

*Elementi di Informatica*  
( *Lezione II, parte I* )  
**Sistemi di numerazione:**  
*binario, ottale ed esadecimale*

*Il sistema di numerazione  
posizionale decimale*

Nella numerazione posizionale ogni cifra del numero assume un valore in funzione della “*posizione*”:

221 in notazione compatta, cioè

$$2 \times 100 + 2 \times 10 + 1 \times 1$$

ovvero, con la notazione esplicita

$$2 \times 10^2 + 2 \times 10^1 + 1 \times 10^0$$

## *Sistema posizionale*

- Ogni numero si esprime come la somma dei prodotti di ciascuna cifra per la base elevata all'esponente che rappresenta la *posizione* della cifra:

$$221 = 2 \times 10^2 + 2 \times 10^1 + 1 \times 10^0$$

## *Sistema posizionale (cont.)*

- La *notazione posizionale* può essere usata con qualunque base creando così differenti sistemi di numerazione.
  - Per ogni base di numerazione si utilizza un numero di cifre uguale alla base.
- In informatica si utilizza prevalentemente la numerazione:
  - binaria,
  - ottale,
  - esadecimale.
- Il sistema di numerazione romano non è posizionale:
  - Ad esempio, XIII vs. CXII.

## *Sistema di numerazione decimale*

➤ La numerazione *decimale* utilizza una notazione posizionale basata su 10 cifre (da 0 a 9) e sulle potenze di 10.

- Il numero 234 può essere rappresentato esplicitamente come:

$$2 \times 10^2 + 3 \times 10^1 + 4 \times 10^0$$

## *Sistema di numerazione binario*

➤ Il sistema di numerazione *binario* utilizza una notazione posizionale basata su 2 cifre (0 e 1) e sulle potenze di 2.

- Il numero 1001 può essere rappresentato esplicitamente come:

$$\begin{aligned} 1001_2 &= 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \\ &= 9_{10} \end{aligned}$$

## *Sistema di numerazione ottale*

➤ Il sistema di numerazione *ottale* utilizza una notazione posizionale basata su 8 cifre (da 0 a 7) e sulle potenze di 8.

- Il numero 534 può essere rappresentato esplicitamente come:

$$534_8 = 5 \times 8^2 + 3 \times 8^1 + 4 \times 8^0 = 348_{10}$$

## *Sistema di numerazione esadecimale*

➤ La numerazione *esadecimale* utilizza una notazione posizionale basata su 16 cifre (da 0 a 9 ed i caratteri A, B, C, D, E, F) e sulle potenze di 16.

- Il numero **B7FC**<sub>16</sub> può essere rappresentato esplicitamente come:

$$\begin{aligned} (11) \times 16^3 + 7 \times 16^2 + (15) \times 16^1 + (12) \times 16^0 \\ = 47100_{10} \end{aligned}$$

## *Conversione da base n a base 10*

- Per convertire un numero da una qualunque base alla base 10 è sufficiente rappresentarlo esplicitamente:

$$1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13_{10}$$

$$710_8 = 7 \times 8^2 + 1 \times 8^1 + 0 \times 8^0 = 456_{10}$$

$$A51_{16} = (10) \times 16^2 + 5 \times 16^1 + 1 \times 16^0 = 2641_{10}$$

## *Conversione da base 10 a base n*

- Per convertire un numero ad una base  $n$  qualsiasi occorre trovare tutti i resti delle successive divisioni del numero per la base  $n$ .
- Come esempio si vuole trovare il valore binario del numero **210**
    - Basterà dividere 210 per la base 2,
    - ...

## *Conversione da base 10 a base 2*

210	2	resto	0
105	2		1
52	2		0
26	2		0
13	2		1
6	2		0
3	2		1
1	2		1



➤ Leggendo la sequenza dei resti dal basso verso l'alto, si ottiene il numero:

**11010010<sub>2</sub>**

## *Verifica di correttezza*

➤ Per una verifica di correttezza basta riconvertire il risultato alla base 10:

$$\begin{aligned} 11010010_2 &= 1 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + \\ &\quad 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + \\ &\quad 1 \times 2^1 + 0 \times 2^0 = 210_{10} \end{aligned}$$

## *Costruzione dei numeri binari*

➤ Per costruire la successione dei numeri binari si può seguire il seguente schema:

