I sommatori

Prof. Alberto Borghese
Dipartimento di Informatica
borghese@di.unimi.it

Università degli Studi di Milano

Riferimenti: Appendice B5 prima parte.

Sommario

Addizionatori

Addizionatori ad anticipazione di riporto

Implementazione di funzioni algebriche

And, Or, Not per ottenere:

Operazioni algebriche (somme, prodotti, sottrazioni e divisioni) su numeri binari.

Operazioni logiche su numeri binari.

Operazione di somma

```
1110      ← Riporto          111
  1011 +  ← Addendo 1       01011 +
   110 =  ← Addendo 2       00110 =
-------                     -------
10001                       10001
```

3 Attori: addendo 1, addendo 2, riporto.

Gli addendi sono presenti all'inizio
Il riporto viene generato via via che la somma viene svolta

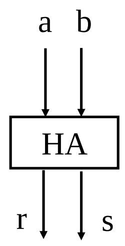
Viene eseguita sequenzialmente da dx a sx.

(Half) Adder a 1 bit

Tabella della verità della somma:

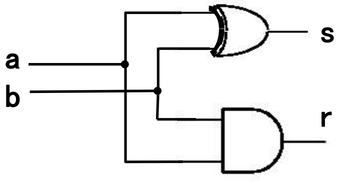
a b	somma	riporto
0 0	0	0
0 1	1	0
1 0	1	0
1 1	0	1

a +
b =



a b	xor
0 0	0
0 1	1
1 0	1
1 1	0

$s = a \oplus b$
$r = ab$



La somma è diventata un'operazione logica!

Cammini critici:
Somma = 1;
Riporto = 1;

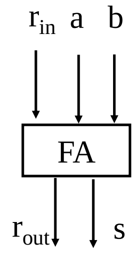
Complessità
Somma = 1 porta;
Riporto = 1 porta;

Full Adder a 1 bit

Tabella della verità della somma completa:

a b r_{in}	somma	riporto
0 0 0	0	0
0 1 0	1	0
1 0 0	1	0
1 1 0	0	1
0 0 1	1	0
0 1 1	0	1
1 0 1	0	1
1 1 1	1	1

r_{in} +
a +
b



$s = m_1 + m_2 + m_4 + m_7$
$r = m_3 + m_5 + m_6 + m_7$

Full Adder a 1 bit – espressione logica di s

Tabella della verità della somma completa:

$$s = m_1 + m_2 + m_4 + m_7$$

a b r_{in}	somma	riporto
0 0 0	0	0
0 1 0	1	0
1 0 0	1	0
1 1 0	0	1
0 0 1	1	0
0 1 1	0	1
1 0 1	0	1
1 1 1	1	1

$$s = \overline{a}\, b\, \overline{r_{in}} + a\, \overline{b}\, \overline{r_{in}} + \overline{a}\, \overline{b}\, r_{in} + a\, b\, r_{in} =$$

$$= (\overline{a}b + a\overline{b})\overline{r_{in}} + (\overline{a}\,\overline{b} + ab)\, r_{in} =$$

$$= (a \oplus b)\, \overline{r_{in}} + (\overline{a}\,\overline{b} + ab)\, r_{in} =$$

$$= (a \oplus b)\, \overline{r_{in}} + \overline{(a \oplus b)}\, r_{in}$$

XOR

a b	y	$\overline{y}$
0 0	0	1
0 1	1	0
1 0	1	0
1 1	0	1

$XOR(a,b) = (\overline{a}b + a\overline{b})$

$!XOR(a,b) = XNOR(a,b) = (\overline{a}\,\overline{b} + ab)$

Full Adder a 1 bit – espressione logica di r_{out}

Tabella della verità della somma completa:

$$r = m_3 + m_5 + m_6 + m_7$$

a b r_{in}	somma	riporto
0 0 0	0	0
0 1 0	1	0
1 0 0	1	0
1 1 0	0	1
0 0 1	1	0
0 1 1	0	1
1 0 1	0	1
1 1 1	1	1

$$r_{out} = a\, b\, \overline{r_{in}} + \overline{a}\, b\, r_{in} + a\, \overline{b}\, r_{in} + a\, b\, r_{in} =$$

1) $ab + (a \oplus b)\, r_{in}$

2) $a\, r_{in} + (a \oplus r_{in})\, b$

Quale è meglio?

