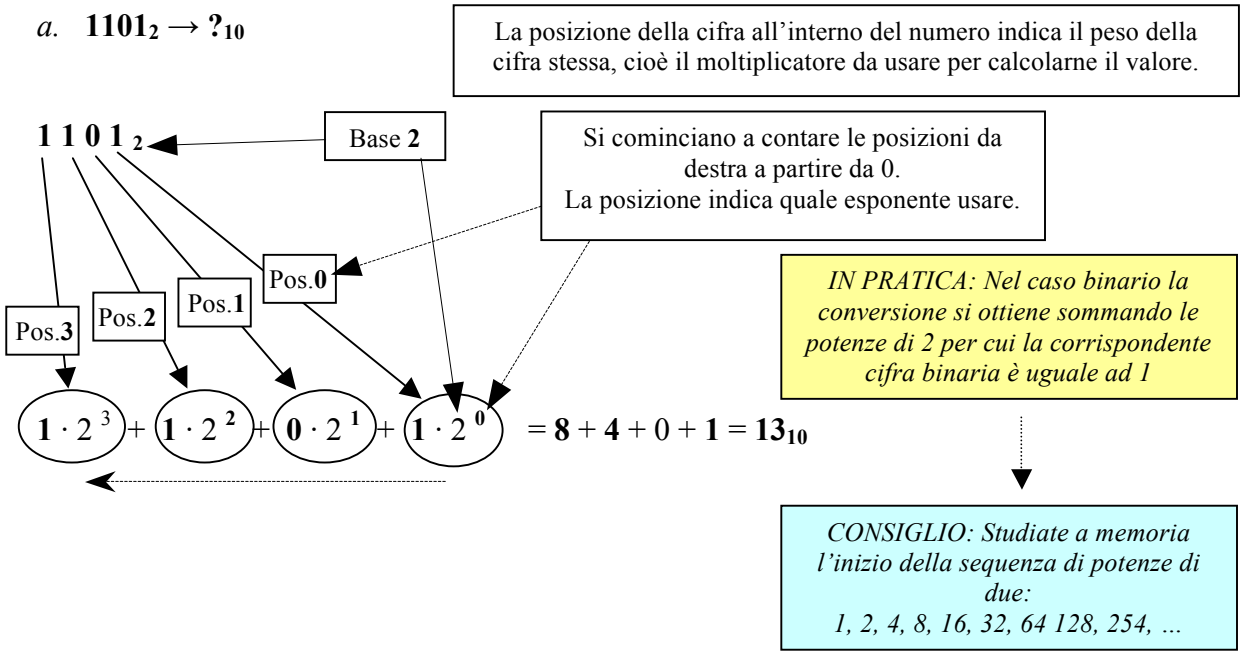
Esercitazione 1 del 07/10/2011

1. Conversione binario → decimale

a. **$1101_2 \rightarrow ?_{10}$**



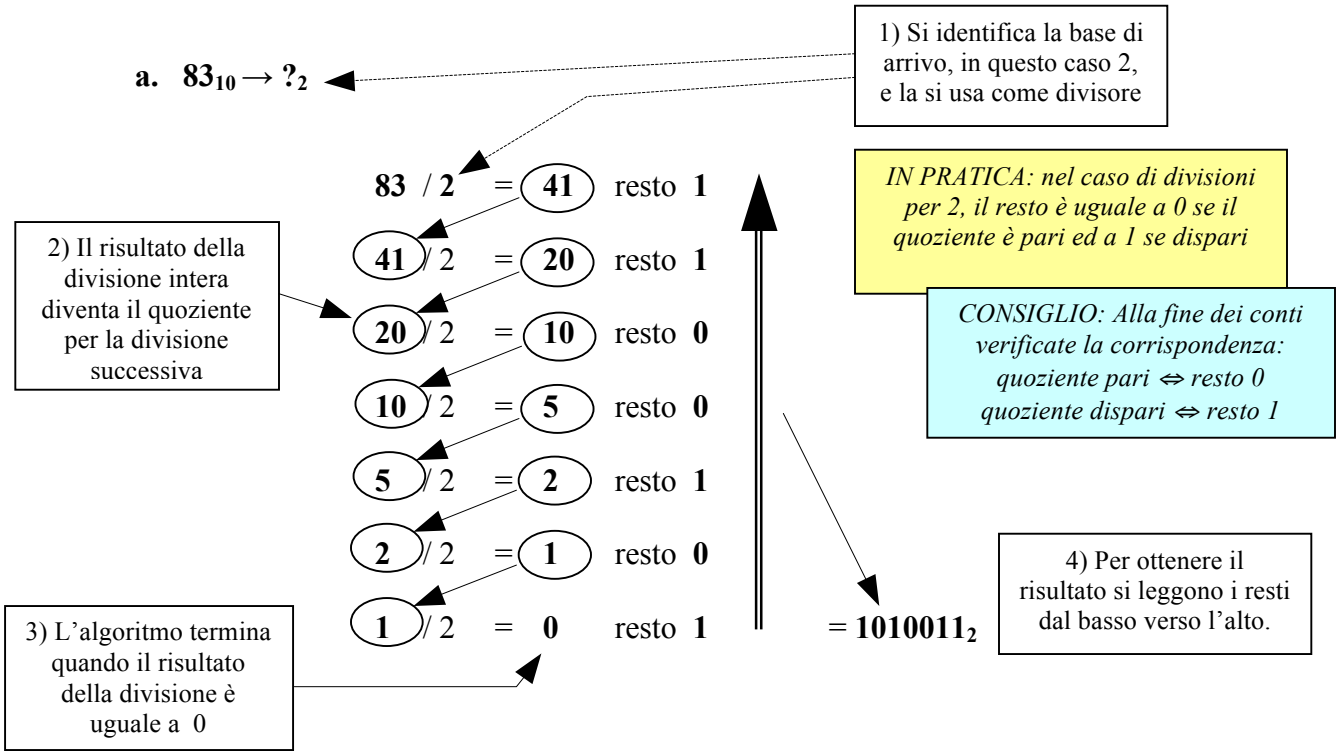
b. **10010101_2** $= \mathbf{1} \cdot 2^7 + \mathbf{0} \cdot 2^6 + \mathbf{0} \cdot 2^5 + \mathbf{1} \cdot 2^4 + \mathbf{0} \cdot 2^3 + \mathbf{1} \cdot 2^2 + \mathbf{0} \cdot 2^1 + \mathbf{1} \cdot 2^0$
$= 128 + 16 + 4 + 1 = \mathbf{149_{10}}$

c. **1001001_2** $= \mathbf{1} \cdot 2^6 + \mathbf{0} \cdot 2^5 + \mathbf{0} \cdot 2^4 + \mathbf{1} \cdot 2^3 + \mathbf{0} \cdot 2^2 + \mathbf{0} \cdot 2^1 + \mathbf{1} \cdot 2^0$
$= 64 + 8 + 1 = \mathbf{73_{10}}$

d. **101111101_2** $= \mathbf{1} \cdot 2^8 + \mathbf{0} \cdot 2^7 + \mathbf{1} \cdot 2^6 + \mathbf{1} \cdot 2^5 + \mathbf{1} \cdot 2^4 + \mathbf{1} \cdot 2^3 + \mathbf{1} \cdot 2^2 + \mathbf{0} \cdot 2^1 + \mathbf{1} \cdot 2^0$
