Rappresentazione di numeri Relativi ℤ

La scelta più ovvia: Modulo e Segno (MS)

- Si AGGIUNGE l'informazione del segno alla rappresentazione del valore assoluto del numero
- Solitamente si antepone il simbolo + (opzionale) oppure −

  $+143,\ 12,\ -38$
- Questa soluzione è buona per gli uomini, meno per i sistemi elettronici, che usano rappresentazioni comunque binarie

Il caso binario (MS)

- L'uso di un simbolo binario $\{0,1\}$ torna utile per indicare il segno

- cifre binarie del valore assoluto
- cifra binaria che indica il segno
  - 0 : POSITIVO
  - 1 : NEGATIVO

  } convenzione più diffusa

- Vantaggi: intuitiva
  - facili le operazioni di cambio-segno
  - estrazione del valore assoluto
  - gestione della moltiplicazione con segno
- Aspetti problematici:
  - lo 0 ha 2 rappresentazioni, non c'è biunivocità tra rappresentante e rappresentato
  - somme e sottrazioni richiedono l'analisi dei segni
- Range rappresentato con n bit

$$-(2^{n-1}-1) \leq x \leq +(2^{n-1}-1)$$

Simmetrico

N° totale di codici $2^n - 1$

(si è perso un codice nel doppio 0)

Rappresentazioni che mantengono le proprietà della somma

- Cioè in cui nella rappresentazione è contenuto il valore originale, con il suo segno (C1)

- Una strategia con qualche vantaggio è la seguente:

$$\begin{cases} \text{se numero positivo: codifica binaria} \\ \text{se numero negativo: rappresentazione ottenuta } (B^n-1)+x \end{cases} \quad (1)$$

Per evitare che una stessa rappresentazione venga attribuita a due valori diversi (uno positivo e uno negativo) occorre stabilire dei range per i valori rappresentabili.

- Per chiarire questo punto, prendiamo come esempio il caso DECIMALE con $n=2$.

Le rappresentazioni disponibili sono 100.
Applicando la legge di rappresentazione (1) otterremo (considerando che $10^2-1=99$)

x	X
-50	49 *
-49	50
-48	51
⋮	⋮
-2	97
-1	98
0	0; 99
1	1
2	2
3	3
⋮	⋮
48	48
49	49
50	50 *

* NON UNIVOCO

- valori rappresentabili in modo univoco

$$-\frac{B^n}{2}+1 \leq x \leq \frac{B^n}{2}-1$$

- ϕ ha 2 rappresentazioni

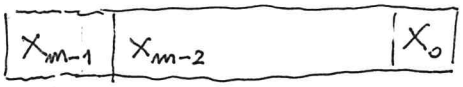
* NON UNIVOCO

il caso binario (C1)

- Con $B=2$ e m bit, la legge di rappresentazione diventa

$$\begin{cases} 0 \le x \le 2^{m-1}-1 : & X = x \quad \text{si osservi che} \quad X_{m-1} = \emptyset \\ -(2^{m-1}-1) \le x \le -1 : & X = (2^m - 1) + x \quad \text{ora} \quad X_{m-1} = 1 \end{cases}$$

Si osservi che $2^m - 1 = \underset{(m-1)}{1\ 1\ 1 \ldots 1}\ \underset{(0)}{}$ numero binario formato da m "1"



↳ non è il "bit del segno" ma se 0 $x \ge 0$
1 $x < \emptyset$

- Vantaggi
  - facile cambio segno (si invertono i bit)
  - facile valore assoluto [se $(X_{m-1}=1)$ si invertono i bit]
  - la somma di due numeri si esegue sempre con un sommatore (e la differenza con un sottrattore) con un aggiustamento se un addendo negativo (o entrambi)

  > somma di positivi: rappresentabile se $S_{m-1} = \emptyset$
  risultato OK

  > positivo con negativo: sempre rappresentabile
  $S = x \overset{\oplus}{+} y \overset{\ominus}{+} 2^m - 1$
  se $x+y < \emptyset$ ok
  se $x+y \ge \emptyset$ si somma 1
  (e non si considera il riporto)

  > negativo con negativo
  $S = x + y + 2^m - 1 + 2^m - 1$
  si somma 1
  se $S_{m-1} = 1$ ok
  (rappresentabile)

- Problematiche
  - Solito problema dello $\emptyset$
  - Algoritmo della somma più semplice di MS ma non diretto

Rappresentazione in TRASLAZIONE (T)

- Considerando un range di B^m valori, da (B pari)

$$-\frac{B^m}{2} \leq x \leq +\frac{B^m}{2} - 1$$

posso ottenere una rappresentazione binaria su m bit sommando $\frac{B^{m-1}}{2}$. Quindi, indipendentemente dal segno di x

$$X = x + \frac{B^m}{2} \qquad 0 \leq X < B^m - 1$$

Il caso binario (T)

- Su m bit $\frac{B^m}{2} = (1000\ldots0)_2$

Ancora, una volta X_{m-1} contiene l'informazione del segno (0 negativo; 1 positivo)

Vantaggi: intuitivo
facile la somma (basta togliere $B^m/2$)

Rappresentazione in complemento alla Base

- Possiamo estendere l'operazione di modulo (e divisione intera) anche in $\mathbb{Z}$.

Siano $a, b \in \mathbb{Z}$ con $b \neq 0$

- allora esisterà sempre e sarà unica la coppia $q \in \mathbb{Z}$ e $0 \leq r < |b|$ tale che

$$q \cdot b + r = a .$$

Per la dimostrazione seguiamo quanto fatto con $a \geq 0$.

