Arithmetic and Logic Unit e moltiplicatore

M. Favalli

Engineering Department in Ferrara

Sommario

Sommario

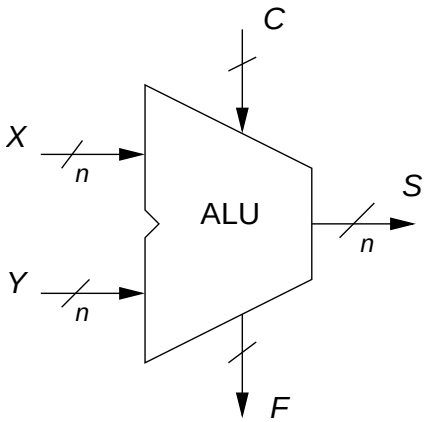
1 Arithmetic and Logic Unit - ALU

2 Moltiplicatore

Introduzione

- La ALU é un componente in grado di eseguire diversi tipi di operazione di tipo aritmetico e logico su due operandi di n bit
- Il tipo di operazioni viene determinato dalla configurazione dei bit di controllo
- Le prime CPU avevano la parte di elaborazione dati su una ALU (attualmente ne sono presenti piú di una)
- La ALU oltre a produrre un risultato di n bit produce anche diversi segnali di flag che rappresentano eccezioni o condizioni particolari sul risultato
- La sua struttura riflette un compromesso fra costo e prestazioni

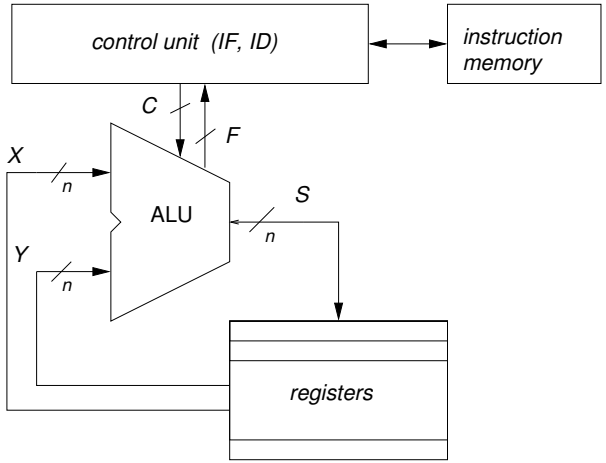
Schema



$S = S(X, Y, C)$ e $F = F(X, Y, S)$

Si noti che le operazioni aritmetiche sono in modulo 2^n

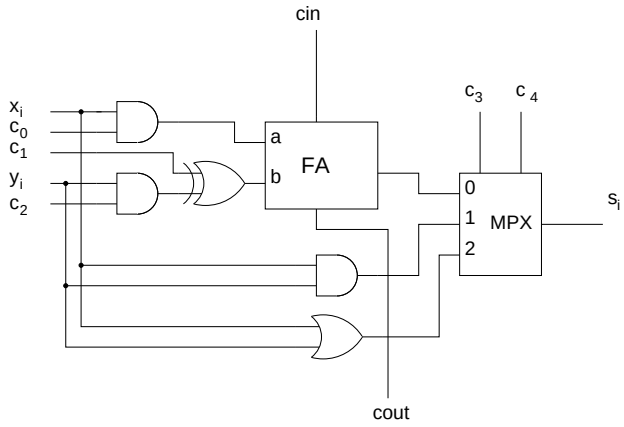
Utilizzo della ALU in una semplice CPU



Struttura

- Sono stati proposti e realizzati diversi tipi di ALU
- Forse il tipo piú diffuso é quello di tipo bit - sliced in cui una ALU a *n* bit viene costruita a partire da *n* slice, ovvero *n* ALU a 1 bit ciascuna
- La ALU é essenzialmente costruita intorno a un *n* bit adder di tipo ripple-carry
- Vantaggi di modularitá e costo e svantaggi di prestazioni

1-bit ALU



1-bit ALU - funzioni aritmetiche

In questo caso il carry-in é significativo

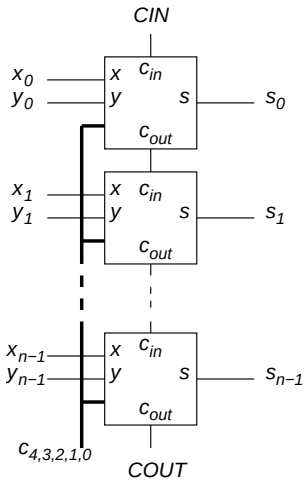
c_4c_3	$c_2c_1c_0$	a	b	s	c_{out}
00	000	0	0	c_{in}	0
00	001	x_i	0	$x_i \oplus c_{in}$	$x_i c_{in}$
00	010	0	1	$1 \oplus c_{in}$	c_{in}
00	011	x_i	1	$x_i' \oplus 1 \oplus c_{in}$	$x_i' + c_{in}$
00	100	0	y_i	$y_i \oplus c_{in}$	$y_i c_{in}$
00	101	x_i	y_i	$x_i \oplus y_i \oplus c_{in}$	$x_i y_i + x_i c_{in} + y_i c_{in}$
00	110	0	y_i'	$y_i' \oplus c_{in}$	$y_i' c_{in}$
00	111	x_i	y_i'	$x_i \oplus y_i' \oplus c_{in}$	$x_i y_i' + x_i c_{in} + y_i' c_{in}$

