



Moltiplicatori HW e ALU

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

Università degli Studi di Milano

Riferimenti: Appendice B5 prima parte.
Per approfondimenti, Capitolo 7 del Fummi, Sami, Silvano.



Sommario

Moltiplicatori

ALU



Moltiplicazione mediante shift



Lo shift di un numero a dx, di k cifre, corrisponde ad una divisione per la base elevata alla k-esima potenza.

Lo shift di un numero a sx, di k cifre, corrisponde ad una moltiplicazione per la base elevata alla k-esima potenza.

Esempio nel caso decimale:

$$\begin{aligned}213_{10} \times 10 &= 2130_{10} \\ \mathbf{21}3_{10} &= (2 \times 10^2 + \mathbf{1} \times 10^1 + 3 \times 10^0) \times \mathbf{10^1} = \\ &= (2 \times 10^2 + \mathbf{1} \times 10^1 + 3 \times 10^0) \times \mathbf{10^1} = \\ &= (2 \times 10^2 \times \mathbf{10^1} + \mathbf{1} \times 10^1 \times \mathbf{10^1} + 3 \times 10^0 \times \mathbf{10^1}) = \\ &= (2 \times 10^3 + \mathbf{1} \times 10^2 + 3 \times 10^1) = 2130 \text{ cvd.}\end{aligned}$$

$$\begin{aligned}213_{10} / 10 &= 21,3_{10} \\ \mathbf{21}3_{10} &= (2 \times 10^2 + \mathbf{1} \times 10^1 + 3 \times 10^0) / \mathbf{10^1} = \\ &= (2 \times 10^2 + \mathbf{1} \times 10^1 + 3 \times 10^0) \times \mathbf{10^{-1}} = \\ &= (2 \times 10^2 \times \mathbf{10^{-1}} + \mathbf{1} \times 10^1 \times \mathbf{10^{-1}} + 3 \times 10^0 \times \mathbf{10^{-1}}) = \\ &= (2 \times 10^1 + \mathbf{1} \times 10^0 + 3 \times 10^{-1}) = 21,3 \text{ cvd.}\end{aligned}$$



Moltiplicazione mediante shift



Lo shift di un numero a dx, di k cifre, corrisponde ad una divisione per la base elevata alla k-esima potenza.

Lo shift di un numero a sx, di k cifre, corrisponde ad una moltiplicazione per la base elevata alla k-esima potenza.

Esempio nel caso binario:

$$\begin{aligned}23 * 4 = 92 &\Rightarrow 10111 * 100 = 1011100 \\ &\textit{Esprimendo l'operazione in decimale:} \\ &= (1x2^4 + 0x2^3 + 1x2^2 + 1x2^1 + 1x2^0) \times 2^2 = \\ &= (1x2^6 + 0x2^5 + 1x2^4 + 1x2^3 + 1x2^2) = 64 + 16 + 8 + 4 = 92 \text{ cvd.}\end{aligned}$$

$$\begin{aligned}23 / 4 = 5,75 &\Rightarrow 10111 / 100 = 101,11 \\ &\textit{Esprimendo l'operazione in decimale:} \\ &= (1x2^4 + 0x2^3 + 1x2^2 + 1x2^1 + 1x2^0) \times 2^{-2} = \\ &= (1x2^2 + 0x2^1 + 1x2^0 + 1x2^{-1} + 1x2^{-2}) = 5,75 \text{ cvd.}\end{aligned}$$



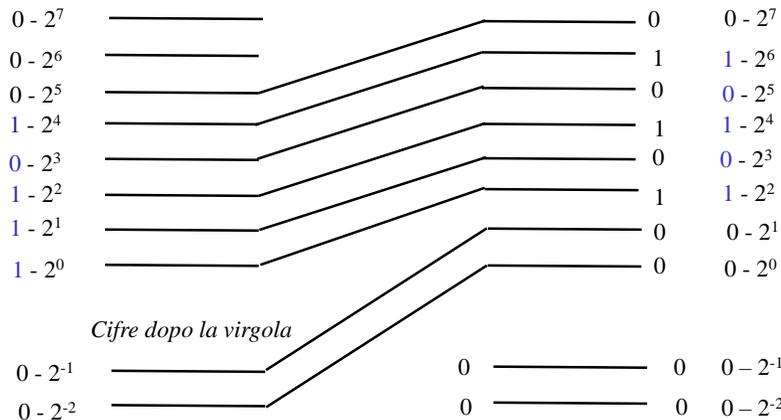
Moltiplicazione mediante shift



Codifica su 8 + 2 cifre:

$$23 * 4 = 92 \Rightarrow 10111 * 100 = 1011100$$

$$23_{10} = 00010111,00_2$$



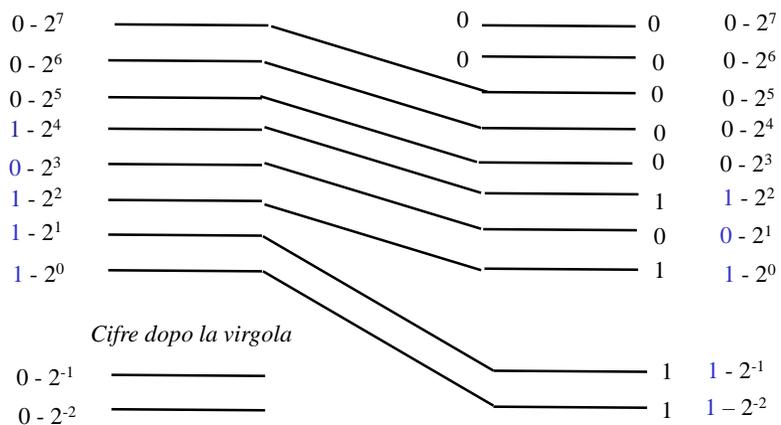
Divisione mediante shift



Codifica su 8 + 2 cifre:

$$23 / 4 = 5,75 \Rightarrow 10111 / 100 = 101,1100$$

$$23_{10} = 00010111,00_2$$





Moltiplicazione decimale

$$\begin{array}{r} \text{Moltiplicando} \longrightarrow 278 \times \\ \text{Moltiplicatore} \longrightarrow 423 = \end{array}$$

Prodotti parziali \longrightarrow

$$\begin{array}{r} \text{-----} \\ 834 + \\ 556 - \\ 1112 - - \\ \text{-----} \end{array}$$

$$\text{prodotto} \longrightarrow 117594$$

$$278 \times 423 = 278 \times (4 \times 10^2 + 2 \times 10^1 + 3 \times 10^0) = 278 \times (4 \times 10^2) + 278 \times (2 \times 10^1) + 278 \times (3 \times 10^0)$$

Somma dei prodotti parziali



Moltiplicazione binaria - I

$$\begin{array}{r} \text{Moltiplicando} \longrightarrow 11011 \times \\ \text{Moltiplicatore} \longrightarrow 11(1) = \end{array}$$

$$\begin{array}{r} 11011 \times 27_{10} \\ 111 = 7_{10} \end{array}$$

1° prodotto parziale

$$11011 +$$

$$\begin{array}{r} \text{-----} \\ 111111 \\ 11011 + \\ 11011 - \\ 11011 - - \\ \text{-----} \end{array}$$

$$10111101 \quad 189_{10}$$



Moltiplicazione binaria - II



$$\begin{array}{r} \text{Moltiplicando} \longrightarrow 11011x \\ \text{Moltiplicatore} \longrightarrow 1(1)1 = \end{array}$$

$$\begin{array}{r} 11011x \quad 27_{10} \\ 111 = \quad 7_{10} \\ \hline 111111 \\ 11011+ \\ 11011- \\ 11011- - \\ \hline 10111101 \quad 189_{10} \end{array}$$

$$\begin{array}{r} \hline 11011+ \\ 11011- \\ \hline \end{array}$$

prodotti parziali

Il secondo prodotto parziale è incolonnato alle potenze di 2^1 (la cifra del moltiplicatore ha peso 2^1)

Ho generato 2 addendi (2 prodotti parziali)
Provvedo subito alla loro somma



Moltiplicazione binaria - III



$$\begin{array}{r} \text{Moltiplicando} \longrightarrow 11011x \\ \text{Moltiplicatore} \longrightarrow 111 = \end{array}$$

$$\begin{array}{r} 11011x \quad 27_{10} \\ 111 = \quad 7_{10} \\ \hline 111111 \\ 11011+ \\ 11011- \\ 11011- - \\ \hline 10111101 \quad 189_{10} \end{array}$$

$$\begin{array}{r} 11111 \\ 11011+ \\ 11011- \\ \hline \end{array}$$

2 prodotti parziali

1^a Somma parziale \longrightarrow $1010001 +$

$$P = P_0 + P_1 + P_2 = (P_0 + P_1) + P_2 = S_0 + P_2$$



Moltiplicazione binaria - IV



$$\begin{array}{l} \text{Moltiplicando} \longrightarrow 11011 \text{ x} \\ \text{Moltiplicatore} \longrightarrow \quad \textcircled{1}11 = \end{array}$$

$$\begin{array}{r} 11011 \text{ x } 27_{10} \\ 111 = 7_{10} \\ \hline 111111 \\ 11011+ \\ 11011- \\ 11011- - \\ \hline 10111101 \quad 189_{10} \end{array}$$

$$\begin{array}{r} \hline 11111 \\ 11011+ \\ 11011- \\ \hline 1010001+ \\ \textcircled{11011} - \end{array}$$

prodotti parziali

1^a Somma parziale

Il terzo prodotto parziale è incolonnato alle potenze di 2^2 (la cifra del moltiplicatore ha peso 2^2)



Moltiplicazione binaria - V



$$\begin{array}{l} \text{Moltiplicando} \longrightarrow 11011 \text{ x} \\ \text{Moltiplicatore} \longrightarrow \quad 111 = \end{array}$$

$$\begin{array}{r} 11011 \text{ x } 27_{10} \\ 111 = 7_{10} \\ \hline 111111 \\ 11011+ \\ 11011- \\ 11011- - \\ \hline 10111101 \quad 189_{10} \end{array}$$

$$\begin{array}{r} \hline 11111 \\ 11011+ \\ 11011- \\ \hline 10000 \\ 1010001+ \\ 11011- - \end{array}$$

prodotti parziali

1^a Somma parziale

Somma finale = prodotto $\longrightarrow 10111101$



Moltiplicazione binaria - I



	Moltiplicando →	1 1 0 1 1 x	27 x
	Moltiplicatore →	1 0 1 1 =	11 =

		1 1 1 1 1	
		1 1 0 1 1 +	27 +
		1 1 0 1 1 -	27 - =

		1 0 1 0 0 0 1	297

Prodotti parziali → (black arrows pointing to the three partial products: 11111, 11011+, 11011-)
Riporto → (blue arrow pointing to the carry 1 in the second row of the partial products)
Somma parziale → (yellow arrow pointing to the final sum 1010001)



Moltiplicazione binaria - II



	Moltiplicando →	1 1 0 1 1 x	27 x
	Moltiplicatore →	1 0 1 1 =	11 =

		1 1 1 1 1	
		1 1 0 1 1 +	27 +
		1 1 0 1 1 -	27 - =

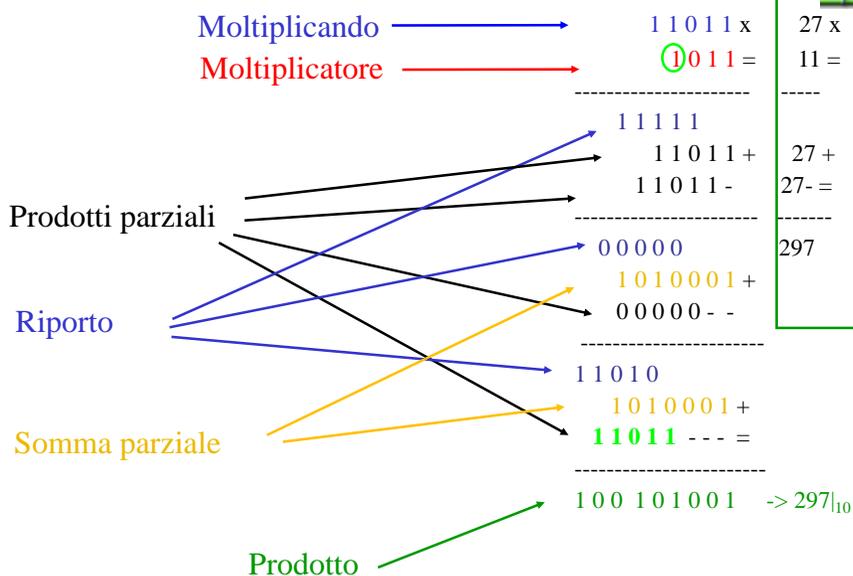
		0 0 0 0 0	297
		1 0 1 0 0 0 1 +	
		0 0 0 0 0 - -	

