

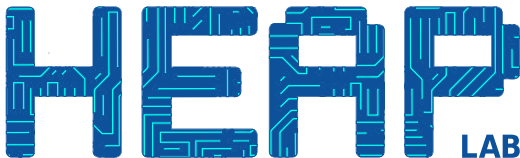
# Esercitazione di Informatica A

Codifiche numeriche:  
Rappresentazioni binarie

Stefano Cherubin

`<nome>.<cognome>@polimi.it`

Esercitazione 2  
28 Settembre 2018



## Algoritmo

Dividere ripetutamente per la base (2). Scrivere i resti della divisione in ordine inverso.

# Conversione da base $n$ a decimale

## Algoritmo

Moltiplicare il valore di ciascuno dei bit di modulo per la base (2) elevata alla rispettiva posizione. Sommare il tutto.

0	0	1	0	1	0	1	0
	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
$\pm$	modulo						

# Numeri binari relativi (1/2)

+42

## Modulo e segno

0	0	1	0	1	0	1	0
---	---	---	---	---	---	---	---

±

modulo

## Complemento a 2

0	0	1	0	1	0	1	0
---	---	---	---	---	---	---	---

# Numeri binari relativi (2/2)

−42

## Modulo e segno

1	0	1	0	1	0	1	0
---	---	---	---	---	---	---	---

±

modulo

Cambiare il bit di segno, mantenendo uguale il modulo.

## Complemento a 2

1	1	0	1	0	1	1	0
---	---	---	---	---	---	---	---

Invertire tutti i bit. Poi sommare 1

# Conversione da binario a decimale (C2)

## Algoritmo per la rappresentazione in complemento a 2

Moltiplicare il valore di ciascuno dei bit per la base (2) elevata alla rispettiva posizione. Solo il bit più significativo (più a sinistra) avrà contributo negativo. Sommare il tutto.

0	0	1	0	1	0	1	0
$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
$\pm$							modulo

# Intervallo di rappresentazione

Su  $N$  bit si possono avere  $2^N$  rappresentazioni di numeri binari. Il dominio di dimensione  $2^N$  può essere usato per rappresentare numeri

$$\left[0; 2^N - 1\right]$$

Se introduciamo i numeri negativi occorre rappresentare in qualche modo l'informazione sul segno del numero. Per fare ciò si utilizza un bit, per convenzione il più significativo. Resta un bit in meno per rappresentare il modulo. Quindi il dominio sarà

$$\left[-\left(2^{N-1} - 1\right); 2^{N-1} - 1\right]$$

Con la rappresentazione in complemento a due, il valore che in modulo e segno sarebbe uno 0 di segno negativo (10000000...) rappresenta invece  $-2^{N-1}$ . Il dominio diventa quindi

$$\left[-2^{N-1}; 2^{N-1} - 1\right]$$

# Carry, overflow, underflow

## Carry bit

Il bit di carry si porta a 1 quando la somma tra due numeri (positivi o negativi) supera il numero di bit del risultato.

## Overflow bit

Il bit di overflow si porta a 1 quando:

- la somma di due numeri negativi fornisce in output un numero positivo (*underflow*)
- la somma di due numeri positivi fornisce in output un numero negativo (*overflow*)

Sono bit indipendenti.

Approfondimento consigliato<sup>1</sup>

---

<sup>1</sup>[http://teaching.idallen.com/dat2343/10f/notes/040\\_overflow.txt](http://teaching.idallen.com/dat2343/10f/notes/040_overflow.txt)



Trova la rappresentazione in complemento a 2 di lunghezza fissa 8 bit dei seguenti numeri espressi in decimale

- +14
- -66
- -31

## Complemento a 2: +14

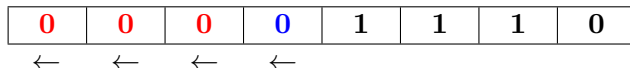
Trova la rappresentazione in complemento a 2 di lunghezza fissa 8 bit di

- +14

$$\begin{array}{r|l} 14 & 2 \\ \hline 7 & \mathbf{0} \uparrow \\ | +14 | = & 3 \quad \mathbf{1} \uparrow = 1110 \\ & 1 \quad \mathbf{1} \uparrow \\ & 0 \quad \mathbf{1} \uparrow \end{array}$$

