Corso Integrato di Statistica Informatica e Analisi dei dati Informatica - 4b

Dr Carlo Meneghini
Dip. di Fisica "E. Amaldi"
via della Vasca Navale 84
meneghini@fis.uniroma3.it
st. – 83 -
tel.: 06 55177217
Home page
http://www.fis.uniroma3.it/~meneghini

Codificare quantità, suoni e caratteri

Sistemi di numerazione posizionale

Ingredienti:
- insieme di simboli (cifre) C_i
- base **B** = numero di simboli

C_i = cifre: valori interi progressivi

Cifra più significativa — Cifra meno significativa

$$C_n \ldots C_2\, C_1\, C_o$$

$$C_n \cdot B^n + \ldots C_2 \cdot B^2 + C_1 \cdot B^1 + C_o \cdot B^0 = \text{quantità}$$

Attenzione: C_n è il valore numerico della cifra n-esima

Il sistema Binario

E un sistema di numerazione posizionale

che utilizza 2 cifre: 0, 1

Base = 2

$1\,1\,0\,1\,1\,0\,1^{bin}$

$1\cdot2^6+ 1\cdot2^5+ 0\cdot2^4+ 1\cdot2^3+ 1\cdot2^2+ 0\cdot2^1+ 1\cdot2^0 =$

$= 1\cdot64+1\cdot32+0\cdot16+1\cdot8+1\cdot4+ 0\cdot2+1\cdot1 =$

$= 109^{dec}$

Sistemi di numerazione per PC

B = 2: sistema binario (bin)

B = 10: sistema decimale (utenti umani)

B = 8: ottale (oct)

B = 16: esadecimale (HEX)

Dec	Binario	ottale	HEX
0	0 0000 0000	0	0
1	0 0000 0001	1	1
2	0 0000 0010	2	2
3	0 0000 0011	3	3
4	0 0000 0100	4	4
5	0 0000 0101	5	5
6	0 0000 0110	6	6
7	0 0000 0111	7	7
8	0 0000 1000	10	8
9	0 0000 1001	11	9
10	0 0000 1010	12	A
15	0 0000 1111	17	F
16	0 0001 0000	20	10
255	0 1111 1111	377	FF
256	1 0000 0000	400	100

Proprietà dei sistemi di numerazione posizionale

- in base **B** un 1 seguito da n-0 rappresenta $\mathbf{B^N}$

  es.: N=3

  binario (**B**=2): $1000 = \mathbf{2}^3 = 8$

  Hex (**B**=16): $1000 = \mathbf{16}^3 = 4096$

- con N cifre posso rappresentare $\mathbf{B^N}$ valori diversi, compreso lo 0 e quindi contare fino a $\mathbf{B^N}$-1

  es.: N=3

  binario: $111^{bin} = \mathbf{2}^3\text{-}1 = 7$

  decimale $999^{dec} = \mathbf{10}^3\text{-}1 = 999$

  Hex: $FFF^{hex} = \mathbf{16}^3 - 1 = 4095$

Conversione decimale -> binario

Scrivere 81^{dec} in notazione binaria

1	2	5	10	20	40	81	N
1/2	2/2	5/2	10/2	20/2	40/2	81/2	base = 2
0	1	2	5	10	20	40	quoziente (N/B)
1	0	1	0	0	0	1	Resto

$81^{dec} = 1010001^{bin}$

Scrivere 55^{dec} in notazione binaria

1	3	6	13	27	55	N
1/2	3/2	6/2	13/2	27/2	55/2	base = 2
0	1	3	6	13	27	quoziente (N/B)
1	1	0	1	1	1	Resto

$55^{dec} = 110111^{bin}$

Conversione: decimale -> base *B*

1	10	81	N
	10/8	81/8	Base = 8
0	1	10	quoziente (N/B)
1	2	1	Resto

$81^{dec} = 121^{oct}$

5	81	N
16		Base = 16
0	5	quoziente (N/B)
5	1	Resto

$81dec = 51^{hex}$

6	55	N
8		Base = 8
0	6	quoziente (N/B)
6	7	Resto

$55^{dec} = 67^{oct}$

14	229	N
16		Base = 16
0	14	quoziente (N/B)
14	5	Resto

$229dec = E5^{hex}$

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

Cambiamento di base: tabelle di conversione

Da bin a oct

Si convertono terzine di cifre binarie

001	000	110	101
1	0	6	5

$0010\ 0011\ 0101^{bin} = 1065^{oct}$

bin	oct
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

tabella di conversione bin/oct

Da bin a hex

Si convertono quartine di cifre binarie

0010	0011	0101
2	3	5

$0010\ 0011\ 0101^{bin} = 235^{hex}$

	00	01	10	11
00	0	1	2	3
01	4	5	6	7
10	8	9	A	B
11	C	D	E	F

tabella di conversione bin/hex

bin	hex	bin	hex
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

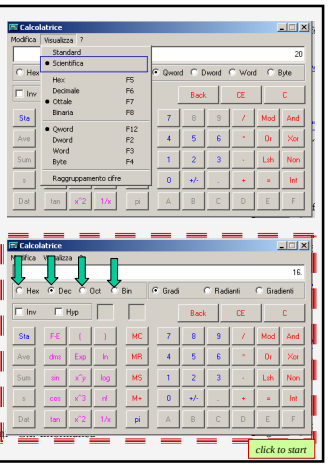
tabella di conversione bin/hex

dec	15
bin	0000 0000 0000 1111
Hex	0 0 0 F

dec	3 855
bin	0000 1111 0000 1111
Hex	0 F 0 F

dec	3 850
bin	0000 1111 0000 1010
Hex	0 F 0 A

dec	44 800
bin	1010 1111 0000 0000
Hex	A F 0 0



Rappresentazione dei numeri interi relativi

Problema: come codificare il segno (numeri interi "relativi") senza aggiungere un simbolo (-) al codice binario?

rappresentazione in modulo e segno

rappresentazione in complemento a 2

rappresentazione in eccesso 2^{N-1}

Rappresentazione in modulo e segno

Il bit piú significativo (MSB) rappresenta il segno, i restanti N-1 bit rappresentano il valore assoluto del numero

1	0	1	0	1	1	0	1

MSB

segno: 1 = + 0= -

valore assoluto

0	0	1	0	1	1	0	1

45^{dec}

1	0	1	0	1	1	0	1

-45^{dec}

8 bit: codificano i numeri

da $-(2^7 - 1) = -127$ a $+(2^7 - 1) = 127$

In totale? = **255**

Problemi

Efficienza:

$+0^{dec} = 0000\ 0000^{bin}$

$-0^{dec} = 1000\ 0000^{bin}$

Lo zero è rappresentato con due diverse codifiche

posso codificare solo 255 valori anziché 256

calcolo

$x + (-x) \neq 0$

Non valgono le normali regole di somma

Decimale con segno

$7^{dec} +$

$-7^{dec} =$

-0^{dec}

Binario mod+sgn

$0000\ 0111^{bin} +$

$1000\ 0111^{bin} =$

$1000\ 1110^{bin}$

-14^{dec}

Rappresentazione in complemento a 2

Il bit piú significativo (MSB) eguale a 1 indica un peso negativo (-1). Gli altri bit hanno i pesi positivi che gli spettano in base alla posizione.

1	1	0	1	0	0	1	1

$-1\cdot2^7+1\cdot2^6+0\cdot2^5+1\cdot2^4+0\cdot2^3+0\cdot2^2+1\cdot2^1+1\cdot2^0 =$

$- 128 + 64 + 0 + 16 + 0 + 0 + 2 + 1 =$ **$- 45^{dec}$**

$1000\ 0000 = -128^{dec}$; $1111\ 1111 = -1^{dec}$

$0000\ 0000 = 0^{dec}$; $0111\ 1111 = 127^{dec}$

Lo zero è rappresentato con una sola codifica

NOTA: Con N bit si possono rappresentare 2^N valori interi con segno da -2^{N-1} a $2^{N-1}-1$
Con 8 bit posso codificare 256 valori tra -128 e 127

13

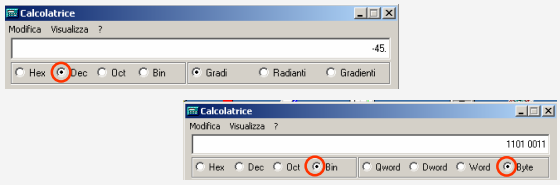
Rappresentazione in complemento a 2

Il numero X si rappresenta con lo stesso codice binario del numero assoluto:

$2^N - X$

es.: -45^{dec}, utilizzando 8 bit si ottiene codificando il valore:

$2^8 - 45 = 211^{dec} = 1101\ 0011^{bin}$



Codifica: Esempio: -45

45^{dec}	0	0	1	0	1	1	0	1
binario complementare	1	1	0	1	0	0	1	0
+1								1
	1	1	0	1	0	0	1	1

+ $- 45^{dec}$

C. Meneghini - C.I.-Informatica 15

Codifica: Esempio: -40

40^{dec}	0	0	1	0	1	0	0	0
binario complementare	1	1	0	1	0	1	1	1
+1								1
riporti	0	0	0	0	1	1	1	
	1	1	0	1	1	0	0	0

+ $- 40^{dec}$

C. Meneghini - C.I.-Informatica 16

Proprietà della rappresentazione in compelmento a 2

a) X e -X hanno codifiche molto diverse:

$+45^{dec} =$ **$0010\ 1101$**

$-45^{dec} =$ **$1101\ 0011$**

b) Per le somme algebriche valgono le stesse regole della somma binaria (trascurando il riporto sul bit piú significativo)

$X + (-X) = 0$

Riporto da trascurare

Riporti

Decimale con segno: $45^{dec} + (-45^{dec}) = -00^{dec}$

$1\ 1111\ 111$

$0010\ 1101^{bin} +$

$1101\ 0011^{bin} =$

$0000\ 0000^{bin}$

17

⚠ Errori di calcolo: **overflow** ⚠

NOTA: Con N bit si possono rappresentare 2^N valori interi con Segno da -2^{N-1} a $2^{N-1}-1$

N = 7

$93^{dec} + 35^{dec} = 128^{dec}$

$0101\ 1101^{bin} + 0010\ 0011^{bin} = 1000\ 0000^{bin}$

$128 > 2^7-1$

⚠ **$= -128^{dec}$**

$-93^{dec} + -75^{dec} = -168^{dec}$

$1010\ 0011^{bin} + 1011\ 0101^{bin} = 0101\ 1000^{bin}$

$-168 < -2^7$

⚠ **$= -88^{dec}$**

Si ha un errore di **OVERFLOW** quando il numero di bit non è sufficiente alla rappresentazione del numero

Nota: effetto degli errori di overflow:

Non è sempre valida la proprietà associativa:
$(A + B) + C = A + (B + C)$

esempio: usiamo una una notazione a 8 bit con segno (complemento a 2)

siano: A = 40, B = -99, C = 100: **A + B + C = 50**

A = 0010 1000^{bin}
B = 1010 0110^{bin}
C = 0110 0100^{bin}

A+B
0010 1000^{bin} +
1010 0110^{bin} =
1100 1110^{bin}
-128 + 64 + 8 + 4 + 2 = -50^{dec}

(A+B)+C
1100 1110^{bin} +
0110 0100^{bin} =
0011 0010^{bin}
32 + 16 + 2 = 50^{dec}

-128 64 32 16 8 4 2 1

Nota: effetto degli errori di overflow:

esempio: usiamo una una notazione a 8 bit con segno (complemento a 2)

siano: A = 40, B = -90, C=100:

(A + B) + C ? (A + C) + B

A = 0010 1000^{bin}
B = 1010 0110^{bin}
C = 0110 0100^{bin}

A+C
0010 1000^{bin} +
0110 0100^{bin} =
1000 1100^{bin}
-128 + 8 + 4 = -116^{dec}

40^{dec} +
100^{dec} =
140^{dec}

ST0P: OVERFLOW ERROR

-128 64 32 16 8 4 2 1

Rappresentazione in eccesso 2^N-1

$C_{N-1} \ldots C_2\ C_1 = C_n \cdot 2^{N-1} + \ldots C_2 \cdot 2 + C_1 \cdot 1 - 2^{N-1}$

Il valore di un numero binario a N bit in notazione in eccesso si ottiene calcolando il valore binario del numero e sottraendo il valore 2^{N-1}

0000 0000 = -128
0000 0001 = $2^0 - 2^7$ = 1 -128 = - 127
0000 1000 = $2^3 - 2^7$ = 16 -128 = - 112
0111 1111 = ... = 127 -128 = - 1
1111 1111 = ... = 255 -128 = 127
1000 0000 = $2^7 - 2^7 = 0$
1000 0001 = $2^7 + 1 - 2^7 = 1$
1001 0000 = $2^7 + 2^3 - 2^7$ = 16

Nota: Il bit piú significativo è
0 per i numeri negativi (da -2^{N-1} a -1)
1 per i numeri positivi (da 0 a 2^{N-1} -1)

La rappresentazione in eccesso si utilizza per la **caratteristica** nella rappresentazione dei numeri reali

Rappresentazione dei numeri frazionari **virgola fissa**

$\ldots C_1\ C_o\ ,\ C_{-1}\ C_{-2} \ldots$

$\ldots C_1 \cdot B^1 + C_o \cdot B^0\ ,\ C_{-1} \cdot B^{-1} + C_{-2} \cdot B^{-2} \ldots$

1101,0011

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

8 + 4 + 1 , 1/8 + 1/16

8 + 4 + 1 , 0.125 + 0.0625

13,1875^{dec}

Codifica binaria in **virgola fissa**

.09375	.1875	.325	.75	.5	0		N
		2					base
0.1875	0.325	0.75	1.5	1			Prodotto
0	0	0	1	1			parte intera

.09375^{dec} = 0.00011^{bin}

0.2	0.4	0.8	0.6	0.2	0.4	0.8	N
		2					base
0.4	0.8	1.6	1.2	0.4	0.8	1.6	Prodotto
0	0	1	1	0	0	1	parte intera

0.2^{dec} = 0.0011001100110011....^{bin} periodico in notazione binaria

Errori di Troncamento

$0.2^{dec} = 0,\overline{0011}^{bin}$

$\text{Errore}(\%) = \frac{V_{appr.} - V_{esatto}}{V_{esatto}}$

.5
.25
.125
.0625
.03125 0.0011^{bin} = 0.1875^{dec} 4 bit
.015625
.0078125
.00390625
.001953125 0.0011 0011^{bin} = 0.19921875^{dec} 8 bit
.0009765625
.00048828125
.000244140625
..... 0.0011 0011 0011^{bin} = 0.199951171875 ^{dec} 12 bit

Errore ~ 7 %
Errore ~ 0.4 %
Errore ~ 0.02 %

C. Meneghini - C.I.-Informatica

Rappresentazione dei numeri frazionari: **virgola mobile**

Numeri grandi: 3 450 000 000 000 Numeri piccoli: 0.000 000 000 645

Problema: rappresentare numeri molto grandi o molto piccoli ottimizzando il numero di difre da scrivere

Numeri grandi: 3 450 000 000 000 = $3.45 \cdot 10^{12}$
Numeri piccoli: 0.000 000 000 645 = $6.45 \cdot 10^{-11}$

$m \cdot B^n = m\ E\ n$

mantissa esponente (Caratteristica)

Notazione scientifica

Numeri grandi: 3 450 000 000 000 = 3,45 E 12
Numeri piccoli: 0.000 000 000 645 = 6,45 E -11

25

Mantissa Normalizzata

$m \cdot B^n$

- $0 \le m < 1$
- la prima cifra della mantissa sempre diversa da 0

$3\ 450\ 000^{dec} \Rightarrow 0.345\ E\ 7^{dec}$ $0.345 \cdot 10^7$

se $m \neq 0$ si ha : $B^{-1} \le m < 1$

$101101^{bin} \Rightarrow 0{,}101101 \cdot 2^6\ (bin)$

$0.0010011^{bin} \Rightarrow 0.10011 \cdot 2^{-2}\ (bin)$

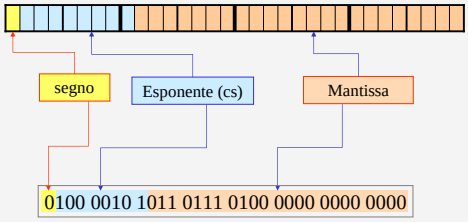
Nota: In codice binario, nella notazione con mantissa normalizzata, la prima cifra dopo la virgola è sempre **1**

per il Computer è piú efficiente sottintendere la prima cifra non nulla:

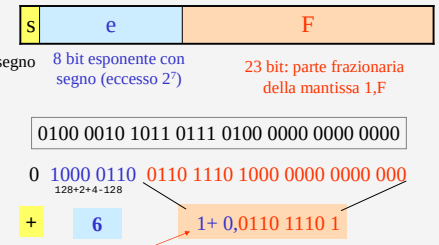
$101101^{bin} \Rightarrow (1+0{,}01101)\ E\ 5^{bin}$

$(1+0{,}01101) \cdot 2^{5\ bin}$

Rappresentazione in virgola mobile **numeri reali**



Word = 32 bit (4 byte)



28

Codifica di numeri reali

Qualunque sia la codifica adottata, la rappresentazione dei numeri reali può essere approssimata

Es.: decimale

Numero	Rappresentazione			
1/3	0.33	0.33333	0.3333333333	0.333333333333333333…
π	3.14	3.14159	3.1415926536	3.14159265358979238…
$2^{1/2}$	1.41	1.41421	1.4142135624	1.41421356237309504 8…
e	2.72	2.71828	2.7182818285	2.71828182845904523 5…
sin(10)	0.17	0.17365	0.1736481777	0.17364817766693034885

La mantissa ha una dimensione finita, ció puó comportare errori di arrotondamento

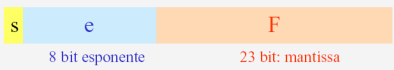
$$\text{Errore}(\%) = \frac{V_{appr} - V_{esatto}}{V_{esatto}}$$

35

Precisione: errori di troncamento

La precisione è determinata dal numero di cifre frazionarie che si possono rappresentare

π	err.(%)
3.14	$5 \cdot 10^{-4}$
3.141593	$1 \cdot 10^{-7}$
3.141592654	$1 \cdot 10^{-10}$
3.1415926535898	$2 \cdot 10^{-15}$
3.14159265358979323846264338 32....	$3 \cdot 10^{-26}$....



8 bit esponente -128 127

23 bit: mantissa $2^{23} = 8\ 388\ 608^{dec}$

Il numero di bit dell'esponente determina l'intervallo dei numeri rappresentabili

Il numero di bit della mantissa determinano la precisione

s	e	F
	8 bit esponente	23 bit: mantissa 2^{23} = 8 388 608^{dec}

Esempi (singola precisione: 32 bit)

5.000 001 + 5.0 = 10.0	5.000 001 - 5.0 = $9.53674316 \cdot 10^{-7}$
4.000 005 + 4.0 = 8.000 00477	4.000 005 - 4.0 = $4.76837158 \cdot 10^{-6}$
400.005 + 4 = 404.005005	400.005 + 4.000 005 = 404.005 005

il numero piú piccolo ε che si deve aggiungere a X perchè sia X + ε ≠ X è: ε ~ $2.38 \cdot 10^{-7}$

Programma: precisione.exe
su www.fis.uniroma3.it/~meneghini

esempio
3. + **X** = 3.000 000 24 per
2E-7 < **X** < 3E-7

Conseguenze

A = 5.000001 B = 5.0

Calcolo algebrico:
$(A-B) \cdot 10^6 = 1$

Calcolo numerico (singola precisione):
0.95367432

Err. = 0.05%

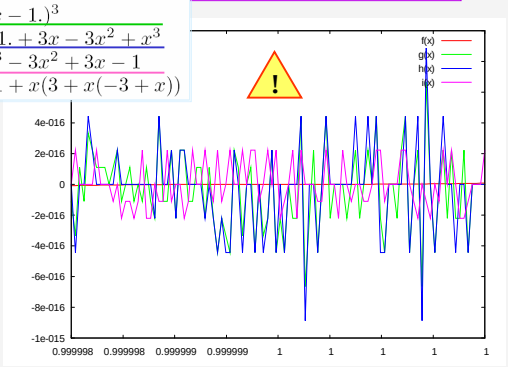
Soluzione:
doppia precisione (64 bit)

Precisione	singola	doppia
N. di bit totali	32	64
N. Bit di Segno	1	1
N. bit Mantissa	23	52
N. Bit esponente	8	11
Esponente massino	127	1023
Esponente minimo	-126	-1022

Programma: precisione2.exe
su www.fis.uniroma3.it/~meneghini

Precisione numerica

$$f(x) = (x-1.)^3$$
$$g(x) = -1. + 3x - 3x^2 + x^3$$
$$h(x) = x^3 - 3x^2 + 3x - 1$$
$$i(x) = -1 + x(3 + x(-3 + x))$$



Esempio tratto dal corso di Davide Manca – Politecnico di Milano

Precisione numerica

x	$(x-1.)^3$	$-1.+3x-3x^2+x^3$	x^3-3x^2+3x-1	$-1+x(3+x(-3+x))$	err%
1.00001000	1E-15	0.000000E+00	0.000000E+00	0.000000E+00	100
1.00001300	2.197E-15	2.442491E-15	2.664535E-15	2.220446E-15	-11
1.00001600	4.096E-15	3.774758E-15	3.996803E-15	4.218847E-15	7.84
1.00001900	6.859E-15	7.327472E-15	7.105427E-15	6.883383E-15	-6.83
1.00002200	1.0648E-14	1.065814E-14	1.065814E-14	1.065814E-14	-0.10
1.00002500	1.5625E-14	1.554312E-14	1.554312E-14	1.576517E-14	0.52
1.00002800	2.1952E-14	2.198242E-14	2.176037E-14	2.198242E-14	-0.14
1.00003100	2.9791E-14	3.019807E-14	3.019807E-14	2.997602E-14	-1.37
1.00003400	3.9304E-14	3.907985E-14	3.907985E-14	3.952394E-14	0.57
1.00003700	5.0653E-14	5.062617E-14	5.062617E-14	5.062617E-14	0.05
1.00004000	6.4E-14	6.439294E-14	6.439294E-14	6.417089E-14	-0.61
1.00004300	7.9507E-14	7.926992E-14	7.904788E-14	7.949197E-14	0.30
1.00004600	9.7336E-14	9.703349E-14	9.725554E-14	9.725554E-14	0.31
1.00004900	1.17649E-13	1.176836E-13	1.176836E-13	1.174616E-13	-0.03
1.00005200	1.40608E-13	1.405542E-13	1.407763E-13	1.407763E-13	0.04
1.00005500	1.66375E-13	1.660894E-13	1.660894E-13	1.660894E-13	0.17
1.00005800	1.95112E-13	1.949552E-13	1.949552E-13	1.953993E-13	0.08
1.00006100	2.26981E-13	2.271516E-13	2.273737E-13	2.269296E-13	-0.08

Precisione della macchina

Si definisce come il più grande numero ε che si può aggiungere a 1 senza modificare il risultato

$$1 + \varepsilon = 1$$

Singola precisione: ε ~ 10^{-7}

Doppia precisione: ε ~ 10^{-16}

	eps	ris	
1	1.00E-14	diverso	=SE(D4=E4+D4,"uguale","diverso")

	eps	ris	
1	1.00E-16	uguale	=SE(D4=E4+D4,"uguale","diverso")

	D	E	F	G	H	I
3		eps	ris			
4	1	1.00E-14	diverso	=SE(D4=E4+D4,"uguale","diverso")		
5						
6	9.99E-01					
7		= ((1-E4) - 1)/E4				

Errori di arrotondamento nel calcolo numerico: Fatti

Durante la guerra del Golfo, il 25 Febbraio del 1991, un missile americano da difesa Patriot lasciò passare un missile Scud nemico. Il missile Scud colpì un accampamento militare uccidendo 28 soldati. Il problema fu individuato nel calcolo della differenza tra due numeri in virgola mobile ottenuti convertendo e scalando un numero intero di un registro di temporizzazione. (google: failure + arithmetic + errors + patriot)

Il clock interno (tempo in decimi di secondo) è codificato in virgola fissa usando 24 bit:

$0.1^{dec} = 0.0001100110011001100110011001100\ldots$

Usando 24 bit il clock del sistema misura intervalli di $0.000\ 1100\ 1100\ 1100\ 1100\ 1100^{bin}$ decimi di secondo con un errore:

$\Delta = 0.000\ 0000\ 0000\ 0000\ 0000\ 0000\ 1100\ 1100\ldots^{bin} \sim$

$\Delta \sim 0.000000095^{dec}$ $(9.5 \cdot 10^{-8})$ decimi di secondo.

Il tempo è calcolato dall'accensione delle strumento accumulando i cicli di lettura e l'errore si accumula.

In 100 ore l'errore totale è:

$\Delta_{tot} = 0.000000095 * 3600000 = 0.34$ s

La velocità del missile è 1676 m s^{-1}, in 0.34 s il missile da intercettare ha percorso ~570 m e non era piú intercettabile !!!

Fatti

Il 4 Giugno 1996 venne lanciato il primo razzo Ariane 5. Tutto andò bene per 36 secondi. Poi improvvisamente il razzo deviò dalla sua traiettoria e si autodistrusse. Il problema risiedeva nel Sistema di Riferimento Inerziale che generò un "operation exception" convertendo un numero in doppia precisione a 64 bit in un intero a 16 bit. Il numero risultó maggiore di 32767 (215-1) Ciò produsse l'invio di un messaggio errato di diagnostica al computer di bordo che lo interpretò come un dato relativo alle condizioni di volo..

Verso il 1982, la borsa di Vancouver introdusse un nuovo indice inizializzandolo al valore di: 1 000.000 (tre cifre decimali). L'indice veniva aggiornato dopo ogni transizione condotta sulla piazza di Vancouver. L'indice era basato sulla quotazione di circe 1.400 titoli cosicché in media il valore dell'indice veniva ricalcolato circa 3.000 volte al giorno sempre con una precisione di tre cifre decimali. Dopo 22 mesi di attività, l'indice raggiunse quota: 520. La causa di tutto ciò non era dovuta ad un periodo di recessione economica, bensì al fatto che ogni nuovo valore calcolato veniva troncato alla terza cifra decimale anziché essere arrotondato. Ad esempio il valore effettivo: 540.32567 veniva troncato a 540.325 anziché essere arrotondato a 540.326. Così facendo l'indice perdeva in media 1 punto al giorno quindi circa 20 punti al mese. La sostituzione dell'operazione di troncamento con quella di arrotondamento portò direttamente l'indice ad assumere il valore corretto: 1098.892.

Codifica dei caratteri

Un computer lavora con numeri in codice binario. Per trattare lettere e caratteri assegna ad ogni simbolo un diverso codice binario.

52 caratteri alfabetici	A, B, ... a, b...
10 cifre	0, 1, 2, ...
9 segni di punteggiatura	. , ; : ? ! ' " *spazio*
8 operazioni e simboli matematici	+ - * / ^ = < >
6 parentesi	( ) { } [ ]
altri	# $ % & @ | ~ \ _

Servono circa 100 simboli !

Quanti bit di informazione servono per codificare i caratteri in notazione binaria?

La quantità di informazione contenuta in un simbolo alfanumerico è:

$\log_2 100 = 6.64 \sim 7$

Utilizzando 7 bit è possibile codificare
2^7=128 simboli diversi

44

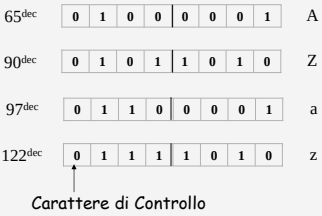
Il Codice ASCII

American Standard Code for Information Interchange

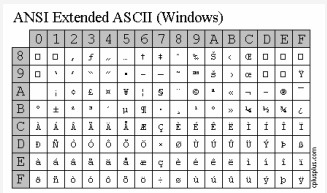
Il codice ASCII standard codifica 128 caratteri (cifre, lettere, simboli, caratteri di controllo) usando un codice a **8 bit**

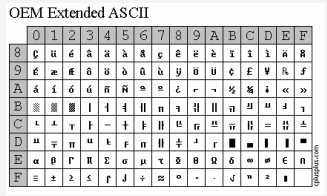
Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

C. Meneghini - C.I.-Informatica 40



Il bit piú significativo puó essere usato come "bit di controllo" o per codificare altri 128 caratteri. Si ha cosi il codice ASCII esteso in cui i primi 128 simboli (0-127) corrispondono al codice ASCII standard e gli altri 128 (128-255) simboli aggiuntivi (lettere accentate, simboli tecnici, etc...)





C. Meneghini - C.I.-Informatica 42

Problema: come rappresentare alfabeti estesi quali l'alfabeto Cirillico o gli ideogrammi cinesi? Come includere simboli speciali per la matematica, l'algebra, la fisica delle particelle, etc...?

UNICODE

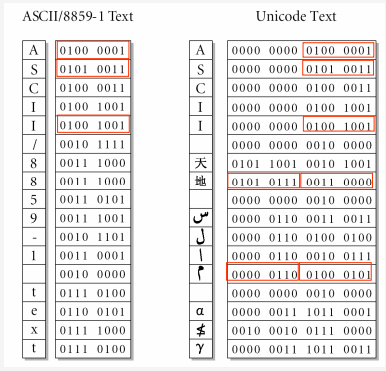
Unicode assegna un numero univoco a ogni carattere, indipendentemente dalla piattaforma (S.O.), indipendentemente dall'applicazione, indipendentemente dalla lingua
www.unicode.org

UNICODE è un codice universale disponibile in tre formati:

32-bit (UTF-32): 2^{32} = 4294967296 simboli;

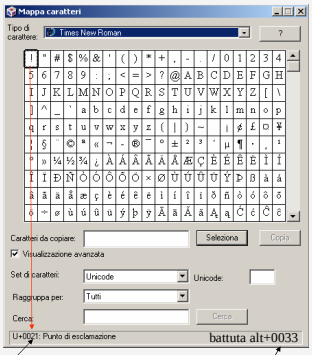
16-bit (UTF-16): 2^{16} = 65536 simboli;

8-bit (UTF-8) = 256 simboli, compatibile con il codice ASCII



from: www.unicode.org/versions/Unicode4.1.0

Come trovare i caratteri sul PC



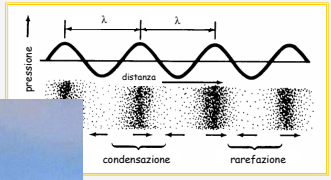
HEX $0021^{hex} = 33^{dec}$ dec.

® U+0174 battuta alt+0174

~ U+007E battuta alt+0126

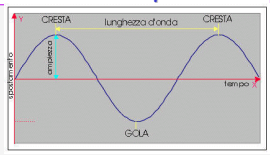
é U+00E9 battuta alt+0233

Codifica dei suoni: audio digitale

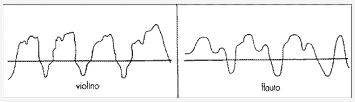


Caratteristiche dei suoni

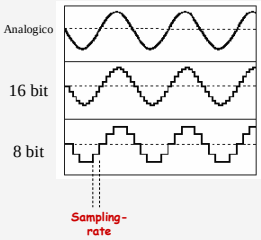
- Intensità: dipende dall'ampiezza
- Altezza: determinata dalla frequenza



- Timbro: qualità del suono

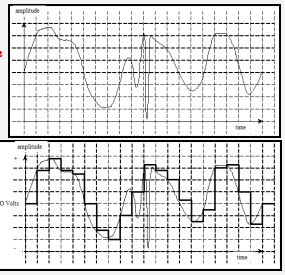


Audio digitale



16 bit 44.1 kHz

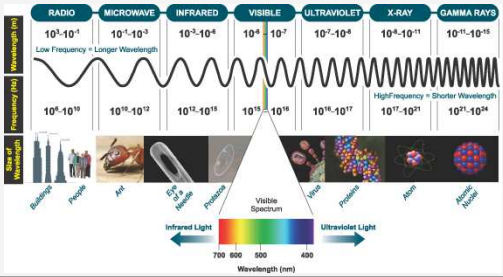
bit-rate



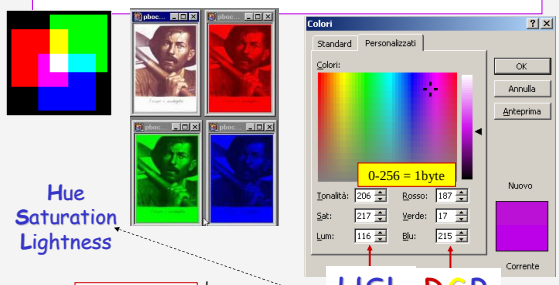
Codifica delle immagini: RGB

Luce visibile: regione dello spettro elettromagnetico compresa tra 400 e 700 nm

$c = \lambda \nu$ $\lambda = c / \nu$ $\nu = c / \lambda$



Sintesi additiva dei colori RGB



0-256 = 1byte

Hue Saturation Lightness

HSL RGB

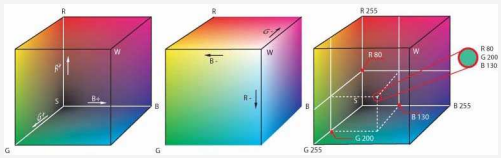
187.017.215 dec.

1011 1011. 0001 0001 . 1101 0111 bin.

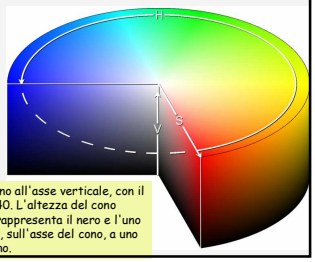
BB . 11 . D7 hex.

codice: 3 byte = 24 bit = 2^{24} = 16 777 216 colori

RGB



HSL, HSB, HSV ... sono sinonimi



Hue Saturation Value
tonalità, saturazione e luminosità

La tonalità H viene misurata da un angolo intorno all'asse verticale, con il rosso a 0 gradi, il verde a 120 e il blu a 240. L'altezza del cono rappresenta la luminosità (B) con lo zero che rappresenta il nero e l'uno il bianco. La saturazione (S) invece va da zero, sull'asse del cono, a uno sulla superficie del cono.