

# Parte III

## Indice

---

- ◆ Rappresentazione dei valori frazionari
  - in virgola fissa
  - in virgola mobile
- ◆ Esercizi

# Rappresentazione dei valori frazionari

---

- ◆ I valori frazionari sono del tipo:

$xxxxxxx...xxxx, yyyyyy...yyyy$

dove le  $x$  indicano i bit dedicati per rappresentare la parte intera e le  $y$  i bit della parte frazionaria

- ◆ I numeri frazionari possono essere rappresentati in due modi:
  - in virgola fissa (*fixed point*)
  - in virgola mobile (*floating point*)

# Rappresentazione in virgola fissa

---

*parte intera*

*parte frazionaria*

- ◆ Il numero di bit della parte frazionaria e quello della parte intera sono costanti
- ◆ Per le operazioni aritmetiche valgono le regole adottate comunemente per i numeri decimali frazionari
- ◆ Si ricordi che bisogna *incolonnare* correttamente *i due operandi* (virgola sotto virgola)

# Rappresentazione in virgola fissa

---

◆ Nelle operazioni valgono, immutate, le considerazioni viste in precedenza:

- se in una somma si ha un riporto oltre la cifra più significativa si ha overflow
- se in una differenza si ha un prestito oltre la cifra più significativa si ha overflow

◆ Esempi Si esegua la somma delle seguenti coppie di valori:

$$10010,010 + 11001,01 +$$

$$\underline{00111,0001} = \underline{01111,11} =$$

$$11001,0101 \quad \boxed{1}01001,00$$

– Primo caso: OK  overflow

– Secondo caso: riporto oltre l'ultima cifra (overflow)

# Rappresentazione in virgola fissa

---

- ◆ Nel caso si vogliano rappresentare valori reali con segno, viene riservato il MSB proprio per esso

<i>segno</i>	<i>parte intera</i>	<i>parte frazionaria</i>
--------------	---------------------	--------------------------

- ◆ Per il calcolo dell'*operazione* di CA2:
  - si scambiano gli zeri e gli uno e si aggiunge un 1 **in corrispondenza del LSB**
  - si applica la regola pratica vista per i valori interi **a partire dal LSB**

# Rappresentazione in virgola fissa

---

## ◆ Esempio

Si esegua la seguente somma in MS e CA2:

$$-5,75_{10} + 1,25_{10}$$

rappresentando su 3 bit la parte intera (più 1 bit per il segno) e su 3 quella frazionaria

– In MS:

$$\begin{array}{r|l} 1101,110 & + \quad (-5,75) \\ 0001,010 & = \quad (+1,25) \\ \hline 1100,100 & \quad \quad (-4,5) \end{array}$$

Il risultato avrà il segno dell'operando con modulo maggiore (quindi sarà negativo) e il modulo verrà calcolato sottraendo il modulo minore dal maggiore

# Rappresentazione in virgola fissa

---

## ◆ Esempio (*segue*)

– In CA2:

Prima si calcola la rappresentazione in complemento a 2 di  $-5,75_{10}$  e poi si esegue la somma con la rappresentazione in CA2 di  $+1,25_{10}$

$$\overbrace{0101,110}_{+5,75}_{CA2} \xrightarrow{CA2} \overbrace{1010,010}_{-5,75}_{CA2}$$

– Eseguendo la somma si ha:

$$\begin{array}{r} 1010,010 + \quad (-5,75) \\ 0001,010 = \quad (+1,25) \\ \hline 1011,100 \quad \quad (-4,5) \end{array}$$

# Rappresentazione in virgola mobile

---

- ◆ In molte applicazioni nel campo scientifico è spesso richiesto di rappresentare *grandezze* estremamente *piccole* o grandezze estremamente *grandi*
- ◆ Il *numero di bit usati* all'interno di un sistema di elaborazione per rappresentare delle grandezze è necessariamente *limitato*
- ◆ Il problema che si pone è quello di riuscire a rappresentare sulla stessa macchina valori che possono essere di ordini di grandezza *completamente differenti* tra loro