0	0	0	0	=	0
0	0	0	1	=	1
0	0	1	0	=	2
0	0	1	1	=	3
0	1	0	0	=	4
0	1	0	1	=	5
0	1	1	0	=	6
0	1	1	1	=	7

## *I primi 32 numeri binari*

0	0	0	0	0	0	0	0	=	0
0	0	0	0	0	0	0	1	=	1
0	0	0	0	0	0	1	0	=	2
0	0	0	0	0	0	1	1	=	3
0	0	0	0	0	1	0	0	=	4
0	0	0	0	0	1	0	1	=	5
0	0	0	0	0	1	1	0	=	6
0	0	0	0	0	1	1	1	=	7

0	0	0	0	1	0	0	0	=	8
0	0	0	0	1	0	0	1	=	9
0	0	0	0	1	0	1	0	=	10
0	0	0	0	1	0	1	1	=	11
0	0	0	0	1	1	0	0	=	12
0	0	0	0	1	1	0	1	=	13
0	0	0	0	1	1	1	0	=	14
0	0	0	0	1	1	1	1	=	15

## *I primi 32 numeri binari (cont.)*

0 0 0 1 0 0 0 0 = 16
0 0 0 1 0 0 0 1 = 17
0 0 0 1 0 0 1 0 = 18
0 0 0 1 0 0 1 1 = 19
0 0 0 1 0 1 0 0 = 20
0 0 0 1 0 1 0 1 = 21
0 0 0 1 0 1 1 0 = 22
0 0 0 1 0 1 1 1 = 23

0 0 0 1 1 0 0 0 = 24
0 0 0 1 1 0 0 1 = 25
0 0 0 1 1 0 1 0 = 26
0 0 0 1 1 0 1 1 = 27
0 0 0 1 1 1 0 0 = 28
0 0 0 1 1 1 0 1 = 29
0 0 0 1 1 1 1 0 = 30
0 0 0 1 1 1 1 1 = 31

## *Operazioni binarie*

$$\begin{array}{r} 10110101+ \\ 1000110 = \\ \hline 11111011 \end{array}$$

$$\begin{array}{r} 00110011+ \\ 00111000 = \\ \hline 01101011 \end{array}$$



## *Operazioni binarie (cont.)*

1101 x	10011 x
11 =	10 =
<hr/>	<hr/>
1101	00000
1101	10011
<hr/>	<hr/>
100111	100110

## *Esercizi*

Eseguire le seguenti operazioni direttamente in binario, convertire in decimale e verificare il risultato:

- $110000 + 1001010$ ;
- $1001010 + 1111111 + 10$ ;
- $100110 \times 111100$ ;
- $001001 \times 111$ .

*Fine*

*Elementi di Informatica*

*( Lezione II , parte II )*

*Analogico vs. Digitale*

## *I segnali per comunicare*

➤ ***ANALOGICO***

➤ ***DIGITALE***

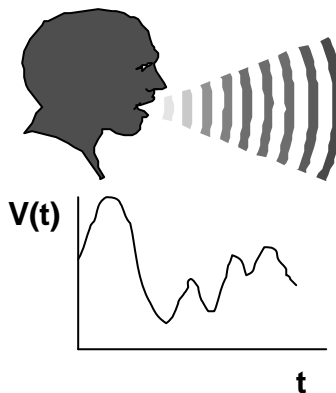
Gli esseri umani ed i computer utilizzano differenti tipi di segnali per comunicare.

## *Informazione analogica*

La voce umana e la trasmissione dei segnali di radio e televisione sono comunicazioni di tipo

***ANALOGICO***

dove le grandezze fisiche sono funzioni continue nel tempo e possono assumere *infiniti* valori.



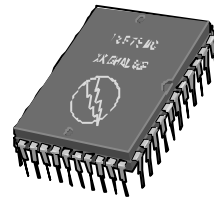
# *Informazione digitale*

La trasmissione dei segnali nei computer ed in genere nei circuiti elettronici avviene in modo

## ***DIGITALE***

poiché le grandezze fisiche sono rappresentate da *stati discreti*.

- Nei circuiti di memoria di un computer lo 0 viaggia come un segnale a basso voltaggio e spegne gli interruttori (transistor), al contrario l'1 viaggia ad alto voltaggio e li accende.