1-bit ALU - funzioni logiche

Si tratta di operazioni logiche bit a bit il cui risultato non dipende dal carry-in

c_4c_3	$c_2c_1c_0$	s
01	- - -	$x_i y_i$
10	- - -	$x_i + y_i$

ALU a *n* bit



Aritmetica in complemento a 2

ALU a n bit: funzioni aritmetiche

- Fino a questo momento i simboli $\{X, Y, S\}$ sono stati utilizzati per denotare parole di n bit
- Nel caso delle funzioni aritmetiche della ALU, le configurazioni binarie di tali parole codificano numeri interi rappresentati in complemento a 2 che verranno denotati come $(X)_2$, $(Y)_2$, $(S)_2$, lo stesso vale per costanti rappresentate su n bit
- Nel caso di espressioni il pedice viene riportato solo all'esterno delle parentesi: $(X + Y + 9)_2 = (X)_2 + (Y)_2 + (9)_2$
- Le notazioni $(W)_2^{c1}$ e $(W)_2^{c2}$ sono utilizzate per denotare il complemento a 1 e il complemento a 2 di $(W)_2$: $(W)_2^{c1} = (2^n - W - 1)_2$ e $(W)_2^{c2} = (2^n - W)_2$

ALU a n bit: funzioni aritmetiche

$c_4 c_3\, c_2 c_1 c_0$	$S\ (CIN = 0)$	$S\ (CIN = 1)$
00 000	$(0)_2$	$(1)_2$
00 001	$(X)_2$	$(X + 1)_2$
00 010	$(2^n - 1)_2$	$(0)_2$
00 011	$(X)_2^{c1}$	$(X^{c1} + 1)_2 = (X)_2^{c2} = = (-X)_2$
00 100	$(Y)_2$	$(Y + 1)_2$
00 101	$(X + Y)_2$	$(X + Y + 1)_2$
00 110	$(Y)_2^{c1}$	$(Y^{c1} + 1)_2 = (Y)_2^{c2} = = (-Y)_2$
00 111	$(X + Y^{c1})_2$	$(X + Y^{c1} + 1)_2 = = (X + Y^{c2})_2 = (X - Y)_2$

ALU a n bit

Funzioni logiche

vskip 2mm

$c_4 c_3\, c_2 c_1 c_0$	S
01 - - -	XY
10 - - -	$X + Y$

Per la complementazione si puó utilizzare il complemento a 1

Bit di flag

I bit di flag forniscono indicazioni sul risultato

- bit di zero: vale 1 se il risultato $S = 0_2$
- bit di paritá: fornisce la paritá sul risultato
- bit di carry: CARRY OUT
- bit di overflow: vale 1 se il risultato non é contenuto in n bit
- bit di segno: vale 1 se il risultato non é contenuto in n bit

Overflow

Si ha overflow se il segno del risultato é diverso da quello atteso sulla base del tipo di operazione e dei segni degli operandi:

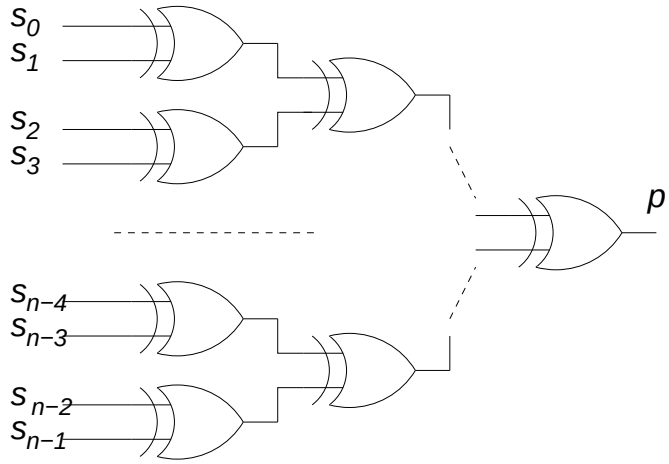
- $X > 0$, $Y > 0$ e $S < 0$ con una somma aritmetica
- $X < 0$, $Y < 0$ e $S > 0$ con una somma aritmetica
- $X > 0$, $Y < 0$ e $S < 0$ con una sottrazione aritmetica
- $X < 0$, $Y > 0$ e $S > 0$ con una sottrazione aritmetica

