NUMERI NEGATIVI

sabato 1 ottobre 2022 19:46

È NECESSARIO PREVEDERE UNA codifica anche PER numeri negativi!

MODULO E SEGNO

AL NUMERO BINARIO VERO E PROPRIO, VIENE ANTEPOSTO UN BIT CHE, ASSUME IL VALORE 0 SE IL NUMERO È POSITIVO OPPURE IL VALORE 1 SE IL NUMERO È NEGATIVO.

$$0000 = 0_{(10)} \qquad 1000 = 0_{(10)}$$

Ho due modi per scrivere lo zero. ED È FASTIDIOSO PERCHÉ STIAMO SPRECANDO SPAZIO, OSSIA STIAMO RAPPRESENTANDO DUE VOLTE LO STESSO NUMERO.

QUINDI È INEFFICIENTE E POSSIAMO FARE ALTRO.

COMPLEMENTO A UNO

UN NUMERO VIENE NEGATO INVERTENDO TUTTI I BIT.

0	0	0	0	1	1	0	0	= +12
1	1	1	1	0	0	1	1	= -12

SI CHIAMA "COMPLEMENTO A 1" PERCHÉ SE SOMMO UN NUMERO n CON IL SUO COMPLEMENTATO, HO UNA MASCHERA DI TUTTI UNO.
MA ESISTONO ANCORA DUE ZERI.

$12 - 12 = 0$, E 0 È ANCHE UGUALE A 11111111 (-0).

ORA EFFETTUIAMO LA SOMMA.

$-1 + 2 = 1$

CON 8 BIT: 1 = 00000001 PERCIÒ -1 = 11111110

$$\begin{array}{r} 11111110\ + \\ 00000010\ = \\ \hline 1\underbrace{00000000}\ \end{array}$$

IO HO FISSATO IL NUMERO DEI MIEI BIT A 8, E SE USO 8 BIT, IL MIO RISULTATO È ZERO!

SIGNIFICA CHE SE USO QUESTA RAPPRESENTAZIONE E RIMUOVO IL NUMERO FISSO DI CIFRE, IL RISULTATO È SBAGLIATO.

QUINDI, SE SI VERIFICA UN RIPORTO CHE ARRIVA FINO ALLA FINE, È NECESSARIO SOMMARE AL RISULTATO OTTENUTO ANCHE IL BIT DI RIPORTO!

QUINDI, SE SI VERIFICA UN RIPORTO CHE ARRIVA FINO ALLA FINE, È NECESSARIO SOMMARE AL RISULTATO OTTENUTO ANCHE IL BIT DI RIPORTO!

END-AROUND CARRY.

00000000 + 1 = 00000001

MA PROGETTARE CIRCUITI COSÌ DIVENTA PIÙ COMPLICATO, PERCHÉ OGNI VOLTA MI DEVO CHIEDERE SE È USCITO UN BIT CHE NON RIESCO A RAPPRESENTARE.

LA STESSA COSA VALE PER LE SOTTRAZIONI.

PERÒ IL COMPLEMENTO A UNO, OGGI NON VIENE PIÙ USATO NELLE RAPPRESENTAZIONI, MA SI USA:

COMPLEMENTO A DUE!

- La rappresentazione che consente l'implementazione hardware più semplice ed efficiente
- Largamente utilizzata dalle architetture convenzionali

SE IO HO UN VALORE X POSITIVO, IL VALORE CORRISPONDENTE DI X NEGATIVO, SU m BIT FISSATI, È DATO DA: $2^m - x$

PER RAPPRESENTARE IL NUMERO NEGATIVO, NON CI METTO L'UNO DAVANTI, NON INVERTO I BIT, MA DEVO CALCOLARE QUESTO $[2^m - x]$!

MA PER UN ELABORATORE È VELOCE, MA PER NOI È SCOMODO!

ABBIAMO DETTO NEL COMPLEMENTO A 1 DI UN NUMERO X, SE LO SOMMO COL SUO CORRISPONDENTE POSITIVO, IL RISULTATO È UNA MASCHERA DI TUTTI UNO!

SE NELLA MIA RAPPRESENTAZIONE A m CIFRE, LA MIA PAROLA È COMPOSTA DA TUTTI UNO, QUEL NUMERO È PROPRIO $2^m - 1$:

$2^3 - 1 = 7 \qquad 7_{(10)} \longrightarrow \underbrace{111}_{m=3}{}_{(2)}$

SE IO PRENDO UNA RAPPRESENTAZIONE IN COMPLEMENTO A 1 E SOMMO UN NUMERO X NEGATIVO, IL RISULTATO SARÀ SEMPRE UNA MASCHERA DI TUTTI UNO $(2^m - 1)$!

$$X + \overline{X} = 2^m - 1$$

Valore negativo di x!

PASSAGGI ALGEBRICI: **PORTO 1 A SINISTRA E X A DESTRA!**

$$x + \bar{x} + 1 = 2^m$$

$$\boxed{\bar{x} + 1 = 2^m - x}$$ **DEF. DI COMPLEMENTO A 2!**

SIGNIFICA CHE: SE VOGLIO CALCOLARE IL VALORE NEGATIVO DI X IN COMPLEMENTO A 2 NE CALCOLO IL COMPLEMENTO A 1 E CI SOMMO 1.

[C.2 = C.1 + 1]

QUESTO È ESATTAMENTE QUELLO CHE FA LA CPU!

ESEMPIO

$X = (0110)_2 = (6)_{10}$	$\bar{X} = (1001)_2$ = C.1	$\bar{x} + 1 = 1001 + 1 = 1010 = (-6)_{10}$ = C.2 — SE LETTO COME C.2

SE 1010 LO INTERPRETO COME NUMERO POSITIVO, ALLORA VALE $(10)_{10}$!

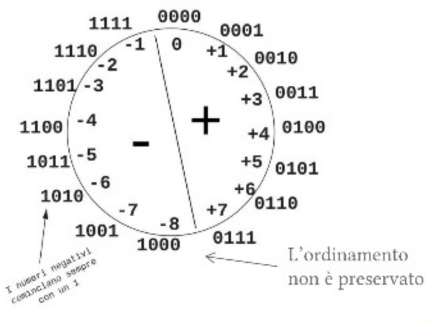
$$\begin{array}{cc|cc|c} 0 & 1 & 1 & 0 & (6) \\ \underline{1} & \underline{0} & 1 & 0 & (-6) \end{array}$$

↳ questi due bit sono uguali!

↳ questi bit sono stati invertiti!

Per calcolare il complemento a due di un numero, si parte dal bit meno significativo. Si lasciano inalterati tutti i bit fino a quando non si trova il primo uno. Quindi, si invertono tutti i bit rimanenti.

4 bit.



Perché non ho due zeri.

- Vi è un solo zero, ma si codifica il -8 e non l'8 (il *numero strano*)
- Con n cifre, si rappresentano i numeri nell'intervallo $[-2^{n-1}, 2^{n-1} - 1]$