Applichiamo un bit in più per il segno: 01110

Estendiamo in segno fino a raggiungere 8 bit



## Complemento a 2: -66

$$\begin{array}{r|l} 66 & 2 \\ \hline 33 & \mathbf{0} \uparrow \\ 16 & \mathbf{1} \uparrow \\ 8 & \mathbf{0} \uparrow \\ 4 & \mathbf{0} \uparrow \\ 2 & \mathbf{0} \uparrow \\ 1 & \mathbf{0} \uparrow \\ 0 & \mathbf{1} \uparrow \end{array} \quad = 1000010$$

Applicare un bit in più per il segno:  $+66 = 01000010$

Invertire tutti i bit e poi sommare 1

$$\begin{array}{rcccccccc} 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & + \\ & & & & & & & 1 & = \\ \hline 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \end{array}$$

## Complemento a 2: -31

$$\begin{array}{r|l}
 31 & 2 \\
 \hline
 15 & \mathbf{1} \uparrow \\
 7 & \mathbf{1} \uparrow \\
 3 & \mathbf{1} \uparrow \\
 1 & \mathbf{1} \uparrow \\
 0 & \mathbf{1} \uparrow
 \end{array}
 \quad = 11111$$

Applichiamo un bit in più per il segno:  $+31 = 011111$

Invertire tutti i bit e poi sommare 1

$$\begin{array}{r}
 \begin{array}{cccccc}
 0 & 1 & 1 & 1 & 1 & 1 \\
 \hline
 1 & 0 & 0 & 0 & 0 & 0
 \end{array}
 \quad + \\
 \begin{array}{cccccc}
 & & & & & 1 \\
 \hline
 & 1 & 0 & 0 & 0 & 0 & 1
 \end{array}
 \quad = \\
 \begin{array}{cccccc}
 \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1}
 \end{array}$$

$\leftarrow \quad \leftarrow \quad \leftarrow$

Qual è il numero minimo di bit necessari a rappresentare i seguenti numeri relativi?

- $-70$
- $+25$
- $+256$
- $-256$

## Numero minimo di bit: -70

Qual è il numero minimo di bit necessari a rappresentare

- -70

Calcoliamo  $|-70| = 70$  in base 2

1000110

Servono quindi 7 bit per rappresentare il modulo di -70. Occorre un bit in più per il segno, quindi arriviamo a 8 bit.

Infatti con 8 bit possiamo rappresentare numeri da

$$[-2^{N-1} ; 2^{N-1} - 1]$$

$$[-2^{8-1} ; 2^{8-1} - 1]$$

$$[-128 ; 127]$$

$$-70 \in [-128; 127]$$

## Numero minimo di bit: +25

Qual è il numero minimo di bit necessari a rappresentare

- +25

Calcoliamo  $|+25| = 25$  in base 2

11001

Servono quindi 5 bit per rappresentare il modulo di +25. Occorre un bit in più per il segno perché operiamo nel dominio dei numeri relativi, quindi arriviamo a 6 bit.

Infatti con 6 bit possiamo rappresentare numeri da

$$[-2^{6-1} \ ; \ 2^{6-1} - 1]$$

$$[-32 \ ; \ 31]$$

$$+25 \in [-32; 31]$$

# Numero minimo di bit: +256

Qual è il numero minimo di bit necessari a rappresentare

- +256

Calcoliamo  $|+256| = 256$  in base 2

100000000

Servono quindi 9 bit per rappresentare il modulo di +256. Occorre un bit in più per il segno, quindi arriviamo a 10 bit.

Infatti con 10 bit possiamo rappresentare numeri da

$$[-2^{10-1} \quad ; \quad 2^{10-1} - 1]$$

$$[-512 \quad ; \quad 511]$$

$$+256 \in [-512; 511]$$



# Numero minimo di bit: -256

Qual è il numero minimo di bit necessari a rappresentare

- -256

Calcoliamo  $|-256| = 256$  in base 2

100000000

Servono quindi 9 bit per rappresentare il modulo di -256. Occorrerebbe un bit in più per il segno, quindi arriveremmo a 10 bit MA...

