

Entità e rappresentazioni

- ◆ E' importante distinguere il concetto di entità da quello di rappresentazione
- ◆ Una *rappresentazione* è un modo per **descrivere** un' *entità*, ma non è l'entità stessa
- ◆ Nell'ambito dei sistemi numerici non bisogna confondere quella che è l'entità *numero* o *valore* con la sua *rappresentazione*
- ◆ Dato il valore “*sedici*”, la sua rappresentazione nel sistema decimale è 16_{10} mentre nel sistema binario è 10000_2 ; 16_{10} e 10000_2 sono due **rappresentazioni differenti** della **stessa entità**

Sistemi numerici

- ◆ Il sistema numerico che siamo abituati ad utilizzare è un sistema:
 - *decimale*: usa 10 cifre
 - *posizionale*: ad ogni “posizione” è associato un “peso”, infatti esistono:
 - » *la posizione delle unità*,
 - » *la posizione delle decine*,
 - » *la posizione delle centinaia*,
 - » etc.
- ◆ Un esempio di *sistema numerico non posizionale* è quello romano in cui non esistono le posizioni delle unità, decine, centinaia, etc.

Esempio Nel numero $IV = 4$

“I” non è una decina, “V” non è un’unità (altrimenti si leggerebbe 15)

Sistemi numerici

- ◆ Nei sistemi numerici posizionali un valore N può essere rappresentato nei seguenti modi:

$$N = d_{n-1}d_{n-2} \dots d_1d_0, d_{-1} \dots d_{-m}$$

$$N = d_{n-1} \cdot r^{n-1} + \dots + d_0 \cdot r^0 + d_{-1} \cdot r^{-1} + \dots + d_{-m} \cdot r^{-m}$$

$$N = \sum_{i=-m}^{n-1} d_i \cdot r^i$$

- ◆ Dove:

- d rappresenta la singola cifra (*digit*)
- r è la radice o base del sistema
- n è il numero di cifre della parte intera
- m è il numero di cifre della parte frazionaria

Sistemi numerici

◆ Esempi:

- base decimale

$$123,45 = 1 \cdot 10^2 + 2 \cdot 10^1 + 3 \cdot 10^0 + 4 \cdot 10^{-1} + 5 \cdot 10^{-2}$$

- base binaria

$$101,01 = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2}$$

◆ Proprietà principali di un sistema numerico posizionale:

- ogni numero intero può essere rappresentato (rango illimitato)
- a ogni numero intero corrisponde un solo insieme ordinato di cifre (rappresentazione unica)

Sistema decimale

- ◆ Nel sistema numerico decimale:
 - la base $r = 10$
 - le cifre $d = 0, 1, \dots, 9$

- ◆ Un valore N si rappresenta nel S.N. decimale come:

$$N = d_{n-1} \cdot 10^{n-1} + \dots + d_0 \cdot 10^0 + \\ + d_{-1} \cdot 10^{-1} + \dots + d_{-m} \cdot 10^{-m}$$

- ◆ Tutte le volte che si farà riferimento ad un valore senza specificarne la base, lo si considererà in base 10
- ◆ In caso contrario la base verrà specificata come pedice della cifra di peso più basso: es. 1001011_2

Sistema binario

◆ Nel sistema numerico binario:

- la base $r = 2$
- le cifre $d = 0, 1$

◆ Un valore N si rappresenta nel S.N. binario come:

$$N = d_{n-1} \cdot 2^{n-1} + \dots + d_0 \cdot 2^0 + \\ + d_{-1} \cdot 2^{-1} + \dots + d_{-m} \cdot 2^{-m}$$

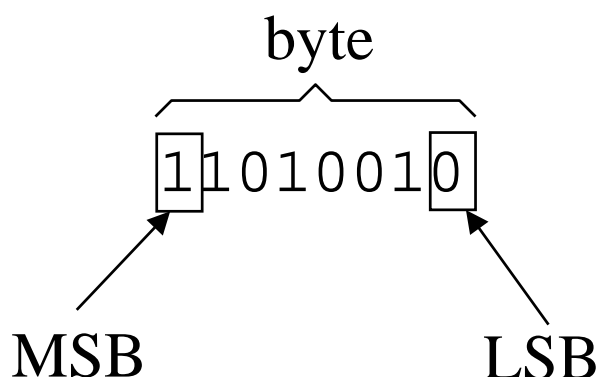
Esempio:

$$101_2 = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 5_{10}$$

- ◆ Cifra binaria: “*bit*” (*binary digit*)
- ◆ Questo sistema numerico è usato, in genere, nei calcolatori e nelle macchine numeriche

Sistema binario

- ◆ Una sequenza di *otto bit* consecutivi è detta *byte*
- ◆ Esempio: 11010010
- ◆ Il bit più a sinistra è detto *Most Significant Bit* (MSB), mentre quello più a destra è detto *Least Significant Bit* (LSB)



Sistema ottale

- ◆ Nel sistema numerico ottale:
 - la base $r = 8$
 - le cifre $d = 0, 1, \dots, 7$
- ◆ Un valore N si rappresenta nel S.N. ottale come:

$$N = d_{n-1} \cdot 8^{n-1} + \dots + d_0 \cdot 8^0 + \\ + d_{-1} \cdot 8^{-1} + \dots + d_{-m} \cdot 8^{-m}$$

Esempio:

$$127_8 = 1 \cdot 8^2 + 2 \cdot 8^1 + 7 \cdot 8^0 = 87_{10}$$

◆ NOTA

La sequenza di cifre 847 *non può essere* un valore ottale in quanto in tale sistema non esiste la cifra 8!

