UNIVERSITÀ DEGLI STUDI ROMA TRE

Dipartimento di Ingegneria
Industriale, Elettronica e Meccanica

Collegio Didattico di Ingegneria Elettronica

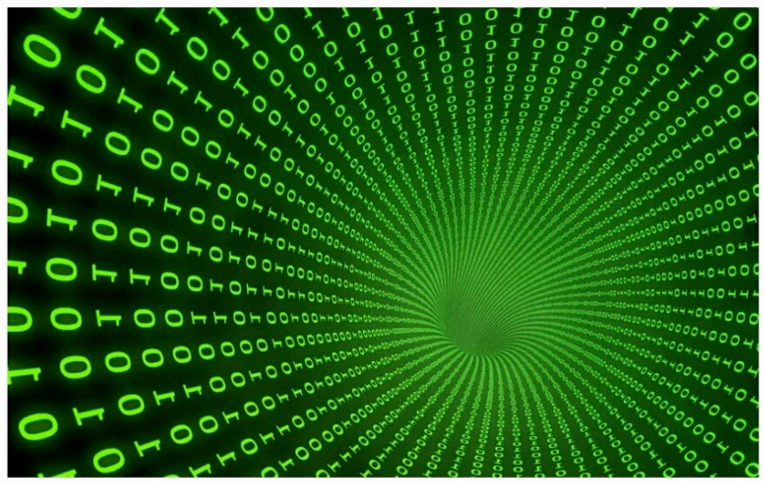
Elementi di Informatica

Rappresentazione Binaria

Giordano Da Lozzo e Giuseppe Sansonetti

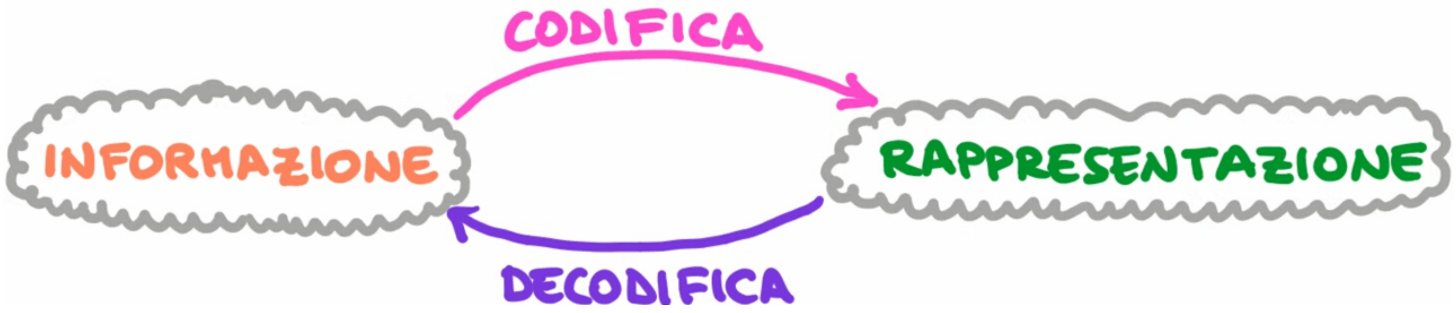
Come sono rappresentati i dati in un calcolatore?

Memoria → Celle di memoria → Bit



Ciascun dato deve essere rappresentato attraverso una sequenza di 0 e 1 di lunghezza finita.

Conversione di dati in sequenze di bit



- Numeri naturali $\mathbb{N}$
- Numeri interi relativi $\mathbb{Z}$

Oggi

- Numeri reali $\mathbb{R}$ $\Rightarrow$ Esprimi il numero come $(-1)^{s} \times (1.M_2) \times 2^{E_2 - 127}$
  $\Rightarrow$ Ottieni la rappresentazione SE_2M_2
- Caratteri $\Rightarrow$ Codifica il carattere con un numero intero (codice ASCII) $\Rightarrow$ Converti il codice in formato binario $\Rightarrow$ Rappresentazione
- Suoni, Immagini, Video, … $\Rightarrow$ …

Numeri naturali

(Passaggio da un sistema di numerazione decimale ad un sistema di numerazione binario senza curarsi dell’esigenza di utilizzare un numero finito di bit)

Notazione decimale – anche detta: notazione in base 10

E’ una notazione posizionale: le cifre hanno un valore dipendente dalla loro posizione (es. 563 è diverso da 635).

