Sistemi di numerazione e rappresentazione binaria dei numeri interi

CORSO DI ARCHITETTURA DEGLI ELABORATORI E LABORATORIO
MODULO LABORATORIO

GABRIELLA VERGA

Il Sistema Decimale

E' un sistema di numerazione posizionale in base 10

10 SIMBOLI: 0,1,2,3,4,5,6,7,8,9

E' posizionale: ha importanza la posizione assunta da ogni cifra all'interno di un numero.

ESEMPIO: 102

2 unità

0 decina

1 centinaia

Sistemi di numerazione Additivo

La Numerazione romana non è posizionale ma **ADDITIVA** perché il valore complessivo del numero è dato dalla somma dei valori dei simboli, indipendentemente dalla loro posizione.

I = 1, IV = 4; V = 5; IX = 9; X = 10; XL = 40; L = 50; XC = 90; C = 100

Notiamo..

Sistema Posizionale:

- *Con un numero LIMITATO DI SIMBOLI (10) è possibile rappresentare QUALUNQUE QUANTITA’*
- *Il sistema è estremamente economico e flessibile*

Sistema Addizionale

- *Al crescere della quantità hanno sempre bisogno di NUOVI SIMBOLI*

Sistemi di numerazione posizionali

Un sistema di numerazione è definito da:

- Un intero **B** detto **base**;

- Un insieme di B simboli **S_B = {s_0, … , s_{B-1}}** ognuno dei quali rappresenta le quantità **0,1,2,…..,B-1**

Un numero a n cifre $p_{(n-1)}$ $p_{(n-2)}$ ….. p_1 p_0 con $p_{(i)} \in S_B$ e i = 0,….,n-1 può essere rappresentato come SOMMA DI POTENZE DELLA BASE:

$$\sum_{i=0}^{n-1} (p_{(i)} \cdot B^i)$$

Esempi

BASE 2 (binaria)

$(\mathbf{1100})_2 = 1 * 2^3 + 1 * 2^2 + 0 * 2^1 + 0 * 2^0 = 8 + 4 + 0 + 0 = (\mathbf{12})_{10}$

$(\mathbf{00110010})_2 = 0 * 2^7 + 0 * 2^6 + 1 * 2^5 + 1 * 2^4 + 0 * 2^3 + 0 * 2^2 + 1 * 2^1 + 0 * 2^0 = 1 * 2^5 + 1 * 2^4 + 1 * 2^1 = 32 + 16 + 2 = (\mathbf{50})_{10}$

BASE 8 (ottale)

$(\mathbf{121})_8 = 1 * 8^2 + 2 * 8^1 + 1 * 8^0 = 64 + 16 + 1 = (\mathbf{81})_{10}$

BASE 16 (esadecimale)

$(\mathbf{128})_{16} = 1 * 16^2 + 2 * 16^1 + 8 * 16^0 = 256 + 32 + 8 = (\mathbf{296})_{10}$

Conversione da base 10 a base B

La conversione di un numero da base 10 a base B usa la tecnica delle divisioni successive:

1) Sia N il numero (in base 10) da convertire;

2) Si calcola la divisione intera N = N / B e si mette da parte il resto R della divisione;

3) Se N > 0 si va al passo 2;

4) Se N = 0 si riportano i vari RESTI da destra verso sinistra: essi rappresentano il numero convertito in base B.

Esempi:

DA BASE 10 A BASE 2 →

14 = 1110

327 = 101000110

DA BASE 10 A BASE 8

65 = 101

DA BASE 10 A BASE 16

4091 = FFB

479 = 1DF

Esempio: conversione in base 2

CONVERTIRE 13 in base 2 (da numero decimale a numero binario)

Operazione	Quoziente	Resto
13		
/2	6	1
/2	3	0
/2	1	1
/2	0	1

$(\mathbf{1101})_2 = 1 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0 = 8 + 4 + 0 + 1 = \mathbf{13}$

Esempio: conversione in base 8

CONVERTIRE 1158 in base 8 (da numero decimale a numero ottale)

Operazione	Quoziente	Resto
1158		
/8	144	6
/8	18	0
/8	2	2
/8	0	2

$(\mathbf{2206})_8 = 2 * 8^3 + 2 * 8^2 + 0 * 8^1 + 6 * 8^0 =$

$2 * 512 + 2 * 64 + 6 = 1024 + 128 + 6 = (\mathbf{1158})_{10}$

Metodo alternativo

CONVERSIONE IN BASE 2

CONVERTIRE 112 in base 2 (da numero decimale a numero binario)