# Rappresentazione in virgola mobile

---

- ◆ La notazione vista in precedenza non può soddisfare questi requisiti perché troppo “rigida”
- ◆ La notazione maggiormente usata nei calcolatori per rappresentare valori reali è nota con il nome di *rappresentazione in virgola mobile* (o *floating point*)
- ◆ In tale rappresentazione la posizione della virgola non è fissa, bensì varia al fine di avere sempre una rappresentazione del numero in notazione scientifica

# Rappresentazione in virgola mobile

---

- ◆ Nella notazione scientifica un valore è rappresentato da:
  - un'unica cifra a sinistra della virgola
  - una parte frazionaria
  - un esponente al quale si deve elevare la base del numero

Esempio  $546,768_{10} \rightarrow 5,46768_{10} \cdot 10^2$

Analogamente con i numeri binari:

$$1011,0110_2 \rightarrow 1,0110110_2 \cdot 2^3$$

*Nota*

Nell'esempio precedente l'esponente della base 2 è stato rappresentato per semplicità grafica in base 10; in realtà il calcolatore rappresenta in base 2 anche l'esponente

# Rappresentazione in virgola mobile

---

- ◆ In generale, un valore in virgola mobile si può esprimere secondo la seguente formula:

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

( $S$  = Segno,  $E$  = Esponente,  $M$  = Mantissa)

- ◆ In generale, possiamo scrivere la notazione scientifica per la base 2 nella seguente forma:

$$1, x_1 x_2 \dots x_n \cdot 2^{y_1 y_2 \dots y_m}$$

dove le  $x$  rappresentano la parte frazionaria e le  $y$  l'esponente a cui elevare la base 2

# Rappresentazione in virgola mobile

---

- ◆ Il numero di bit con cui viene rappresentato un valore in virgola mobile all'interno del calcolatore è comunque **limitato**
- ◆ Il numero di bit da assegnare alla parte frazionaria (detta mantissa) e all'esponente è regolamentato da uno standard per l'aritmetica in virgola mobile (*IEEE P754*)

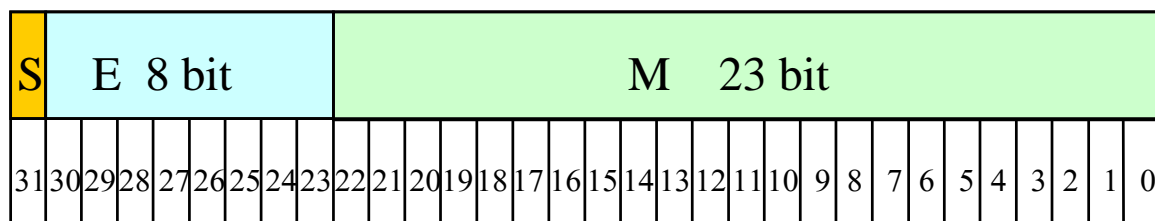
# Rappresentazione in virgola mobile

---

◆ Lo standard *IEEE* P754 prevede che, quando per rappresentare un valore reale sono usati 32 bit (singola precisione), questi siano ripartiti secondo il seguente schema:

- 1 bit per il segno  $S$
- 8 bit per l'esponente  $E$
- 23 bit per la mantissa  $M$

Graficamente:



# Rappresentazione in virgola mobile

---

- ◆ Questa “distribuzione” dei bit tra esponente e mantissa consente di rappresentare valori in un ampio intervallo
- ◆ L’esponente deve poter assumere valori negativi; rappresentazioni possibili: MS, CA2, *polarizzata*
- ◆ Nella *rappresentazione polarizzata* una costante viene sottratta al valore contenuto nell’esponente per ricavare il valore “vero” dell’esponente stesso
- ◆ La rappresentazione polarizzata è più veloce nelle operazioni di confronto