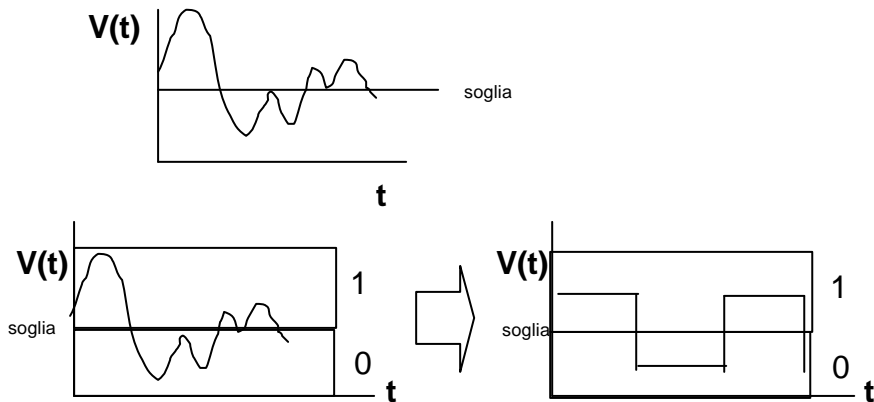


# *Digitalizzazione dei segnali*

➤ I segnali elettrici continui (analogici) vengono convertiti in segnali digitali.

➤ La conversione comporta un certo grado di approssimazione.

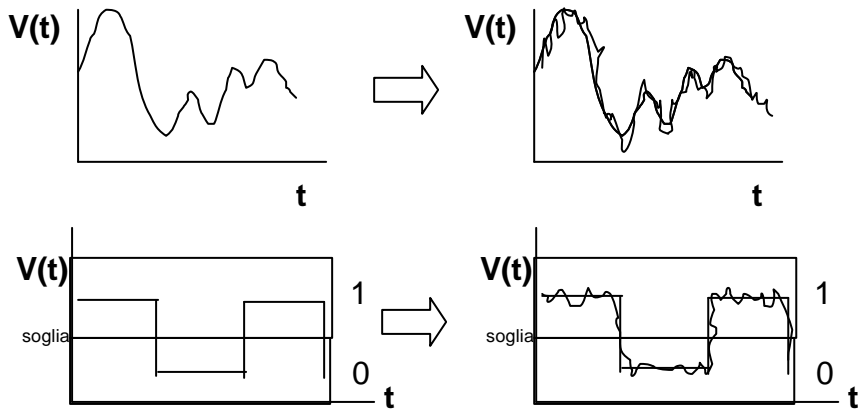
## *Da Analogico a Digitale*



## *Precisione dei segnali*

- I segnali digitali sono meno affetti da disturbi di trasmissione.
- La minore sensibilità al rumore consente di replicare perfettamente il segnale.

## *Precisione dei segnali (cont.)*



## *La rappresentazione delle informazioni*

- Tutte le informazioni sono rappresentate in forma binaria o digitale utilizzando due soli simboli: 0 ed 1.
- Con una cifra binaria si possono quindi rappresentare soltanto due informazioni.

## *La rappresentazione delle informazioni (cont.)*

- Le ragioni di questa scelta sono prevalentemente di tipo tecnologico:
  - Due possibili stati di polarizzazione di una sostanza magnetizzabile;
  - Passaggio/non passaggio di corrente attraverso un conduttore;
  - Passaggio/non passaggio della luce attraverso una fibra ottica.

## *Il bit*

- Unità fisica di informazione che vale 0 oppure 1.

- Il nome proviene da Binary Digit.

- Si utilizzano i multipli del bit:

- |        |    |          |                  |              |
|--------|----|----------|------------------|--------------|
| • Kilo | Kb | $2^{10}$ | ~ un migliaio    | (1024)       |
| • Mega | Mb | $2^{20}$ | ~ un milione     | (1024x1024)  |
| • Giga | Gb | $2^{30}$ | ~ un miliardo    | (1Mb x 1024) |
| • Tera | Tb | $2^{40}$ | ~ mille miliardi | (1Gb x 1024) |

*Fine*

*Elementi di Informatica*

*( Lezione II , parte III )*

*Rappresentazione delle informazioni :*

*La codifica dei testi*



## *Rappresentazione dei caratteri*

- Cos'è un carattere ?
  - Si tratta di un simbolo, in qualche modo astratto.
    - Per esempio una “A” è la rappresentazione grafica convenzionale (detta anche *glifo*) del concetto di carattere “a maiuscola”.
- Dobbiamo trovare una “*convenzione*” con cui realizzare una rappresentazione comprensibile al computer dei caratteri.

## *Codifica binaria*

- Per poter rappresentare le informazioni è necessario utilizzare *sequenze* di bit.
  - Utilizzando due bit si possono rappresentare quattro informazioni diverse:  
00    01    10    11
- Il processo che fa corrispondere ad una informazione una configurazione di bit prende il nome di *codifica dell'informazione*.

## *Sequenze di bit*

Numero di bit nella sequenza	Informazioni rappresentabili
2	4
3	8
4	16
5	32
6	64
7	128
8	256

In generale, con  $n$  bit si possono rappresentare  $2^n$  differenti informazioni.

## *Il byte*

➤ Un gruppo di 8 bit viene denominato Byte.

- Corrisponde ad un carattere.
- Unità di misura della capacità di memoria.

➤ Si utilizzano i multipli del Byte:

- Kilo     KB      $2^{10}$      ~ un migliaio     (1024)
- Mega    MB      $2^{20}$      ~ un milione     (1024x1024)
- Giga    GB      $2^{30}$      ~ un miliardo    (1MBx1024)
- Tera    TB      $2^{40}$      ~ mille miliardi (1GBx1024)

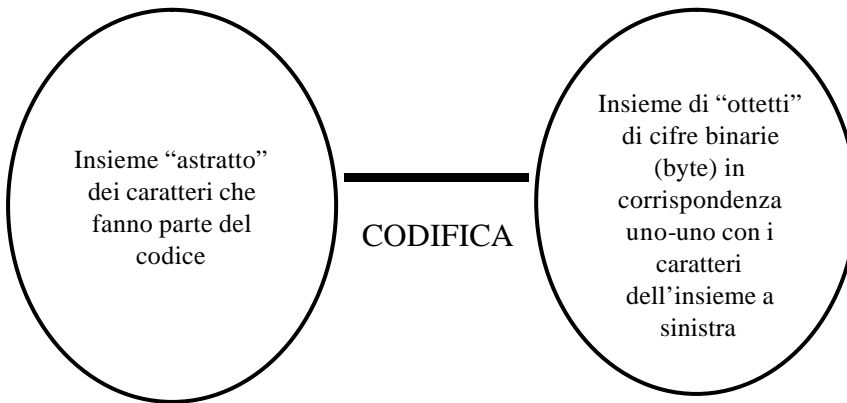
## *I caratteri utilizzati nella comunicazione scritta*

- 52 lettere alfabetiche maiuscole e minuscole
- 10 cifre (0, 1, 2, ..., 9)
- Segni di punteggiatura (, . ; : ! " ' ? ^ \ ...)
- Segni matematici (+, -, ×, ±, {, [, >, ...)
- Caratteri nazionali (à, è, ì, ò, ù, ç, ñ, ö, ...)
- Altri segni grafici (©, ←, ↑, Ⓜ, @, € ...)
- In totale 220 caratteri circa.

## *Codice*

- Si pone quindi la necessità di codificare in numeri binari almeno 220 caratteri.
- La sequenza di bit necessaria a rappresentare 220 simboli deve essere composta da 8 bit e prende il nome di **CODICE**.

## *La fase di codifica*



## *Il codice ASCII*

.....			0011 0000	48	0
0100 0001	65	A	0011 0001	49	1
0100 0010	66	B	0011 0010	50	2
0100 0011	67	C	0011 0011	51	3
.....			.....		
0101 1000	88	X	0011 1010	58	:
0101 1001	89	Y	0011 1011	59	;
0101 1010	90	Z	0011 1100	60	<
.....			0011 1101	61	=
0110 0001	97	a	.....		
0110 0010	98	b	1010 0100	164	ñ
0110 0011	99	c	1000 0111	135	ç

American  
Standard Code  
for Information  
Interchange

## *Il codice ASCII (cont.)*

➤ I caratteri ASCII da 0 a 127:

- I primi 32 (numerati da 0 fino a 31) sono “caratteri di controllo” *non stampabili*,
- I successivi 95 simboli (numerati da 32 fino a 126) sono caratteri *stampabili*,
- Il 128-esimo simbolo è ancora un “carattere di controllo” *non stampabile*.