Sia OV il bit di overflow:

$$OV = c_4' c_3' (c_2 c_1' c_0 (X_{n-1}' Y_{n-1}' S_{n-1} + X_{n-1} Y_{n-1} S_{n-1}') + c_2 c_1 c_0 (X_{n-1}' Y_{n-1} S_{n-1} + X_{n-1} Y_{n-1}' S_{n-1}'))$$

Bit di paritá

Serve per proteggere i dati che vengono scritti nei registri o in memoria



Bit di segno e bit di zero

- Il bit di segno é banalmente il bit di maggior peso del risultato (che é significativo solo nel caso di operazioni aritmetiche)
- Il bit dizero assume il valore 1 solo quando tutti i bit del risultato hanno il valore 0. Puó essere ottenuto tramite un NOR a n ingressi

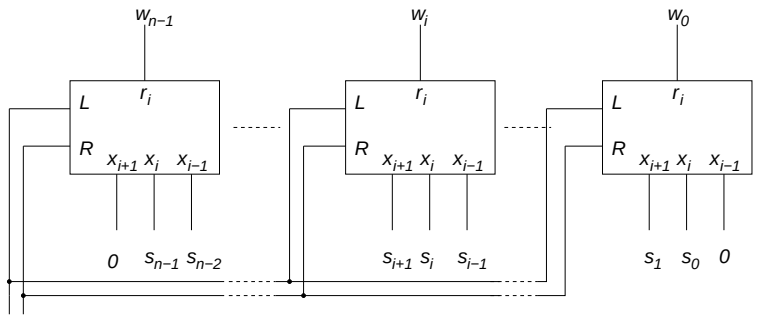
Sommario

Moltiplicazione

- La ALU illustrata non conteneva le operazioni di moltiplicazione e divisione
- Vedremo come caso particolare la moltiplicazione e divisione per 2 (applicata ai numeri naturali)
- Nel caso generale vedremo il prodotto di numeri naturali

Moltiplicazione e divisione per 2

Come noto moltiplicazione e divisione per 2 possono essere rispettivamente eseguite mediante uno shift a sinistra e uno a destra di una posizione
Circuito (da mettere in uscita alla ALU) in grado di eseguire tali operazioni



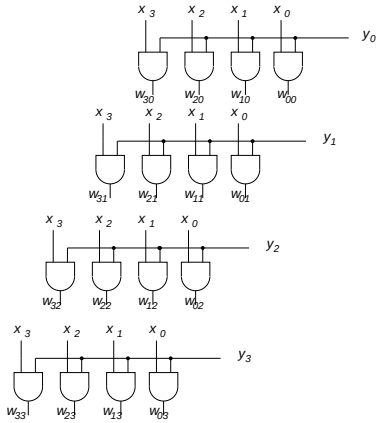
Moltiplicazione fra numeri naturali

L'algoritmo é quello utilizzato in base 10 (somma di prodotti parziali):

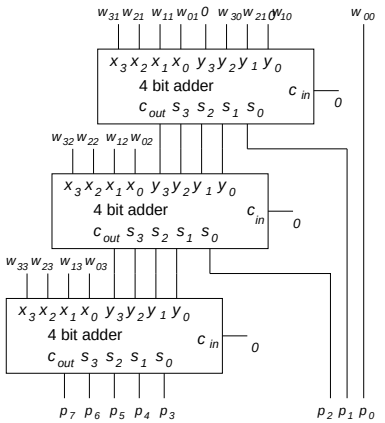
			x_3	x_2	x_1	x_0	$X *$
			y_3	y_2	y_1	y_0	$Y =$
			x_3y_0	x_2y_0	x_1y_0	x_0y_0	$W_0 +$
		x_3y_1	x_2y_1	x_1y_1	x_0y_1	-	$W_1 +$
	x_3y_2	x_2y_2	x_1y_2	x_0y_2	-	-	$W_2 +$
x_3y_3	x_2y_3	x_1y_3	x_0y_3	-	-	$W_3 =$	
p_7	p_6	p_5	p_4	p_3	p_2	p_1	p_0

Moltiplicazione fra numeri naturali

Logica AND (prodotti parziali)



Sommatori



Conclusioni

- Le unitá aritmetiche sono un componente critico del data-path delle ALU
- Gli esempi riportati corrispondono alle soluzioni piú semplici
- Nelle CPU reali si utilizzano diverse tecniche per migliorare le prestazioni di ALU e moltiplicatori che non si riescono a trattare in questo ambito