Implementazione circuitale

$s = (a \oplus b)\,\overline{r_{in}} + \overline{(a \oplus b)}\,r_{in}$

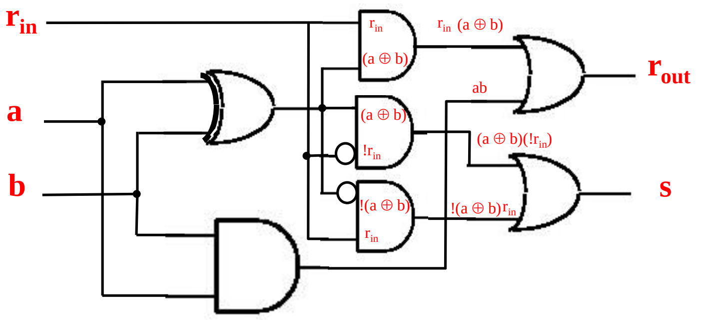
$r_{out} = ab + (a \oplus b)\,r_{in}$

Complessità: 7 porte logiche
(Riutilizzo l'XOR 2 volte)

$s = (a \oplus b)\,\overline{r_{in}} + \overline{(a \oplus b)}\,r_{in}$

$r_{out} = a\,r_{in} + (a \oplus r_{in})\,b$

Complessità: 8 porte logiche
(Riutilizzo l'XOR 1 volta)



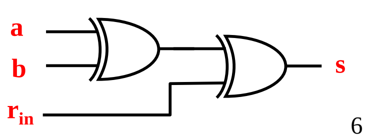
Complessità: 7 porte logiche.
Cammini critici: s -> 3; r_{out} -> 3

Semplificazione circuitale

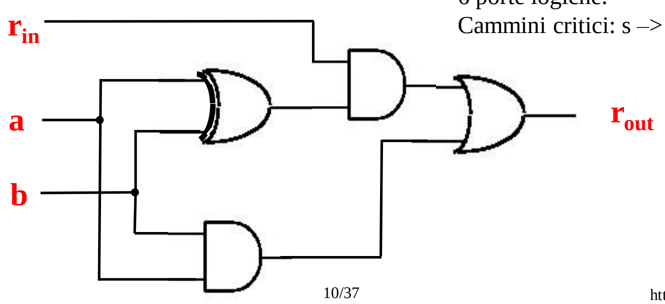
$s = (a \oplus b)\,\overline{r_{in}} + \overline{(a \oplus b)}\,r_{in} = a \oplus b \oplus r_{in}$

$z \triangleq (a \oplus b) \Rightarrow z\,\overline{r_{in}} + \overline{z}\,r_{in} = (z \oplus r_{in}) = ((a \oplus b) \oplus r_{in}) = a \oplus b \oplus r_{in}$

$r_{out} = ab + (a \oplus b)\,r_{in}$



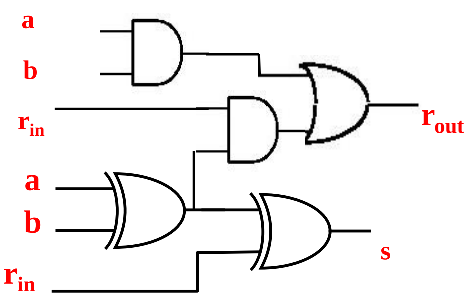
6 porte logiche.
Cammini critici: s –> 2; r_{out} -> 3



