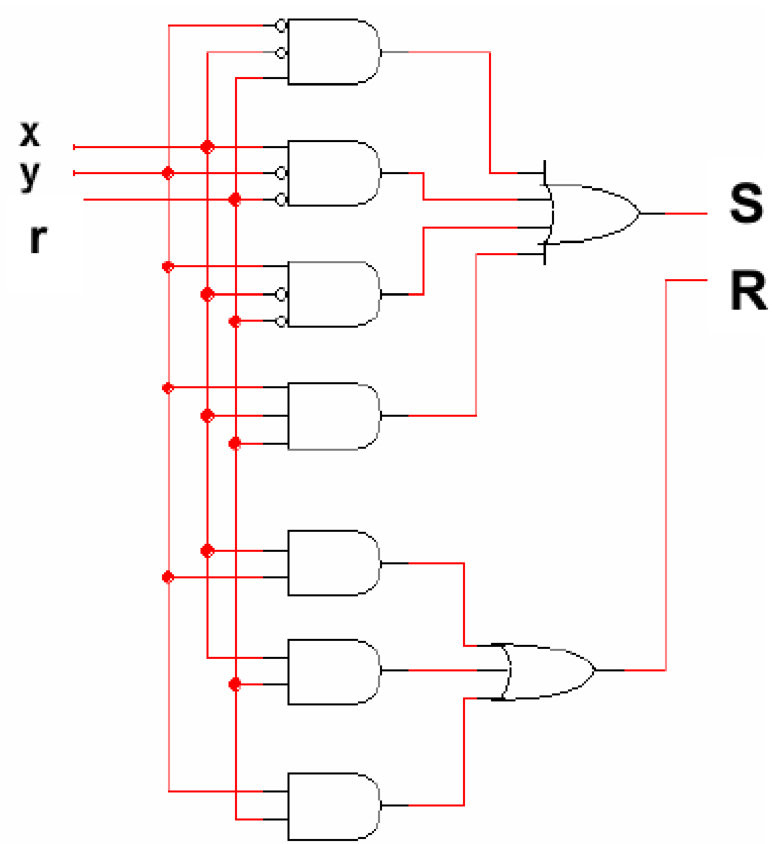
R

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

Full adder (2/2)

$$S = \bar{x}\bar{y}r + \bar{x}y\bar{r} + x\bar{y}\bar{r} + xyr$$

$$R = \bar{x}yr + x\bar{y}r + xy\bar{r} + xyr = xy + xr + yr$$



Addizionatore binario

- E’ possibile isolare il fattore (x ⊕ y)

$$S = \overline{x}\,\overline{y}r + \overline{x}y\overline{r} + x\overline{y}\,\overline{r} + xyr = (\overline{x}y+x\overline{y})\overline{r} + (\overline{x}\,\overline{y}+xy)r = (x\oplus y)\overline{r} + \overline{(x\oplus y)}r$$
$$= (x \oplus y) \oplus r$$

$$R = \overline{x}yr + x\overline{y}r + xy\overline{r} + xyr = (\overline{x}y+x\overline{y})r + xy = (x\oplus y)r + xy$$

- Da cui:

$$S = (x \oplus y) \oplus r = P \oplus r$$
$$R = x \cdot y + r \cdot (x \oplus y) = G + r \cdot P$$