$= 256 + 64 + 32 + 16 + 8 + 4 + 1 = \mathbf{381_{10}}$

Massimo Marchi – marchi@dsi.unimi.it

2. Conversione decimale → binario



b. $417_{10} \rightarrow ?_2$

417	**/ 2 =**	**208**	resto	**1**
208	**/ 2 =**	**104**	resto	**0**
104	**/ 2 =**	**52**	resto	**0**
52	**/ 2 =**	**26**	resto	**0**
26	**/ 2 =**	**13**	resto	**0**
13	**/ 2 =**	**6**	resto	**1**
6	**/ 2 =**	**3**	resto	**0**
3	**/ 2 =**	**1**	resto	**1**
1	**/ 2 =**	**0**	resto	**1**

$\mathbf{417_{10} = 110100001_2}$

c. $3652_{10} \rightarrow ?_2$

3652	**/ 2 =**	**1826**	resto	**0**
1826	**/ 2 =**	**913**	resto	**0**
913	**/ 2 =**	**456**	resto	**1**
456	**/ 2 =**	**228**	resto	**0**
228	**/ 2 =**	**114**	resto	**0**
114	**/ 2 =**	**57**	resto	**0**
57	**/ 2 =**	**28**	resto	**1**
28	**/ 2 =**	**14**	resto	**0**
14	**/ 2 =**	**7**	resto	**0**
7	**/ 2 =**	**3**	resto	**1**
3	**/ 2 =**	**1**	resto	**1**
1	**/ 2 =**	**0**	resto	**1**