Semplificazione ulteriore

$s = a \oplus b \oplus r_{in}$

$r_{out} = ab + (a \oplus b)\, r_{in}$



5 porte logiche.
Cammini critici: s –> 2; r_{out} -> 3

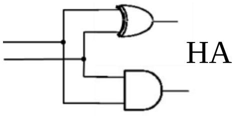
s - rilevatore di (dis)parità

r_{out} – riporto se generato (a = b =1) o se propagato ($a \oplus b$ =1) $r_{out} = r_{in}$

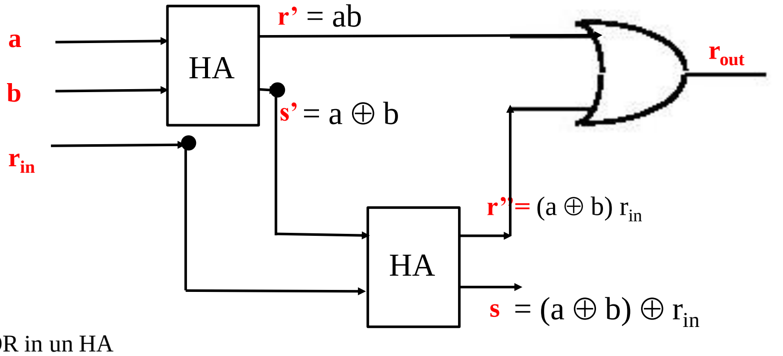
Circuito costruito con HA

$s = a \oplus b \oplus r_{in}$

$r_{out} = ab + (a \oplus b)\, r_{in}$



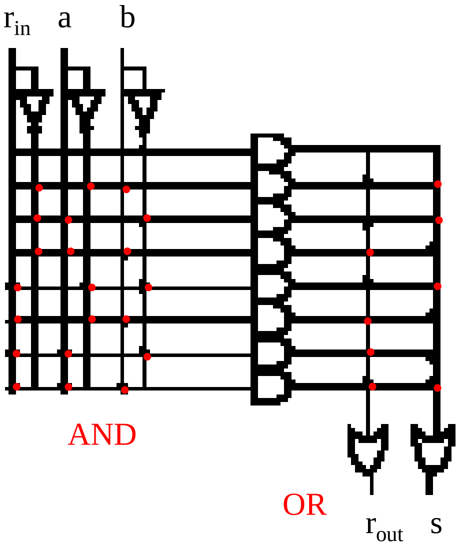
5 porte logiche.
Cammini critici: s –> 2; r_{out} -> 3



Raggruppo
AND e XOR in un HA

Implementazione mediante PLA

a b r_{in}	somma	r_{out}
0 0 0	0	0
0 1 0	1	0
1 0 0	1	0
1 1 0	0	1
0 0 1	1	0
0 1 1	0	1
1 0 1	0	1
1 1 1	1	1



SOP: costruisco i mintermini e li sommo

Esercizi con ROM e PLA

Implementare il circuito del Full Adder mediante ROM

Scrivere il circuito che esegue la somma di: 3 + 4 in base 2. Riportare <u>tutte le uscite</u> delle porte logiche.

Scrivere il circuito che esegue la seguente sottrazione: 5-2 in base 2. Riportare <u>tutte le uscite</u> delle porte logiche.

Sommario

Addizionatori

Addizionatori ad anticipazione di riporto

Operazione di somma

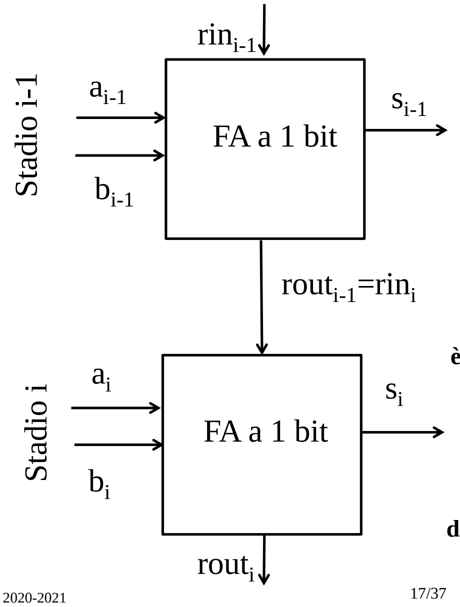
```
 1110        ← Riporto
01011 +      ← Addendo 1
00110 =      ← Addendo 2
-------
10001
```

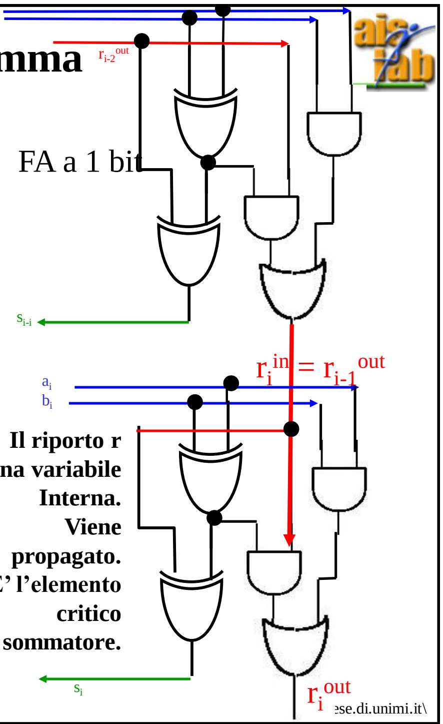
Per ogni bit ho 3 Attori: addendo 1, addendo 2, riporto.

Gli addendi sono presenti all'inizio
Il riporto viene generato via via che la somma viene svolta

Viene eseguita sequenzialmente da dx a sx.

Circuito della somma





Il riporto r è una variabile Interna. Viene propagato. E' l'elemento critico del sommatore.

Cammini critici

Per ogni stadio:
Somma: 2
Riporto: 3

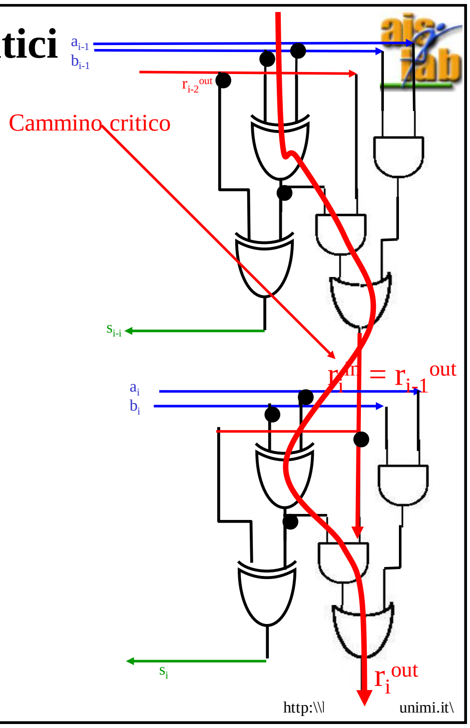
Per due stadi:
Somma: 3 + 2 = 5 (devo aspettare r_{in})
Riporto: 3 + 3 = 6

Riporto per N stadi: $r_{out3} = 3 * N$

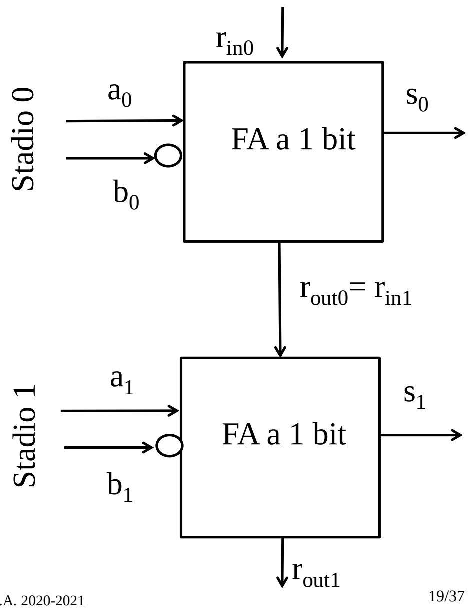
1110
01011 +
00110 =

10001

Funzionamento sequenziale



Circuito della sottrazione



Sommo i seguenti 2 numeri 11 + (-13):
$A = 01011_2 = 11_{10}$
$B = 10011_2 = -13_{10}$