Sistema esadecimale

- ◆ Nel sistema numerico esadecimale:
 - la base $r = 16$
 - le cifre $d = 0, 1, \dots, 9, A, B, C, D, E, F$
- ◆ Un valore N si rappresenta nel S.N. esadecimale come:

$$N = d_{n-1} \cdot 16^{n-1} + \dots + d_0 \cdot 16^0 + \\ + d_{-1} \cdot 16^{-1} + \dots + d_{-m} \cdot 16^{-m}$$

Esempio:

$$A1_{16} = A \cdot 16^1 + 1 \cdot 16^0 = 161_{10}$$

- ◆ Spesso si usa il pedice $_H$ al posto del pedice $_{16}$ per indicare la base esadecimale

Sistema esadecimale

◆ Si noti la corrispondenza:

$$A_H = 10_{10} = 1010_2$$

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

$$C_H = 12_{10} = 1100_2$$

$$D_H = 13_{10} = 1101_2$$

$$E_H = 14_{10} = 1110_2$$

$$F_H = 15_{10} = 1111_2$$

◆ Più è grande il valore della base, più è compatta la rappresentazione di uno stesso valore (ossia il numero risultante è composto da meno cifre):

Esempio

$$B6_{16} = 266_8 = 10110110_2$$

Conversione tra sistemi numerici

Da base qualsiasi a base 10

- ◆ Consideriamo per il momento solo valori interi (o parti intere di valori frazionari)
- ◆ Si può applicare direttamente la definizione:

$$N = d_{n-1} \cdot r^{n-1} + \dots + d_0 \cdot r^0$$

Esempi:

$$1010_2 = (1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0)_{10} = 10_{10}$$

$$26_8 = (2 \cdot 8^1 + 6 \cdot 8^0)_{10} = 22_{10}$$

$$431_5 = (4 \cdot 5^2 + 3 \cdot 5^1 + 1 \cdot 5^0)_{10} = 116_{10}$$

Conversione tra sistemi numerici

- ◆ Un altro metodo deriva dalla possibilità di riscrivere la definizione come segue:

$$N = (((\dots((d_{n-1} \cdot r + d_{n-2}) \cdot r + d_{n-3}) \dots)) \cdot r + d_0)$$

Operativamente:

- si prende la cifra più significativa, la si moltiplica per la base e la si somma alla cifra successiva (ossia quella di peso immediatamente inferiore)
- il risultato così ottenuto lo si moltiplica nuovamente per la base sommandovi la cifra successiva
- si procede in questo modo fino ad arrivare all'ultima cifra (quella di peso zero)

Conversione tra sistemi numerici

◆ Esempi:

$$1101_2 = (((1 \cdot 2 + 1) \cdot 2 + 0) \cdot 2 + 1) = 13_{10}$$

$$435_8 = ((4 \cdot 8 + 3) \cdot 8 + 5) = 285_{10}$$

$$431_5 = ((4 \cdot 5 + 3) \cdot 5 + 1) = 116_{10}$$

Conversione tra sistemi numerici

Da base 10 a base qualsiasi

- ◆ La definizione di valore può essere riscritta come:

$$N = (d_0 + r \cdot (d_1 + r \cdot (d_2 + \dots + r \cdot (d_{n-2} + r \cdot d_{n-1}) \dots)))$$

- se dividiamo il valore N per la base r si ottiene un quoziente dato da:

