

## Soluzione degli esercizi in virgola mobile

Si presentano, nel seguito, le soluzioni commentate degli esercizi relativi ai numeri in virgola mobile svolti durante le lezioni. Si ringrazia lo studente Michele ? (mmelectronics@sitoverde.com) per aver messo nero su bianco quanto da me raccontato a lezione.

## Soluzione esercizio 1

Per l'esponente, essendo  $n=8$ , in eccesso 128 l'intervallo da considerare è:

$$[-2^{n-1}, 2^{n-1}-1].$$

Quindi per l'esponente l'intervallo è:  $[-128, 127]$

Il numerale positivo più grande rappresentabile sarà quindi:

$$0, \underbrace{11111111}_{\text{sgn}} \underbrace{11111111}_{\text{esponente}} \underbrace{11111111}_{\text{mantissa}}$$

Poiché la mantissa è normalizzata tra 1 e 2, la stringa 11111111 può essere approssimata a 2 per cui il numero positivo più grande rappresentabile è (esponente \* mantissa)  $2^{127} * 2 = 2^{128}$ .

Il numerale positivo più piccolo rappresentabile sarà:

$$0 \text{ 00000000 } 00000000$$

In questo caso essendo la mantissa pari a 1,00000000, il numero positivo più piccolo è  $2^{-128} * 1$  (esponente \* mantissa).

Quindi gli estremi dell'intervallo rappresentabile sono  $2^{128}$  e  $2^{-128}$ .

Per simmetria dell'intervallo dei numeri negativi, l'insieme dei numeri rappresentabili è:  $(-2^{128}, -2^{-128}] \cup [2^{-128}, 2^{128})$ . Gli intervalli sono aperti in quanto la mantissa è leggermente più piccola di 2.

Per quanto riguarda l'ordine di grandezza decimale, sapendo che  $2^{10} \approx 10^3$  è possibile calcolare gli ordini di grandezza mediante la seguente proporzione:

$$10 : 3 = 128 : x$$

$$x = (128 * 3) / 10 \approx 38$$

Gli ordini di grandezza in decimale sono quindi  $10^{-38}$  e  $10^{38}$ .

Il numero da rappresentare è:

$$\begin{array}{ccccccc} & F & & F & & 5 & & A & & B & & 9 \\ & 1 \end{array}$$

dove il 1° bit è il segno.

Lasciando invariati i bit fino al primo 1 e complementando poi i successivi si ottiene il modulo del numero in questione:

$$00000001010010101000111$$

Si può notare come si potrebbe arrivare allo stesso risultato ignorando tutti i bit più significativi a 1 tranne l'ultimo, cioè considerando solo il complemento di:

$$\overline{11111111010101010111001}$$

Volendo normalizzare il numero complementato 00000001010010101000111, è necessario spostare la virgola a sinistra di 15 posti.

Quindi dividendo e moltiplicando per  $2^{15}$  si ottiene:

$$1,010010101000111 * 2^{15}$$

L'esponente in eccesso 128 è quindi  $15+128 = 143$ .

Il numero in notazione 16 bit è:

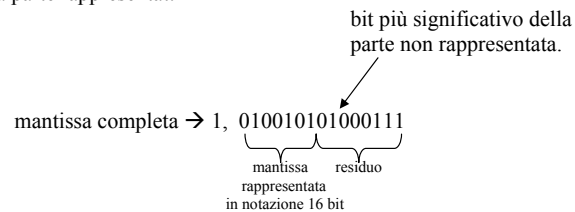
$$1 \text{ 10001111 } 0100101$$

Notiamo che la parte residua non rappresentata del numero è:

$$0,000000001000111 * 2^{15}$$

e corrisponde quindi all'errore assoluto ( $\epsilon_a = n - n'$ ), dato dalla differenza fra il numero da rappresentare ed il numero rappresentato.

Per quanto riguarda l'errore relativo invece, è possibile calcolarlo valutando la distanza fra il bit più significativo della parte non rappresentata ed il bit più significativo della parte rappresentata:



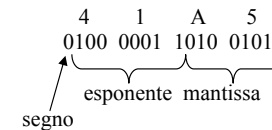
Come si può vedere, la distanza del bit più significativo della parte non rappresentata dista 9 posizioni dal bit a sinistra della virgola per cui l'errore relativo è:

$$\epsilon_r = 2^{-9}$$

L'errore relativo massimo, essendo la mantissa normalizzata, è costante per l'insieme dei numeri rappresentati ed è pari alla potenza associata al bit meno significativo della mantissa, cioè  $2^{-7}$ .