## *Dove trovo le tabelle ASCII ?*

➤ Andate su un motore di ricerca (per esempio [www.google.it](http://www.google.it) ) e digitate “tabella ASCII”.

- Un ottimo sito in italiano è:
  - <http://www.cesit.unifi.it/online/principi/asciistd.html>
- Un buon sito in inglese è invece:
  - <http://www.ibilce.unesp.br/courseware/datas/data1.htm>

Tabella ASCII in notazione binaria: la concatenazione del 'nibble' di riga e di quello di colonna dà il codice ASCII in binario.

	0000 0	0001 1	0010 2	0011 3	0100 4	0101 5	0110 6	0111 7	1000 8	1001 9	1010 10	1011 11	1100 12	1101 13	1110 14	1111 15
0000 0																
0001 1																
0010 2	spazio	!	“	#	\$	%	&	‘	( )	*	+	,	-	.	/	
0011 3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0100 4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0101 5	P	Q	R	S	T	U	V	W	X	Y	Z	[ \ ]	^	_		
0110 6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0111 7	p	q	r	s	t	u	v	w	x	y	z	{   }	~			

Tabella ASCII in notazione decimale: la somma dell'indice di riga e di colonna dà il codice ASCII in decimale.

+	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0																
16																
32	spazio	!	“	#	\$	%	&	‘	( )	*	+	,	-	.	/	
48	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
64	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y	Z	[ \ ]	^	_		
96	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
112	p	q	r	s	t	u	v	w	x	y	z	{   }	~			

## *Sequenze di caratteri ASCII*

Dividendo la sequenza in gruppi di byte è possibile risalire ai singoli caratteri:

**01101001 01101100 00100000 01010000 01001111 00101110**  
**01101001 01101100 00100000 01010000 01001111 00101110**  
**i l P O .**

## *Esempi di sequenze*

➤ “*Computer*” in ASCII diventa:

- C=67=01000011,                      o=111=01101111,  
m=109=001101101,                  p=112=01110000,  
u=117=01110101,                    t=116=01110100,  
e=101=01100101,                    r=114=01110010.

- 01000011- 01101111- 01101101- 11100000-01110101-01110100-01100101- 01110010

➤ **Esercizio :**

- *Scrivere “ASCII” in decimale ed in binario.*

## *Numeri e codice ASCII*

- Con il codice ASCII è possibile rappresentare i numeri come sequenza di caratteri. Ad esempio il numero 234 sarà rappresentato come:

**00110010 00110011 00110100**

**2            3            4**

- Con questo tipo di rappresentazione *non* è possibile effettuare operazioni aritmetiche.

## *Rappresentazione di dati alfabetici*

Codifiche standard:

- **ASCII**, 8 bit per carattere, rappresenta 256 caratteri.
- **UNICODE**, 16 bit per carattere
  - ASCII e caratteri etnici ( $2^{16} = 65.536$  simboli).

➤ Codifiche proprietarie:

- **MSWindows**, 16 bit per carattere
  - simile ad UNICODE.



## *Dieci dita e qualche tasto...*

### ➤ La mia tastiera ha meno di cento tasti.

- Come ottenere tutti i simboli desiderati?
  - Usando combinazioni di tasti.
    - ❖ Per esempio, <Shift><tasto> dà la versione maiuscola.
  - Digitando la combinazione :
    - ❖ <Alt><codice ASCII in notazione decimale>
      - In questo caso bisogna usare il *tastierino numerico* per inserire il codice !

## *Rappresentare “testi” nel computer*

### ➤ Cos'è un testo ?

- una sequenza ordinata di “*caratteri*” .
  - Esempio : abgx76 6&&&%%"0??
- una sequenza ordinata di caratteri *formattati*.
  - Esempio : **AA***x***b**()... **BB**

### ➤ È importante notare che, a parte la superficiale similarità, i due concetti definiti sopra sono differenti.

- Per questo richiedono rappresentazioni diverse.

*Testi formattati*

ciao ciao ciao *ciao* ciao χίαο ciao  
ciao ciao ciao ci ao

Sottolineato

*Corsivo*

**Grassetto**

*Grassetto corsivo*

*Fine*