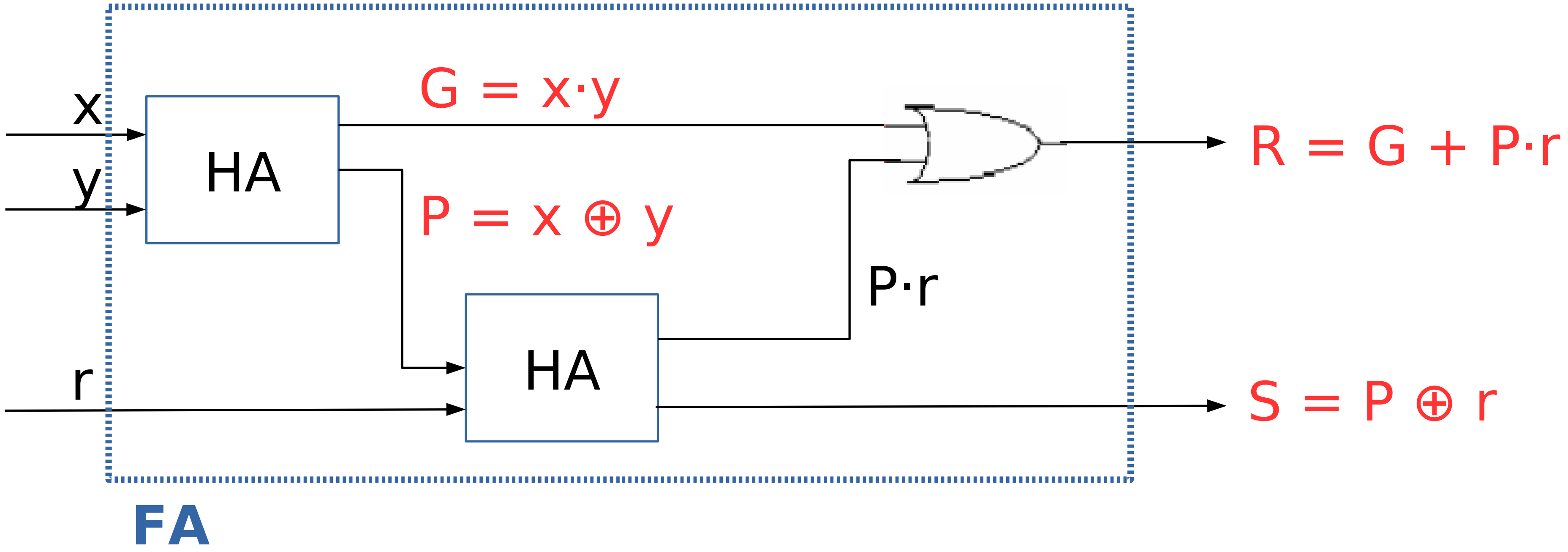
- Dove $P = x \oplus y$ e $G = x \cdot y$ sono rispettivamente le uscite somma e riporto di un half adder

Addizionatore binario

- Pertanto, un addizionatore completo può essere ottenuto a partire da due semiaddizionatori

$$S = (x \oplus y) \oplus r = P \oplus r$$

$$R = x \cdot y + r \cdot (x \oplus y) = G + r \cdot P$$



Addizionatore binario: riporto

- Le diverse componenti dell'espressione di R assumono un significato particolare:
  - **G = x·y "riporto generato"**: indica la creazione di un riporto all'interno dell'addizionatore binario
  - **P = x ⊕ y "riporto propagato"**: indica se, in presenza di un riporto in ingresso, lo stesso verrà propagato in uscita
- Il riporto in uscita può quindi essere espresso come **R=G+P·r**

Addizionatori binari

$n_i = \overline{r_i}$

Non-riporto
Indica assenza di riporto in ingresso (r=0)

$K_i = \overline{a_i}\,\overline{b_i}$

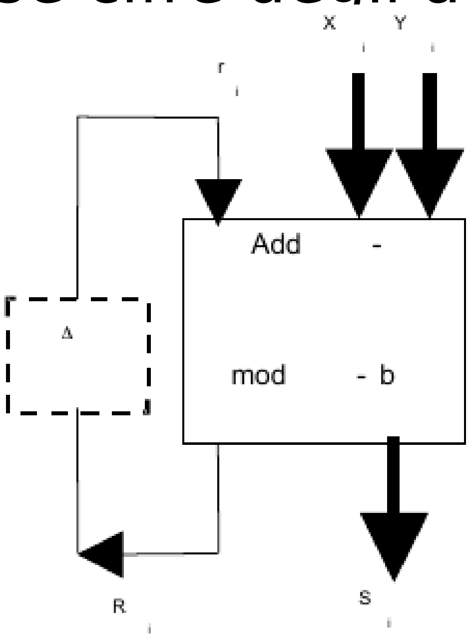
Riporto “ucciso” (“killed”)
Indica che, indipendentemente dalla presenza di un riporto entrante, il riporto in uscita sarà comunque zero

$N_i = K_i + P_i n_i$

Propagazione del non-riporto
Indica assenza di riporto in uscita (R=0)