## Soluzione esercizio 2

Il numero m è:



Quindi l'esponente in eccesso 128 è 10000011.

Complementando il bit più significativo si ha 00000011 che equivale a  $(3)_{10}$ .

La mantissa è 1,0100101 per cui il numero rappresentato è:

1010,0101 approssimato per difetto da 1010, la cui rappresentazione in CP2 con 16 bit è:

0000000000001010

### Soluzione esercizio 3

Il numero n è:

F F 5 A B 9  
~~1111~~ ~~1111~~ 0101 1010 1011 1001

Trascurando i bit piu' significativi posti a 1 (tranne l'ultimo) e lasciando invariati i bit fino al primo 1 e complementando poi i successivi si ottiene il modulo del numero in questione:

01010010101000111

Spostando la virgola a sinistra di 15 posti (dividendo e moltiplicando per  $2^{15}$ ) risulta:  
 $1,010010101000111 * 2^{15}$

Quindi l'esponente da rappresentare è  $(15)_{10}$ .

15 ricade nell'intervallo  $[-16, +15]$  per cui si ricava che il numero  $e$  di bit minimo per la rappresentazione dell'esponente in eccesso  $2^{e-1}$  è dato da:  $e = 5$ .

La mantissa sarà rappresentata da  $15 - e = 10$  bit.

Il numero rappresentato è quindi:

1 11111 0100101010

La parte non rappresentata è  $0,000000000000111 * 2^{15}$  e coincide con l'errore assoluto, mentre l'errore relativo è  $\epsilon_r = 2^{-13}$ .

### Soluzione esercizio 4

I numeri da rappresentare sono:

F A 5 3  
 1111 1010 0101 0011  
 segno      esponente      mantissa

L'esponente in eccesso 64 è 1111010 e in complemento a due equivale a complementare il bit più significativo, quindi 0111010 che in decimale vale 58.

In definitiva  $m = 1,01010011 * 2^{58}$

F 9 F 2  
 1111 1001 1111 0010  
 segno      esponente      mantissa

L'esponente in eccesso 64 è 1111001 e in complemento a due equivale a complementare il bit più significativo, quindi 0111001 che in decimale vale 57.

In definitiva  $n = 1,11110010 * 2^{57}$

Per eseguire la somma è necessario trasformare uno dei due numeri al fine di ottenere lo stesso esponente.

Trasformando il più piccolo (n) si ha:

$n = 0,111110010 * 2^{58}$

A questo punto è possibile sommare le mantisse:

1,01010011+  
 0,11111001  
 10,01001100

Si ha dunque:  $h = m + n = 10,01001100 * 2^{58} = 1,001001100 * 2^{59}$

L'esponente del numero da rappresentare in eccesso 64 è

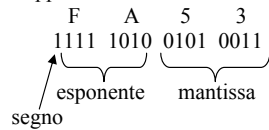
$(59+64)_{10} = (123)_{10} = (1111011)_2$

Il numero nella rappresentazione considerata è quindi

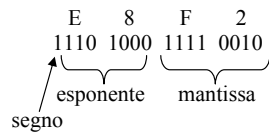
1111 1011 0010 0110  
 segno      esponente      mantissa

che in esadecimale diventa FB26.

Consideriamo ora un'altra coppia di numeri:



L'esponente in eccesso 64 è 1111010 che in complemento a due equivale a complementare il bit più significativo, quindi 0111010 che in decimale vale a 58.  
 In definitiva  $m = 1,01010011 * 2^{58}$



L'esponente in eccesso 64 è 0101000 e in complemento a due equivale a complementare il bit più significativo, quindi 0111001 che in decimale vale a 40.  
 In definitiva  $n = 1,11110010 * 2^{40}$

Come prima, per poter eseguire la somma è necessario trasformare uno dei due numeri al fine di ottenere lo stesso esponente.

Trasformando il più piccolo (n) si ha (spostando la virgola a sinistra di 18 posti):  
 $n = 0,00000000000000000111110010 * 2^{58}$

A questo punto è possibile sommare le mantisse:

$$\begin{array}{r} 1,01010011+ \\ 0,00000000 \\ \hline 1,01010011 \end{array}$$

Si ha dunque:  $h = m + n = 1,01010011 * 2^{58}$  che è proprio uguale al numero più grande.

Quello che si vuole mettere in evidenza in questo esercizio è che se gli esponenti dei due numeri rappresentati m ed n differiscono per un valore maggiore del numero di bit utilizzati per rappresentare l'esponente, il risultato della somma è pari al numero più grande rappresentato (in questo caso m).