$\mathbf{3652_{10} = 111001000100_2}$

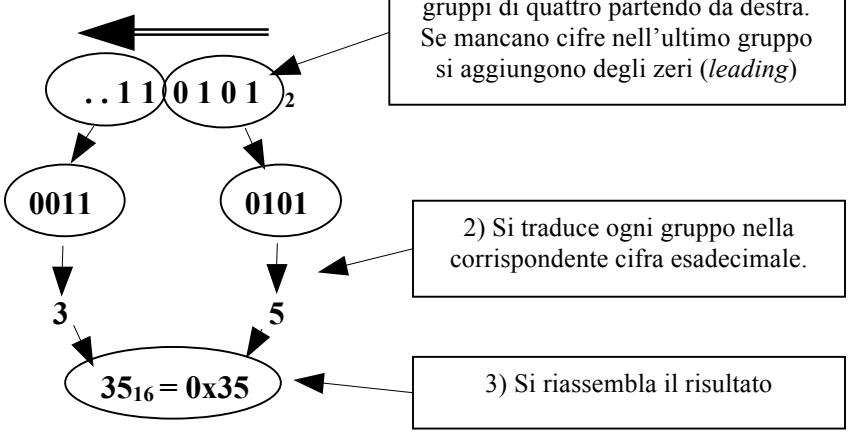
d. $5453_{10} \rightarrow ?_2$

5453	**/ 2 =**	**2726**	resto	**1**
2726	**/ 2 =**	**1363**	resto	**0**
1363	**/ 2 =**	**681**	resto	**1**
681	**/ 2 =**	**340**	resto	**1**
340	**/ 2 =**	**170**	resto	**0**
170	**/ 2 =**	**85**	resto	**0**
85	**/ 2 =**	**42**	resto	**1**
42	**/ 2 =**	**21**	resto	**0**
21	**/ 2 =**	**10**	resto	**1**
10	**/ 2 =**	**5**	resto	**0**
5	**/ 2 =**	**2**	resto	**1**
2	**/ 2 =**	**1**	resto	**0**
1	**/ 2 =**	**0**	resto	**1**

$\mathbf{5453_{10} = 1010101001101_2}$

3. **Conversione binario → esadecimale**

a. $\mathbf{110101_2 \rightarrow ?_{16}}$



CONSIGLIO:Poiché le possibili combinazioni (16) sono poche, conviene imparare a memoria una tabella lookup per la conversione

Tabella lookup

0000 ↔ **0**	1000 ↔ **8**
0001 ↔ **1**	1001 ↔ **9**
0010 ↔ **2**	1010 ↔ **A**
0011 ↔ **3**	1011 ↔ **B**
0100 ↔ **4**	1100 ↔ **C**
0101 ↔ **5**	1101 ↔ **D**
0110 ↔ **6**	1110 ↔ **E**
0111 ↔ **7**	1111 ↔ **F**

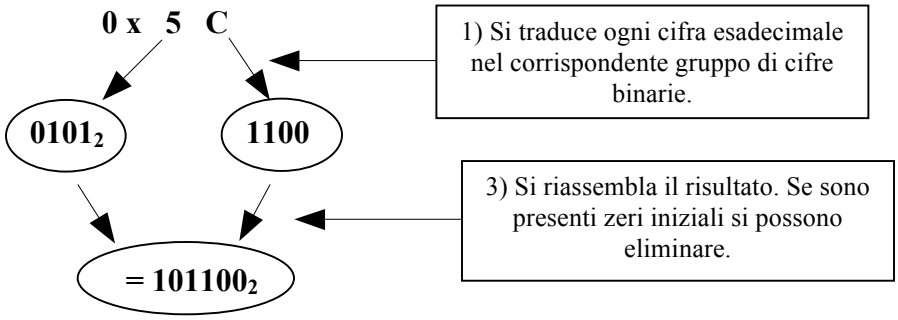
b. $\mathbf{1100011_2} = \mathit{0}110_2 \mid 0011_2 = 0x6 \mid 0x3 = \mathbf{0x63}$

c. $\mathbf{100000110001_2} = 1000_2 \mid 0011_2 \mid 0001_2 = 0x8 \mid 0x3 \mid 0x1 = \mathbf{0x831}$

d. $\mathbf{1001000110100_2} = \mathit{000}1_2 \mid 0010_2 \mid 0011_2 \mid 0100_2 = 0x1 \mid 0x2 \mid 0x3 \mid 0x4 = \mathbf{0x1234}$

4. **Conversione esadecimale → binario**

a. $\mathbf{0x5C \rightarrow ?_2}$



b. **0x4A1** = 0x4 | 0xA | 0x1 = ~~0~~$100_2 \mid 1010_2 \mid 0001_2$ = $\mathbf{10010100001_2}$