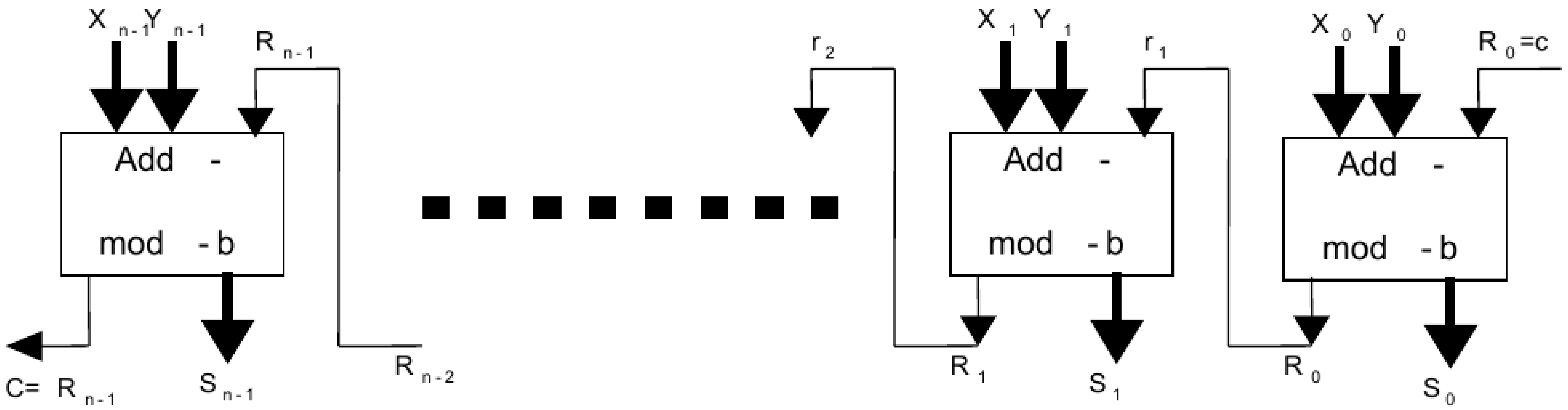
Addizionatori seriali

- Usa un unico addizionatore operante sulla singola cifra
- Opera in momenti successivi su cifre diverse degli addendi
- Richiede un blocco “con memoria”
- E’ lento rispetto ad addizionatori che lavorano in parallelo sulle diverse cifre degli addendi



Addizionatore binario parallelo

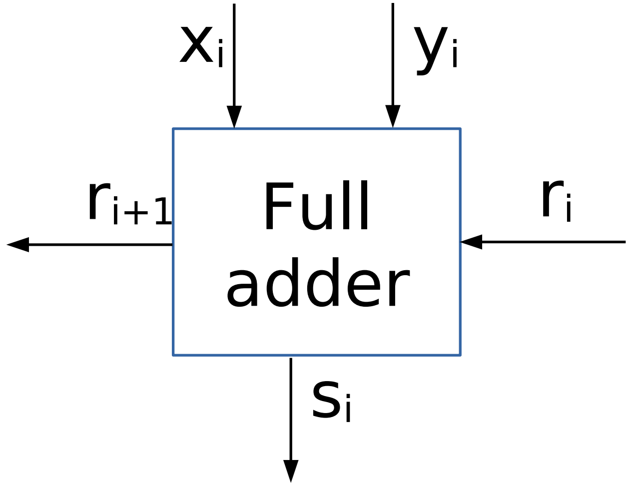
- Opera sulle cifre degli addendi in parallelo
- … anche se il riporto deve propagarsi attraverso l’intera struttura
- Richiede un numero maggiore di risorse rispetto all’addizionatore seriale



Addizionatore parallelo: tempo di risposta

- Gli addizionatori ottenuti collegando in cascata n addizionatori di cifra sono anche chiamati addizionatori a propagazione del riporto (*carry-ripple* o *carry-propagate*)
- ε = tempo di risposta di uno stadio
- Allo stadio i, il riporto uscente o è generato o è ucciso o è propagato
- Tempo di ritardo complessivo: Limite inferiore ε (in tutti gli stadi il riporto è generato o ucciso)
- Tempo di ritardo complessivo: Limite superiore $n\varepsilon$ (un riporto entrante nel primo stadio che è propagato in tutti gli stadi)
- Tempo di ritardo complessivo = $k\varepsilon$ ($k \leq n$), dove k è la più lunga catena di condizioni di propagazione

