



Moltiplicatori e ALU

Prof. Alberto Borghese
Dipartimento di Scienze dell'Informazione

borgnese@di.unimi.it

Università degli Studi di Milano

Riferimenti: Appendice B5 prima parte. Per approfondimenti e HW della moltiplicazione consultare il Fummi.



Sommario

Moltiplicatori

ALU



Moltiplicazione decimale



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

$$\begin{array}{r} \text{Prodotti parziali} \longrightarrow 834 + \\ 556 - \\ 1112 - - \\ \hline \text{prodotto} \longrightarrow 117594 \end{array}$$

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

Somma dei prodotti parziali



Moltiplicazione binaria



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

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

$$\begin{array}{r} 11111 \\ 11011 + \\ 11011 - \\ \hline 1010001 + \\ 11011 - - \\ \hline \text{prodotto} \longrightarrow 10111101 \end{array}$$

Somma parziale $\xrightarrow{1}$ *prodotti parziali*



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:

$$213_{10} / 10 = 21.3_{10}$$

$$213_{10} = (2 \times 10^2 + 1 \times 10^1 + 3 \times 10^0) / 10^1 =$$

$$(2 \times 10^2 + 1 \times 10^1 + 3 \times 10^0) \times 10^{-1} =$$

$$(2 \times 10^2 \times 10^{-1} + 1 \times 10^1 \times 10^{-1} + 3 \times 10^0 \times 10^{-1}) =$$

$$(2 \times 10^1 + 1 \times 10^0 + 3 \times 10^{-1}) = 21.3 \text{ cvd.}$$

Esempio:

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

$$(1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0) \times 2^{-2} =$$

$$(1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}) = 5,75 \text{ cvd.}$$



Moltiplicazione binaria



Moltiplicando \longrightarrow 1 1 0 1 1 x

Moltiplicatore \longrightarrow 1 1 1 =

1 1 0 1 1 x	27 ₁₀	
1 1 1 =	7 ₁₀	1 1 1 1 1
		1 1 0 1 1 +
1 1 1 1 1 1		1 1 0 1 1 -
1 1 0 1 1 +		
1 1 0 1 1 -		1
1 1 0 1 1 - -		1 0 1 0 0 0 1 +
		1 1 0 1 1 - -
1 0 1 1 1 1 0 1	189 ₁₀	
Prodotto \longrightarrow 1 0 1 1 1 1 0 1		



Moltiplicazione binaria



Moltiplicando \longrightarrow 1 1 0 1 1 x 27 x
 Moltiplicatore \longrightarrow 1 0 1 1 = 11 =

Prodotti parziali \longrightarrow 1 1 1 1 1
 \longrightarrow 1 1 0 1 1 + 27 +
 \longrightarrow 1 1 0 1 1 - 27 - =

Riporto \longrightarrow 0 0 0 0 0
 \longrightarrow 1 0 1 0 0 0 1 + 297
 \longrightarrow 0 0 0 0 0 - -

Somma parziale \longrightarrow 1 1 0 1 0
 \longrightarrow 1 0 1 0 0 0 1 +
 \longrightarrow 1 1 0 1 1 - - - =

Prodotto \longrightarrow 1 0 0 1 0 1 0 0 1 $\rightarrow 297_{10}$



Moltiplicazione binaria (su 4 bit)



Moltiplicando \longrightarrow 1 0 1 1 x 11₁₀ x
 Moltiplicatore \longrightarrow 1 0 1 = 5₁₀ =

Prodotti parziali (AND) \longrightarrow 0 0 0 0
 \longrightarrow 1 0 1 1 + 1011*1*2⁰ +
 \longrightarrow 0 0 0 0 - 1011*0*2¹ =

Somma parziale (Sommatori) \longrightarrow 1 0 1 1 +
 \longrightarrow 1 0 1 1 - - 1011*1*2² =

Prodotto \longrightarrow 1 1 0 1 1 1 55₁₀

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



La moltiplicazione binaria



Possiamo vederla come:

Un primo stadio in cui si mette in AND ciascun bit del moltiplicatore con il moltiplicando.