POTENZA	Risultato
2^0	1
2^1	2
2^2	4
2^3	8
2^4	16
2^5	32
2^6	64
2^7	128
2^8	256

$(\mathbf{112})_{10} = 64 + 32 + 16 =$

$= 1 * 2^6 + 1 * 2^5 + 1 * 2^4 =$

$= (\mathbf{1110000})_2$

Esempio: conversione in base 2

CONVERTIRE 112 in base 2 (da numero decimale a numero binario)

Operazione	Quoziente	Resto
112		
/2	56	0
/2	28	0
/2	14	0
/2	7	0
/2	3	1
/2	1	1
/2	0	1

$$(\mathbf{112})_{10} = (\mathbf{1110000})_2$$

Dall’elettricità all’aritmetica (1)

Il calcolatore è una macchina composta da CIRCUITI e COLLEGAMENTI ELETTRICI.

Immaginiamo di poter connettere delle lampadine ai vari collegamenti presenti dentro un computer.

Effettuando delle «istantanee» per valutare la luminosità delle lampadine si nota che la lampadina o E’ ACCESA o SPENTA. **Non esistono luminosità parziali!**

I concetti **ON/OFF** possono essere rappresentati tramite NUMERI:

- OFF = 0
- ON = 1

Dall’elettricità all’aritmetica (2)

Il sistema di numerazione binaria è la soluzione perfetta per rappresentare valori ON/OFF.

Individuata una tipologia di informazione, si possono inserire delle regole non ambigue per rappresentare l’informazione come sequenze binarie.

Il modo più naturale per rappresentare un numero in un calcolatore è tramite una stringa di bit, chiamato numero binario.

Numeri binari

- Numeri binari
- Operazioni Aritmetiche (addizione e sottrazione)
- Numeri in virgola mobile

Numeri interi in BINARIO

$\mathbf{B} = b_{n-1}\, b_{n-2} \, ... \, b_1 \, b_0$ con $b_i \in \mathbf{\{0,1\}}$ e $i = 0,....,n-1$

La stringa B può rappresentare un valore **numerico intero** casuale *val(B)* compreso nell'intervallo **$[0,2^n)$** che si calcola con la seguente formula:

$$val(B) = b_{n-1} * 2^{n-1} + b_{n-2} * 2^{n-2} + ... + b_1 * 2^1 + b_0 * 2^0 = \sum_{i=0}^{n-1} b_i \, 2^i$$

Rappresentazione di numeri interi relativi

Le tre tecniche più importanti sono

1. Segno e valore assoluto
2. Complemento a uno
3. Complemento a due

PROPRIETA' IN COMUNE: il bit più a sinistra rappresenta il segno

Se b_{n-1} vale 0 ➔ val(B) è positivo o nullo.

Se b_{n-1} vale 1 ➔ val(B) è negativo o nullo.

b_{n-1} è detto bit di segno

Le tre tecniche

In tutti e tre i casi i valori positivi si distribuiscono nello stesso modo rispetto alle stringhe di bit che li codificano, mentre per i valori negativi si differenziano.

1. **Segno e valore assoluto**: si commuta il bit di segno b_{n-1} da 0 a 1, mentre gli altri bit restano invariati.
2. **Complemento a uno:** si commuta qualsiasi bit da 0 a 1 e da 1 a 0.
3. **Complemento a due:** si aggiunge 1 al complemento a uno.

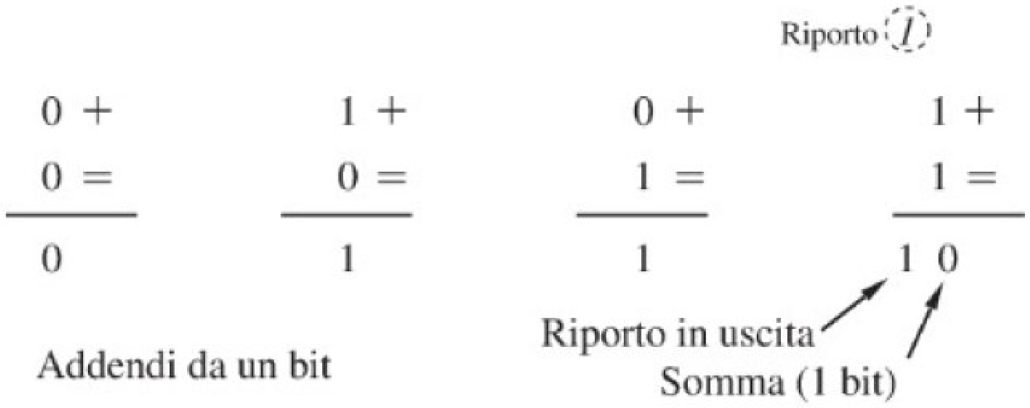
La tecnica di complemento a due anche se sembra la meno intuitiva è quella più usata universalmente nei calcolatori e risulta più efficiente nel calcolo della addizione e sottrazione.

