IEEE 754

· Singola precisione virgola mobile

$$(-1)^S \cdot F \cdot 2^E$$

F mantissa di 23 bit
E esponente di 8 bit
S segno di 1 bit

si può incappare in situazioni di overflow e underflow.
Per ridurre questa situazione si introduce la doppia precisione

Doppia precisione virgola mobile

F mantissa di 52 bit
E esponente di 11 bit
S segno di 1 bit

in realtà essendo il primo bit implicito a 1
la mantissa è da 53 bit

$$(-1)^S \cdot (1+F) \cdot 2^E$$

Questo è il formato utilizzato da IEEE-754 _ mantissa diventa frazione

Cos'è IEEE-754

Lo standard IEEE-754 è lo standard più utilizzato nel campo del calcolo automatico.

Notazione polarizzata

Problema

Problema: un numero negativo apparirà come un numero grande.

Sol: si somma 127 (singola precisione) o 1023 (doppia precisione)

Es: $-1 + 127 = 126 \rightarrow 01111110$ esponente

$1 + 127 = 128 \rightarrow 10000000$ esponente

Si ottiene così: $(-1)^S \cdot (1+F) \cdot 2^{E-P}$ P = polarizzazione

Esempio: $(-0,75)_{10} \rightarrow$? IEEE 754

Singola precisione

1. $-0,75 = 0,5 + 0,25 = 0,11$
2. normalizzo: $0,11 = 01,1 \cdot 2^{-1}$
3. ricavo esponente: $-1 + 127 = 126_{10} = 01111110_2$
4. ricavo frazione: $1,1$
5. ricavo esponente: 1 (negativo)

1 01111110 10000000000000000000000

Doppia precisione

1. $-0,75 = 0,5 + 0,25 = 0,11$
2. normalizzo: $0,11 = 01,1 \cdot 2^{-1}$
3. ricavo esponente: $-1 + 1023 = 1022 = 01111111110$
4. ricavo frazione: $1,1$
5. ricavo esponente: 1 (negativo)

1 01111111110 1000000000000000000000000000000000000000000000000000

Da IEEE 754 a Decimale

$$\underbrace{1}_{\text{segno}}\underbrace{10000001}_{\text{esponente}}\underbrace{01000000000000000000000}_{\text{frazione}}$$

1\. segno esponente frazione

2\. 1 129 $0{,}01_2 = 0{,}25_{10}$

3\. formula: $(-1)^1 \cdot (1 + 0{,}25) \cdot 2^{129-127} = -1{,}25 \cdot 4 = -5{,}0$

oppure

3\. segno: $1 = -$, esponente: $129 - 127 = 2$ spost. frazione: $1\,01{,}000 = -5{,}0$

Operazioni in virgola mobile

N.b.: ho scritto "mantissa" ma dovrebbe essere "frazione"

- Somma

0\. Conversione
1. Scalo a destra mantissa del n° + piccolo fino a quando gli esp. non coincidono
2. Sommo mantisse
3. Normalizzo la somma (1 n° dopo la virgola)
4. Se ho overflow/underflow genero ecc. altrimenti continuo
5. Arrotondo la mantissa
6. Se non è normalizzato torno al punto 3, altrimenti ho finito

Esempio: $(0{,}5)_{10} + (-0{,}4375)_{10} = ?$

0\. Conversioni

- $(0{,}5)_{10} = 0{,}1_2 = 1{,}0 \cdot 2^{-1}$

$$\begin{array}{l|l}
0{,}4375 & 2 = 0{,}875 \\
0{,}875 & 2 = 1{,}75 \\
0{,}75 & 2 = 1{,}5 \\
0{,}5 & 2 = 1{,}0
\end{array}$$