E' equivalente ad effettuare la differenza:
$S = A - B = 11 - 13$

$B = +13_{10} = 01101_2$ =>

complemento a 1 => 10010 $\overline{B} = B$

Sommo 1 per ottenere il complemento a 2:
10010 **+ 1** = $10011_2 = -13_{10}$

Il complemento a 2 di B, -B, si ottiene come:

$-B = (\overline{B} + 1)$

La sottrazione diventa $S = A + (\overline{B} + 1)$

Nella sottrazione avrò $r_{in0} = +1$

I problemi del full-adder

Il full adder con propagazione del riporto è lento.

- Il riporto si propaga sequenzialmente
  caratteristica dell'algoritmo di calcolo

- la commutazione dei circuiti non è istantanea (tempo di commutazione)
  caratteristica fisica dei dispositvi

- Soluzioni
  modificare l'algortimo
  modificare i dispositivi

Prima possibilità: forma tabellare

Riscrivo le equazioni del riporto in modo non sequenziale. Come?

$$\boxed{\{r_{out3}, r_{out2}, r_{out1}, r_{out0}\} = f(r_{in0},\ a_0,\ b_0,\ a_1,\ b_1,\ a_2,\ b_2,\ a_3,\ b_3, \ldots.)}$$

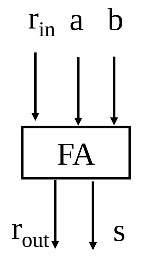
Scrivo la tabella della verità dove in uscita ho gli N riporti ed
In ingresso 2 * N valori (gli N bit dei 2 addendi).

La tabella della verità ha $2^{(2N+1)}$ righe (per N=32, …)

Carry look-ahead (anticipazione di riporto)

Approccio strutturato per diminuire la latenza della somma.

$r_{out} = ab + (a \oplus b)\ r_{in}$



Analisi del singolo stadio.
Quando si genera un riporto in uscita?

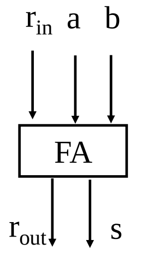
Quando ho almeno due 1, in ingresso; cioè almeno due «1» tra r_{in}, a e b.

11000 riporto
01101 +
00100 =

10001

Propagazione e generazione

Ho riporto quando ho almeno due 1, in ingresso; cioè tra r_{in}, a e b.



Osservazioni:

1) Viene generato un riporto dallo stadio i, qualsiasi sia il riporto in ingresso se a = b = 1 => $g_i = a_i b_i$.

2) Viene generato un riporto allo stadio i, se il riporto in ingresso è = 1 ed una delle due variabili in ingresso è = 1 => se $p_i = (a_i \oplus b_i)$ => viene generato riporto se $p_i r_i^{in} = 1$ (p_i propaga il segnale di riporto r_i^{in}).

a b r_{in}	somma	riporto
0 0 0	0	0
0 1 0	1	0
1 0 0	1	0
1 1 0	0	1
0 0 1	1	0
0 1 1	0	1
1 0 1	0	1
1 1 1	1	1

Quando sia la condizione 1) che la condizione 2) è verificata? Cosa succede se entrambe le condizioni sono verificate?

Esempio

Sono interessato ad r_4^{out}. Supponiamo anche il riporto in ingresso al primo stadio: $r_0^{in} = 0$.

r_{in}	0 0 0 0 0 0 0	0 1 1 1 0 0 0	0 1 1 1 0 0 0
a	1 0 1 0 1 1 0 1 +	1 0 1 0 1 1 0 1 +	1 0 1 1 1 1 0 1 +
b	0 0 0 1 0 0 0 0 =	0 0 0 1 1 0 1 0 =	0 0 0 1 1 0 0 0 =
	----------------	----------------	----------------
	1 0 1 1 1 1 1 1	1 1 1 0 0 1 1 1	1 1 0 1 0 1 0 1

$r_5^{in} = r_4^{out} = 0$

$r_5^{in} = r_4^{out} = 1$

Per propagazione

$p_4 = (a_4 \oplus b_4) r_4^{in}$.