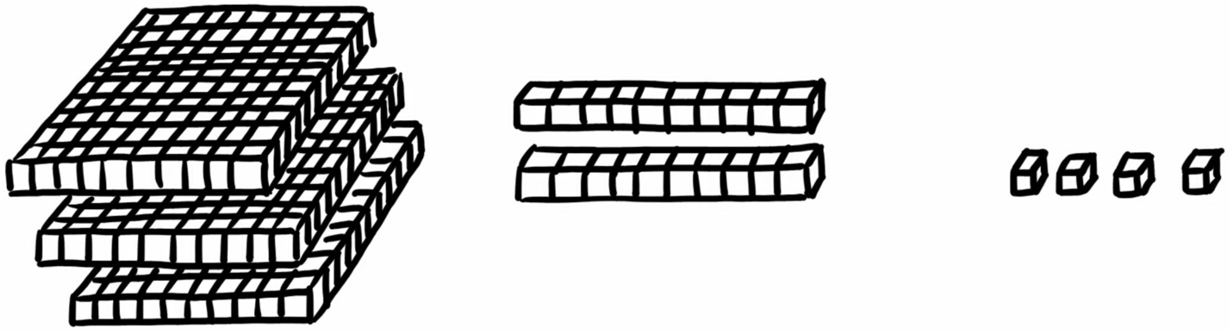
Esistono 10 cifre (0, 1, … 9).

Il valore di una cifra dipende dalla cifra stessa e dalla sua posizione.

Sistema di numerazione creato in India fra il 400 a.C. e il 400 d.C. ed esportato in Europa dagli arabi intorno al 900 d.C.

0	1	2	3	4	5	6	7	8	9
٠	١	٢	٣	٤	٥	٦	٧	٨	٩

Il numero 324



$3 \times 100 \quad + \quad 2 \times 10 \quad + \quad 4 \times 1$

$3 \times 10^2 \quad + \quad 2 \times 10^1 \quad + \quad 4 \times 10^0$

Base 10

$$47358 = 4\times10^4 + 7\times10^3 + 3\times10^2 + 5\times10^1 + 8\times10\hat{}0$$

Le cifre sono fattori per potenze con base 10
(ed esponenti 0, 1, 2, ..., da destra a sinistra).

Che succede se la base non è 10?

- Il numero 532 in base 8 vale $5 \times 8^2 + 3 \times 8^1 + 2 \times 8^0$
  $\Rightarrow$ E' uguale al numero 346 in base 10

- Il numero 10 in base 2 vale $1 \times 2^1 + 0 \times 2^0$
  $\Rightarrow$ E' uguale al numero 2 in base 10

- Il numero 39 in base 12 vale $3 \times 12^1 + 9 \times 12^0$
  $\Rightarrow$ E' uguale al numero 45 in base 10

Rappresentazione in base *b*

Ci sono b cifre disponibili

Base 10 – cifre 0, 1, ..., 9
Base 8 – cifre 0, 1, ..., 7
Base 16 – cifre 0, 1, ..., 9, A, B, C, D, E, F

Il numero $a_n a_{n-1} \dots a_1 a_0$ rappresenta

$$a_n \times b^n + a_{n-1} \times b^{n-1} + \cdots + a_1 b^1 + a_0 b^0$$

- a_n è la cifra più significativa
- a_0 è la cifra meno significativa

Contiamo insieme!

- Q: quali sono i primi numeri in base 8?

- Q: quali sono i primi numeri in base 2?

Contiamo insieme!

- Q: quali sono i primi numeri in base 8?
  - A: 0, 1, 2, …, 6, 7, 10, 11, 12, …, 16, 17, 20, 21, …, 76, 77, 100, 101, …
- Q: quali sono i primi numeri in base 2?
  - A: 0, 1, 10, 11, 100, 101, 110, 111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111, 10000, 10001, …

Domandoni

- Q: come si rappresenta in base 31 il numero che si rappresenta come 32 in base 10?

- Q: quanto vale 2+2 in base 5? E in base 4? E in base 3?