		1 0 1 0 0 0 1	

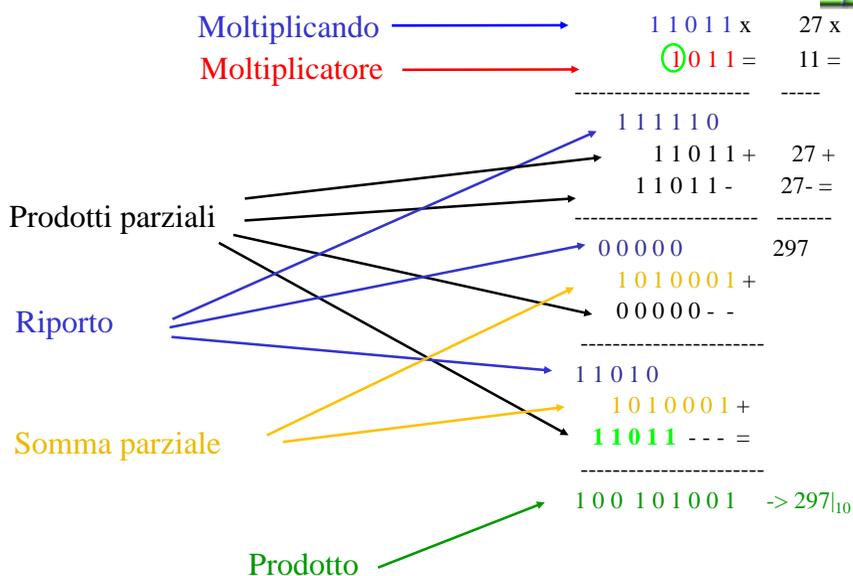
Prodotti parziali → (black arrows pointing to the three partial products: 11111, 11011+, 11011-)
Riporto → (blue arrows pointing to the carry 1 in the second row and the carry 0 in the third row of the partial products)
Somma parziale → (yellow arrow pointing to the final sum 1010001)



Moltiplicazione binaria - III

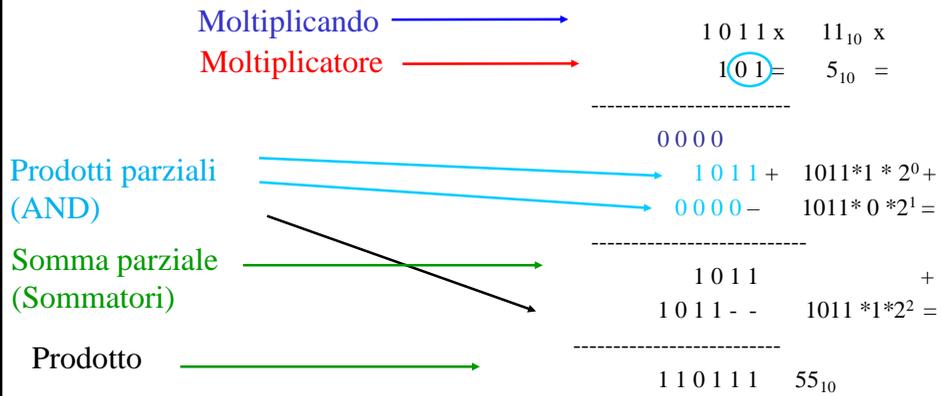


Moltiplicazione: prodotti e somme





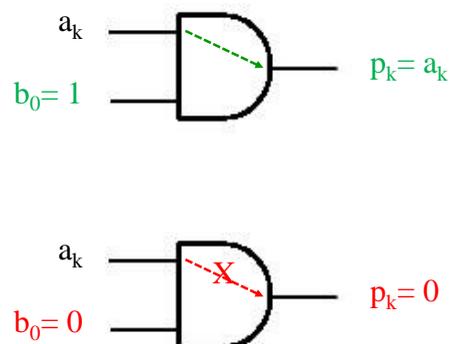
Moltiplicazione binaria (su 4 bit)



Il prodotto parziale è = $\begin{cases} \text{Moltiplicando incolonnato opportunamente} \\ 0 \end{cases}$



Moltiplicazione su 1 bit





La moltiplicazione binaria



Possiamo vederla come:

Un primo stadio in cui si mette in AND ciascun bit del moltiplicatore con il moltiplicando (**prodotto parziale**).

Un secondo stadio in cui si effettuano le **somme dei prodotti parziali** (full adder): somma dei bit su due righe diverse.



La matrice dei prodotti parziali



A e B su 4 bit

Prodotti parziali {					a_3	a_2	a_1	a_0		
					$a_3 b_0$	$a_2 b_0$	$a_1 b_0$	$a_0 b_0$		b_0
				$a_3 b_1$	$a_2 b_1$	$a_1 b_1$	$a_0 b_1$	b_1		
	$a_3 b_2$	$a_2 b_2$	$a_1 b_2$	$a_0 b_2$	b_2					
$a_3 b_3$	$a_2 b_3$	$a_1 b_3$	$a_0 b_3$	b_3						
	2^6	2^5	2^4	2^3	2^2	2^1	2^0			

In binario i prodotti parziali sono degli AND.

Sulla linea tanti AND quanto è la lunghezza di A
Tanti prodotti parziali quanto è la lunghezza di B



La matrice dei prodotti parziali



Prodotti parziali

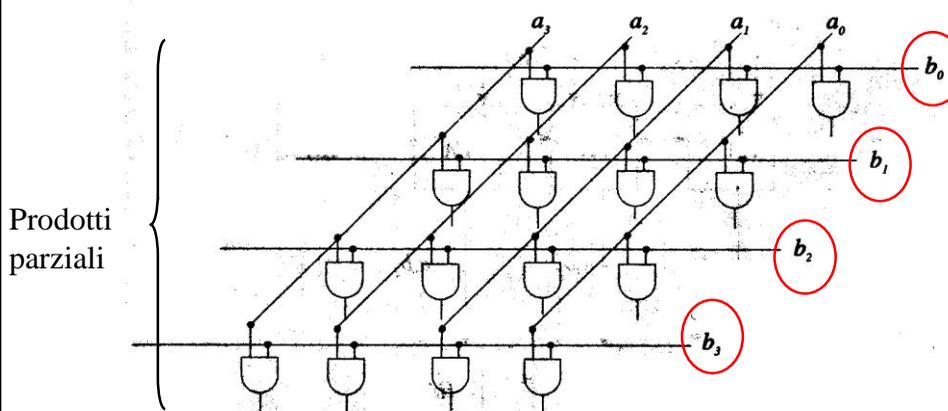
	a_3	a_2	a_1	a_0	
	$a_3 b_0$	$a_2 b_0$	$a_1 b_0$	$a_0 b_0$	b_0
	$a_3 b_1$	$a_2 b_1$	$a_1 b_1$	$a_0 b_1$	b_1
	$a_3 b_2$	$a_2 b_2$	$a_1 b_2$	$a_0 b_2$	b_2
	$a_3 b_3$	$a_2 b_3$	$a_1 b_3$	$a_0 b_3$	b_3

Il bit i -esimo del moltiplicatore, b_i , fa passare 0 o il moltiplicando (prodotto parziale)
 Il prodotto parziale viene incolonnato opportunamente.