- $(-0{,}4375)_{10} = -0{,}0111 = -1{,}11 \cdot 2^{-2}$

1. Scalo mantissa

$1,0 \cdot 2^{-1}$, $-1,11 \cdot 2^{-2} \rightarrow -0,111 \cdot 2^{-1}$

2. Somma mantisse

$1,0_2 = 1_{10}$, $0,111_2 = 0,5 + 0,25 + 0,125 = 0,875$

$$\begin{array}{r} 1,000 - \\ 0,875 = \\ \hline 0,125 \end{array}$$

$\hookrightarrow 0,001_2$

3. Normalizzo la somma

$0,001 \cdot 2^{-1} = 1,000 \cdot 2^{-4}$

4. Controllo overflow/underflow

$-127 \leq -4 \leq 126$ ✓ $-4 + 127 = 123$ ✓ polarizzato

5. Arrotondo mantissa

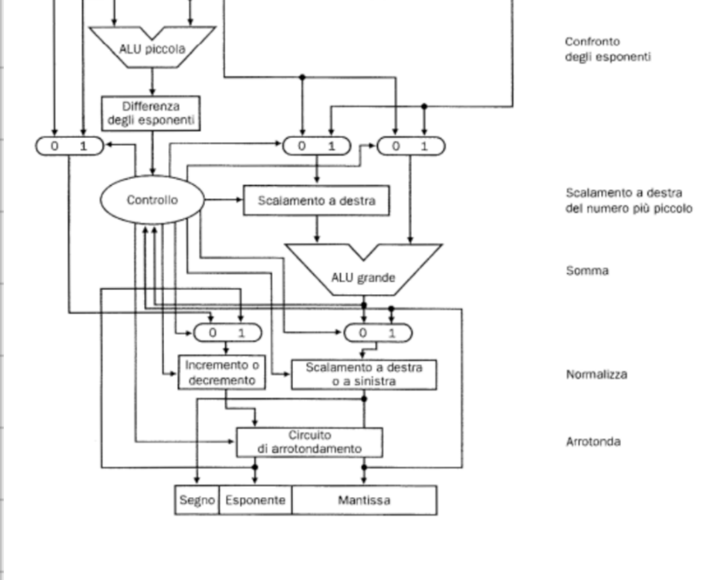
0,0072398

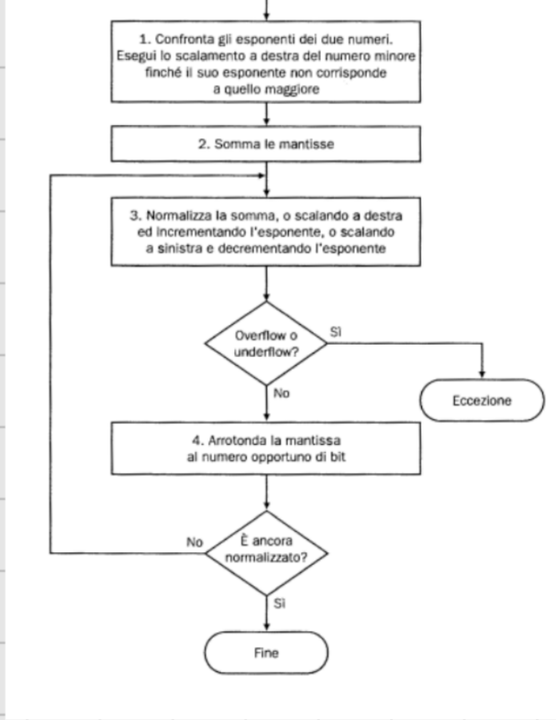
$1,000 \cdot 2^{-4}$ è già arrotondato!

6. Risultato

$1,000 \cdot 2^{-4} = 0,0001 = 0,0625_{10}$

Somma a livello hardware | Flowchart





Moltiplicazione

0. Conv. binaria
1. Calcolo esponente sommandoli
2. Controllo polarizzazione (somme esp. polarizzati - polarizzante)
3. Moltiplicazione mantisse
4. Normalizzazione risultato
5. Determina segno

Esempio: $0,5_{10} \cdot -0,4375_{10}$

0. Conversione

$0,5_{10} = 1,0 \cdot 2^{-1}$ $\qquad$ $-0,4375 = -1,111 \cdot 2^{-2}$

1. Somma esponenti

$-1 + (-2) = -3$ ✓

2. Controllo polarizzazione

$$-126 \le x \le 127 \xrightarrow{+127} 1 \le x+127 \le 254$$

$$(-1+127)+(-2+127)-127 = 126+125-127 = 124 \checkmark$$

3. Moltiplico mantisse

$$1{,}000 \cdot 1{,}110 = 1{,}110 \cdot 2^{-3}$$

4. Normalizzo

$1{,}110 \cdot 2^{-3}$ già normalizzato

5. Determino segno

$$(+)\cdot(-) = - \longrightarrow -1{,}110 \cdot 2^{-3}$$

Virgola mobile in Mips

In Mips posso effettuare:
- somme, sottrazioni, moltiplicazioni, divisioni
- confronti (equal, not equal, greater ecc..)
- branch

in virgola mobile a singola e doppia precisione

I registri sono $f1 - $f9. Possiamo fare lw e sw con lwc1 e swc1.

Esempio

```
lwc1    $f4, x($sp)       #carica num. v.m. 32 bit in F4
lwc1    $f6, y($sp)       #carica num. v.m. 32 bit in F6
add.s   $f2, $f4, $f6     # somma a 32 bit
swc1    $f2, z($sp)       # salvo la somma su z
```

`swc1 $f12, 2($sp) # salva la somma su 2`

Tabella Output Possibili

Single precision		Double precision		Object represented
Exponent	Fraction	Exponent	Fraction	
0	0	0	0	0
0	Nonzero	0	Nonzero	± denormalized number
1–254	Anything	1–2046	Anything	± floating-point number
255	0	2047	0	± infinity
255	Nonzero	2047	Nonzero	NaN (Not a Number)

Legenda:

- **Numeri denormalizzati**: numeri che riempono l'intervallo tra lo 0 e il più piccolo numero normalizzato. Ottenere questo numero genera gradual underflow, ovvero una perdita di precisione lentamente, piuttosto che improvvisa. Hanno il n° prima della virgola = 0

- **Floating-point number**: numeri "normali" Hanno il n° prima della virgola ≠ 0

- **Infinito**: divisione per 0

- **NaN**: numero non valido