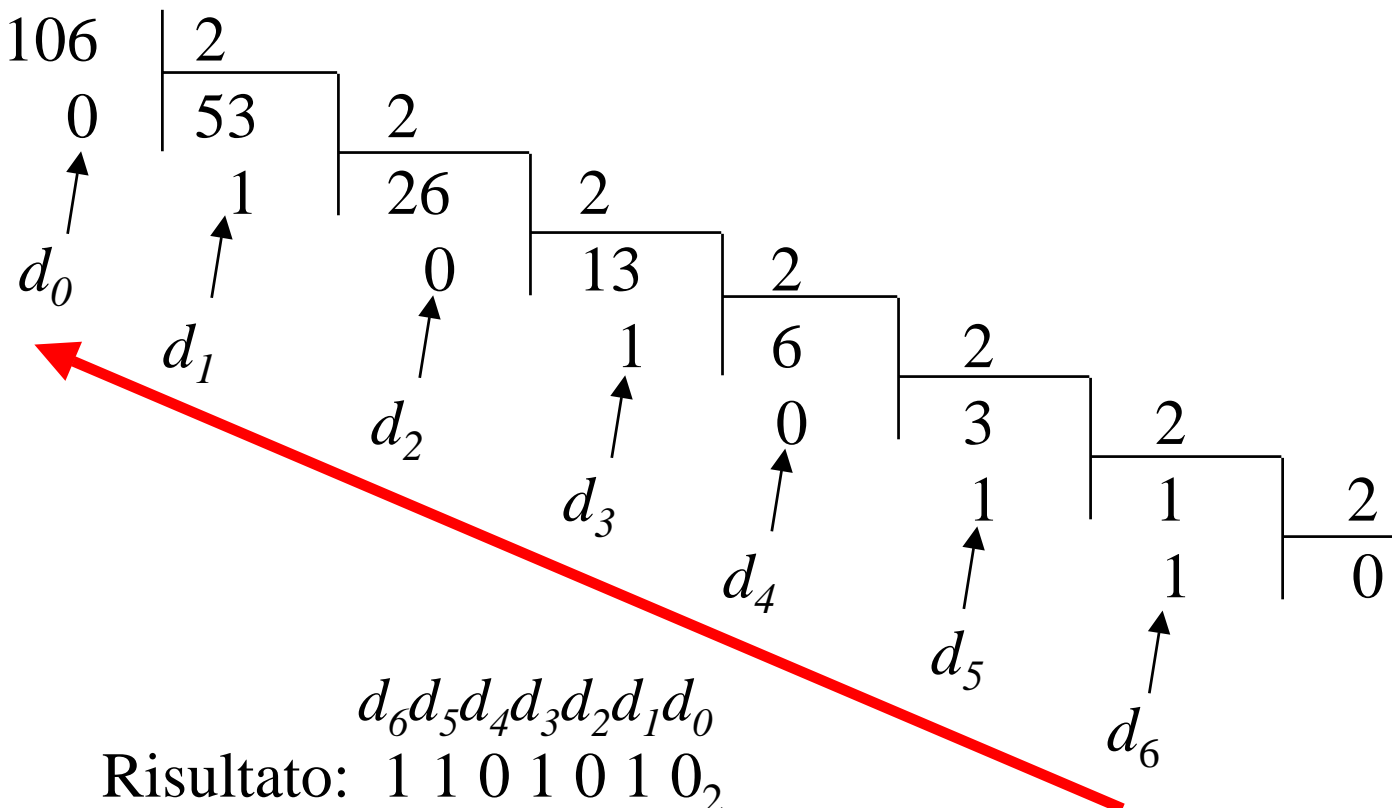
$$(d_1 + r \cdot (d_2 + \dots + r \cdot (d_{n-2} + r \cdot d_{n-1}) \dots))$$

e un resto d_0 che è la cifra di peso inferiore (peso zero) del valore N nella base r

- ◆ Ripetendo il procedimento si ricavano le cifre del valore nella base desiderata (i resti delle divisioni) a partire dal LSB

Conversione tra sistemi numerici

- ◆ Il processo di divisione si arresta quando il quoziente ottenuto è nullo e l'ultimo resto costituisce la cifra di peso maggiore
- ◆ Esempio: dato il valore 106_{10} se ne ricavi la sua rappresentazione in binario:



Conversione tra sistemi numerici

Da base qualsiasi (**N**) a base qualsiasi (**M**)

- ◆ In genere conviene convertire il numero da base **N** a base 10 e il risultato da base 10 a base **M**
- ◆ Nei casi in cui **sia N sia M** siano **potenze di 2**, conviene passare non dalla base 10, ma dalla base 2: la conversione tra una base potenza di 2 e la base 2 è molto veloce se si considera quanto segue

Conversione tra sistemi numerici

- ◆ La definizione di numero binario N può essere riscritta nel seguente modo (in questo esempio si fanno raggruppamenti di 3 cifre):

$$N = \dots + d_5 \cdot 2^5 + d_4 \cdot 2^4 + d_3 \cdot 2^3 + d_2 \cdot 2^2 + d_1 \cdot 2^1 + d_0 \cdot 2^0$$

$$N = \dots + (d_5 \cdot 2^2 + d_4 \cdot 2 + d_3) \cdot (2^3)^1 + (d_2 \cdot 2^2 + d_1 \cdot 2 + d_0) \cdot (2^3)^0$$

$$N = \dots + \beta \cdot 8^1 + \alpha \cdot 8^0$$

- Ne consegue che, per passare **dalla rappresentazione in binario a quella ottale**, è sufficiente raggruppare le cifre a gruppi di tre partendo da destra e convertire i singoli gruppi di cifre ottenuti (se il gruppo più a sinistra ha meno di 3 cifre completarlo a sinistra aggiungendovi gli zeri necessari)