$b_0 (a_3 a_2 a_1 a_0)$ genera P_0
 $b_1 (a_3 a_2 a_1 a_0)$ genera P_1
 $b_2 (a_3 a_2 a_1 a_0)$ genera P_2
 $b_3 (a_3 a_2 a_1 a_0)$ genera P_3



Il circuito che effettua i prodotti



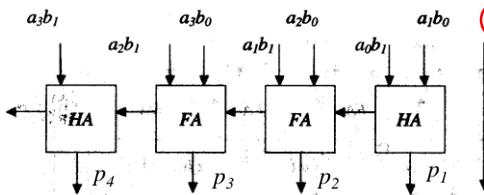
b_k agisce come interruttore, facendo passare 0 o A



Somma delle prime 2 righe dei prodotti parziali - I



$$\begin{array}{r}
 \begin{array}{cccc}
 a_3 & a_2 & a_1 & a_0 \\
 a_3 b_0 & a_2 b_0 & a_1 b_0 & a_0 b_0 \\
 \hline
 a_3 b_1 & a_2 b_1 & a_1 b_1 & a_0 b_1 \\
 \hline
 a_3 b_2 & a_2 b_2 & a_1 b_2 & a_0 b_2 \\
 \hline
 a_3 b_3 & a_2 b_3 & a_1 b_3 & a_0 b_3
 \end{array}
 \end{array}$$



Somma dei primi 2 prodotti parziali

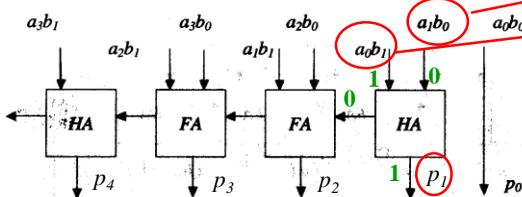
$$\begin{array}{r}
 1101 \times 13 \\
 1011 = 11 \\
 \hline
 1100 \\
 1101 + \\
 1101 - \\
 \hline
 0000 \\
 100111 + \\
 0000 - - \\
 \hline
 1100 \\
 100111 + \\
 1101 - - - = \\
 \hline
 10001111 \rightarrow 143_{10}
 \end{array}$$



Somma delle prime 2 righe dei prodotti parziali - II



$$\begin{array}{r}
 \begin{array}{cccc}
 a_3 & a_2 & a_1 & a_0 \\
 a_3 b_0 & a_2 b_0 & a_1 b_0 & a_0 b_0 \\
 \hline
 a_3 b_1 & a_2 b_1 & a_1 b_1 & a_0 b_1 \\
 \hline
 a_3 b_2 & a_2 b_2 & a_1 b_2 & a_0 b_2 \\
 \hline
 a_3 b_3 & a_2 b_3 & a_1 b_3 & a_0 b_3
 \end{array}
 \end{array}$$

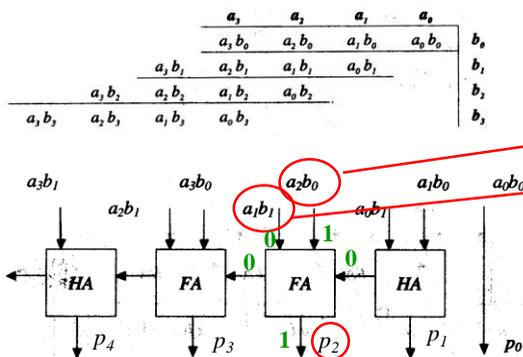


Somma dei primi 2 prodotti parziali →
la somma parziale

$$\begin{array}{r}
 1101 \times 13 \\
 1011 = 11 \\
 \hline
 11000 \\
 1101 + \\
 1101 - \\
 \hline
 0000 \\
 100111 + \\
 0000 - - \\
 \hline
 1100 \\
 100111 + \\
 1101 - - - = \\
 \hline
 10001111 \rightarrow 143_{10}
 \end{array}$$



Somma delle prime 2 righe dei prodotti parziali - III

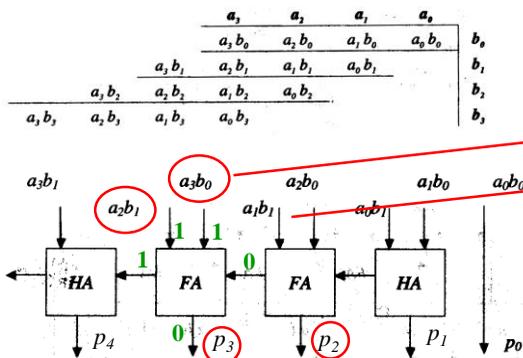


Somma dei primi 2 prodotti parziali →
la somma parziale

$$\begin{array}{r}
 1101 \times \quad 13 \times \\
 1011 = \quad 11 = \\
 \hline
 11000 \\
 1101+ \\
 1101- \\
 \hline
 0000 \\
 100111+ \\
 0000- \\
 \hline
 1100 \\
 100111+ \\
 1101- \\
 \hline
 10001111 \rightarrow 143_{10}
 \end{array}$$



Somma delle prime 2 righe dei prodotti parziali - IV



Somma dei primi 2 prodotti parziali →
la somma parziale

$$\begin{array}{r}
 1101 \times \quad 13 \times \\
 1011 = \quad 11 = \\
 \hline
 11000 \\
 1101+ \\
 1101- \\
 \hline
 0000 \\
 100111+ \\
 0000- \\
 \hline
 11000 \\
 100111+ \\
 1101- \\
 \hline
 10001111 \rightarrow 143_{10}
 \end{array}$$



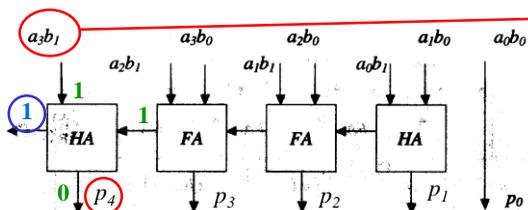
Somma delle prime 2 righe dei prodotti parziali - V



$$\begin{array}{cccc|c}
 & a_3 & a_2 & a_1 & a_0 & b_0 \\
 & a_3 b_0 & a_2 b_0 & a_1 b_0 & a_0 b_0 & b_0 \\
 a_3 b_1 & a_2 b_1 & a_1 b_1 & a_0 b_1 & & b_1 \\
 \hline
 a_3 b_2 & a_2 b_2 & a_1 b_2 & a_0 b_2 & & b_2 \\
 a_3 b_3 & a_2 b_3 & a_1 b_3 & a_0 b_3 & & b_3
 \end{array}$$

$$\begin{array}{r}
 1101 \times 13 \times \\
 1011 = 11 = \\
 \hline
 \end{array}$$

$$\begin{array}{r}
 11000 \\
 1101+ \\
 \hline
 1101- \\
 \hline
 0000 \\
 100111+ \\
 0000- \\
 \hline
 \end{array}$$



Somma dei primi 2 prodotti parziali →
la somma parziale

Dove va il riporto in uscita all'ultimo FA?

$$\begin{array}{r}
 1100 \\
 100111+ \\
 1101- \\
 \hline
 10001111 \rightarrow 143_{10}
 \end{array}$$



Somma della terza riga

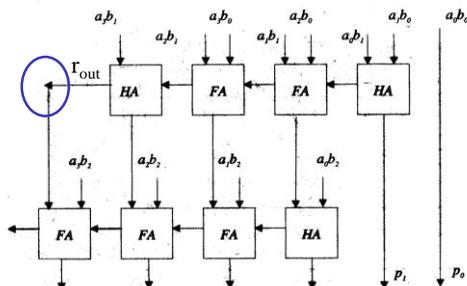


I primi due prodotti parziali sono ottenuti dalla prima batteria di sommatore.

Ogni altro prodotto parziale è sommato da un'ulteriore batteria di sommatore.

$$\begin{array}{r}
 1101 \times 13 \times \\
 1011 = 11 = \\
 \hline
 \end{array}$$

$$\begin{array}{r}
 11000 \\
 1101+ \\
 \hline
 1101- \\
 \hline
 \end{array}$$



$$\begin{array}{r}
 00000 \\
 100111+ \\
 0000- \\
 \hline
 \end{array}$$

$$\begin{array}{r}
 1100 \\
 100111+ \\
 1101- \\
 \hline
 \end{array}$$

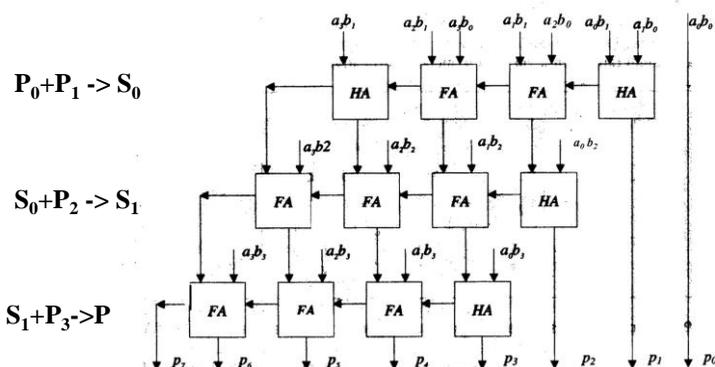
$$\begin{array}{r}
 10001111 \rightarrow 143_{10}
 \end{array}$$



Circuito completo della somma dei prodotti parziali



N-1 batterie di sommatori



Problema: A e B su 4 bit => P su 8 bit (prodotto su 2N bit)



Valutazione della complessità



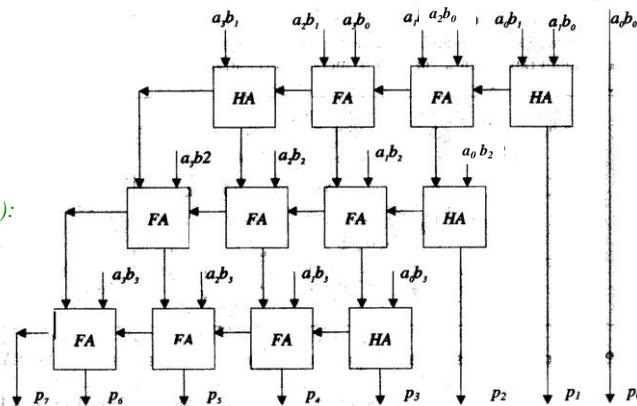
Complessità:

Half Adder: 2 porte
Full Adder: 5 porte

Stadio prodotti (AND):

A su N bit
B su M bit

$N * M$ porte AND



Stadio Somme:

Se $N = M = 4$ numero totale di porte a 2 ingressi = 5

N sommatori per linea
M-1 linee

$CO_{Tot} =$

Numero linee
(M-1)

Numero FA
per linea

Numero HA
per linea

Primo HA
1a linea

Prodotti
Parziali

$$CO_{Tot} = (M-1) * [(N-1) * 5 + 1 * 2] - 2 + M * N$$



Valutazione del cammino critico



Cammini critici:

Half Adder:

Somma - 1 porta

Riporto - 1 porta

Full Adder:

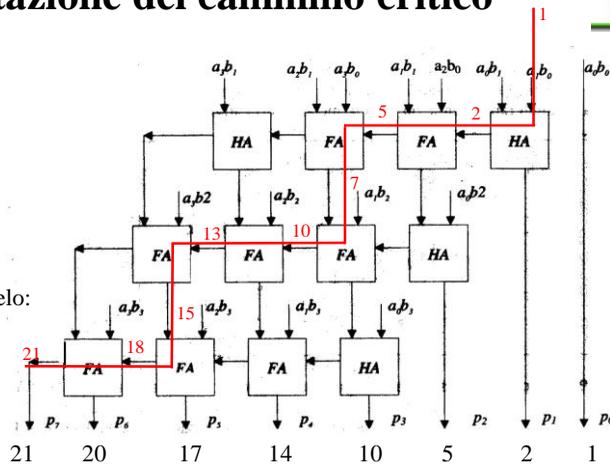
Somma - 2 porte

Riporto - 3 porte

Gli AND operano in parallelo:

ritardo 1.

HA e FA non sono equivalenti per il cammino critico



Se $N = M = 4$ cammino critico totale = 21

$$CC_{Tot} = 8 + (M-4)*(2+3) + 12 + 1$$



Osservazioni



Cammini critici:

Half Adder:

Somma - 1 porta

Riporto - 1 porta

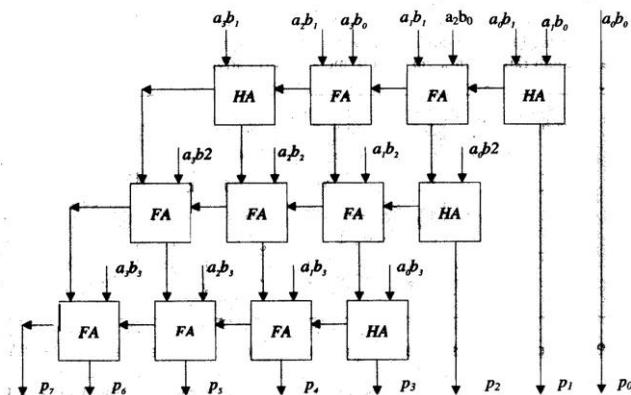
Full Adder:

Somma - 2 porte

Riporto - 3 porte

Gli AND operano in parallelo:

ritardo 1.