$r_5^{in} = r_4^{out} = 1$

Per generazione

$g_4 = a_4 b_4$

Sviluppo della funzione logica riporto

$r_i^{out} = ab + (a \oplus b)\, r_i^{in}$

$r_i^{out} = g_i + p_i\, r_i^{in}$

$r^{out}_0 = g_0 + p_0 r^{in}_0$

$r^{out}_1 = g_1 + p_1 r^{in}_1 = g_1 + p_1 g_0 + p_1 p_0 r^{in}_0$

r^{out}_1 r^{in}_0

111
1001 +
0010 =

1100

$g_0 = 0$
$p_0 = p_1 = 1$

r^{out}_1

110
1001 +
0011 =

1100

$g_0 = 1$
$p_1 = 1$

r^{out}_1

1 0
1010 +
0011 =

1100

$g_1 = 1$

Sviluppo della funzione logica riporto

$r_i^{out} = ab + (a \oplus b)\, r_i^{in}$

$r_i^{out} = g_i + p_i\, r_i^{in}$

$r_0 = g_0 + p_0 r_0$

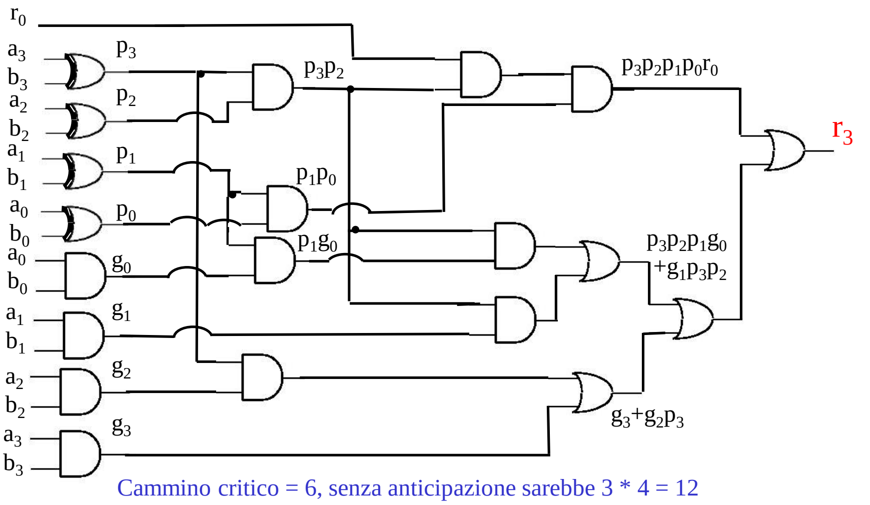
$r_1 = g_1 + p_1 r_0 = g_1 + p_1 g_0 + p_1 p_0 r_0$

$r_2 = g_2 + p_2 r_1 = g_2 + p_2(g_1 + p_1 g_0 + p_1 p_0 r_0) = g_2 + p_2 g_1 + p_2 p_1 g_0 + p_2 p_1 p_0 r_0.$

$r_3 = g_3 + p_3 r_2 = g_3 + p_3(g_2 + p_2 g_1 + p_2 p_1 g_0 + p_2 p_1 p_0 r_0) = g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 g_0 + p_3 p_2 p_1 p_0 r_0.$

Determinazione del cammino critico.