Conversione tra sistemi numerici

◆ Esempio

$$10011001_2 = [010][011][001]_2 = 231_8$$

◆ Analogamente si può procedere per passare dalla rappresentazione binaria a quella in base 2^n , in questo caso i raggruppamenti avvengono a gruppi di n cifre

◆ Esempio:

$$10011001_2 = [1001][1001]_2 = 99_H$$

$$11010010_2 = [1101][0010]_2 = D2_H$$

◆ Chiaramente è valido anche il viceversa: ad ogni cifra in base 2^n corrispondono n bit in binario

Rappresentazione di valori

- ◆ La rappresentabilità dei valori è legata al numero di cifre disponibili
- ◆ Nei sistemi di elaborazione, come in generale in tutte le applicazioni pratiche, il numero di cifre impiegate nella rappresentazione di valori numerici è limitato
- ◆ Si ha *overflow* (o trabocco) quando si è nell'impossibilità di rappresentare il risultato di una operazione (ad esempio una somma o una sottrazione) con il numero di cifre a disposizione

Rappresentazione di valori

- ◆ Consideriamo il sistema binario
- ◆ Il numero di configurazioni diverse con n bit (“distribuzione con ripetizione di 2 elementi di classe n ”) è 2^n ; quindi con n bit si possono rappresentare 2^n valori; il numero N più grande rappresentabile con n bit è:

$$N = 2^n - 1$$

(2^n numeri, il primo è lo 0) ed è costituito da tutti uno:

$$1111111\dots 1_2$$

Rappresentazione di valori

- ◆ Richiedono n cifre (bit) per essere rappresentati tutti i numeri interi X compresi nell'intervallo:

$$\underbrace{1000\dots000}_{n \text{ bit}} \leq X \leq \underbrace{1111\dots111}_{n \text{ bit}}$$

ossia: $2^{n-1} \leq X \leq 2^n - 1$

questa può essere scritta anche:

$$2^{n-1} - 1 < X \leq 2^n - 1$$

- ◆ Sommando 1 ai termini della disuguaglianza:

$$2^{n-1} < X + 1 \leq 2^n$$

e passando ai logaritmi in base 2:

$$n - 1 < \log_2(X + 1) \leq n$$

Rappresentazione di valori

◆ Scindendo la disequaglianza:



$$n \geq \log_2(X + 1)$$



e $n - 1 < \log_2(X + 1)$ diventa

$$n < \log_2(X + 1) + 1$$



Le due disequaglianze  e  affermano che n debba essere almeno grande quanto $\log_2(X + 1)$, ma più piccolo della stessa quantità + 1

◆ Riassumendo:

$$n = \lceil \log_2(X + 1) \rceil$$

dove l'operatore $\lceil \]$ indica l'arrotondamento al valore intero immediatamente superiore (un numero intero resta invariato)

Rappresentazione di valori

– Esempi:

$$\lceil 17,85 \rceil = 18$$

$$\lceil 7,2 \rceil = 8$$

$$\lceil 9 \rceil = 9$$

◆ Per una generica base r vale la relazione generale:

$$n = \lceil \log_r (X + 1) \rceil$$

Rappresentazione di valori

- ◆ Per il sistema binario vale la seguente tabella riassuntiva:

n numero di bit	2^n numero di valori	$2^n - 1$ max valore rappresentabile
1	2	1
2	4	3
3	8	7
4	16	15
5	32	31
6	64	63
...

Rappresentazione di valori

- ◆ Se si indica con D il numero di cifre decimali che occorrono per rappresentare un numero N e con B il numero di bit necessari per rappresentare lo stesso numero si ha:

$$D = \lceil \log_{10}(N + 1) \rceil$$

$$B = \lceil \log_2(N + 1) \rceil$$

- ◆ Il rapporto:

$$\frac{D}{B} = \frac{\lceil \log_{10}(N + 1) \rceil}{\lceil \log_2(N + 1) \rceil}$$

- ◆ non è costante al variare di N

Rappresentazione di valori

◆ Tuttavia, essendo $2^{10} = 1024 \cong 1000$ si può dire che il sistema binario richiede circa 10 cifre ogni 3 cifre utilizzate nella rappresentazione del valore in base 10

◆ Il prefisso *Kilo*, nella terminologia informatica, non indica quindi 1000, bensì:

$$2^{10} = 1024$$

◆ Il prefisso *Mega* indica la quantità:

$$2^{20} = 1048576$$

◆ Il prefisso *Giga* indica:

$$2^{30} = 1073741824$$

Conversione di valori frazionari

Da base qualsiasi a base 10

- ◆ Data la rappresentazione in base N di un valore numerico frazionario, per ottenere la corrispondente rappresentazione decimale si può applicare la definizione

Esempio (N=2):

$$\begin{aligned} 0,1011_2 &= 0 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} + 1 \cdot 2^{-4} = \\ &= 0,6875_{10} \end{aligned}$$

