

Modulo: Approfondimenti sui Sistemi Aritmetici
di un computer: tipo intero [P2_02]

Unità didattica: Rappresentazione dei numeri e
cambiamento di base [1-AT]

Titolo: Richiami sui sistemi di numerazione

Argomenti trattati:

- ✓ Rappresentazione posizionale dei numeri
- ✓ Sistemi di numerazione binario, ottale, decimale, esadecimale
- ✓ Algoritmi di conversione di un numero da una base di numerazione ad un'altra

Prerequisiti richiesti: insiemi numerici della matematica (\mathbb{N} , \mathbb{Q} , \mathbb{R})

Rappresentazione posizionale

Un numero si rappresenta come una **combinazione lineare*** di potenze successive della **base** del sistema di numerazione

$$1936.27_{10} = 1 \cdot 10^3 + 9 \cdot 10^2 + 3 \cdot 10^1 + 6 \cdot 10^0 + 2 \cdot 10^{-1} + 7 \cdot 10^{-2}$$

base 10

$$1011.01_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2}$$

base 2

$$1A3C.F7_{16} = 1 \cdot 16^3 + 10 \cdot 16^2 + 3 \cdot 16^1 + 12 \cdot 16^0 + 15 \cdot 16^{-1} + 7 \cdot 16^{-2}$$

base 16

unità

* Una combinazione lineare di $\{b_1, b_2, \dots, b_n\}$ con coefficienti $\{\alpha_1, \alpha_2, \dots, \alpha_n\}$ è $\alpha_1 b_1 + \alpha_2 b_2 + \dots + \alpha_n b_n$

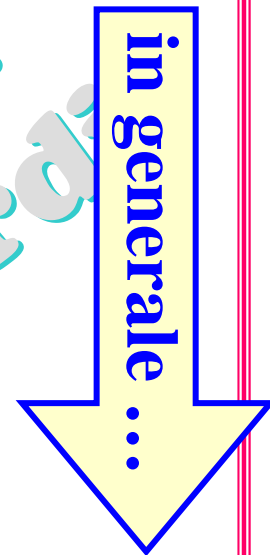
Alfabeto numerico

cifre	
0	9
1	8
2	7
3	6
4	5

Ogni riga contiene le
due cifre complementari

cifre	
0	1

cifre	
0	F
1	E
2	D
3	C
4	B
5	A
6	9
7	8



cifre	
0	
1	
...	
$\beta-1$	

Tipo Intero

(prof. M. Rizzardi)

base 10

base 2

base 16

base β

Cambiamento di base

da base β a base 10

$\beta=2$

$$\begin{aligned} 1011.01_2 &= 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} = \\ &= 8 + 2 + 1 + 1/4 = 11.25_{10} \end{aligned}$$

$\beta=16$


$$\begin{aligned} 1A3C.F7_{16} &= 1 \cdot 16^3 + 10 \cdot 16^2 + 3 \cdot 16^1 + 12 \cdot 16^0 + 15 \cdot 16^{-1} + 7 \cdot 16^{-2} \\ &= 4096 + 2560 + 48 + 12 + 15/16 + 7/256 \\ &\approx 6716.96_{10} \end{aligned}$$

da base 10 a base β ($\beta=2$)

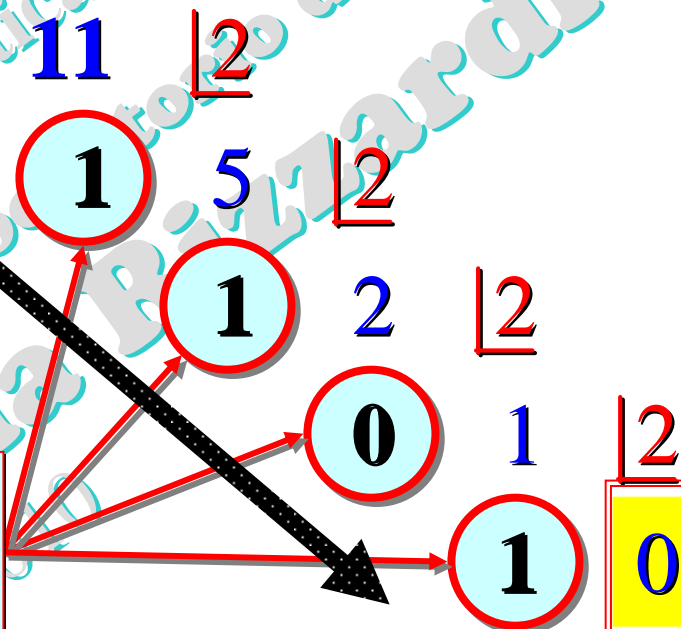
$$11.27_{10} = 1011.0100\dots_2$$

parte intera

(Algoritmo delle divisioni successive)