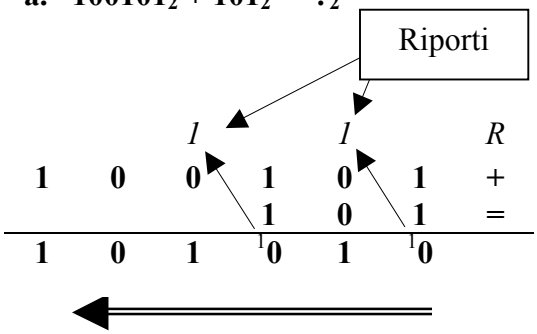
c. **0xEDC** = 0xE | 0xD | 0xC = $1110_2 \mid 1101_2 \mid 1100_2$ = $\mathbf{111011011100_2}$

d. **0x3010** = 0x3 | 0x0 | 0x1 | 0x0 = ~~00~~$11_2 \mid 0000_2 \mid 0001_2 \mid 0000_2$ = $\mathbf{11000000010000_2}$

Massimo Marchi – marchi@dsi.unimi.it

5. Somme binarie

a. $100101_2 + 101_2 = ?_2$



La somma binaria si esegue esattamente come quella decimale classica. Si allineano a destra i numeri da sommare quindi si procede sommando ogni coppia di bit e l'eventuale riporto, da destra verso sinistra. Gli eventuali riporti si sommano sulla coppia di bit successiva a sinista

In base 2, occorre ricordarsi che:
1 + **1** = **0** riporto **1**
1 + **1** + **1** = **1** riporto **1**

$100101_2 + 101_2 = 101010_2$

CONSIGLIO: verificate sempre i risultati traducendo i numeri binari in decimale e rifacendo i calcoli in questa base

b. $1111011_2 + 10101000_2 = 100100011_2$

1	1	1		1	1	1			R
		1	1	0	1	0	1	1	+
	1	0	1	0	1	1	1	0	=
1	1 0	1 0	1 0	1	1 1	0	0	1	

c. $110_2 + 101011111_2 = 101100101_2$

		1	1	1		1	1		R
					1	0	1	1	+
1	0	0	1	1	1	0	1	1	=
1	0	1	1 0	1 0	1 0	1	1 1	1 0	

d. $110111100_2 + 101100001_2 = 1100011101_2$

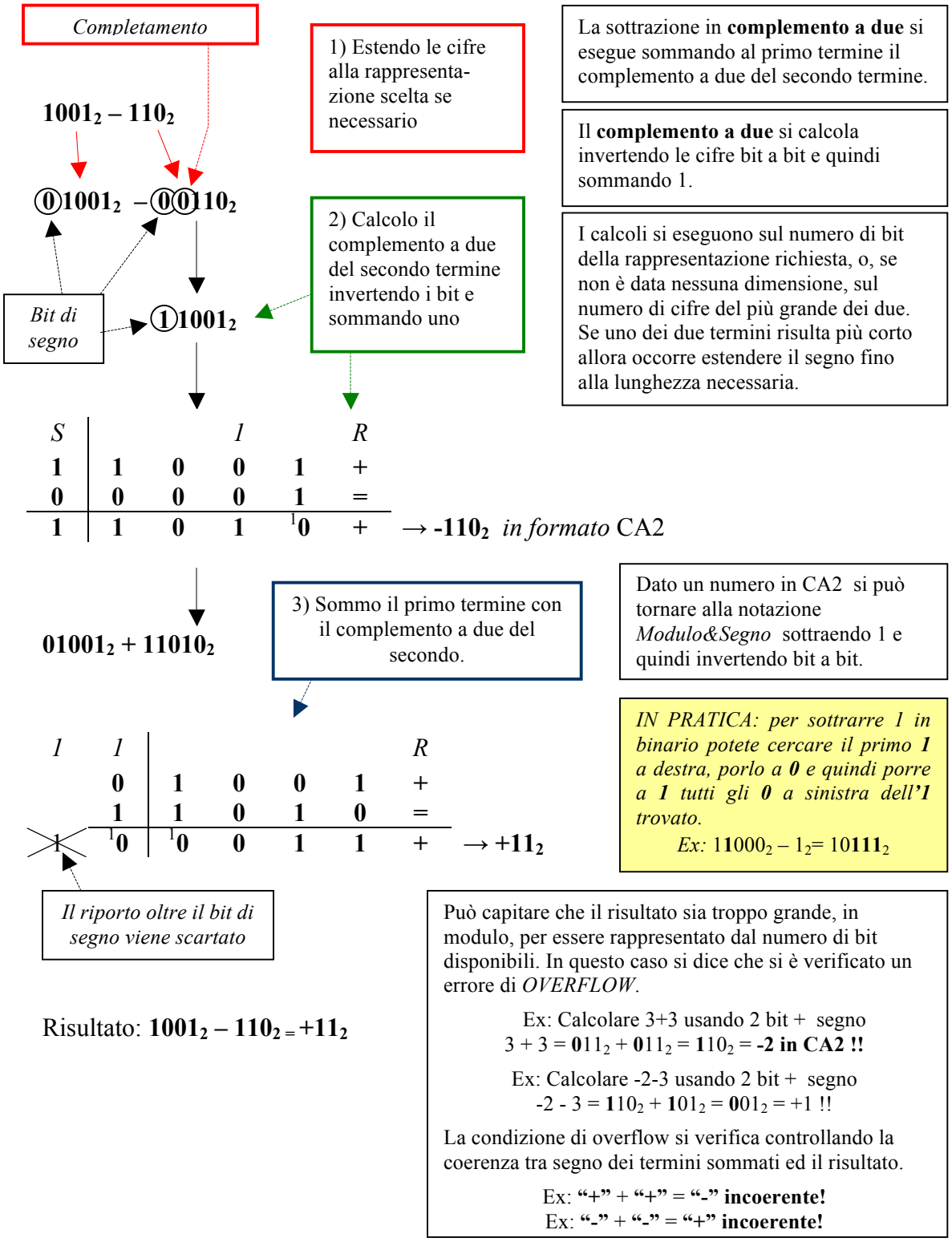
1	1	1				1				R
	1	1	1	0	1	0	1	0	0	+
	1	1	1	1	0	0	1	0	1	=
1	1 1	1 1	1 0	1	1	1	1 0	0	1	