Architettura **modulare**, ogni schiera di sommatore lavora sul risultato della schiera superiore e fornisce l'input alla schiera inferiore

Quanto si guadagna sostituendo ai sommatore a propagazione di riporto sommatore ad anticipazione di riporto?



Sommario



Moltiplicatori

ALU



Funzione della ALU



E' integrata nel processore, all'inizio degli anni 90 è stata rivoluzionaria la sua introduzione con il nome di co-processore matematico.

Esegue le operazioni aritmetico-logiche.

E' costituita da circuiti combinatori. Utilizza i blocchi di base già visti.

Opera su parole (MIPS 32 bit).

Le ALU non compaiono solamente nei micro-processori.



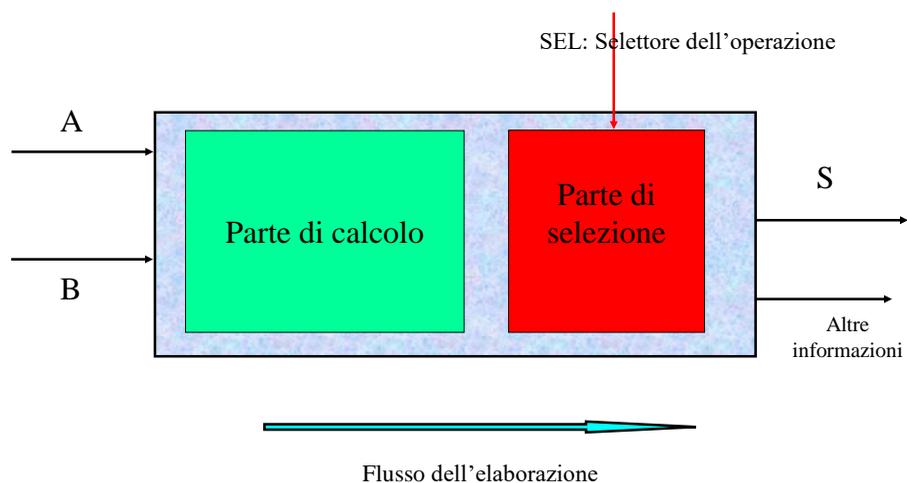
Problematiche di progetto



- Velocità (Riporto).
- Costo.
- Precisione.
- Affidabilità
- Consumo energetico.



Struttura a 2 livelli di una ALU



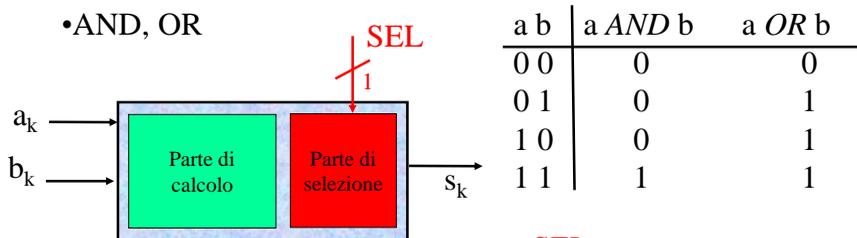
Esempio di esecuzione condizionata.



Progettazione della ALU – 1 bit

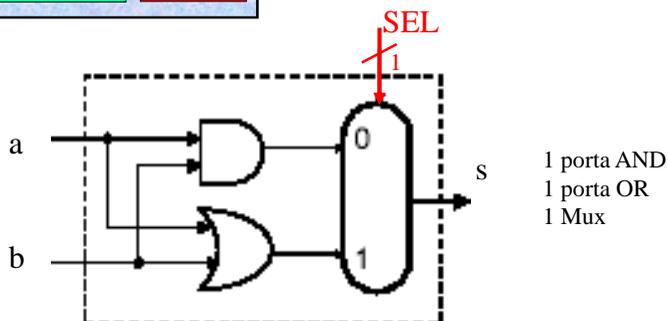


•AND, OR



SEL = 0
s = AND(a,b)

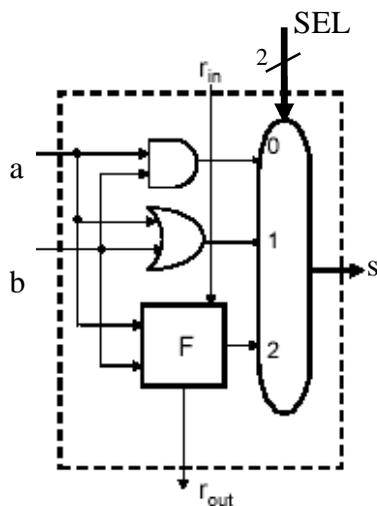
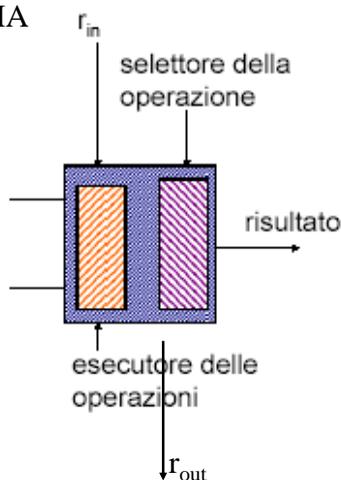
SEL = 1
s = OR(a,b)



La nuova struttura della ALU – 1 bit



•AND
•OR
•SOMMA



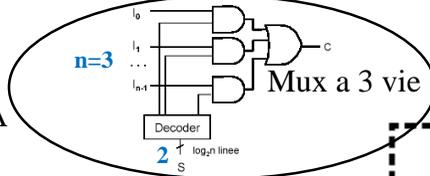
Perchè SEL non viene messo in ingresso?