$$11_{10} = 1011_2$$


**resti della
divisione**



```
void char_bit_divisioni(unsigned char n, unsigned char bit[8])
{
    char k=0, j;
    do
    {
        bit[k++] = n%2;
        n = n/2;
    } while (n>0);
    for (j=k; j<8; j++)
        bit[j]=0;
}
```

numero intero positivo = 11

Gli 8 bit di 11 sono:

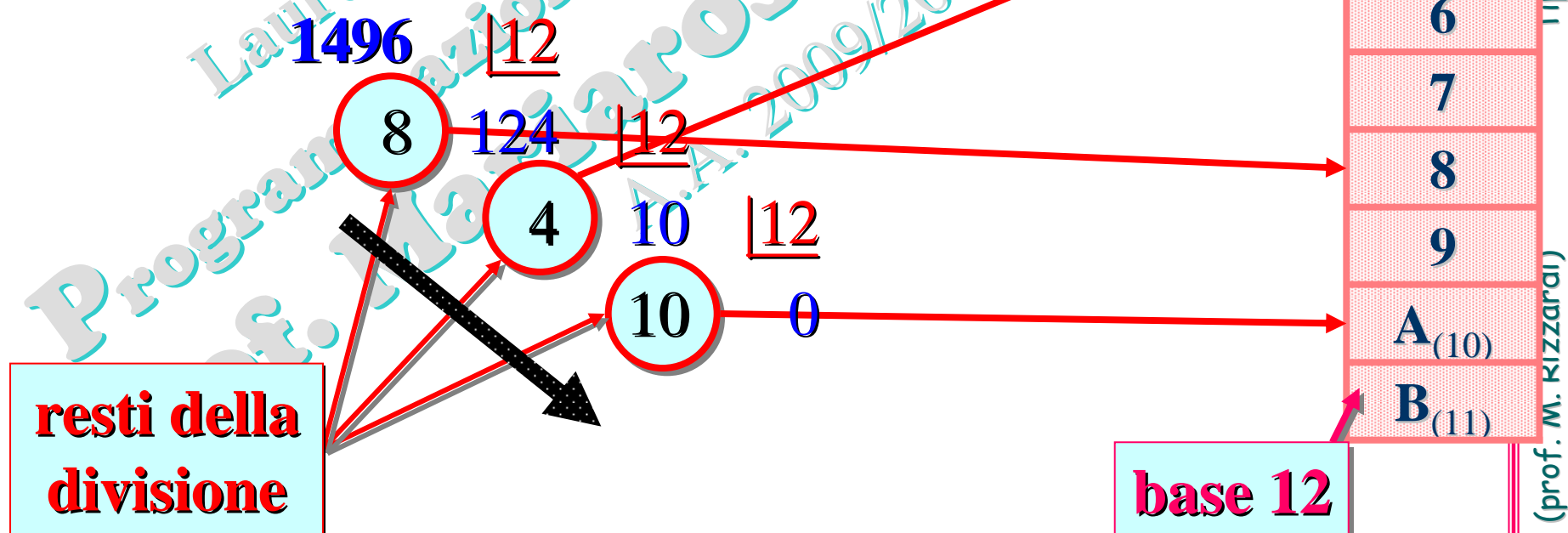
0000 1011

Algoritmo generale di cambiamento di base: esempio

da base 10 a base 12

parte intera (Alg. divisioni successive)

$1496_{10} = A48_{12}$

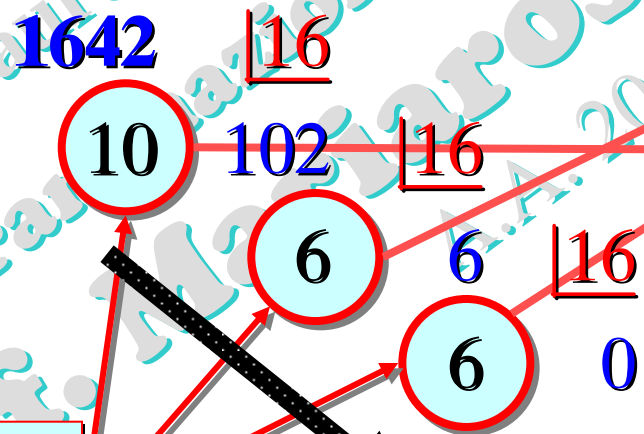


Algoritmo generale di cambiamento di base: esempio

da base 10 a base 16

parte intera (*Alg. divisioni successive*)

