Architettura degli elaboratori

UniVR - Dipartimento di Informatica

Fabio Irimie

1° Semestre 2023/2024

Indice

1 Numeri razionali a virgola mobile

Gli standard della virgola mobile sono: IEEE 754/85 e IEEE 754/19. Questo standard è stato rivisto molte volte e ora viene usato da tutte le codifiche per i numeri in virgola mobile.

Il numero viene separato in due parti: Mantissa (M) e una base (b) con un esponente (e).

$$N = +- M * b^{+-E}$$

Questo permette di dividere il numero in modo da poter scegliere quanti bit dedicare alla mantissa e quanti all'esponente.

Ci sono 2 problemi però:

- bisogna scegliere la base in cui fare la codifica (base 2)
- divisione bit tra M e E (23 M, 8 E, 1 S)
- rappresentazione univoca (1.0...)
- bisogna trovare un modo per rappresentare gli errori

Un numero in base 10 si può rappresentare in più modi> $120_{10} = 12 * 10^1 = 120 * 10^0 = 1.20 * 10^2$

Se la mantissa e la base sono in base 2 le operazioni tra numeri sono agevolate.

$0110 * 2 = 1100$ è uno shift a sinistra in binario.

$1010/2 = 0101$ è uno shift a destra in binario.

1.1 Divisione tra bit con mantissa e base diversa

Un numero è rappresentabile in 2 modi:

- 32 bit (singola precisione / float)
- 64 bit (doppia precisione / double)

Prendiamo in considerazione 32 bit, ora dobbiamo decidere quanti bit dedicare alla mantissa e alla base

2^{+-E}

$|E| = 4bit = 2^{+7}$

$5bit = 2^{+15}$

$6bit = 2^{+31}$

$7bit = 2^{+63}$

$8bit = 2^{+127}$

L'impatto dei bit sull'esponente è doppiamente esponenziale, quindi cresce tantissimo. Tra tutti i bit a disposizione ne dedichiamo 8 all'esponente, $32-8 = 24$ bit rimanenti, quindi 23 bit vengono assegnati alla mantissa e 1 bit viene assegnato al segno.

Per la rappresentazione univoca la mantissa si codifica in virgola fissa. Cioè si parte da una mantissa con un punto fisso e dividendo o moltiplicando (shift) si può spostare la virgola per arrivare alla forma 1.00000... e questa forma è la rappresentazione univoca. Questa operazioe si chiama normalizzazione e visto che la rappresentazione è sempre la stessa l'1. non viene rappresentato, quindi viene inserito nella mantissa solo tutto ciò che viene dopo l'1. .

Se lavorassimo con un esponente in complemento a due ci sarebbe il seguente problema: 00000000000...0 $= 1 * 2^0 = 1$

Allora si è deciso di codificare l'esponente in Eccesso 127. Quindi per rappresentare lo zero si usa come esponente il minore numero possibile: $1 * 2^{127} = 0$ Per codificare i numeri si somma 127 al numero desiderato e visto che i numeri possibili ora vanno da -127 a +127 se codifichiamo il risultato in modulo avremo dei numeri da 0 a 256.

Esempio 1.1
1 01110111 0110...0
$M = -(1 + 1/4 + 1/8) * 2 = -(11/8) * 2^E$
$E = (1 + 2 + 4 + 16 + 32 + 64) - 127 = 119 - 127 = -8$
$N = -11/8 * 2^{-8}$

Esercizio 1.1
Codifica $+(4 + \frac{1}{2} + \frac{1}{16}) * 2^{+34}$

- 0 00000000 0...0 $= +0$
- 1 00000000 0...0 $= -0$

Quando l'esponente è tutto 1 e la mantissa tutta 0 allora equivale a $infinito$ + o - in base al primo bit. Se invece la mantissa è diversa da 0 con esponente tutti 1 allora rappresenta un errore NaN.

Somma: