SISTEMA NUMERICO

Il più comune è il sistema posizionale, cioè le cifre (digit) hanno un valore che dipende da:

1. cifra stessa
2. dalla sua posizione

Oss. Es. di sistema non posizionale (dove si usano dei simboli) es. num. romani

BASE o RADICE

indica il numero di cifre che compone l'alfabeto del sistema.

NUMERO INTERO

$$N_b = d_{m-1}\, d_{m-2} \cdots d_2\, d_1\, d_0$$

N : è il numero
b : base
m : numero di cifre del numero
d_i : cifra appartenente al sistema base b
d_{m-1} : cifra + significativa (MSD Most Significant Digit)
d_0 : cifra − significativa (LSD Least Significant Digit)

NUMERI CON FRAZIONARIA

La virgola separa la parte intera dalla parte frazionaria.

$$N_b = d_{m-1}\, d_{m-2} \cdots d_2\, d_1\, d_0, d_{-1}\, d_{-2} \cdots d_{m-1}\, d_{-m}$$

$m+m$ = numero di cifre che compone il numero
d_{-m} = cifra − significativa (LSD)

SISTEMA BINARIO

- 2 cifre = $\{0,1\}$
- posizionale
- base 2
- il pedice $b=2$

Es. $1010_1{}_2$: si è binario
11001 : Non lo posso sapere.

Oss. Senza pedice, per convenzione, un numero senza pedice è considerato DECIMALE.

Note -
4 bit = NIBBLE
8 bit = BYTE
16 bit = WORD
32 bit = DOUBLE WORD
64 bit = QUAD WORD

SISTEMA OTTALE (usato in AUTOMAZIONE)

- 8 cifre = {0,1,2,3,4,5,6,7}
- posizionale
- base 8
- il pedice b=8

Oss. notare che è multiplo di 2

Es. 1367_8: sì è un numero ottale

1728_8: sicuramente NO !!!

SISTEMA DECIMALE

- 10 cifre = {0,1,2,3,4,5,6,7,8,9}
- posizionale
- base 10
- il pedice b=10

SISTEMA ESADECIMALE

- 16 cifre = {0,1,2,3,4,5,6,7,8,9,A(10),B,C,D,E,F(15)}
- posizionale
- base 16
- pedice b=16

Es. 180: potrebbe essere, ma non ne sono sicuro.

$13A0_{16}$: sicuramente è esadecimale.

—o—

NUMERI INTERI

numeri decimali => numeri binari / ottali / esadecimali

① Si divide N_{10} per la base del numero di destinazione, fino ad ottenere un quoziente = 0

② i resti che si hanno durante queste divisioni (<b) sono le cifre della conversione, l'ultimo resto è la cifra più significativa.

Es. dato un numero 831_{10} trovare il corrispondente in binario:

831	2	1
415	2	1
207	2	1
103	2	1
51	2	1
25	2	1
12	2	0
6	2	0
3	2	1
1	2	1
0		

MSB

RESTI

DIVISORE

DIVIDENDO

Risposta è: $831_{10} = 1100111111_2$

Oss. dividendo per 2 ottengo resti di 0,1 cioè le cifre del binario.

numeri < binari / ottali / esadecimali => numeri decimali

Metodo teorico-matematico

$$\sum_{i=0}^{n-1} X_i \cdot b^i$$

X : cifra o digit del sistema
i : posizione
b : base

Es. Passare da 10011_2 al corrispondente decimale

$$10011 = 1\cdot 2^4 + 0\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 =$$

(i=4, i=0)

$$1\cdot 16 + 0 + 0 + 1\cdot 2 + 1\cdot 1 = 16+2+1 = 19_{10}$$

METODO PRATICONE PER IL BYTE (8 bit)

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	← potenza di 2 delle prime 8 cifre
128	64	32	16	8	4	2	1	← valore o peso in base alla posizione
0	0	0	1	0	0	1	1	← numero binario da convertire
			16 +			2+1	= 19_{10}	← Somma dei valori che corrispondono ai bit = "1"

NUMERI CON PARTE FRAZIONARIA

numero decimale => numero < binario / ottale / esadecimale

① La parte intera si gestisce come sopra.

② La parte frazionaria: si divide invece che moltiplicare per la base

i) Si prende la parte frazionaria: $0,\ldots\ldots$

ii) Si moltiplica per la base di arrivo: $0,\ldots\ldots \cdot (\text{base di arrivo})$
la parte intera rappresenta la cifra da mettere, in ordine dopo la virgola

iii) riprendo la parte frazionaria del p.to ii) e applico i pti i) e ii)

Continuo fino a quando:

- raggiungo la precisione voluta
- il numero, dopo la moltiplicazione è intero.

Esempio: convertire il valore $23,35_{10}$ in un numero binario.