Valutazione ALU a 1 bit

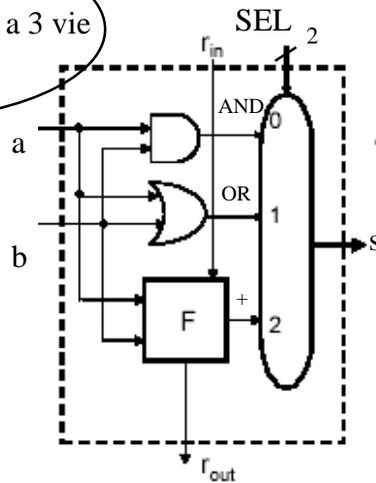


- AND
- OR
- SOMMA



Complessità 1° livello (calcolo): $5+2 = 7$
 Complessità 2° livello (mux): $3*1+(3+2) = 8$
 (Decoder + AND (semaforo) + OR
 (congiunzione))

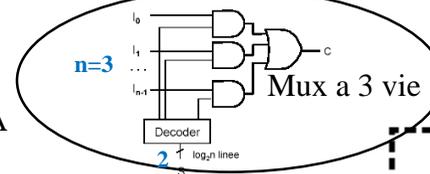
Complessità totale: $7+8 = 15$



Valutazione ALU a 1 bit



- AND
- OR
- SOMMA



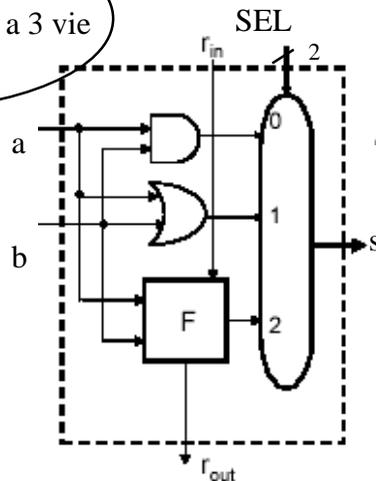
CC 1° livello: 2 per s_{out} , 3 per r_{out}

CC 2° livello: 4 (1 Decoder + (1 AND -
 semaforo + 2 OR (congiunzione))

CC complessivo: 2 (calcolo) + [1 AND
 (semaforo)+ 2 (OR - selezione)] = 5!!

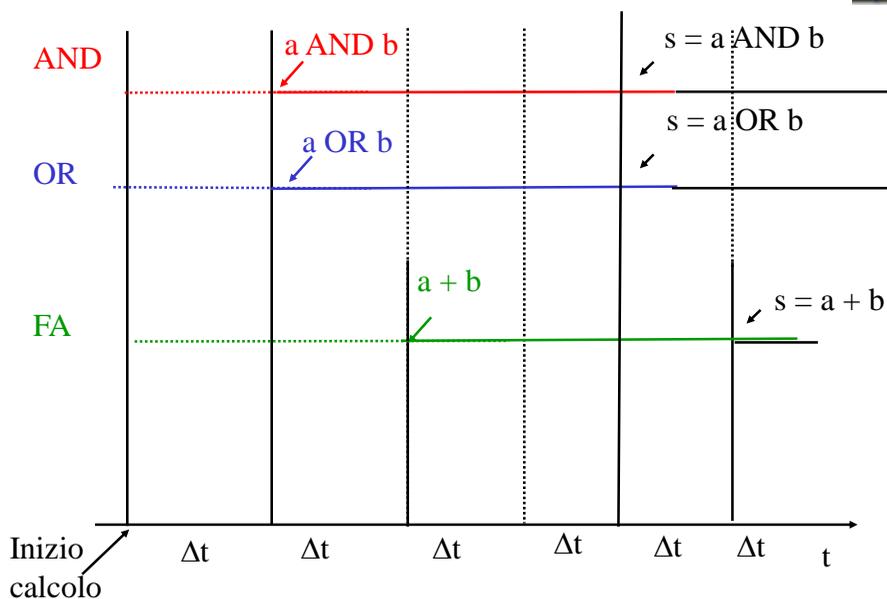
*Il CC del decoder non viene contato: gli
 AND del decoder interni al mux sono attivati
 in parallelo ai circuiti di calcolo.*

Il CC considerato è quello della somma per valutare il tempo necessario perchè commuti s.





I cammini critici all'interno della ALU



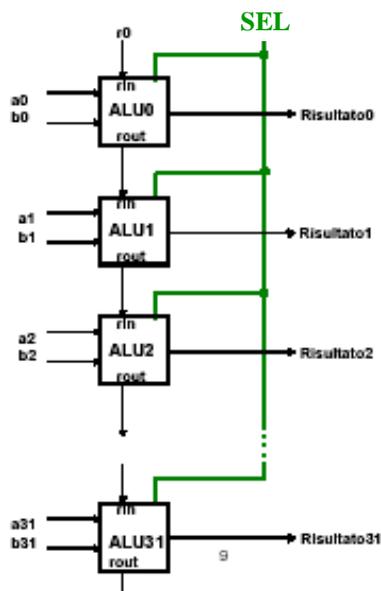
ALU a 32 bit



Come collegare le ALU ad 1 bit?

Flusso di calcolo

Perchè non si può parallelizzare?





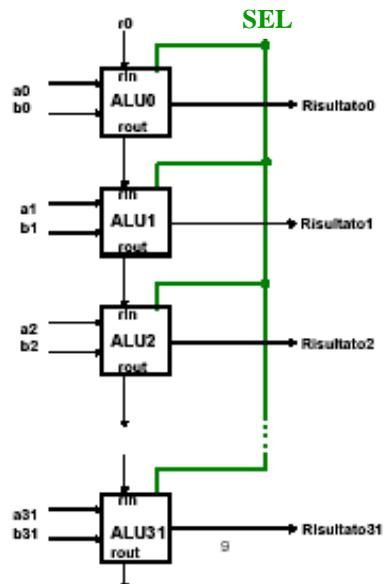
Valutazione ALU a 32 bit



Complessità: $15 \times 32 = 480$ porte logiche

Cammino critico: 3×31 (propagazione riporti) + 3 (semaforo + OR uscita mux di s_{31}) = 96 porte logiche

per 4 operazioni su 32 bit



Sottrazione



In complemento a 2 diventa un'addizione: $a - b = a + \bar{b} + 1 = 1 + a + \bar{b}$

Esempio: $s = 3 - 4$; su 3 bit

3 -> 011	011 +
-4 -> 100 in complemento a 2	100 =
-1 -> 111 in complemento a 2	111

Posso scrivere il numero negativo in complemento a 2 come somma:

	4 -> 100	numero positivo: \bar{b}
Passo I - Complemento a 1	011+	complemento a 1: $\bar{\bar{b}}$
Passo II - Sommo + 1	1=	sommo 1: 1=
Risultato - Complemento a 2	100	risultato -b

Posso quindi scrivere: $-b = \bar{b} + 1$



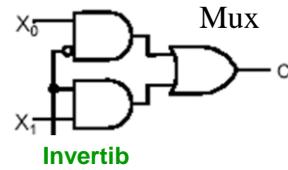
Sottrazione



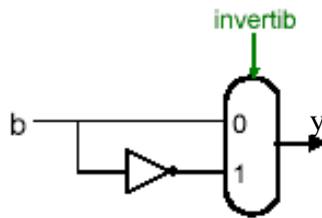
In complemento a 2 diventa un'addizione: $a - b = a + \bar{b} + 1$

Serve:

- a) un inverter (NOT).
- b) la costante 1



a)



Iff invertib
 $y = !b$

Aggiunge 2 porte logiche al cammino critico di ogni stadio.
Aggiungere 2 porte per ogni stadio.

b) Da dove prendo la costante 1?