Adder con anticipo del riporto



$$r_{i+1} = G_i + P_i\, r_i$$
$$s_i = P_i \oplus r_i$$

$$G_i = x_i\, y_i$$
$$P_i = x_i \oplus y_i$$

$$r_{i+1} = G_i + P_i\, r_i = G_i + P_i\,(G_{i-1} + P_{i-1}\, r_{i-1}) = G_i + P_i\, G_{i-1} + P_i\, P_{i-1}\, r_{i-1} = \ldots\ldots$$
$$= G_i + P_i\, G_{i-1} + P_i\, P_{i-1}\, G_{i-2} + \ldots + P_i\, P_{i-1} \ldots P_1\, G_0 + P_i\, P_{i-1} \ldots P_0\, r_0$$

- Per ogni stadio i, r_{i+1} si può ottenere dai bit degli addendi X ed Y e dal riporto entrante nella catena r_0 , <u>dopo 3 ritardi di porta</u>
- Dopo un ulteriore ritardo di porta, sono disponibili i bit di somma

Adder con anticipo del riporto
Es. Riporti in un sommatore a 4 bit

$r_{i+1} = G_i + P_i\, r_i$

$s_i = P_i \oplus r_i$

$G_i = x_i\, y_i$

$P_i = x_i \oplus y_i$

$r_1 = G_0 + P_0 r_0$

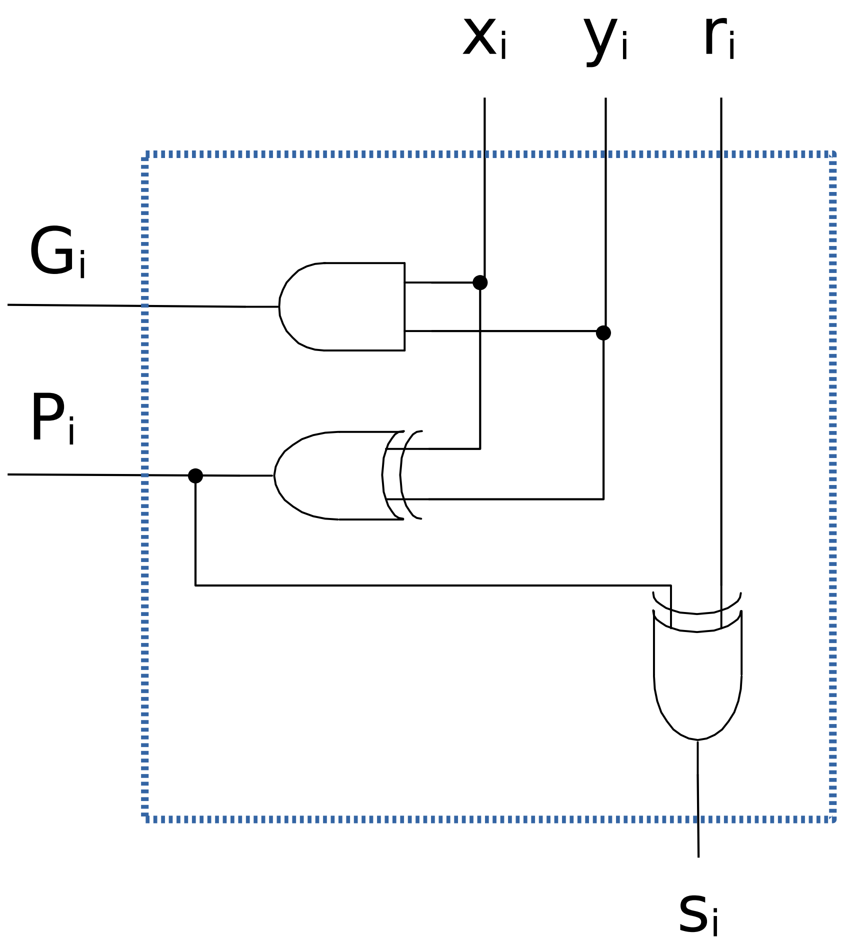
$r_2 = G_1 + P_1\, r_1 = G_1 + P_1(G_0 + P_0\, r_0) = G_1 + P_1 G_0 + P_1 P_0 r_0$