PARTE INTERA

23	2	1
11	2	1
5	2	1
2	2	0
1	2	1
0		

10111_2

PARTE FRAZIONARIA

(base di arrivo) (cifre in ordine dopo la ",")

$0,35 \cdot 2 = 0,70$ 0
$0,70 \cdot 2 = 1,40$ 1
$0,40 \cdot 2 = 0,80$ 0
$0,80 \cdot 2 = 1,60$ 1
$0,60 \cdot 2 = 1,20$ 1

... ci fermiamo alla 5ª cifra dopo la virgola

RISULTATO: $23,35_{10} \rightarrow 10111,01011$

numero $\left\{\begin{array}{l} \text{binario} \\ \text{ottale} \\ \text{esadecimale} \end{array}\right. \Longrightarrow$ numero decimale

① La parte intera come i numeri interi

② La parte frazionaria: si moltiplica la cifra per la base con esponente negativo

Esempio: $10111,01011_2 \longrightarrow \sum_{i=0}^{m-1} X_i b^i + \sum_{u=-1}^{-m} X_u \cdot b^{-u}$

(intera / frazionaria)

$2^4\ 2^3\ 2^2\ 2^1\ 2^0$

$16\ 8\ 4\ 2\ 1$

$16 + 4+2+1 = 23_{10}$

$0\cdot 2^{-1} + 1\cdot 2^{-2} + 0\cdot 2^{-3} + 1\cdot 2^{-4} + 1\cdot 2^{-5}$

$0{,}25 + 0{,}0625 + 0{,}03125$

RIS.: $23{,}34375_{10}$

Polinomio: $X_{m-1} b^{m-1} + X_{m-2} b^{m-2} + \ldots + X_1 b^1 + X_0 b^0 + X_{-1} b^{-1} + X_{-2} b^{-2} + \ldots + X_{m-1} b^{m-1} + X_{-m} \cdot b^{-m}$

Qualche informazione:

[1] Con n cifre, in base b, si possono rappresentare valori da 0 a $(b^n - 1)$

es. 3 cifre in base 10: 0 a $(10^3 - 1) = 1000 - 1 = 999$ (con lo 0 mille numeri distinti)

[2] binario → ottale
ottale → binario : si raccolgono i bit a gruppi di 3, partendo dalla virgola. Per la parte intera verso Sx completando con 0. Per la parte frazionaria verso dx e completo con 0.

$010\,110\,111,011\,100_2$

$2\ 6\ 7,3\ 4_8$

3 Esadecimale → binario
binario → Esadecimale

10110111,0111

11 → B 7 , 7

Si raccolgono a partire dalla virgola: gruppi di 4 bit, verso SX per la parte intera e verso dx per la parte frazionaria. Se mancano cifre a sx o dx si completano con degli 0.

0 1 2 3 4 5 6 7 8 9 A B C D E F → 10
11
12

PREFISSI MULTIPLI DEI NUMERI BINARI

S.I.

K = Kilo = 10^3 => Kb = Kbit = 10^3 bit = 1000 bit
M = Mega = 10^6 => Mb = Mbit = 10^6 bit
G = Giga = 10^9 => Gb = Gbit = 10^9 bit
T = Tera = 10^{12} => Tb = Tbit = 10^{12} bit

Questo sistema viene usato nella telecomunicazione per misurare la quantità di dati trasmessi nell'unità di tempo.

Es. 25 Mbps => 25 Mb per secondo
↑
b minuscolo significa bit

Oss. Attenzione perché a volte viene usata l'U.M. MBps (B = byte = 8 bit)
il rapporto è 8 Mbps = 1 MBps
↑
(B volte maggiore)

Nell'informatica l'istituto IEC (comitato elettrotecnico internazionale) ha adottato i seguenti prefissi binari:

U.M. bibit
Ibinary

PREFISSI	U.M.	ABBREV.	ESPONENTE	DECIMALE
Ki	Kibibit	Kibit	2^{10}	1024 bit
Me	Mebibit	Mibit	2^{20}	
Gi	Gibit	Gibit	2^{30}	
Te	Tebit	Tibit	2^{40}	

Oss. 1 Mibit = 1024 Kibit

CALCOLI ARITMETICI CON NUMERI BINARI

Somma

$$\begin{array}{r} 0+\\ 0=\\ \hline 0 \end{array} \qquad \begin{array}{r} 0+\\ 1=\\ \hline 1 \end{array} \qquad \begin{array}{r} 1+\\ 0=\\ \hline 1 \end{array} \qquad \begin{array}{r} 1+\\ 1=\\ \hline 10 \end{array}$$

↑ RIPORTO (CARRY)

Sottrazione

$$\begin{array}{r} 0-\\ 0=\\ \hline 0 \end{array} \qquad \begin{array}{r} 0-\\ 1=\\ \hline \end{array} \qquad \begin{array}{r} 1-\\ 0=\\ \hline 1 \end{array} \qquad \begin{array}{r} 1-\\ 1=\\ \hline 0 \end{array}$$

(1)

(1) vediamo come funziona $0-1=?$

caso 1

$$\begin{array}{r} {\scriptstyle 0\,1}\\ 1\cancel{1}0-\\ 1=\\ \hline 1 \end{array}$$

fase 1 - chiedo un prestito a un bit + signif.

fase 2 - il bit "1" che fa il prestito diventa "0" e il bit "0" che lo riceve diventa "10" e faccio il calcolo.

caso 2

$$\begin{array}{r} {\scriptstyle 0\,1}\\ 1\cancel{0}0-\\ 1=\\ \hline \end{array} \quad \Downarrow \quad \begin{array}{r} 0\,1\,10-\\ 1=\\ \hline 0\,1\,1 \end{array}$$

fase 1 - chiedo un prestito a cifra + Sign.

fase 2 - la cifra che fa il prestito va a "0" le altre vanno a "1" e l'ultimo che riceve il prestito "10"

prova: $100 \to 4$ $001 \to 1$ $4-1=3 \to 011$

Moltiplicazione

$$\begin{array}{cccc} 0\cdot & 0\cdot & 1\cdot & 1\cdot \\ 0= & 1= & 0= & 1= \\ \hline 0 & 0 & 0 & 1 \end{array}$$