Conversione di valori frazionari

- ◆ Un altro modo consiste nel partire dalla cifra meno significativa, dividerla per N e sommare la cifra che la precede
 - Il procedimento continua finché non si arriva al punto decimale

Esempio (N=2):

$$0,1011_2 = \left(\left(\left(\left(1 \cdot \frac{1}{2} + 1 \right) \cdot \frac{1}{2} + 0 \right) \cdot \frac{1}{2} + 1 \right) \cdot \frac{1}{2} = 0,6875_{10}$$

Conversione di valori frazionari

Da base 10 a base qualsiasi

- ◆ Dato un valore frazionario $N < 0$, lo si esprima in base **b**:

$$N = a_{-1} \cdot b^{-1} + a_{-2} \cdot b^{-2} + \dots + a_{-m} \cdot b^{-m}$$

si osserva che moltiplicandolo per **b** si ricava come valore intero la prima cifra della parte fratta (a_{-1})


- ◆ Se si scarta la parte intera appena moltiplicata e si moltiplica la parte frazionaria restante nuovamente per **b** si ottiene (a_{-2}) e così via

Conversione di valori frazionari

- ◆ Riassumendo, per convertire un numero dalla base 10 alla base N :
 - si moltiplica per N la parte frazionaria del numero decimale estraendo la parte intera
 - si ripete il procedimento sulla parte frazionaria rimasta finché il risultato della moltiplicazione è 1.000...0 oppure non si raggiunge il numero di cifre che si intende utilizzare

Conversione di valori frazionari

◆ Esempio:

$$\begin{array}{r} 0,6875 \quad \times \\ \hline 1,3750 \quad \times \quad 2 \\ \hline 0,7500 \quad \times \quad 2 \\ \hline 1,5000 \quad \times \quad 2 \\ \hline 1,0000 \end{array}$$


Da cui risulta:

$$0,6875_{10} = 0,1011_2$$

Conversione di valori frazionari

Da base qualsiasi (**N**) a base qualsiasi (**M**)

- ◆ In genere conviene convertire il numero da base **N** a base 10 e il risultato da base 10 a base **M**
- ◆ Nei casi in cui **sia N sia M** siano **potenze di 2**, conviene passare non dalla base 10, ma dalla base 2
- ◆ Si faccia attenzione che i gruppi di 2^n cifre devono essere identificati a partire dalla virgola e non dal LSB

Conversione di valori frazionari

◆ Esempi Si convertano in ottale ed esadecimale i seguenti valori frazionari:

$100,001_2$

$= [100], [001]_2$ $4,1_8$

$= [0100], [0010]_2$ $4,2_H$

$1101,01111_2$

$= [001][101], [011][110]_2$ $15,36_8$

$= [1101], [0111][1000]_2$ $D,78_H$

$11010,001_2$

$= [011][010], [001]_2$ $32,1_8$

$= [0001][1010], [0010]_2$ $1A,2_H$

Conversione di valori frazionari

- ◆ Un valore frazionario rappresentabile con un numero finito di cifre in una base N non necessariamente è rappresentabile con un numero finito di cifre anche in una base M diversa da N
- ◆ Quando l'operazione di conversione porta ad avere un numero con infinite cifre decimali, il calcolo termina quando si raggiunge la precisione desiderata

Conversione di valori frazionari

Dato un numero N composto da n cifre dopo la virgola, si definiscono:

◆ Errore assoluto (ε)

la minima quantità rappresentabile in base b con le n cifre, in valore assoluto

Quindi: $\varepsilon = b^{-n}$

◆ Errore relativo (η)

$$\eta = \frac{\varepsilon}{|N|} \cdot 100 \%$$

◆ Esempi

$0,4_{10}$	$\varepsilon = 10^{-1} = 0,1$	$\eta = 25 \%$
$0,1011_2$	$\varepsilon = 2^{-4}$	$\eta = 9,09 \%$
100_{10}	$\varepsilon = 10^{-0} = 1$	$\eta = 1 \%$

Conversione di valori frazionari

- ◆ Esempio Convertire il numero $0,225_{10}$ con un'approssimazione assoluta di $1/32$ ($2^{-5} \Rightarrow 5$ cifre dopo la virgola)

$$0,225_{10} = 0,00111_2$$

- ◆ Quanti bit (n) di parte frazionaria occorrono per convertire un numero in base \mathbf{b} mantenendo una precisione minima assoluta almeno pari a ε ?
Dalla definizione: $\varepsilon = \mathbf{b}^{-n}$, poiché si dice “*almeno pari a*” si ha:

$$\varepsilon \geq \frac{1}{\mathbf{b}^n}$$

da cui:

$$n = \left\lceil \log_{\mathbf{b}} \frac{1}{\varepsilon} \right\rceil$$

Conversione di valori frazionari

- ◆ Esempio Quanti bit occorrono per convertire con una precisione assoluta migliore di $1/100$ il numero $0,136_{10}$ in base 2?