Un secondo stadio in cui si effettuano le somme (full adder) dei bit sulle righe contenenti i prodotti parziali.



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	

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

$b_0 (a_3 a_2 a_1 a_0)$ genera P_0

$b_1 (a_3 a_2 a_1 a_0)$ genera P_1

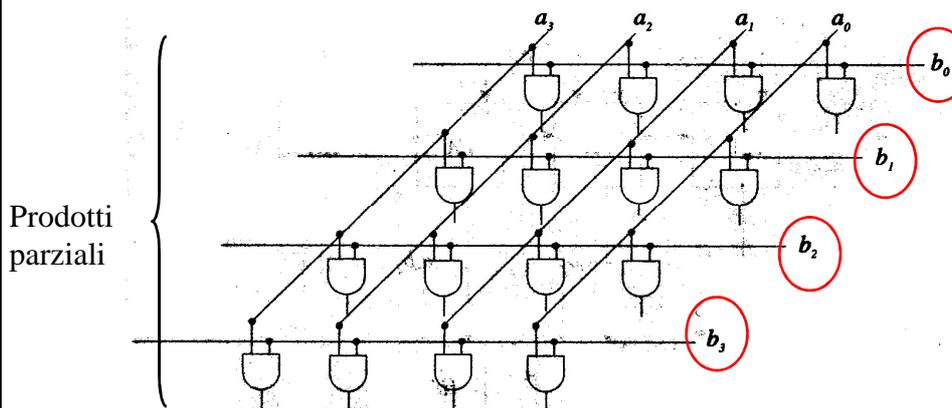
$b_2 (a_3 a_2 a_1 a_0)$ genera P_2

$b_3 (a_3 a_2 a_1 a_0)$ genera P_3

.....

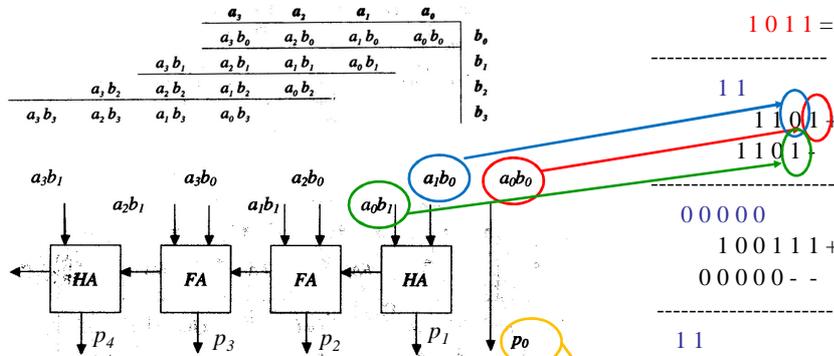


Il circuito che effettua i prodotti





Somma delle prime 2 righe dei prodotti parziali



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

$$\begin{array}{r} 11 \\ 1101 \\ 1101 \\ \hline \end{array}$$

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

$$\begin{array}{r} 11 \\ 100111+ \\ 1101- - - = \\ \hline \end{array}$$

$$10001111 \rightarrow 143_{10}$$

Somma dei primi 2 prodotti parziali:
 Aggiunge il terzo prodotto parziale:

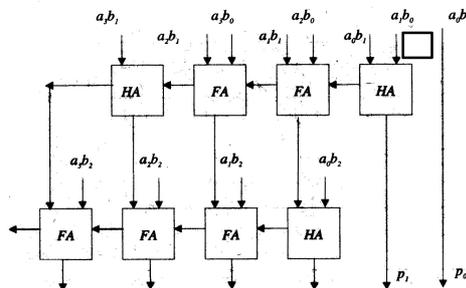
HA e FA non sono equivalenti
 per i diversi cammini critici.