Massimo Marchi – marchi@dsi.unimi.it

6. Sottrazioni binarie (in complemento a due)

a. $1001_2 - 110_2 = ?_2$

COMPLEMENTO A DUE



Risultato: $1001_2 - 110_2 = +11_2$

Massimo Marchi – marchi@dsi.unimi.it

b. $\mathbf{110_2 - 11011_2 = ?_2}$

Uso quattro cifre più il bit di segno:

$\mathbf{110_2 - 11011_2 = 0\ 00110_2 - 0\ 11011_2 = 000110_2 + (\ 100100_2 + 1\ )}$

Calcolo il CA2 di $\mathbf{-1010_2}$:

S						*R*	
1	**0**	**0**	**1**	**0**	**0**	+	
0	**0**	**0**	**0**	**0**	**1**	=	
1	**0**	**0**	**1**	**0**	**1**		→ $\mathbf{-1011_2}$ in CA2

Eseguo la somma tra il primo termine e il CA2 del secondo:

S		*1*				*R*	
0	**0**	**0**	**1**	**1**	**0**	+	
1	**0**	**0**	**1**	**0**	**1**	=	
1	**0**	**1**	^{1}**0**	**1**	**1**		→ $\mathbf{-10101_2}$ in CA2
							(101011→010100+1=10101)

Risultato: $\mathbf{110_2 - 11011_2 = -10101_2}$

c. $\mathbf{10111_2 - 111_2 = ?_2}$

Uso cinque cifre più il bit di segno:

$\mathbf{11001_2 - 111_2 = 0\ 10111_2 - 0\ 00111_2 = 011001_2 + (\ 111000_2 + 1\ )}$

Calcolo il CA2 di $\mathbf{-1001_2}$:

S						*R*	
1	**1**	**1**	**0**	**0**	**0**	+	
0	**0**	**0**	**0**	**0**	**1**	=	
1	**1**	**1**	**0**	**0**	**1**		→ $\mathbf{-111_2}$ in CA2

Eseguo la somma tra il primo termine e il CA2 del secondo:

1	*1*	*1*			*1*		*R*	
	0	**1**	**1**	**0**	**0**	**1**	+	
	1	**1**	**1**	**0**	**0**	**1**	=	
~~1~~	^{1}**0**	^{1}**1**	^{1}**0**	**0**	**1**	^{1}**0**		→ $\mathbf{+10010_2}$

Risultato: $\mathbf{11001_2 - 111_2 = +10010_2}$

d. $\mathbf{1101_2 - 110011_2 = ?_2}$ (*Eseguire i calcoli a 8 bit)*

Uso sette cifre più il bit di segno:

$\mathbf{1101_2 - 110011_2 = 0\ 0001101_2 - 0\ 0110011_2 = 00001101_2 + (\ 11001100_2 + 1_2\ )}$

S								*R*	
1	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**+**	
0	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**=**	
1	**1**	**0**	**0**	**1**	**1**	**0**	**1**		→ **-110011_2** in CA2

Eseguo la somma tra il primo termine e il CA2 del secondo:

S			*1*	*1*		*1*		*R*	
0	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**+**	
1	**1**	**0**	**0**	**1**	**1**	**0**	**1**	**=**	
1	**1**	**0**	**1**	^{1}**1**	^{1}**0**	**1**	^{1}**0**		→ **-100110_2** in CA2
						(11011010→00100101+1=100110)			

Risultato: $\mathbf{1101_2 - 110011_2 = -100110_2}$

Massimo Marchi – marchi@dsi.unimi.it

7. Conversione in floating point secondo lo standard IEEE 754

a. $\mathbf{-20{,}75_{10} = <}s,e,m\mathbf{>}$?