Domandoni

- Q: come si rappresenta in base 31 il numero che si rappresenta come 32 in base 10?
  - A: 11

- Q: quanto vale 2+2 in base 5? E in base 4? E in base 3?
  - A: 4, 10, 11

Sistemi di numerazione più utilizzati

- Base 10

Abbastanza cifre
Non troppe cifre
Ci viene naturale

- Base 2 – facilità di rappresentare due stati, corrispondenti ai due simboli 0 e 1, tramite grandezze elettromagnetiche in un calcolatore

- Base 16: compatta e facilmente convertibile a base 2

Più cifre, più potenza

Base 2	Base 10	Base 16
0	0	0
1	1	1
10	2	2
11	3	3
100	4	4
101	5	5
110	6	6
111	7	7
1000	8	8
1001	9	9
1010	10	A
…	…	…
1111	15	F
…	…	…
1100100	100	64
…	…	…
1111101000	1000	3E8

Rappresentazione binaria, anche detta in base 2

- Ci sono 2 cifre disponibili: 0 e 1

- Il numero $a_n a_{n-1} \dots a_1 a_0$ rappresenta $a_n \times 2^n + a_{n-1} \times 2^{n-1} + \cdots + a_1 \times 2^1 + a_0 \times 2^0$

Conversione da base 2 a base 10

- Data la rappresentazione binaria di un numero, vogliamo costruire la sua rappresentazione decimale (decodifica)

- Algoritmo
  - Input: $(a_n a_{n-1} \dots a_1 a_0)$ — Base 2
  - Output: $(a_n \times 2^n + a_{n-1} \times 2^{n-1} + \cdots + a_1 \times 2^1 + a_0 \times 2^0)$ — Base 10
- Esempi:
  - A: 1011 = $1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0$
    $8 + 0 + 2 + 1 = 11$
  - A: 110010 = $1\times 2^5 + 1\times 2^4 + 0\times 2^3 + 0\times 2^2 + 1\times 2^1 + 0\times 2^0$
    $32 + 16 + 0 + 0 + 2 + 0 = 50$

Conversione da base 10 a base 2

- Data la rappresentazione decimale di un numero, vogliamo costruire la sua rappresentazione binaria (codifica)

- Sfrutta la seguente idea. Se dividi il numero dato per 2, il resto è la cifra meno significativa della rappresentazione binaria del numero.