- Nel caso a, b concordi e positivi la dimostrazione è la stessa data per $\mathbb{N}$.
- Nel caso a, b concordi negativi costruiamo la successione

$b \quad 2b \quad 3b \ldots \quad qb$ tendente a $-\infty$.

qb è il primo termine tale che $qb \leq a$

Allora q è la divisione intera.

$r = a - qb$ con $r \geq 0$ in quanto $qb \leq a$

$r < b$ in quanto $(q-1)b > a$

$a - qb < -b$ (positivo) $|b|$

- Caso a, b discordi con $b > 0$

costruiamo la successione

$-b \quad -2b \quad -3b \ldots \quad qb$ tende a $-\infty$ $\quad qb \leq a$ (negativo)

$r = a - qb$ con $r \geq 0$

$r < b$ in quanto $(q-1)b > a$

ecc.

- Allora per complemento alla base

$X = (x)_{B^m}$ limitando il range $-\frac{B^m}{2} \leq x \leq \frac{B^m}{2} - 1$

oppure si può ridefinire

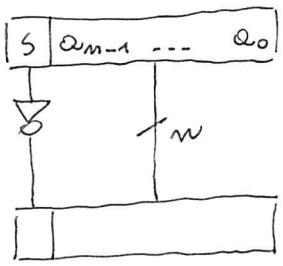
$X = x \quad$ se $x \geq 0$

$X = x + B^m \quad$ se $x \leq 0$

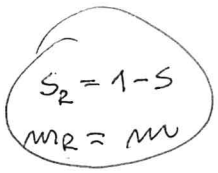
Operazioni su numeri relativi binari

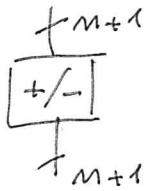
- Cambio segno (prodotto per -1)
- Estrazione del valore assoluto
- Estensione / riduzione della rappresentazione
- Somma e differenza

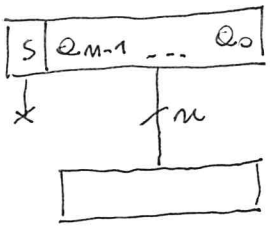
Modulo e segno (n+1 bit)



Cambio segno



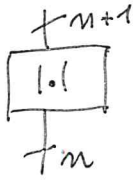


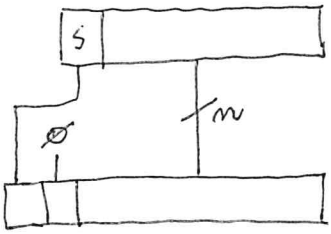


Estrazione del modulo.

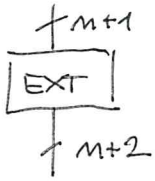
$|x| = m$

Bastano n bit per l'info. senza segno (intero assoluto)



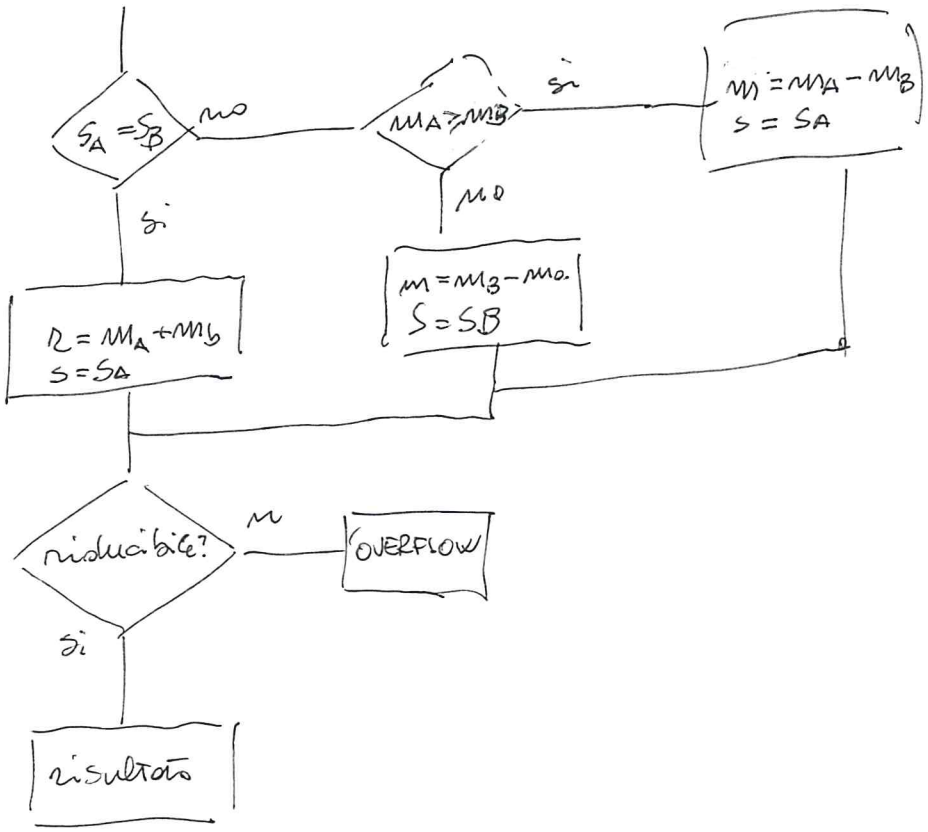


Estensione.
È sufficiente aumentare i bit del modulo, riempiendolo con 0



$S_E = S$

$m_E = m$ con $M_m = 0$


$S_A = S_B$
no
$M_A \geq M_B$
si
$m = M_A - M_B$
$s = S_A$
si
no
$R = M_A + M_B$
$s = S_A$
$m = M_B - M_0$
$S = S_B$
riducibile?
no
OVERFLOW
si
risultato