Per convertire in floating point occorre:
- calcolare il segno
- convertire la parte intera in binario (ex: con il metodo delle divisioni per 2 successive)
- convertire la parte frazionaria in binario (ex: con il metodo delle moltiplicazioni per 2 successive)
- unire i due risultati e normalizzare.
- calcolare la mantissa secondo la precisione voluta (occorre ricordarsi di scartare il primo 1 della normalizzazione)
- calcolare l'esponente della normalizzazione polarizzato e convertirlo in binario secondo la precisione voluta

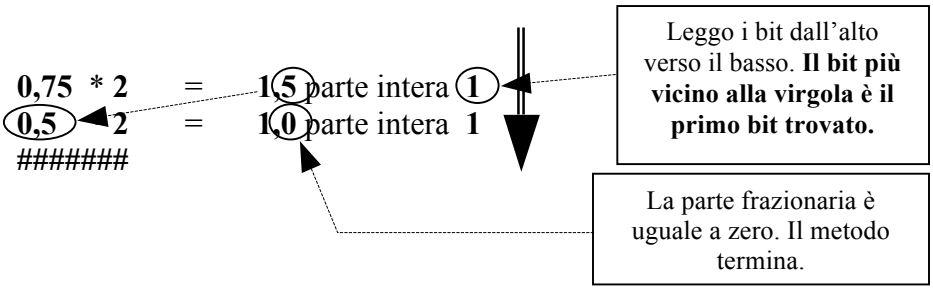
Numero negativo: $\boldsymbol{s = 1}$

Converto 40_{10} in base 2: $\mathbf{20_{10} = 10100_2}$

20	**/ 2**	**=**	**10**	resto **0**
10	**/ 2**	**=**	**5**	resto **0**
5	**/ 2**	**=**	**2**	resto **1**
2	**/ 2**	**=**	**1**	resto **0**
1	**/ 2**	**=**	**0**	resto **1**

Converto $0{,}5_{10}$ in base 2: $\mathbf{0{,}75_{10} = 0{,}11_2}$

La conversione della parte frazionaria **per moltiplicazioni 2** successive si esegue nel seguente modo:
- si moltiplica per 2 la parte frazionaria. La parte intera del risultato costituisce il primo bit della parte frazionaria espressa in binario.
- Si ripete il passo precedente sulla parte frazionaria del risultato. La parte intera del risultato costituirà adesso il secondo bit della parte frazionaria espressa in binario.
- Si ripete il procedimento ricavando i successivi bit fino a che la parte frazionaria risulta uguale a zero (tutti i bit successivi saranno a zero) oppure si è raggiunta la precisione voluta (es. si sono ricavati già i 23 bit necessari per una mantissa in precisione singola).



Unisco i risultati: $\mathbf{20{,}75_{10} = 10100{,}11_2}$

Normalizzo: $\mathbf{10100{,}11_2 = 1{,}010011_2 \cdot (10_2)^4}$

Mantissa a 23 bits: $\mathbf{1{,}010011_2 \rightarrow m = 0100110000\ 0000000000\ 000}$

Polarizzo l'esponente: $\mathbf{4_{10} + 127_{10} = 131_{10} = 128_{10} + 2_{10} + 1_{10} = 10000011_2}$

Esponente a 8 bits: $\mathbf{10000011_2 \rightarrow e = 10000011}$

$\mathbf{-20{,}75_{10} = <1,\ 10000011,\ 01001100000000000000000>}$

Massimo Marchi – marchi@dsi.unimi.it

b. $-7{,}625_{10}$ = <s,m,e>?