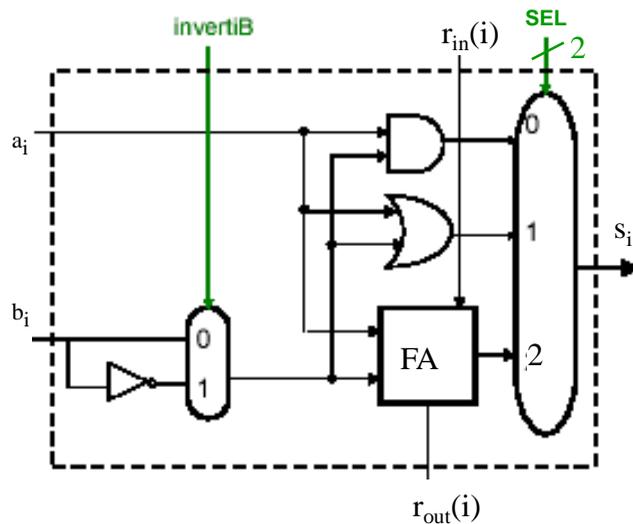


Sottrazione - ALU_i



Operazioni:

- AND
- OR
- SOMMA
- SOTTRAZIONE

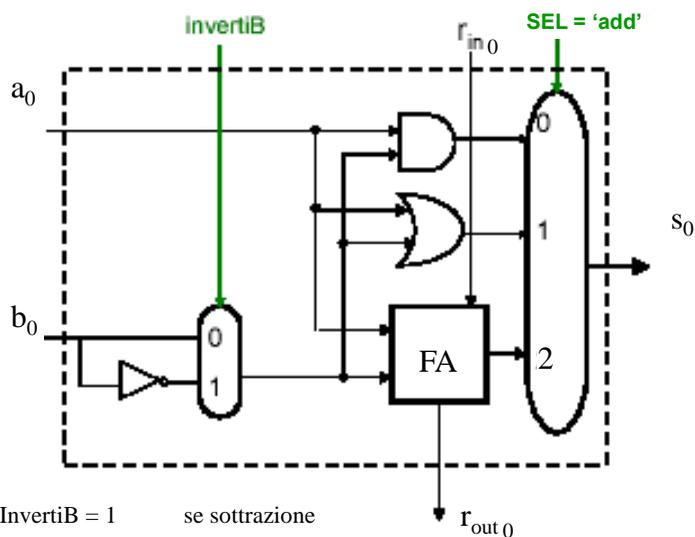


$r_{in}(i) = r_{out}(i-1) \quad i = 1, 2, 3, \dots, 31$
InvertiB = 1

$i \neq 0$ (tranne che nel primo stadio)
se sottrazione



Sottrazione – primo stadio: ALU₀



$r_{in}(0) = \text{InvertiB} = 1$ se sottrazione

(occorre utilizzare un full adder anche per il bit meno significativo con $r_{in0} = 1$).
Effettuo quindi la somma di 1 con la somma della prima coppia di bit.

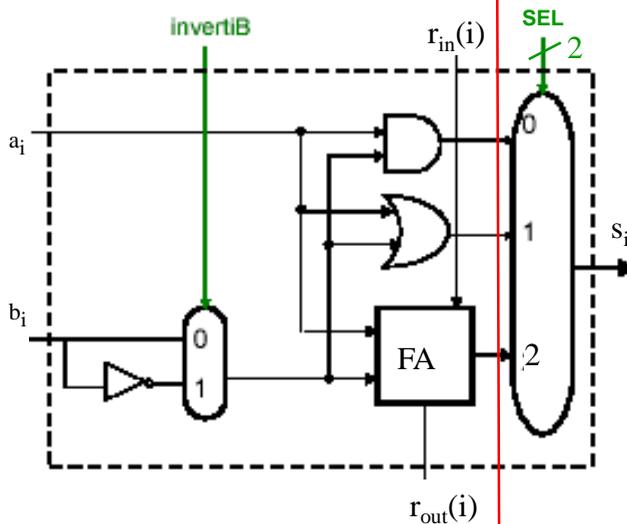


Operazioni - ALU_i



E' possibile programmare questa ALU per eseguire:

- 1) $a \text{ AND } !b$
- 2) $a \text{ OR } !b$
- 3) $a + b$
- 4) $a - b$



La parte di calcolo è comunque separata dalla parte di selezione



Sottrazione: ALU a 32 bit



$r_{in}(0) = \text{InvertiB} = 1$
se sottrazione

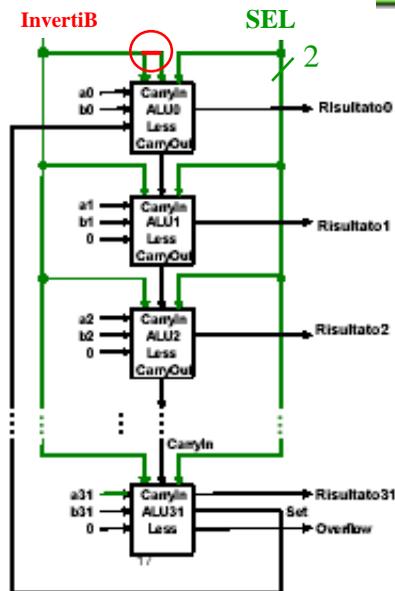
- AND
- OR
- SOMMA
- SOTTRAZIONE

From_UC	SEL	r_0	InvertiB
And	And	0	0
Or	Or	0	0
Somma	Add	0	0
Sottr.	Add	1	1

InvertiB e r_0 sono lo stesso segnale, si può ancora ottimizzare.

$r_{in}(0)$ entra solo in ALU₀

InvertiB entra in tutte le ALU_i



A.A.

49/51

<http://borghese.di.unimi.it/>



ALU a 32 bit con CLA



- Come realizzare una ALU a 32 bit con:
 - Porte OR
 - Porte AND
 - CLA a 4 bit?

Definire complessità e cammino critico

Notate che l'inverter su b aggiunge complessità e cammino critico.

A.A. 2022-2023

50/51

<http://borghese.di.unimi.it/>



Sommario



Moltiplicatori

ALU