# Rappresentazione in virgola mobile

---

- ◆ Nello standard *IEEE* P754 su 32 bit la quantità di polarizzazione è pari a  $127_{10}$  e la rappres. polarizzata è detta *codice eccesso 127*
- ◆ Per garantire l'unicità di rappresentazione, la mantissa viene portata ad avere un valore compreso nell'intervallo  $[1,2)$  e l'esponente viene adattato opportunamente, quindi si hanno sempre numeri della forma  $1.xx...x$
- ◆ L'uno a sinistra della virgola esiste sempre (per definizione): si può recuperare un bit per la mantissa (si guadagna in precisione) se non lo si rappresenta (*hidden bit*)

# Rappresentazione in virgola mobile

---

- ◆ Allora la formula per ricavare il valore di un numero rappresentato in virgola mobile (*IEEE P754*) diventa:

$$(-1)^s \cdot (1 + M') \cdot 2^{(E-127)}$$

dove  $1 + M' = M$

- ◆ In virgola mobile *IEEE P754* la rappresentazione dello 0 “viola” parzialmente la regole espresse in precedenza e corrisponde ad avere tutti i bit a zero

# Rappresentazione in virgola mobile

---

## ◆ Esempio

Rappresentare in formato floating point *IEEE* P754 il numero  $13,25_{10}$

– il numero è positivo  $\Rightarrow$  segno = 0

–  $13,25_{10} = 1101,01_2$

spostando la virgola di 3 posizioni a sinistra il numero si normalizza in:

$1,10101_2 \cdot 2^3$

– l'esponente in codice eccesso 127 è:

$3 + 127 = 130_{10} = 10000010_2$

– risultato:

0 10000010 101010000000000000000000

# Rappresentazione in virgola mobile

---

## ◆ Esempio

Quale valore reale in virgola mobile *IEEE* P754 è rappresentato dalla seguente sequenza di bit?

1 01100000 01000000000000000000000000000000

– segno: MSB = 1  $\Rightarrow$  numero negativo

– esponente:  $01100000_2 = 96_{10}$

essendo in codice eccesso 127

bisogna sottrargli 127:

$$96 - 127 = -31$$

– parte frazionaria della mantissa:

$$01000000000000000000000000000000_2 = 0,25_{10}$$

– Applicando la formula si ottiene:

$$(-1)^1 \cdot (1 + 0,25) \cdot 2^{-31} =$$

$$-1,25 \cdot 2^{-31} =$$

$$-5,82076 \cdot 10^{-10}$$

# Esercizi

---

◆ Esercizio 1 Eseguire le seguenti operazioni in virgola fissa:

$$\begin{array}{r} \text{a)} \quad 01101,001 + \\ 10000,110 = \end{array}$$

$$\begin{array}{r} \text{c)} \quad 10110,010 - \\ 01011,111 = \end{array}$$

$$\begin{array}{r} \text{b)} \quad 001,0010 + \\ 011,0011 = \end{array}$$

$$\begin{array}{r} \text{d)} \quad 1111 - \\ 0101,1 = \end{array}$$

$$\begin{array}{r} \text{a)} \quad 01101,001 + \\ \underline{10000,110} = \\ 11101,111 \end{array}$$

$$\begin{array}{r} \text{c)} \quad 10110,010 - \\ \underline{01011,111} = \\ 01010,011 \end{array}$$

$$\begin{array}{r} \text{b)} \quad 001,0010 + \\ \underline{011,0011} = \\ 100,0101 \end{array}$$

$$\begin{array}{r} \text{d)} \quad 1111 - \\ \underline{0101,1} = \\ 1001,1 \end{array}$$

# Esercizi

---

◆ Esercizio 2 Determinare a quali valori decimali corrispondono le seguenti rappresentazioni floating point *IEEE P754*?

a) 010000000001000000000000000000000000

b) 101111111110000000000000000000000000

a) Il bit più significativo è 0, quindi, il numero rappresentato è positivo.

L'esponente è  $10000000_2 = 128_{10}$  a cui bisogna sottrarre 127. La mantissa è  $00100000000000000000000000_2 = 0,125_{10}$  quindi otteniamo:

$$(-1)^0 \cdot (1 + 0,125) \cdot 2^{128-127} = 2,25_{10}$$

# Esercizi

---

b) Il bit più significativo è 1, quindi, il numero rappresentato è negativo.