Esempi

Stringa (B)				Segno e valore assoluto	Complemento a 1	Complemento a 2
b_3	b_2	b_1	b_0			
0	1	0	0	+4	+4	+4
0	0	1	1	+3	+3	+3
0	0	1	0	+2	+2	+2
0	0	0	1	+1	+1	+1
0	**0**	**0**	**0**	**0**	**0**	**0**
1	0	0	0	-0	-7	-8
1	0	0	1	-1	-6	-7
1	0	1	0	-2	-5	-6
1	0	1	1	-3	-4	-5

Addizione di numeri naturali

Il **riporto in uscita** della cifra precedente viene assegnato come **RIPORTO IN ENTRATA** alla successiva



Addizione e sottrazione di numeri naturali

REGOLE per calcolare addizione e sottrazione algebrica di numeri relativi codificati in complemento a due (con n bit):

1. per calcolare l'addizione algebrica si applica l'algoritmo di addizione in aritmetica binaria naturale, ma si TRASCURA il riporto in uscita dalla posizione dei bit più significativi degli addendi;

2. supponendo che A e B siano minuendo e sottraendo, rispettivamente si calcola la sottrazione algebrica A – B nel seguente modo:
   - prima si complementa a 2 il sottraendo B (B' → -B).
   - si calcola l'addizione algebrica A + B' seconda la regola 1.

Esempi

(a)
$$\begin{array}{r} 0010\ + \\ 0011\ = \\ \hline 0101 \end{array} \qquad \begin{array}{r} (+2) \\ (+3) \\ \hline (+5) \end{array}$$

(b)
$$\begin{array}{r} 0100\ + \\ 1010\ = \\ \hline 1110 \end{array} \qquad \begin{array}{r} (+4) \\ (-6) \\ \hline (-2) \end{array}$$

(c)
$$\begin{array}{r} 1011\ + \\ 1110\ = \\ \hline 1001 \end{array} \qquad \begin{array}{r} (-5) \\ (-2) \\ \hline (-7) \end{array}$$

(d)
$$\begin{array}{r} 0111\ + \\ 1101\ = \\ \hline 0100 \end{array} \qquad \begin{array}{r} (+7) \\ (-3) \\ \hline (+4) \end{array}$$

Estensione e riduzione del segno

Accade spesso che si debba aumentare o diminuire il numero di bit usati per codificare un numero relativo in complemento a due (ad esempio per adattarlo a una parola di memoria o a un registro di processore avente dimensione differente).

Le regole sono:

- **AUMENTARE** o **ESTENDERE**:
  - se n > 0 (inizia con 0) ➔ aggiungere altri bit 0 a sinistra.
  - se n < 0 (inizia con 1) ➔ aggiungere altri bit 1 a sinistra.
- **DIMINUIRE** o **RIDURRE**:
  - se n > 0 (inizia con 0) ➔ si possono togliere bit 0 a sinistra (smettere PRIMA che emerge in testa 1)
  - Se n < 0 (inizia con 1) ➔ si possono togliere bit 1 a sinistra (smettere PRIMA che emerge in testa 0)

ESTENSIONE	RIDUZIONE
011 (3) ➔ 0..011	00011 ➔ 011
101 (-3) ➔ 1..101	11101 ➔ 101

Evento di Trabocco (overflow)

Il risultato di addizione e sottrazione in complemento a due è corretto se è COMPRESO nell'intervallo $[-2^{n-1}, 2^{n-1})$

In caso contrario avviene un **evento di TRABOCCO**.

REGOLA

- *SI PUO' VERIFICARE (non necessariamente) TRABOCCO **SOLO SE** GLI ADDENDI SONO CONCORDI IN SEGNO*
- *SI VERIFICA TRABOCCO **SE E SOLO** SE I DUE ADDENDI SONO CONCORDI IN SEGNO E IL BIT DI SEGNO DELLA SOMMA E' DIVERSO DA QUELLO DEGLI ADDENDI*

ESEMPIO: +7+4 con n = 4 si ottiene 1011 (-5)