$$n = \lceil \log_2 100 \rceil = 7$$

Nota

Si poteva procedere nel seguente modo:

- si vuole: $\frac{1}{100} > \frac{1}{2^n}$ da cui: $2^n > 100$
 - 100: 2 cifre decimali \Rightarrow 6/7/8 binarie
 - $n = 6 \Rightarrow 2^n = 64$ $64 > 100?$ *NO!*
 - $n = 7 \Rightarrow 2^n = 128$ $128 > 100?$ *OK!*
- quindi $n = 7$

Conversione di valori frazionari

◆ Per convertire un numero composto da una parte intera e da una frazionaria occorre convertire separatamente le due parti e “unirle” successivamente

◆ Esempio:

$$17,5_{10} \Rightarrow 17_{10} = 10001_2$$

$$0,5_{10} = 0,1_2$$

$$17,5_{10} = 10001,1_2$$

Codici

- ◆ Un codice è un insieme di simboli usato per rappresentare, secondo una corrispondenza biunivoca prestabilita, un'informazione di qualsiasi genere (parole, numeri, eventi, etc.)
- ◆ I sistemi numerici (S.N.) sono un caso particolare di codice. In generale un S.N. può essere usato come codice, mentre non vale il viceversa

Codici

◆ Codici binari

Un codice è detto binario quando le parole di codice usano come simboli solo le cifre 0 e 1

◆ Tra i codici binari sono molto usati i *codici BCD* (Binary Coded Decimal)

◆ Nei codici BCD un numero è diviso nelle cifre decimali che lo compongono e ogni cifra è codificata separatamente dalle altre su 4 bit

◆ Tra questi il più conosciuto è il *codice 8-4-2-1* (dal peso delle sue cifre)

Codice ASCII

- ◆ Un tipo particolare di codice a 7 bit molto usato nei computer è il *codice ASCII* (American Standard Code for Information Interchange)
- ◆ Il codice ASCII è usato per la codifica dei caratteri; ad es. ogni volta che viene premuto sulla tastiera un tasto, viene inviato al computer il codice ASCII corrispondente al tasto medesimo
- ◆ Analogamente, quando deve essere stampato a video (o su un altro periferico di uscita) un carattere, viene trasmesso all'unità di visualizzazione il codice ASCII del carattere

Codice ASCII

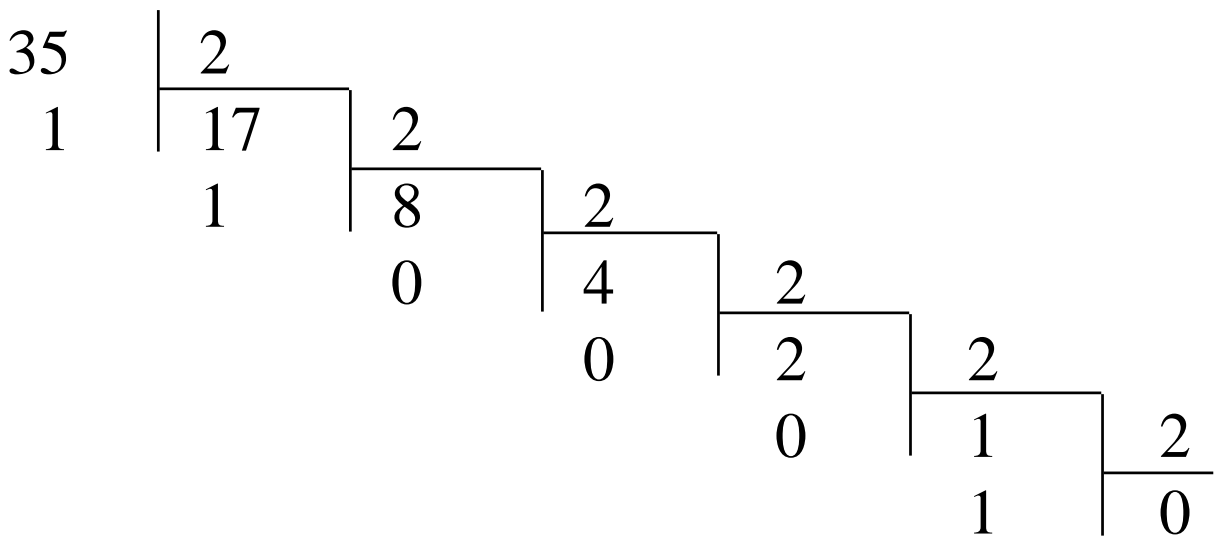
- ◆ Il codice contiene:
 - 26 + 26 lettere (maiuscole + minuscole)
 - 10 cifre decimali (da 0 a 9)
 - segni di interpunzione
 - caratteri di controllo
- ◆ Le cifre sono ordinate per valore
- ◆ Le lettere maiuscole sono ordinate alfabeticamente
- ◆ Le lettere minuscole sono ordinate alfabeticamente (e sono a distanza fissa dalle maiuscole)
- ◆ Le sequenze precedenti non sono contigue (ci sono altri caratteri in mezzo)