Somma della terza riga



I primi due prodotti parziali sono sommati dalla prima batteria di sommatore.
 Ogni altro prodotto parziale è sommato da un'ulteriore batteria di sommatore.



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

$$\begin{array}{r} 11 \\ 1101+ \\ 1101- \\ \hline \end{array}$$

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

$$\begin{array}{r} 11 \\ 100111+ \\ 1101- - - = \\ \hline \end{array}$$

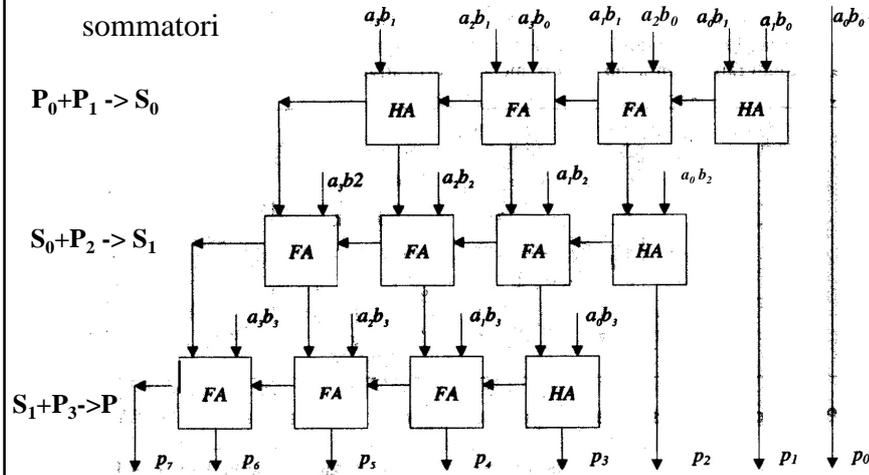
$$10001111 \rightarrow 143_{10}$$



Circuito completo della somma dei prodotti parziali



N-1 batterie di sommatori



Problema: overflow: A e B su 32 bit => P su 64 bit.



Valutazione della complessità



Complessità:

Half Adder: 2 porte
Full Adder: 5 porte

Stadio AND:

A su N bit
B su M bit

N * M porte AND

Stadio OR:

N sommatori per riga

M-1 righe

1° riga

Altre righe (M-2)

$$\text{Numero porte} = (N-2) * 5 + 2 * 2 + (M-2) * [(N-1) * 5 + 1]$$



Numero porte se N = M = 4 -> 48



Valutazione del cammino critico



Cammini critici:

Half Adder:

Somma - 1 porta

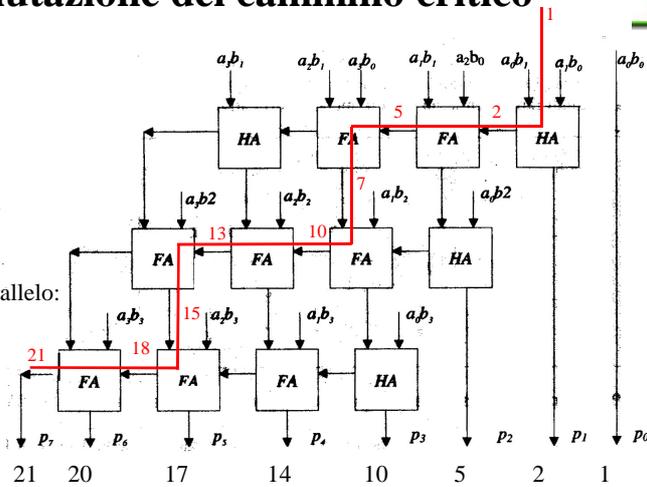
Riporto - 1 porta

Full Adder:

Somma - 2 porte

Riporto - 3 porte

Gli AND operano in parallelo:
ritardo 1.