$$1642_{10} = 66A_{16}$$



resti della
divisione

base 16

cifre
0
1
2
3
4
5
6
7
8
9
A ₍₁₀₎
B ₍₁₁₎
C ₍₁₂₎
D ₍₁₃₎
E ₍₁₄₎
F ₍₁₅₎

tipo intero

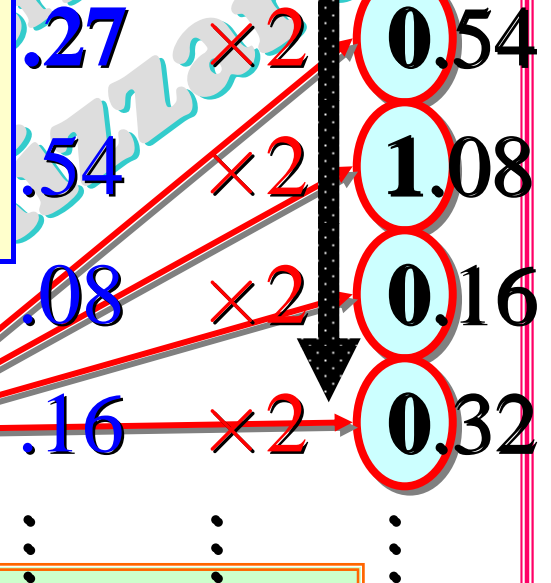
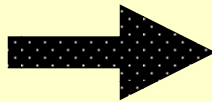
(prof. M. Rizzarai)

da base 10 a base β ($\beta=2$)

$$11.27_{10} = 1011.0100\dots_2$$

parte frazionaria (Alg. multipl. successive)

$$0.27_{10} = 0.0100\dots_2$$



parte intera

```
void fraz_bit_mult(float x, unsigned char n_bit, unsigned char bit[23])
```

```
{ char k;  
  for (k=0; k<n_bit; k++)  
  { x = x * 2; bit[k] = (char) x;  
    x = x - bit[k];  
  }  
  for (j=n_bit; k<23; k++)  
    bit[k] = 0;  
}
```