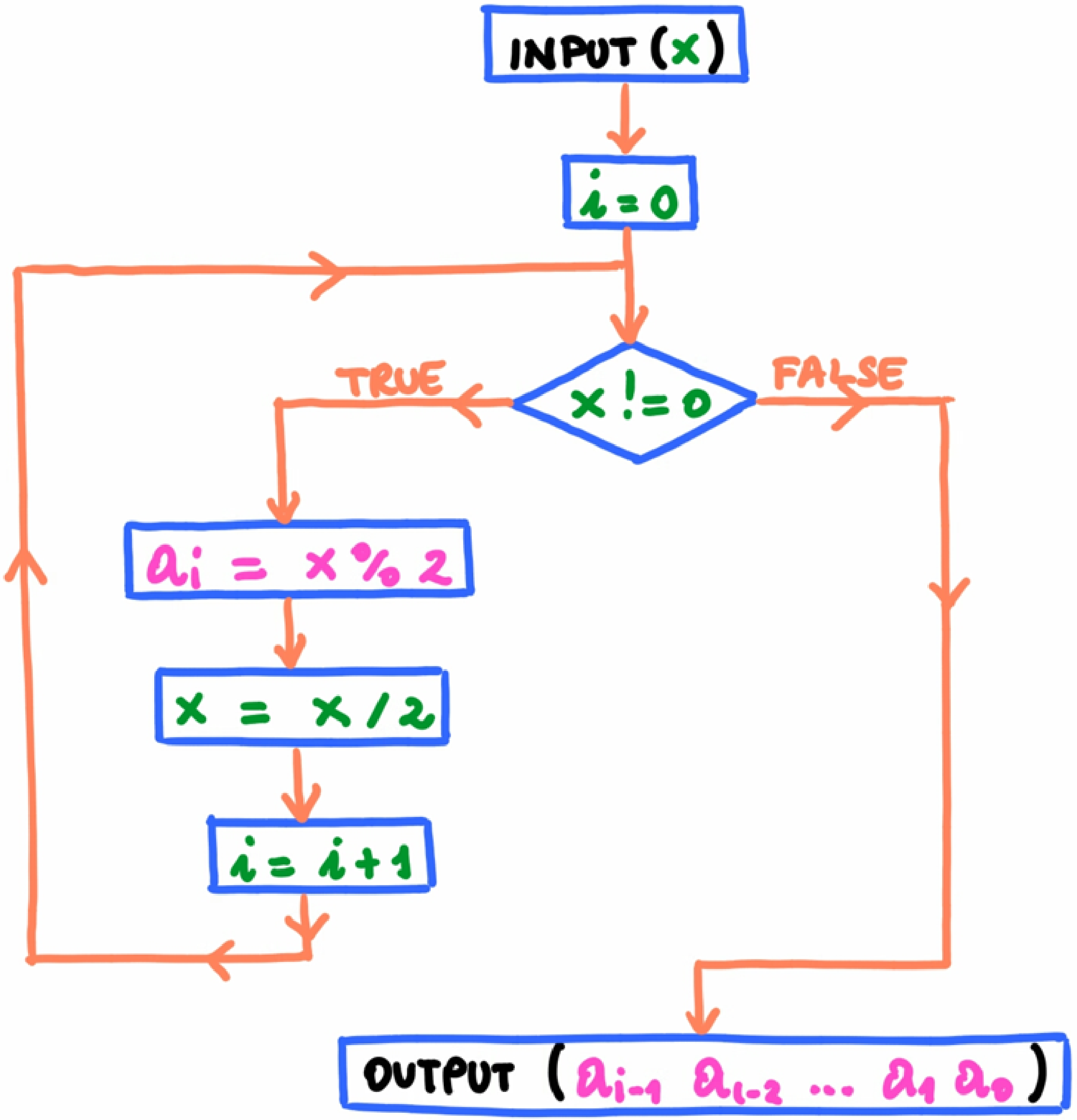
Conversione da base 10 a base 2

1. Input (x)
2. $i = 0$
3. Fintanto che ($x! = 0$)

    3.1 $a_i = x\%2$ (resto divisione intera)

    3.2 $x = x/2$ (divisione intera)

    3.3 $i = i + 1$
4. Output ($a_n a_{n-1} \dots a_1 a_0$)



INPUT (x)
i = 0
x != 0
TRUE
FALSE
a_i = x % 2
x = x / 2
i = i + 1
OUTPUT (a_{i-1} a_{i-2} ... a_1 a_0)

Sia $x = 50$ e $i = 0$

$x > 0 \Rightarrow a_0 = x\%2 = 0,$ $x = x/2{=}25,$ i=1
$x > 0 \Rightarrow a_1 = x\%2 = 1,$ $x = x/2{=}12,$ i=2
$x > 0 \Rightarrow a_2 = x\%2 = 0,$ $x = x/2{=}6,$ i=3
$x > 0 \Rightarrow a_3 = x\%2 = 0,$ $x = x/2{=}3,$ i=4
$x > 0 \Rightarrow a_4 = x\%2 = 1,$ $x = x/2{=}1,$ i=5
$x > 0 \Rightarrow a_5 = x\%2 = 1,$ $x = x/2{=}0,$ i=6

$x = 0 \Rightarrow Output(a_5 a_4 a_3 a_2 a_1 a_0) = 110010$

Operazioni dell’aritmetica binaria

- Ad esempio: somma, differenza, prodotto, …
- Q: come si possono eseguire?

Operazioni dell'aritmetica binaria

- Ad esempio: somma, differenza, prodotto, …

- Q: come si possono eseguire?
- A: puoi convertire gli operandi nella loro rappresentazione decimale, eseguire l'operazione e poi convertire il risultato nella rappresentazione binaria

- Ma puoi anche effettuare le operazioni direttamente sulle rappresentazioni binarie…

Somma

$$
\begin{array}{r}
101000101 \; + \\
1001110 \\
\hline
\end{array}
$$

Somma

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

Somma

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

Somma

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

$$
\begin{array}{cccccccccc}
 & & & & \multicolumn{2}{c}{+1} & & & & \\
1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & + \\
 & & 1 & 0 & 0 & 1 & 1 & 1 & 0 & \\
\hline
 & & & & & 0 & 0 & 1 & 1 &
\end{array}
$$