Cammino critico: 21



Somme parziali e prodotto



Moltiplicando \longrightarrow 1 1 0 1 1 x
 Moltiplicatore \longrightarrow 1 1 1 =

$$\begin{aligned}
 P &= P_0 + P_1 + P_2 = \\
 &= (P_0 + P_1) + P_2 = \\
 &= S_0 + P_2
 \end{aligned}$$

1 1 1 1 1

1 1 0 1 1 +

1 1 0 1 1 -

Somma parziale $\xrightarrow{1}$

1 0 1 0 0 0 1 +

1 1 0 1 1 - -

prodotti parziali

prodotto \longrightarrow 1 0 1 1 1 1 0 1



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.



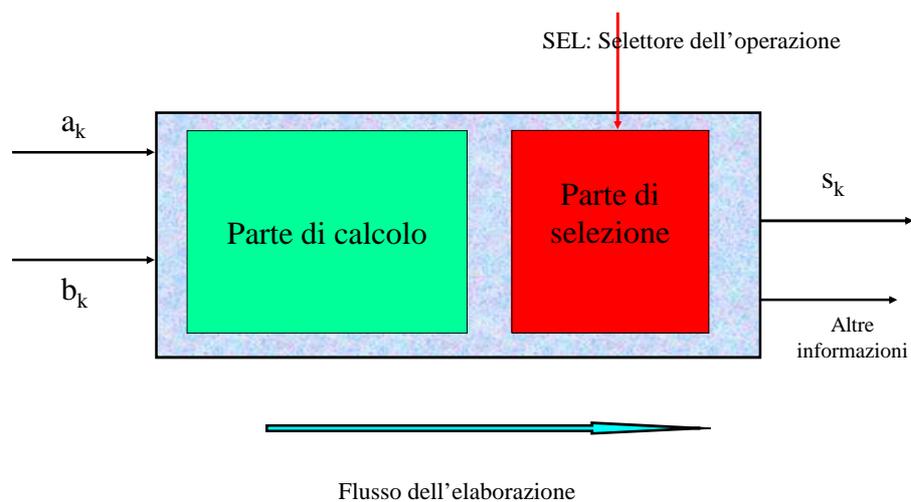
Problematiche di progetto



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



Struttura a 2 livelli di una ALU

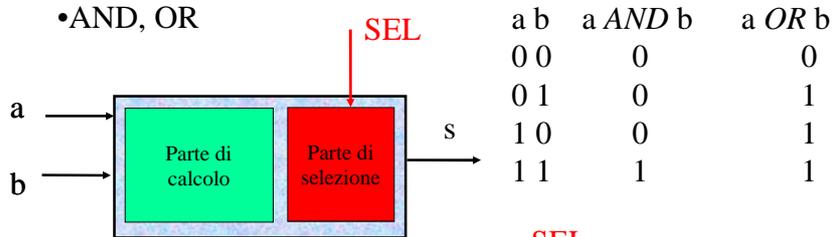




Progettazione della ALU – 1 bit

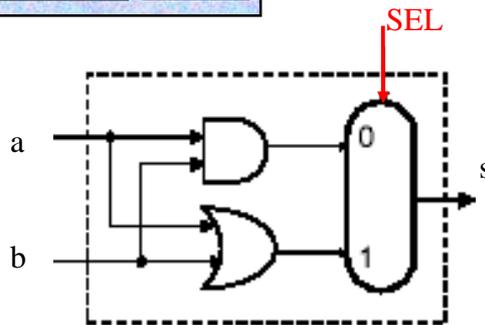


•AND, OR



SEL = 0
s = AND(a,b)

SEL = 1
s = OR(a,b)