Esercizi

◆ Esercizio 1 Si convertano i seguenti numeri rappresentati in base 10 in binario puro:

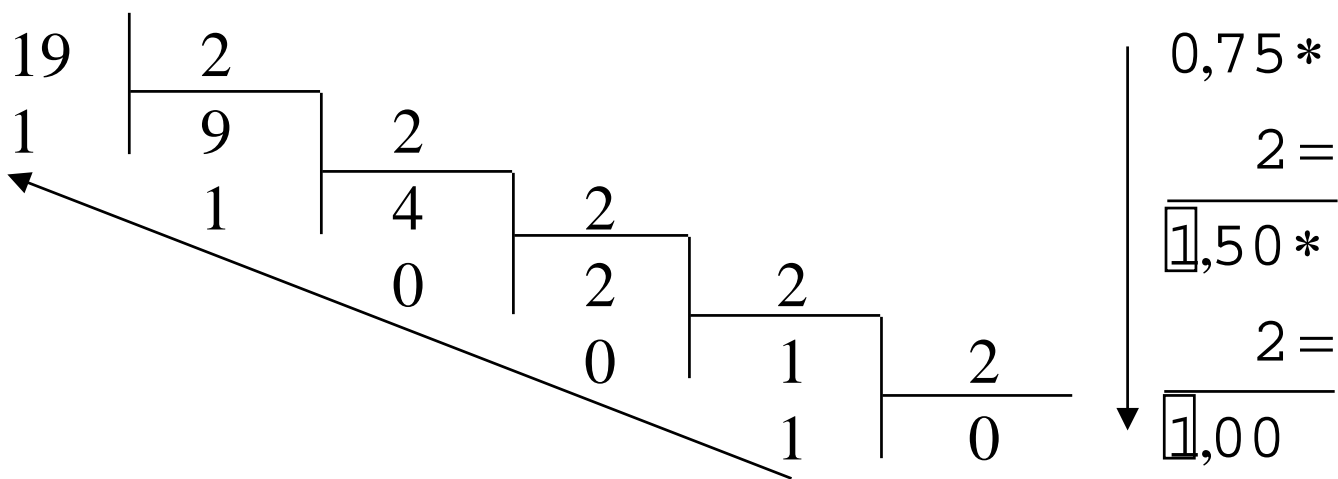
- a) 35 b) 64 c) 48
- d) 19,75 e) 37,25 f) 128,34