Somma

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

Somma

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

Somma

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

Somma

$$
\begin{array}{r}
1\,0\,1\,0\,0\,0\,1\,0\,1 \quad + \\
1\,0\,0\,1\,1\,1\,0 \quad\;\; \\
\hline
1\,0\,0\,1\,0\,0\,1\,1 \quad\;\;
\end{array}
$$

Somma

$$
\begin{array}{r}
101000101 \; + \\
1001110 \\
\hline
110010011
\end{array}
$$

Prodotto

$$
\begin{array}{r}
1\,1\,0\,1\,1\ \times \\
1\,0\,1 \\
\hline
\end{array}
$$

Prodotto

$$
\begin{array}{r}
1\ 1\ 0\ 1\ 1\ \times \\
1\ 0\ 1 \\
\hline
1\ 1\ 0\ 1\ 1
\end{array}
$$

Prodotto

$$
\begin{array}{cccccc}
 & 1 & 1 & 0 & 1 & 1 \times \\
 & & & 1 & 0 & 1 \\
\hline
 & 1 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & \\
\end{array}
$$

Prodotto

$$
\begin{array}{ccccccc}
 & & 1 & 1 & 0 & 1 & 1 & \times \\
 & & & & 1 & 0 & 1 & \\
\hline
 & & 1 & 1 & 0 & 1 & 1 & \\
 & 0 & 0 & 0 & 0 & 0 & & \\
1 & 1 & 0 & 1 & 1 & & & \\
\hline
\end{array}
$$

Prodotto

$$
\begin{array}{cccccccc}
 & & & 1 & 1 & 0 & 1 & 1 \;\times \\
 & & & & & 1 & 0 & 1 \\
\hline
 & & & 1 & 1 & 0 & 1 & 1 \\
 & & 0 & 0 & 0 & 0 & 0 & \\
 & 1 & 1 & 0 & 1 & 1 & & \\
\hline
1 & 0 & 0 & 0 & 0 & 1 & 1 & 1
\end{array}
$$

Differenza

$$
\begin{array}{r}
110101\,- \\
11110 \\
\hline
\end{array}
$$

Differenza

$$
\begin{array}{cccccc}
1 & 1 & 0 & 1 & 0 & 1 \; - \\
  & 1 & 1 & 1 & 1 & 0 \\
\hline
  &   &   &   &   & 1
\end{array}
$$

Differenza

$$\begin{array}{cccccccc} 1 & 1 & 0 & \overset{0}{\cancel{1}} & \overset{+2}{0} & 1 & - \\ & 1 & 1 & 1 & 1 & 0 & \\ \hline & & & & 1 & 1 & \end{array}$$

Differenza

$$\begin{array}{ccccccc}
 & & {\scriptstyle +2} & {\scriptstyle +2} & & & \\
1 & \overset{0}{\cancel{1}} & \overset{1}{\cancel{0}} & \overset{2}{\cancel{1}} & 0 & 1 & - \\
 & 1 & 1 & 1 & 1 & 0 & \\
\hline
 & & & 1 & 1 & 1 &
\end{array}$$

Differenza

$$\begin{array}{cccccccc} 1 & \overset{0}{\cancel{1}} & \overset{1}{\cancel{0}} & 1 & 0 & 1 & - \\ & 1 & 1 & 1 & 1 & 0 & \\ \hline & & 0 & 1 & 1 & 1 & \end{array}$$

Differenza

$$
\begin{array}{ccccccc}
 & +2 & & & & & \\
 & 2 & & & & & \\
\cancel{1} & \cancel{1} & 0 & 1 & 0 & 1 & - \\
 & 1 & 1 & 1 & 1 & 0 & \\
\hline
 & 1 & 0 & 1 & 1 & 1 &
\end{array}
$$

Numeri interi relativi

(nel sistema di numerazione binario, utilizzando un numero finito di bit)

Interi relativi con pochi bit

1. I calcolatori mettono a disposizione un numero limitato di bit per rappresentare ciascun numero $\Rightarrow$ non si possono rappresentare tutti i numeri!

2. Finora abbiamo descritto una maniera per codificare e decodificare numeri naturali. Come possiamo estendere la rappresentazione per gestire i numeri interi relativi?

Rappresentazioni con pochi bit

Assumiamo che la rappresentazione di ciascun numero consista di un numero fisso n di bit (es. $n = 32$).

0	1	1	1	0	0	1	1	1	0	1	0	0	0	0	1	0	1	0	0	0	0	0	1	1	1	0	1	0	1	1	0

Osserva: i numeri naturali rappresentabili secondo la codifica precedentemente descritta sono quelli nell'intervallo $[0; 2^n - 1]$, ovvero 2^n numeri distinti.

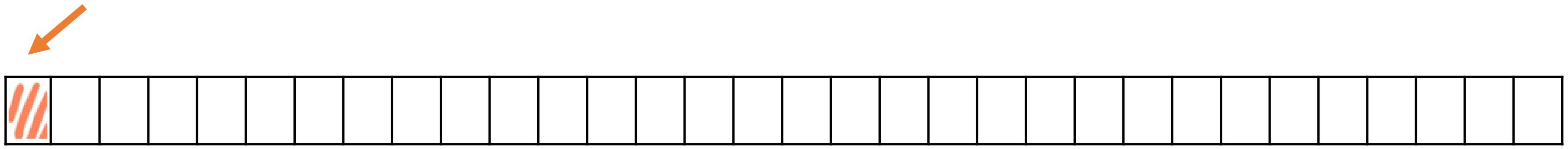
Rappresentazione di interi relativi

Esistono diverse rappresentazioni dei numeri interi relativi

- Modulo e segno
- Complemento a due

Rappresentazione modulo e segno

La cifra più significativa rappresenta il segno, le altre il modulo.



Codifica: dato un numero intero relativo x, la sua cifra più significativa è pari a 1 se x è negativo, è pari a 0 se x è positivo.

La codifica del modulo si ottiene come abbiamo visto per i numeri naturali (ma con $n-1$ cifre).

Decodifica: il segno è $+$ se la cifra più significativa è 0, è $-$ altrimenti.
La decodifica del modulo si ottiene come abbiamo visto sui numeri naturali.

Modulo e segno: esempi (con n=8)

Qual è la rappresentazione modulo e segno del numero -5 (espresso in base 10) con 8 bit?

- Numero negativo ⇒ cifra più significativa è 1
- Il modulo 5 si rappresenta come 1 0 1 in base 2.

  4 2 1

⇒La rappresentazione è 10000101: osserva gli 0 di riempimento sul modulo.

Quale numero si rappresenta come 00001011 in modulo e segno?

- Cifra più significativa è 0 ⇒ numero positivo
- La rappresentazione del modulo è 1011 ⇒ il modulo è 8+2+1=11

  8 4 2 1

⇒ Il numero è +11

Modulo e segno: svantaggi

Doppia rappresentazione per lo zero: sia 0000…0000 che 1000…0000 rappresentano il numero zero.

Possono essere rappresentati solo $2^n - 1$ numeri, quelli nell'intervallo $[-2^{n-1} + 1; 2^{n-1} - 1]$.

Rappresentazione non circolare: sommando 1 all'intero più grande rappresentabile (0111…1111) non si ottiene l'intero più piccolo rappresentabile (1111…1111).

Complemento a due

Codifica: dato un numero intero relativo x, allora se x è positivo la codifica coincide con la codifica del numero naturale x, altrimenti coincide con la codifica del numero naturale $2^n + x$.

Qual è la rappresentazione in complemento a 2 con 8 bit del numero -67?

Il numero è negativo $\Rightarrow$ sommagli 2^n $\Rightarrow -67 + 2^8 = -67 + 256 = 189$

Rappresenta $189 \Rightarrow \underset{128}{1}\ \underset{64}{0}\ \underset{32}{1}\ \underset{16}{1}\ \underset{8}{1}\ \underset{4}{1}\ \underset{2}{0}\ \underset{1}{1}$

Qual è la rappresentazione in complemento a 2 con 8 bit del numero 37?