L'esponente è  $01111111_2 = 127_{10}$  a cui bisogna sottrarre 127. La mantissa è  $11000000000000000000000000_2 = 0,75_{10}$  quindi otteniamo:

$$(-1)^1 \cdot (1 + 0,75) \cdot 2^{127-127} = -1,75_{10}$$

◆ **Esercizio 3** Quante e quali cifre sono necessarie per convertire in virgola fissa senza segno con un errore assoluto inferiore a  $1/50$  i seguenti valori?

a) 1,76

b) 0,12

c) 0,5

d) 2,137

e) 10,15

f) 0,125

# Esercizi

---

- Per convertire con una precisione di almeno  $1/50$  sono necessarie 6 cifre dopo la virgola, infatti:

$$\frac{1}{50} > \frac{1}{2^n} \Rightarrow n = \lceil \log_2 50 \rceil = 6$$

a) Si devono convertire separatamente la parte intera e quella frazionaria.

La parte intera è semplicemente 1 mentre per quella frazionaria si ha il calcolo a lato:

6 cifre per la parte frazionaria,  
1 cifra per la parte intera:

$\Rightarrow$  sono necessarie 7 cifre

$$1,76_{10} = 1,110000_2$$

$$\begin{array}{r} 0,76 * \\ \underline{2=} \\ \boxed{1},52 * \\ \underline{2=} \\ \boxed{1},04 * \\ \underline{2=} \\ \boxed{0},08 * \\ \underline{2=} \\ \boxed{0},16 * \\ \underline{2=} \\ \boxed{0},32 * \\ \underline{2=} \\ \boxed{0},64 \end{array}$$

# Esercizi

---

b) Si devono convertire separatamente la parte intera e quella frazionaria. La parte intera è semplicemente 0 mentre quella frazionaria si ottiene da:

$$\begin{array}{r} 0,12 * \\ \underline{\quad 2 =} \\ \overline{0},24 * \\ \underline{\quad 2 =} \\ \overline{0},48 * \\ \underline{\quad 2 =} \\ \overline{0},96 * \\ \underline{\quad 2 =} \\ \overline{1},92 * \\ \underline{\quad 2 =} \\ \overline{1},84 * \\ \underline{\quad 2 =} \\ \overline{1},68 \end{array}$$

sono quindi necessarie 7 cifre

$$0,12_{10} = 0,000111_2$$

# Esercizi

---

c) In questo caso la conversione della parte intera e di quella frazionaria è immediata e sono necessarie solo due cifre per rappresentare  $0,5_{10} = 0,1_2$

d) Per rappresentare la parte intera sono necessarie 2 cifre binarie in quanto  $2_{10} = 10_2$  mentre per la parte frazionaria si ha:

$$\begin{array}{r} 0,137 * \\ \hline 2 = \\ \underline{0,274 *} \\ 2 = \\ \underline{0,548 *} \\ 2 = \\ \underline{1,096 *} \\ 2 = \\ \underline{0,192 *} \\ 2 = \\ \underline{0,384 *} \\ 2 = \\ \underline{0,768} \end{array}$$

sono quindi necessarie 8 cifre per rappresentare  $2,137_{10} = 10,001000_2$

# Esercizi

---

e) Per rappresentare la parte intera sono necessarie 4 cifre binarie in quanto

$10_{10} = 1010_2$  mentre per la parte frazionaria si ha il calcolo a lato:

$$\begin{array}{r} 0,15 * \\ \underline{2=} \\ 0,30 * \\ \underline{2=} \\ 0,60 * \\ \underline{2=} \\ 1,20 * \\ \underline{2=} \\ 0,40 * \\ \underline{2=} \\ 0,80 * \\ \underline{2=} \\ 1,60 \end{array}$$

sono quindi necessarie 10 cifre

$$10,15_{10} = 1010,001001_2$$

f) In questo caso la conversione della parte intera e di quella frazionaria è immediata e sono necessarie solo quattro cifre per rappresentare

$$0,125_{10} = 0,001_2$$