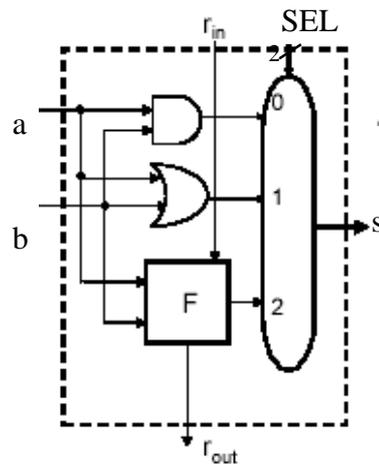
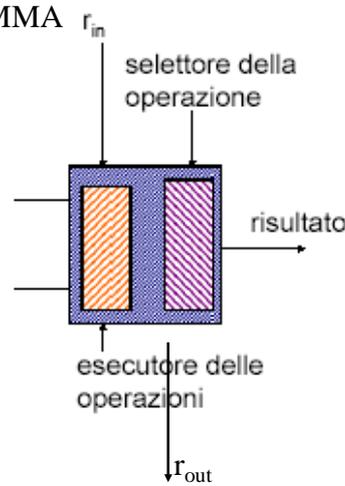
1 porta AND
1 porta OR
1 Mux



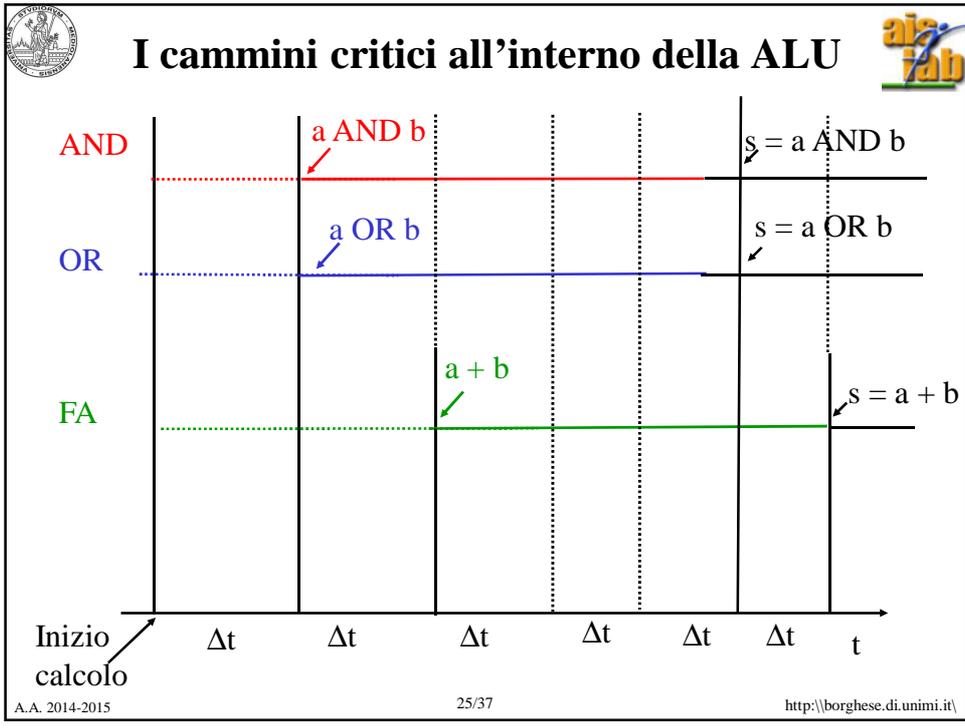
La nuova struttura della ALU – 1 bit



- AND
- OR
- SOMMA



Perchè SEL non viene messo in ingresso?



Valutazione ALU a 1 bit

- AND
- OR
- SOMMA

Complessità 1° livello: $5 + 2 = 7$

Complessità 2° livello: 8

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

CC 2° livello: 4 (2 AND + 2 OR del mux)

CC complessivo: 2 (calcolo) + 2 (OR – selezione)