Il numero è positivo $\Rightarrow$ rappresentalo $\underset{128}{0}\ \underset{64}{0}\ \underset{32}{1}\ \underset{16}{0}\ \underset{8}{0}\ \underset{4}{1}\ \underset{2}{0}\ \underset{1}{1}$

Complemento a due

Algoritmo alternativo per la codifica dei numeri negativi

- Rappresenta il modulo
- Complementa le cifre (1 al posto di 0 e viceversa)
- Somma 1

$\boxed{-67}$
$\Rightarrow 01000011$
$\Rightarrow 10111100$
$\Rightarrow 10111101$

Complemento a due: decodifica

Sfrutta:

I numeri che iniziano con 0 sono positivi o nulli

I numeri che iniziano con 1 sono negativi

Decodifica A: se la cifra più significativa è 0, allora stessa decodifica dei numeri naturali.
Altrimenti:

- Stessa decodifica dei numeri naturali
- Sottrai 2^n

$\boxed{10110001}$
$\Rightarrow +177$
$\Rightarrow +177 - 256 = -79$

Decodifica B: se la cifra più significativa è 0, allora stessa decodifica dei numeri naturali.
Altrimenti:

- Complementa i bit
- Somma uno
- Stessa decodifica dei numeri naturali
- Cambia segno

$\boxed{10110001}$
$\Rightarrow 01001110$
$\Rightarrow 01001111$
$\Rightarrow 79$
$\Rightarrow -79$

Complemento a due: vantaggi

Una sola rappresentazione per lo zero (0000…0000); il numero 1000…0000 rappresenta -2^{n-1} e non 0

Possono essere rappresentati 2^n numeri, quelli nell'intervallo $[-2^{n-1}; 2^{n-1} - 1]$.

Le somme si fanno come per i numeri naturali, anche se i numeri sono negativi, ignorando una eventuale cifra di riporto oltre i bit usati.

$$
\begin{array}{rl}
01100111 \; + & (103) \\
10100101 & (-91) \\
\hline
\cancel{1}00001100 & (12)
\end{array}
$$

Stessa cosa per le sottrazioni

Complemento a due: circolarità

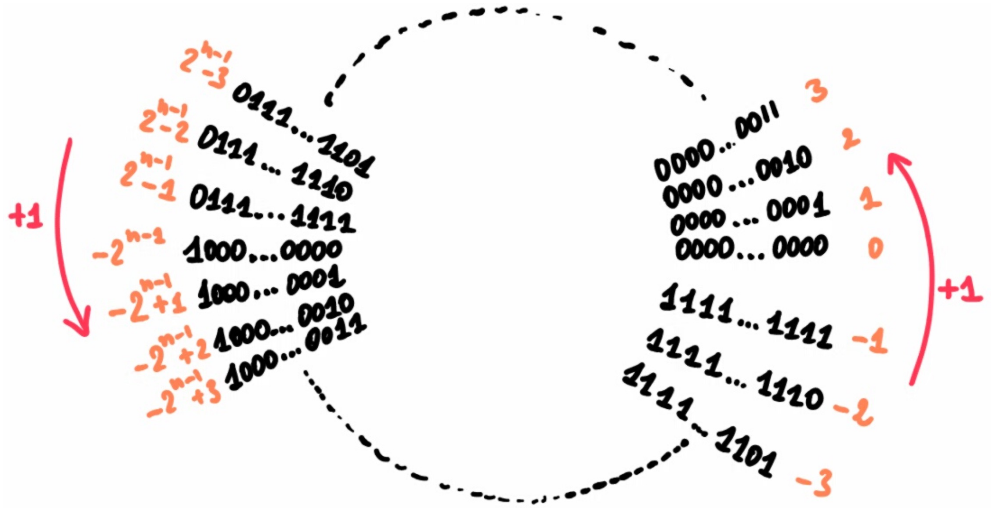
I numeri che iniziano con 0 sono positivi o nulli, dal più piccolo (0000…0000) al più grande (0111…1111). I numeri che iniziano con 1 sono negativi, dal più piccolo (1000…0000) al più grande (1111…1111)

Sommando 1 a -1=1111…1111

Si ottiene 0=0000…0000

Sommando 1 a $2^{n-1}-1 = 011\ldots1111$

Si ottiene $-2^{n-1} = 1000\ldots000$



Ciascun numero x rappresenta una classe di equivalenza di numeri interi: tutti i numeri $x + j * 2^n, con\ j \in \mathbb{Z}$

Altre risorse

- Bellini, Guidi: Linguaggio C — Appendice D1-D4