Numero negativo: $s = 1$
Converto 7_{10} in base 2: $7_{10} = 111_2$
Converto $\cdot{,}625_{10}$ in base 2: $0{,}625_{10} = 0{,}101_2$

0,625 * 2 = 1,25 parte intera **1**
0,25 * 2 = 0,5 parte intera **0**
0,5 * 2 = 1,0 parte intera **1**
#########

Unisco i risultati: $7{,}625_{10} = 111{,}101_2$
Normalizzo: $111{,}101_2 = 1{,}11101_2 \cdot (10_2)^2$ *(shift a sinistra di 2 posizioni: exp= 2)*
Mantissa a 23 bit: $1{,}11101_2 \rightarrow m = 1110100000\ 0000000000\ 000$
Polarizzo l'esponente: $2_{10} + 127_{10} = 129_{10} = 10000001_2$

129 / 2 = 64 resto **1**
64 / 2 = 32 resto **0**
32 / 2 = 16 resto **0**
16 / 2 = 8 resto **0**
8 / 2 = 4 resto **0**
4 / 2 = 2 resto **0**
2 / 2 = 1 resto **0**
1 / 2 = 0 resto **1**

Esponente a 8 bits: $1111101_2 \rightarrow e = 01111101$

$-7{,}625_{10}$ = <1, 10000001, 11101000000000000000000>

c. $0{,}4375_{10}$ = <s,m,e>?

Numero positivo: $s = 0$
Converto 0_{10} in base 2: $0_{10} = 0_2$

Converto $\cdot{,}4375_{10}$ in base 2: $0{,}4375_{10} = 0{,}0111_2$

0,4375	*** 2**	=	**0,875**	parte intera **0**
0,875	*** 2**	=	**1,75**	parte intera **1**
0,75	*** 2**	=	**1,5**	parte intera **1**
0,5	*** 2**	=	**1,0**	parte intera **1**

#########

Unisco i risultati: $0{,}0111_{10} = 0{,}0111_2$
Normalizzo: $0{,}0111_2 = 1{,}11_2 \cdot (10_2)^{-2}$ *(shift a destra di 2 posizioni: exp=-2)*
Mantissa a 23 bit: $1{,}11_2 \rightarrow m = 1100000000\ 0000000000\ 000$
Polarizzo l'esponente: $-1_{10} + 127_{10} = 125_{10} = 1111101_2$

125	**/ 2**	=	**62**	resto **1**
62	**/ 2**	=	**31**	resto **0**
31	**/ 2**	=	**15**	resto **1**
15	**/ 2**	=	**7**	resto **1**
7	**/ 2**	=	**3**	resto **1**
3	**/ 2**	=	**1**	resto **1**
1	**/ 2**	=	**0**	resto **1**

Esponente a 8 bits: $1111110_2 \rightarrow e = 01111110$

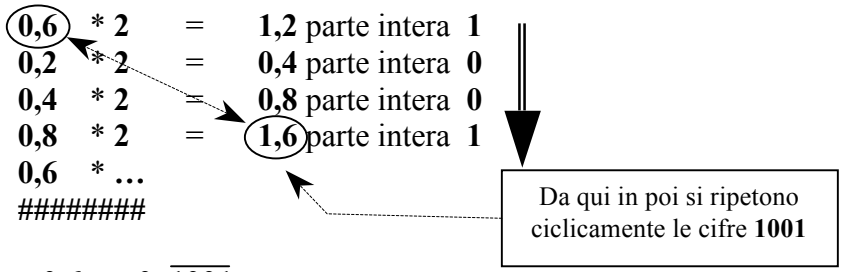
$0{,}4375_{10}$ <0, 01111110, 11000000000000000000>

d. $-0{,}6_{10}$ = <s,m,e>?

Numero negativo: ***s* = 1**

Converto 0_{10} in base 2: $\mathbf{0_{10} = 0_2}$

Converto $,3_{10}$ in base 2: $\mathbf{0{,}6_{10} = 0, \overline{1001}_2}$



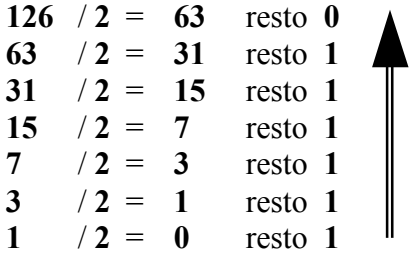
Unisco i risultati: $\mathbf{0{,}6_{10} = 0, \overline{1001}_2}$

Normalizzo: $\mathbf{0, \overline{1001}_2 = 0, 1001\ \overline{1001}_2 = 1{,}001\ \overline{1001}_2 \cdot (10_2)^{-1}}$

Mantissa a 23 bit: ~~1~~**, 001 1001 1001 1001 1001 1001 ~~1001~~**

→ *m* = 00110011001100110011001

Polarizzo l'esponente: $\mathbf{-1_{10} + 127_{10} = 126_{10} = 1111110_2}$



Tronco la rappresentazione periodica della mantissa alla lunghezza fissa di 23 bit