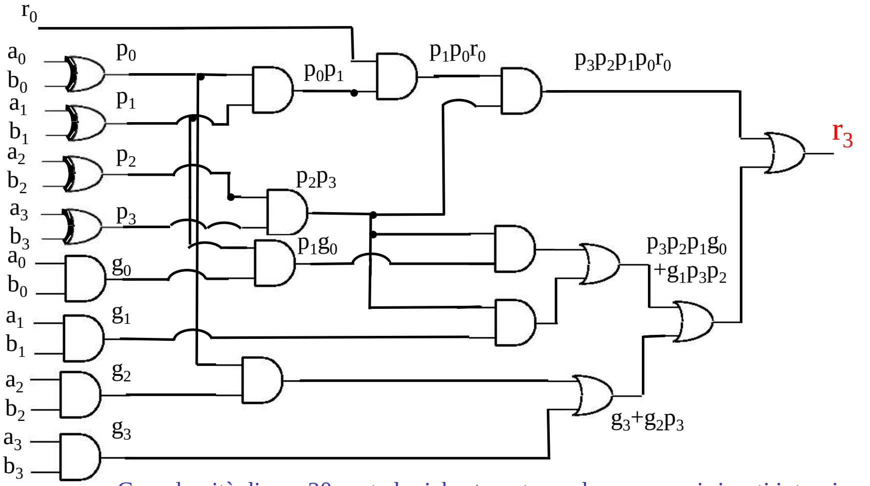
$r_3 = g_3 + p_3 r_2 = g_3 + p_3(g_2 + p_2 g_1 + p_2 p_1 g_0 + p_2 p_1 p_0 r_0) =$
$g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 g_0 + p_3 p_2 p_1 p_0 r_0$



Cammino critico = 6, senza anticipazione sarebbe 3 * 4 = 12

Determinazione la complessità.

$r_3 = g_3 + p_3 r_2 = g_3 + p_3(g_2 + p_2 g_1 + p_2 p_1 g_0 + p_2 p_1 p_0 r_0) =$
$g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 g_0 + p_3 p_2 p_1 p_0 r_0.$



Complessità di r_3 = 20 porte logiche + porte per la somma e i riporti interni

Complessità aggiuntiva per gli altri bit di riporto

$r_2 = g_2 + p_2r_1 = g_2 + p_2(g_1 + p_1g_0 + p_1p_0r_0) =$
$g_2 + p_2g_1 + p_2p_1g_0 + p_2p_1p_0r_0$

In rosso le porte già presenti nel circuito di r_{out3}

Complessità aggiuntiva pari a 6 porte logiche.

$r_1 = g_1 + p_1r_0 = g_1 + p_1g_0 + p_1p_0r_0$

Complessità aggiuntiva pari a 2 porte logiche.

Complessità aggiuntiva totale: 8 porte logiche.

Complessità aggiuntiva per i bit di somma

$$s_k = (a_k \oplus b_k) \oplus r_{k\,in} = p_k \oplus r_{k\,in}$$

Ogni bit di somma aggiunge una porta logica XOR =>
La complessità aumenta di N * 1 = 4 porte logiche.

Un CLA su 4 bit ha quindi una complessità di 20 + 6 + 2 + 4 = 32 porte logiche.

Un sommatore a propagazione di riporto ha una complessità di 4*5 = 20 porte logiche.

Quanto si guadagna con l'anticipazione del riporto per N stadi?

Cammino critico per le variabili interne:

r_0^{out} => 3
r_1^{out} => 4
r_2^{out} => 5

Cammino critico per le variabili esterne:

r_3^{out} => 6
s_3 => 6 NB la prima porta XOR è in comune con r_2^{out}
s_2 => 5 NB la prima porta XOR è in comune con r_1^{out}
s_1 => 4 NB la prima porta XOR è in comune con r_0^{out}
s_0 => 2

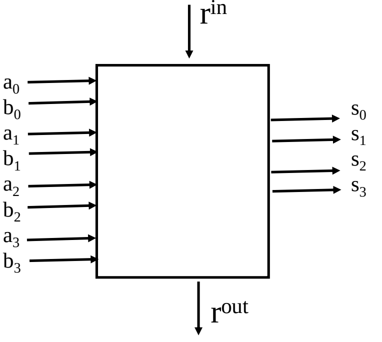
Cammino critico scala come $CC_{(1\ stadio)}$ *log(N)

Addizionatori modulari

La complessità del circuito è tollerata per piccoli n.