a) $35_{10} = 100011_2$

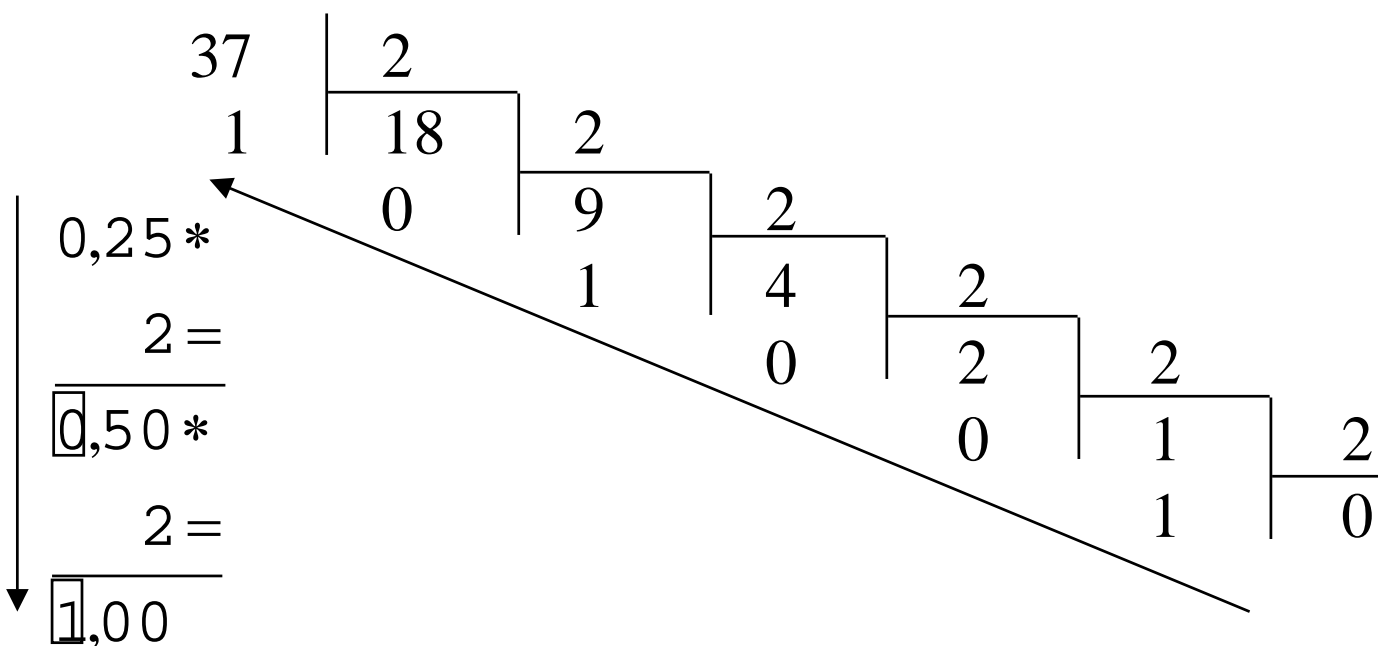


Esercizi

d) $19,75_{10} = 10011,11_2$



e) $37,25_{10} = 100101,01_2$



Esercizi

◆ Esercizio 2 A quali valori decimali corrispondono i seguenti numeri in binario puro?

- a) 00101,001
- b) 10110,1
- c) 1110,0011
- d) 1111,11
- e) 0,101
- f) 100,1001

a) $101,001_2 =$

$$= 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} =$$
$$= 5,125_{10}$$

b) $10110,1_2 =$

$$= 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 + 1 \cdot 2^{-1} =$$
$$= 22,5_{10}$$

c) $1110,0011_2 =$

$$= 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 + 0 \cdot 2^{-1} +$$
$$+ 0 \cdot 2^{-2} + 1 \cdot 2^{-3} + 1 \cdot 2^{-4} = 14,1875$$

Esercizi

d) $1111,11_2 =$

$$= 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2} =$$
$$= 15,75_{10}$$

e) $0,101_2 =$

$$= 0 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} = 0,625_{10}$$

f) $100,1001_2 =$

$$= 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 + 1 \cdot 2^{-1} + 10 \cdot 2^{-2} +$$

◆ **Esercizio 3** Si convertano i seguenti numeri binari nelle rappresentazioni ottali e esadecimali:

– a) 10110010

b) 11010001

– c) 11111101

d) 10100111

– e) 10101010

f) 1110101

Esercizi

a) $10110010_2 = 262_8 = B2_H$

b) $11010001_2 = 321_8 = D1_H$

c) $11111101_2 = 375_8 = FD_H$

d) $10100111_2 = 247_8 = A7_H$

e) $10101010_2 = 252_8 = AA_H$

f) $1110101_2 = 165_8 = 75_H$

◆ Esercizio 4 Convertire i seguenti numeri in base 2:

– a) 34_5

b) 15_7

c) 23_8

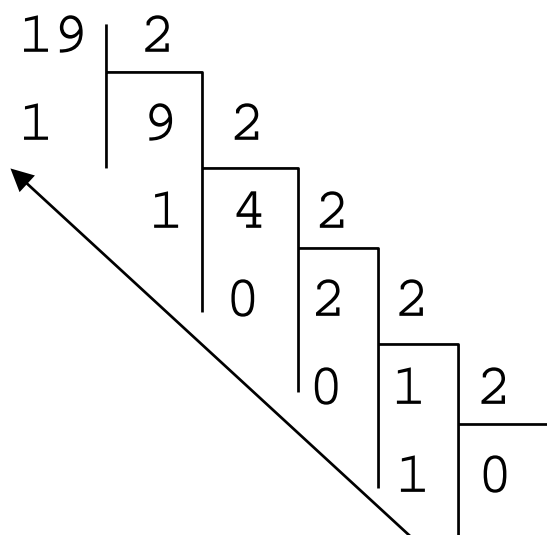
– d) $A2_H$

e) 213_3

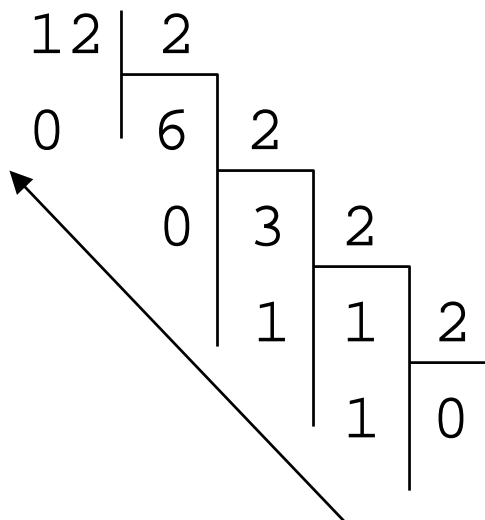
f) 34_{10}

Esercizi

a) $34_5 = 3 \cdot 5^1 + 4 \cdot 5^0 = 19_{10} = 10011_2$



b) $15_7 = 1 \cdot 7^1 + 5 \cdot 7^0 = 12_{10} = 1100_2$



Esercizi

c) $23_8 = 10\ 011_2$

d) $A2_H = 1010\ 0010_2$

e) Non possibile perché 213_3 non può essere una rappresentazione in base 3

f) $34_{10} = 100010_2$