Gli AND del selettore sono attivati in parallelo ai circuiti di calcolo

A.A. 2014-2015 26/37 <http://borghese.di.unimi.it/>



Sommario



ALU ad 1 bit

ALU a 32 bit

Comparazione, Overflow, Test di uguaglianza

Tecnologie di costruzione di una ALU



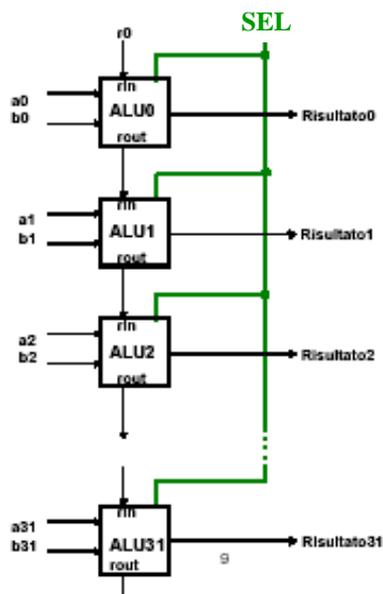
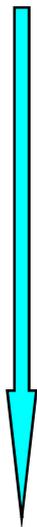
ALU a 32 bit



Come collegare le
ALU ad 1 bit?

Flusso di calcolo

Perchè non si può
parallelizzare?





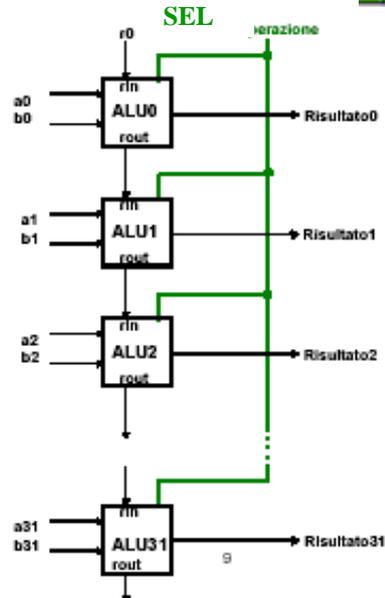
Valutazione ALU a 32 bit



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

Cammino critico: $3 \times 31 +$
 $2 (s_{out}) +$
 $[1 + 2]$ (selezione) =
98 porte logiche

per 4 operazioni



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: $\bar{b} +$
Passo II - Sommo + 1	1 =	sommo 1: 1 =
Risultato - Complemento a 2	100	risultato -b

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



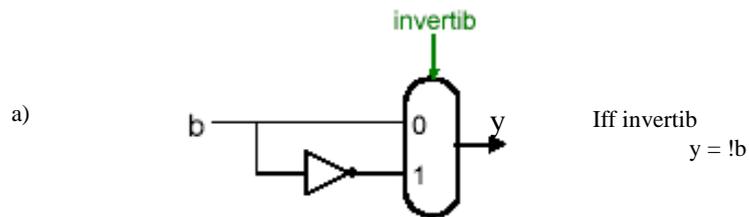
Sottrazione



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

Serve:

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

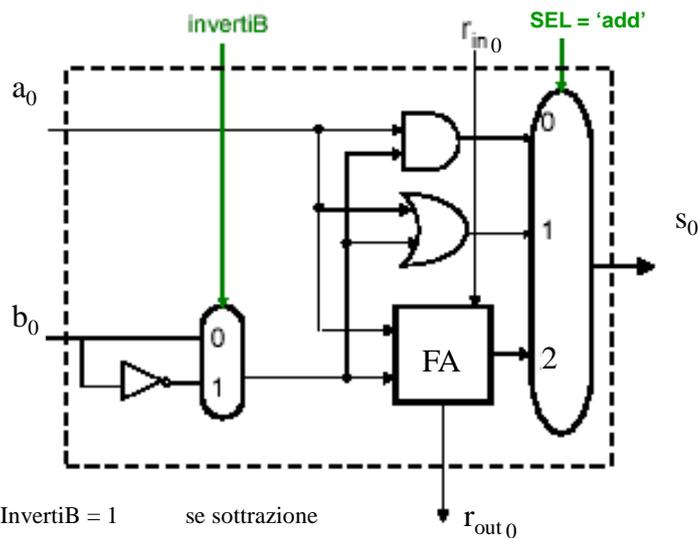


Aggiunge 2 porte logiche al cammino critico.