...nei casi speciali delle esatte potenze di 2 ( $256 = 2^8$ ) ci troviamo agli estremi dell'intervallo rappresentabile. È opportuno ricordare che, in questi casi, per rappresentare una potenza esatta di 2 con segno negativo si può usare un bit in meno.

$$[-2^{9-1} ; 2^{9-1} - 1]$$

$$[-256 ; 255]$$

$$-256 \in [-256; 255]$$

Si eseguano le seguenti operazioni in binario

- $36 + 11$
- $38 - 83$

Si eseguano le seguenti operazioni in binario su 6 bit e si dica in quali occasioni si presenta overflow o underflow e si indichi il bit di carry

- $15 - 18$
- $-27 - 10$
- $-1 - 1$

# Operazioni in binario: 36+11

Si esegua la seguente operazione in binario

- $36 + 11$

$$\begin{array}{rcccccc} 1 & 0 & 0 & 1 & 0 & 0 & + \\ 0 & 0 & 1 & 0 & 1 & 1 & = \\ \hline 1 & 0 & 1 & 1 & 1 & 1 & \end{array}$$

$2^5 \quad 2^4 \quad 2^3 \quad 2^2 \quad 2^1 \quad 2^0$

$$1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 32 + 8 + 4 + 2 + 1 = 47$$

# Operazioni in binario: 38-83

Si esegua la seguente operazione in binario

- $38 - 83$

- $= 38 + (-83)$

$$\begin{array}{rcccccccc} 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & + \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & = \\ \hline 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \end{array}$$

# Operazioni in binario: 15-18

Si esegua la seguente operazione in binario su 6 bit e si dica se si presenta overflow o underflow e si indichi il bit di carry

- $15 - 18$ 
  - $= 15 + (-18)$

$$\begin{array}{rcccccc} 0 & 0 & 1 & 1 & 1 & 1 & + \\ 1 & 0 & 1 & 1 & 1 & 0 & = \\ \hline 1 & 1 & 1 & 1 & 0 & 1 & \end{array}$$

c

Operandi discordi  $\implies$  nessun rischio di overflow/underflow

- control bits
  - Carry 0
  - Overflow 0
- NO overflow
- NO underflow

# Operazioni in binario: -27-10

Si esegua la seguente operazione in binario su 6 bit e si dica se si presenta overflow o underflow e si indichi il bit di carry

- $-27 - 10$ 
  - $= -27 + (-10)$

$$\begin{array}{rcccccc} & 1 & 0 & 0 & 1 & 0 & 1 & + \\ & 1 & 1 & 0 & 1 & 1 & 0 & = \\ \hline \textcolor{red}{1} & 0 & 1 & 1 & 0 & 1 & 1 & \\ \text{c} & & & & & & & \end{array}$$

Operandi concordi  $\implies$  rischio di overflow/underflow

Operandi negativi, risultato positivo  $\implies$  underflow

- control bits
  - Carry  $\textcolor{red}{1}$
  - Overflow  $\textcolor{blue}{1}$
- NO overflow
- $\textcolor{blue}{SÌ}$  underflow

# Operazioni in binario: -1-1

Si esegua la seguente operazione in binario su 6 bit e si dica se si presenta overflow o underflow e si indichi il bit di carry

- $-1 - 1$

- $= -1 + (-1)$

$$\begin{array}{rcccccc} & 1 & 1 & 1 & 1 & 1 & 1 & + \\ & 1 & 1 & 1 & 1 & 1 & 1 & = \\ \hline \textcolor{red}{1} & 1 & 1 & 1 & 1 & 1 & 0 & \\ \text{c} & & & & & & & \end{array}$$

Operandi concordi  $\implies$  rischio di overflow/underflow

Operandi negativi, risultato negativo  $\implies$  NO underflow

- control bits
  - Carry  $\textcolor{red}{1}$
  - Overflow 0
- NO overflow
- NO underflow

Queste slides contengono elementi tratti da materiale di Gerardo Pelosi redatto per il corso di Fondamenti di Informatica per Ingegneria dell'Automazione a.a. 2014/15.

## Grazie per l'attenzione!

Licenza Beerware<sup>2</sup>

Queste slides sono opera di Stefano Cherubin. Mantenendo questa nota, puoi fare quello che vuoi con quest'opera. Se ci dovessimo incontrare e tu ritenessi che quest'opera lo valga, in cambio puoi offrirmi una birra.

---

<sup>2</sup><http://people.freebsd.org/~phk/>