Codifica

Rappresentazione di numeri in memoria

Rappresentazione polinomiale dei numeri

Un numero decimale si rappresenta in notazione polinomiale moltiplicando ciascuna cifra a sinistra della virgola per una potenza crescente della base, mentre le cifre della parte a destra della virgola vengono moltiplicate per le potenze decrescenti della base ad esempio:

12.34	**=**	**$1 \cdot 10^{1}$**	**+**	**$2 \cdot 10^{0}$**	**+**	**$3 \cdot 10^{-1}$**	**+**	**$4 \cdot 10^{-2}$**
	=	10	+	2	+	0.3	+	0.04

Trasformazione in decimale di numeri binari con la virgola

Per trasformare in decimale il numero binario con la virgola:

101,011

lo si rappresenta in forma polinomiale:

$1\cdot2^2$	+	$0\cdot2^1$	+	$1\cdot2^0$	+	$0\cdot2^{-1}$	+	$1\cdot2^{-2}$	+	$1\cdot2^{-3}$
4	+	0	+	1	+	0	+	0,25	+	0,125

Quindi:

101,011 = 5,375

Trasformazione in binario di numeri con la virgola

La parte intera si trasforma in binario con le divisioni per 2, mentre la parte a destra della virgola si ottiene con moltiplicazioni per 2, la parte intera del prodotto è la cifra binaria da scegliere. Il procedimento termina quando la parte decimale diventa 0. (segue esempio)

Esempio di conversione in binario di numeri con la virgola

Convertire in binario il numero: 6,75.

La parte intera si converte in binario mediante divisioni successive per 2, ottenendo la rappresentazione di 6: 110.

La parte decimale si moltiplica per 2:

- $0{,}75 \cdot 2 = 1{,}5$ - la parte intera 1 è la prima cifra dopo la virgola del numero binario.

Si prosegue moltiplicando per 2 la parte decimale di 1,5, cioè 0,5:

- $0{,}5 \cdot 2 = 1{,}0$ - la parte intera 1 è la seconda cifra dopo la virgola del numero binario.

Il procedimento termina perché la parte a destra della virgola è 0.

Quindi:

$$(6{,}75)_{10} = (110{,}11)_2$$

Numeri periodici

La conversione in binario di un numero con la virgola potrebbe produrre un numero periodico, anche se il numero decimale non lo è.

Si voglia convertire in binario il numero

$$(6,6)_{10}$$

La parte intera si codifica mediante le divisioni per 2 e si trova che

$$(6)_{10} = (110)_2$$

segue

Numeri periodici

Per convertire in binario la parte frazionaria del numero, cioè 0.6, si segue il seguente procedimento:

Parte decimale per 2	Cifra binaria dopo la virgola
0,6·2 = 1,2	1
0,2·2 = 0,4	0
0,4·2 = 0,8	0
0,8·2 = 1,6	1

La parte intera di ogni prodotto è la prossima cifra dopo la virgola, mentre la parte decimale viene usata per svolgere un altro prodotto.
Il processo si arresta quando si trova una parte decimale già ottenuta.
Nell'esempio precedente, la parte decimale 0.6 era già stata ottenuta.

$$(6.6)_{10} = (110,1001)_2$$

Rappresentazione dei numeri reali

In qualsiasi base sia espresso, un numero si può rappresentare nella forma:

$$\pm M \cdot B^{\pm E}$$

Dove:

- $\pm$M è la mantissa con il segno
- B è la base
- $\pm$E è l'esponente con segno

Ad esempio:

$$123{,}45 = 0{,}12345 \cdot 10^{3}$$

0,12345 è la mantissa

3 è l'esponente

Virgola mobile

La rappresentazione del numero nel formato $\pm M \cdot B^{\pm E}$ prende il nome di Floating Point (virgola mobile).

In particolare si preferisce la forma normalizzata, cioè quella in cui la parte intera è formata dalla sola cifra binaria 1 che, essendo nota, non viene memorizzata.

Segno (1 bit)	Esponente	Mantissa

Il segno può assumere i valori 0 per indicare il segno positivo della mantissa e 1 per indicare il segno negativo.
L'esponente viene codificato in eccesso 127.