- b) Da dove prendo la costante 1?



Sottrazione - ALU₀



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

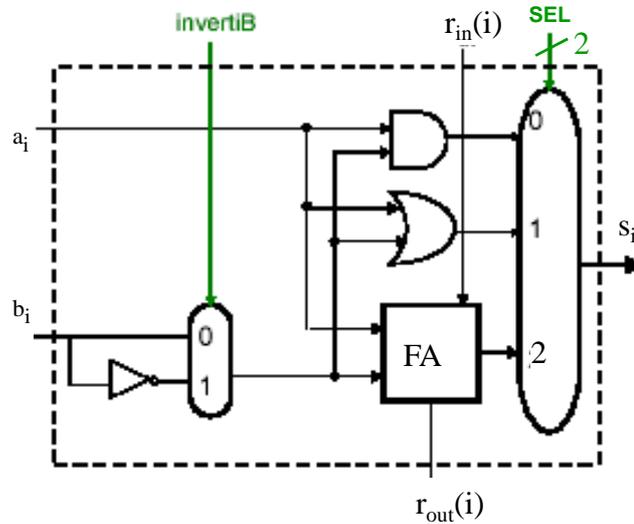
(occorre utilizzare un full adder anche per il bit meno significativo con $r_{in0} = 1$).
Effettuo quindi la somma di 1 con la somma della prima coppia di bit.



Sottrazione - ALU_i



- AND
- OR
- SOMMA
- SOTTRAZIONE



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

$i \neq 0$
 se sottrazione



Operazioni particolari - ALU_i

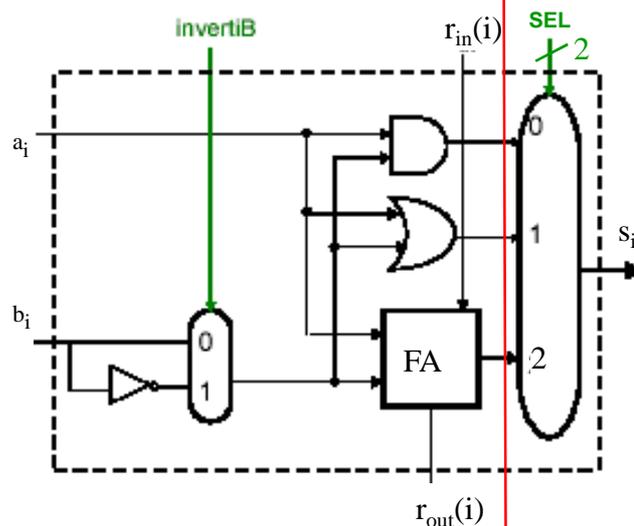


E' possibile programmare questa ALU per eseguire

a AND !b

oppure:

a OR !b



InvertiB = 1
 SEL = AND, OR

La parte di calcolo è comunque separata dalla parte di selezione



Sottrazione: ALU a 32 bit



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

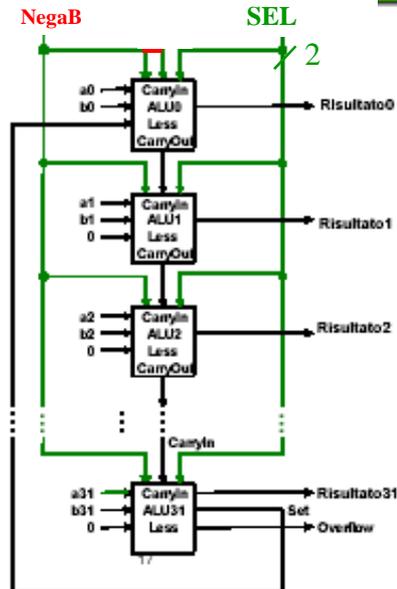
- AND
- OR
- SOMMA
- SOTTRAZIONE

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 ALU_0

InvertiB entra in tutte le ALU_i



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



Sommario



Moltiplicatori

ALU