$r_3 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 r_0$

$r_4 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 r_0$

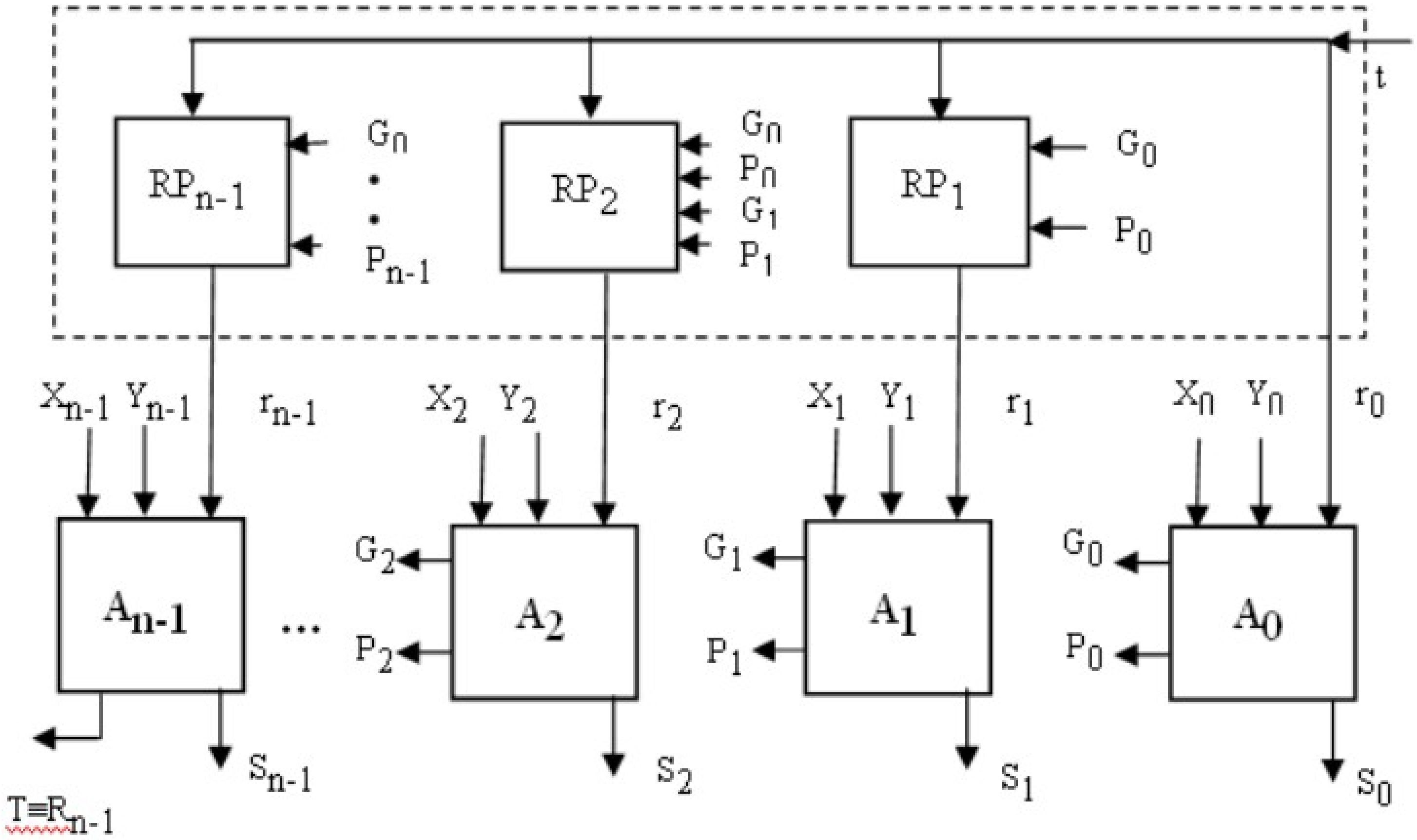
Adder con anticipo del riporto

- L'addizionatore carry lookahead (con anticipo del riporto) usa il seguente blocco al posto del full adder...



Addizionatori carry lookahead

- ...e una rete di lookahead per calcolare i riporti r_{i+1} $(0 \leq i \leq n\text{-}1)$ in base all'espressione ricavata in precedenza



Addizionatori carry lookahead

- Il fan-in di una porta è limitato nella pratica, quindi per n elevato non è possibile ottenere tutti gli r_i (soprattutto per i elevato) dopo tre ritardi di porta
- Ad esempio, un adder a 64 bit viene suddiviso in blocchi di quattro cifre (bit), la parte di "*look*" lungo cui si propaga il riporto è lunga 64/4=16 stadi
  - Il ritardo di propagazione dipende linearmente dal numero di blocchi