Esponente a 8 bits: $\mathbf{1111110_2 \rightarrow e = 01111110}$

$-0{,}6_{10}$ = <1, 01111110, 00110011001100110011001 >

Alcune configurazioni con significato speciale (singola precisione):

0	<0,00000000,00000000000000000000000>
+/-∞	<[*segno*],11111111,00000000000000000000000>
NaN	<[*segno*],11111111, [*mantissa ≠ 0*]>
N.ro denormalizzato	<[*segno*],00000000, [*mantissa denormalizzata≠0*]> *(vedi di seguito)*

Alcuni casi notevoli (singola precisione):

1 = $1{,}0 \cdot (10_2)^0$	<0,01111111,00000000000000000000000>	*(E=0+127)*
MaxFloat = $1{,}11..1 \times 2^{+127}$	<0,11111110,11111111111111111111111>	*(E=+127+127)*
MinFloat = $1{,}0 \times 2^{-126}$	<0,00000001,00000000000000000000000>	*(E=-126+127)*
MinFloatDeN = $0{,}0..01 \times 2^{-126}$	<0,00000000,00000000000000000000001>	

Massimo Marchi – marchi@dsi.unimi.it

Problema dell'approssimazione di numeri piccoli: il formato IEEE 754 denormalizzato.

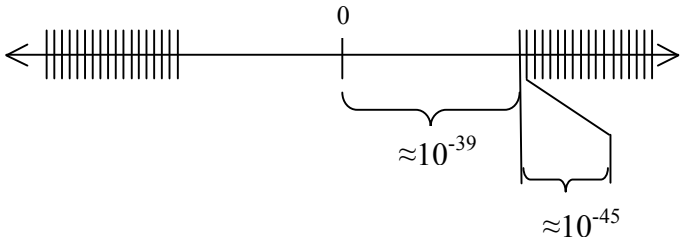
Il più piccolo numero positivo rappresentabile in formato IEEE 754 standard a precisione singola è (*si ricordi che l'esponente uguale a **0** è usato per indicare valori speciali come lo **zero***):

$<0,00000001,00000000000000000000000> = 1{,}0_2 \cdot 2^{-126} \approx 1{,}17 * 10^{-38}$

Il più piccolo numero positivo successivo:

$$<0,00000001,00000000000000000000001> = 1{,}00000000000000000000001_2 \cdot (2_{10})^{-126}$$
$$= 2^{-126} + 2^{-149}$$
$$\approx 1{,}17 \cdot 10^{-38} + 1{,}40 \cdot 10^{-45}$$

Ciò significa che usando il formato IEEE 754 standard, in prossimità dello zero avremo un cambiamento nella linearità della quantizzazione: da **0** al più piccolo successivo ci sarà un salto dell'ordine di 10^{-38} mentre per passi successivi i salti saranno dell'ordine di 10^{-45}.



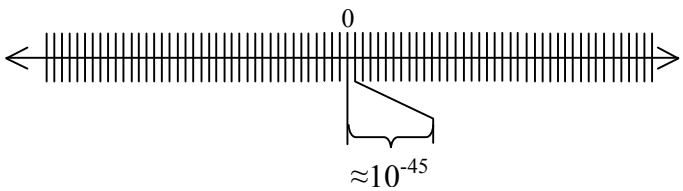
Il più piccolo numero positivo rappresentabile in formato IEEE 754 denormalizzato a precisione singola è ($0.f \times 2^{-126}$):

$$<0,00000000,00000000000000000000001> = 0{,}00000000000000000000001_2 \cdot 2^{-126}$$
$$= 2^{-149} \approx 1{,}40 * 10^{-45}$$

Il più piccolo numero positivo successivo:

$$<0,00000000,00000000000000000000010> = 0{,}00000000000000000000010_2 \cdot 2^{-126}$$
$$= 2 * 2^{-149}$$
$$\approx 2{,}80 \cdot 10^{-45}$$

Usando il formato IEEE 754 denormalizzato, in prossimità dello zero la quantizzazione risulta lineare: da **0** al più piccolo successivo ci sarà un salto dell'ordine di 10^{-45} così come per passi successivi. In aggiunta il salto tra 0 ed il primo positivo successivo risulta minore, e quindi migliore in termini di precisione ottenibile nei calcoli, rispetto al formato IEEE 754 standard.



Lo svantaggio maggiore del formato denormalizzato rispetto al formato standard è che richiede per l'esecuzione dei calcoli aritmetici di algoritmi più complicati.

Per saperne di più: ***http://stevehollasch.com/cgindex/coding/ieeefloat.html***

Massimo Marchi – marchi@dsi.unimi.it