Standard IEEE754

Segno	Esponente	Mantissa		Nr. bit	Nr. Byte
1 bit	8 bit	23 bit	Singola precisione	32	4
1 bit	11 bit	52 bit	Doppia precisione	64	8
1 bit	15 bit	64 bit	Precisione Estesa	80	10

Nella rappresentazione in singola precisione, il valore nel campo esponente

- 127 indica l'esponente 0
- tra 1 e 126 indica l'esponente negativo
- tra 128 e 255 indica l'esponente positivo

Sono previsti tre codici speciali

- Mantissa = 0 e esponente = FF $\rightarrow$ infinito
- Mantissa $\neq$ 0 e esponente = FF $\rightarrow$ NaN (Not a Number)
- Tutti i bit = 0 $\rightarrow$ 0.0

Standard IEEE754 - Esempio

$$(25.75)_{10} = (11001.11)_2$$

$(11001.11)_2$ in forma normalizzata è: $1.100111 \cdot 2^4$

Il bit 1 della parte intera non è memorizzato.

L'esponente 4, in codice eccesso 127 è e=131

$$(131)_{10} = (1000\ 0011)_2$$

In singola precisione il numero si codifica:

Segno	Esponente	Mantissa
0	1000 0011	100 1110 0000 0000 0000 0000
1 bit	8 bit	23 bit

Standard IEEE754 - Esempio

Conversione da semplice precisione a decimale:

Segno	Esponente	Mantissa
1	1000 0001	100 0111 0000 0000 0000 0000

- Il bit di segno indica che il numero è negativo.
- L'esponente $(1000\ 0001)_2 = 129$, quindi l'esponente è $129-127 = 2$
- La mantissa, in forma normalizzata, è 1.100 0111

Quindi il numero rappresentato è:

$$1.100\ 0111 \cdot 2^2 = 110.00111$$

Corrispondente a $(-6.21875)_{10}$

Rappresentazione di numeri con segno

Il bit più significativo (MSB) di un operando è interpretato come segno del numero.

Si assume che se il MSB è

- 1 il numero è negativo
- 0 il numero è positivo

I restanti bit sono rappresentati in complemento a 2. Non costituiscono il valore assoluto del numero.

Complemento a 10

L'ipotesi fondamentale è il numero di cifre riservate a memorizzare il numero. Si supponga di avere a disposizione 2 cifre decimali. Si possono cioè memorizzare i numeri da 0 a 99.

Si decide che i numeri da 0 a 49 siano positivi e i numeri da 50 a 99 siano negativi.

Esempio: Sia A = 27 e B = 15. La sottrazione A-B viene ricondotta alla somma: A + (-B).

Il numero negativo -B viene rappresentato in complemento a 100, cioè –B=(100-15)=85. Infatti 85 rientra nell'intervallo dei numeri negativi.

Quindi la sottrazione A-B=A+(-B) è:

	27	+
	85	=
1	12	

Il risultato è 12. Il riporto si perde, perché, per ipotesi, si possono memorizzare solo 2 cifre.

Complemento a 2

Per rappresentare il sottraendo in complemento a 2:

- Si inverte ciascuna cifra binaria del numero
- Si aggiunge 1

Esempio:

Si assuma di avere 4 bit a disposizione per memorizzare il numero. Il complemento a 2 del numero 0001 si ottiene:

(invertire ciascun bit) 1110

(sommare 1) 1111

Il complemento a due di 0001 è 1111, che corrisponde alla codifica del numero negativo -1 in base 10

sottrazione

Si abbiano a disposizione 4 bit.

Si possono rappresentare 16 numeri.

- Gli 8 numeri da 0000 a 0111 si intendono positivi
- Gli 8 numeri da 1000 a 1111 si intendono negativi.

Esempio (risultato positivo)

Sia A = 0101 (=5_{10}) e B = 0011 (=3_{10})

La codifica in complemento a 2 di 0011 è:

(inversione delle cifre) 1100

(somma di 1) 1101

Quindi:

5+	0101	+
-3	1101	=
2	(1) 0010	

- La cifra (1) si perde, perché per ipotesi si hanno a disposizione solo 4 bit
- Il segno 0 del risultato indica che il numero è positivo

Esempio (risultato negativo)

Sia A = 0011 (=3_{10}) e B = 0101 (=5_{10})

La codifica in complemento a 2 di 0101 è:

(inversione delle cifre) 1010

(somma di 1) 1011

Quindi:

3+	0011	+
-5	1011	=
-2	1110	

Il bit del segno indica che il numero è negativo. Per trovare il valore assoluto del numero bisogna decodificare il valore 1110 in complemento a 2:
(inversione delle cifre) 0001
(somma di 1) 0010.
Il valore assoluto è 2