Circuiti sommatori indipendenti si hanno per 4 bit.

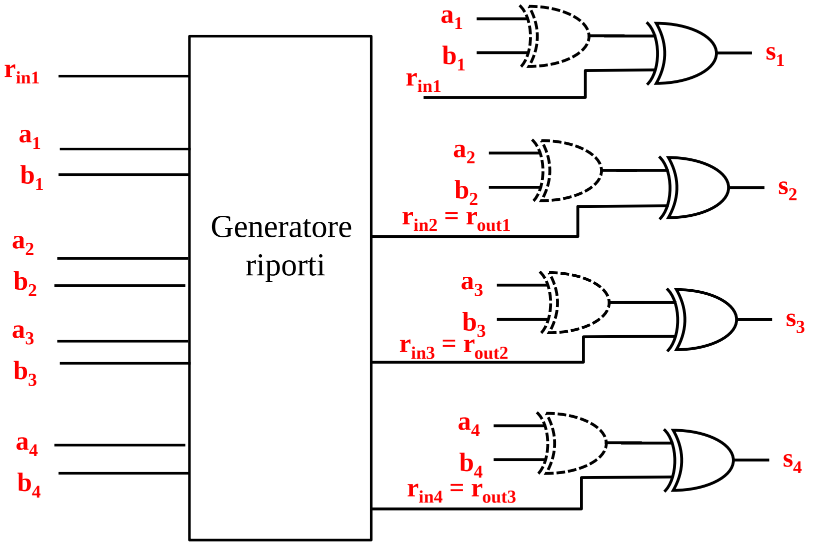
Moduli elementari.



Come si ottiene la somma?

Collegando in cascata i moduli (sommatori elementari).
Cammino critico = 6 (CC di un modulo a 4 bit) * N/4. Per 32 bit, 48.
Per confronto, senza parallelizzazione, sommatore a propagazione di riporto, per 32 bit, N * 3 = 96.

Architettura interna

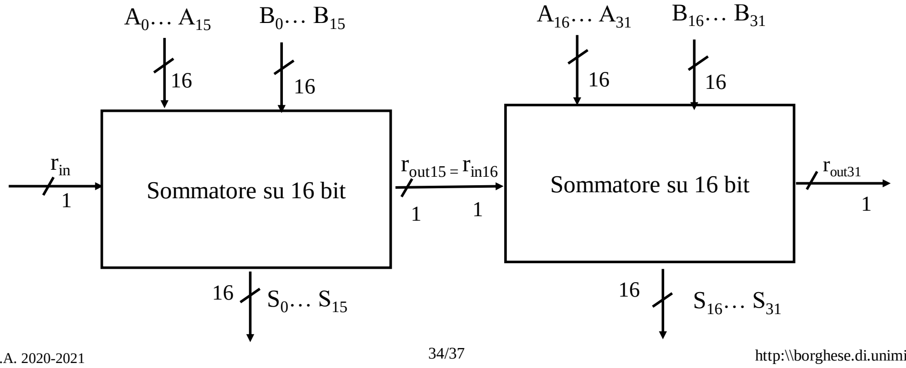


Addizionatori modulari::esempio

Occorre sommare 2 variabili, A e B, su N = 32 bit
Ho a disposizione due sommatori su 16 bit.

$CC = CC_{(1\ stadio)} * \log(N) = 3 * 4 = 12$

Come si ottiene la somma?
Fondamento delle estensioni architetturali SSE



OR su più bit

1	0	0	1

OR

1	1	0	0

=

1	1	0	1

Ogni bit viene elaborato separatamente

AND su più bit

1	0	0	1

AND

1	1	0	0

=

1	0	0	0

Ogni bit viene elaborato separatamente

Sommario

Addizionatori

Addizionatori ad anticipazione di riporto