per la parte intera si può usare `floor()`
in `math.h`

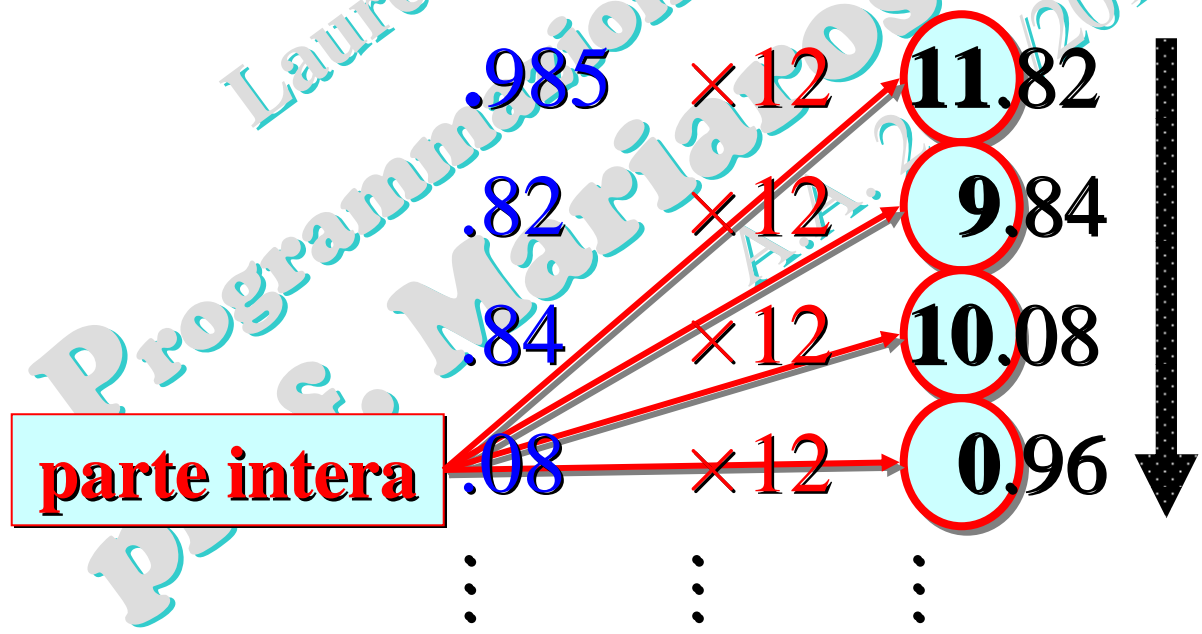
numero decimale positivo = 0.27

I 6 bit frazionari di 0.27 sono:
0. 0100 01

Algoritmo generale di cambiamento di base: esempio

da base 10 a base 12

parte frazionaria (*A. moltiplicazioni successive*)
 $0.985_{10} = 0.B9A0..._{12}$



cifre
0
1
2
3
4
5
6
7
8
9
A ₍₁₀₎
B ₍₁₁₎

base 12

Gli algoritmi delle **divisioni successive** e delle **moltiplicazioni successive** sono generali!

Unendo l'**algoritmo delle divisioni successive** (per la parte intera di un numero razionale positivo) e l'**algoritmo delle moltiplicazioni successive** (per la sua parte frazionaria) si può generare la rappresentazione di un qualsiasi numero razionale rispetto ad una **qualunque base di numerazione**.

Particolarmente semplice risulta la conversione tra la base 2 ed una base potenza di 2

Esempio: da base 2 a base 16

$$16 = 2^4$$

si suddividono i bit
in gruppi di 4

Si aggiungono zeri non significativi fino ad ottenere un numero di cifre, per la parte intera e quella frazionaria, multiplo di 4.

$$1011.01_2 = \boxed{1011}.\boxed{0100}_2 = \boxed{B}.\boxed{4}_{16}$$

$$101011.011_2 = \boxed{0010}\boxed{1011}.\boxed{0110}_2 = \boxed{2}\boxed{B}.\boxed{6}_{16}$$

Si sostituisce ai gruppi di 4 bit la corrispondente cifra esadecimale.

cifre

0 0000

1 0001

2 0010

3 0011

4 0100

5 0101

6 0110

7 0111

8 1000

9 1001

A 1010

B 1011

C 1100

D 1101

E 1110

F 1111

Esempio: da base 2 a base 8

$$8=2^3$$

si suddividono i bit
in gruppi di 3

Si aggiungono zeri non significativi fino ad ottenere un numero di cifre, per la parte intera e quella frazionaria, multiplo di 3.

cifre

0=000

1=001

2=010

3=011

4=100

5=101

6=110

7=111

$$1011.01_2 = \underline{001} \underline{011} . \underline{010}_2 = 1 \ 3 . 2_8$$

$$101111.011_2 = \underline{101} \underline{111} . \underline{011}_2 = 5 \ 7 . 3_8$$

Si sostituisce ai gruppi di 3 bit la corrispondente cifra ottale.

da base 16 a base 2

cifre

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

$$16 = 2^4$$

Ad ogni cifra esadecimale si
sostituisce il corrispondente
gruppo di 4 bit.

$$\text{B7F.C4}_{16} = 1011 \ 0111 \ 1111 . 1100 \ 0100_2$$

$$1\text{A8.CD}_{16} = \cancel{0001} \ 1010 \ 1000 . 1100 \ 1101_2$$

Si eliminano
gli zeri non
significativi.

da base 8 a base 2

$$8=2^3$$

Ad ogni cifra ottale si
sostituisce il corrispondente
gruppo di 3 bit.

cifre

0=000

1=001

2=010

3=011

4=100

5=101

6=110

7=111

6 7 4 . 2 4₈ = 110 111 100 . 010 100₂

1 5 6 . 3 4₈ = 001 101 110 . 011 100₂

Si eliminano
gli zeri non
significativi.

Esercizi

1 Scrivere due *function C* di conversione di un intero positivo (**int**) da base 10 a base 2 mediante l'*algoritmo delle divisioni successive* realizzato rispettivamente:

1. usando gli operatori di quoziente e resto della divisione intera;
2. usando gli operatori bitwise.

2 Scrivere una *function C* di conversione di un intero positivo da base 2 a base 10 mediante l'*algoritmo delle divisioni successive* che generi un array di caratteri contenenti le cifre decimali.

3 Ripetere l'esercizio precedente nel caso che l'input sia un array di caratteri contenenti i bit del numero. [liv. 2]