

Informatica

Dati e sistema binario

Thomas Barbera
thomas.barbera@unimib.it

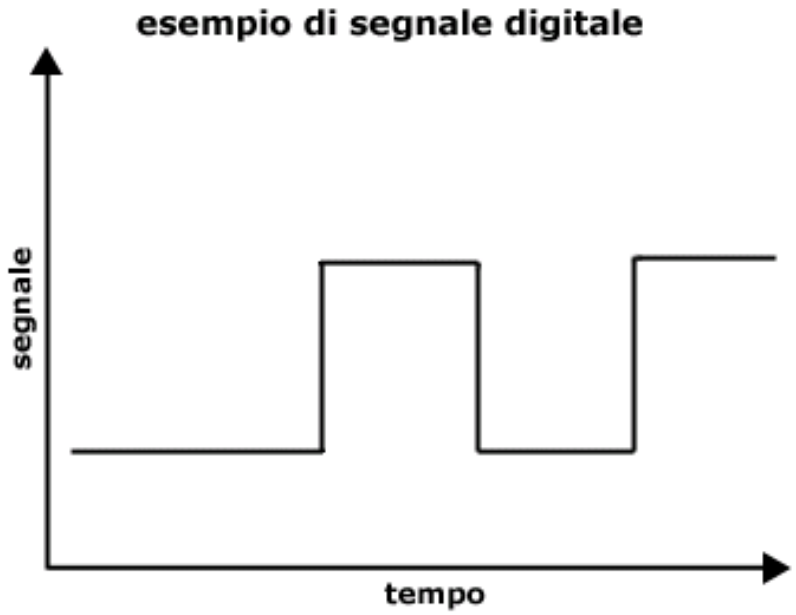
Riadattamento delle slides di Marco Viviani e Simone Zini





Sistemi analogici e digitali





Sistemi analogici e digitali



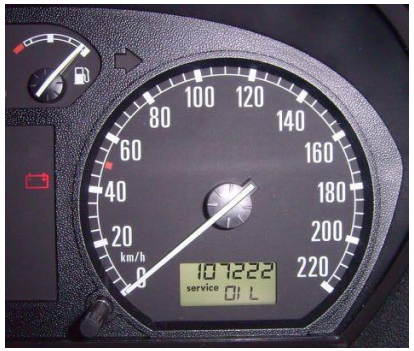
Analogici



Digitali



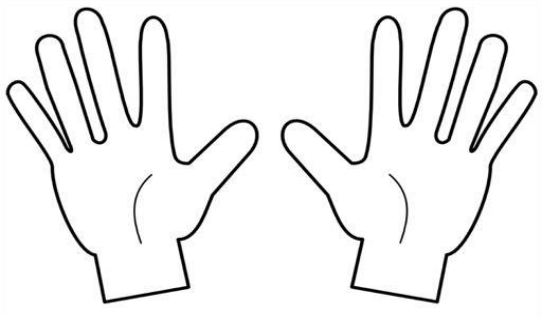








I sistemi digitali e l'informatica



- La parola **digitale** deriva dal latino *digitālis*, da *digitus* (dito)
  - *Digit*: cifra/dito in inglese
- In informatica la rappresentazione dei dati è digitale perché essi vengono rappresentati in **forma numerica**
- La **rappresentazione digitale** ha consentito la realizzazione di calcolatori moderni

I computer e il sistema binario

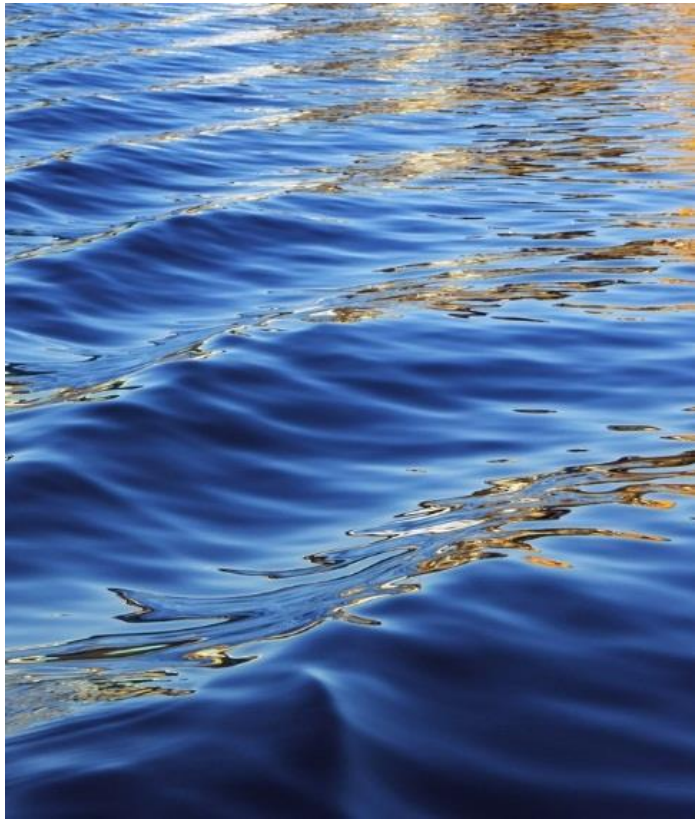
- Nei computer **tutto è rappresentato da un segnale elettrico binario** che può «esistere» solo in due stati fisici ben definiti:
  - **acceso** (nel caso si registri un passaggio di corrente elettrica)
  - **spento** (se non si registra il passaggio di corrente elettrica)

- Questi stati fisici sono rappresentati nel linguaggio del computer attraverso un **sistema binario** che consente al processore di un qualunque sistema informatico di eseguire le operazioni alla base dei moderni software e applicativi

- Il sistema binario è un **sistema di numerazione**



I sistemi di numerazione



Sistemi di numerazione

- Per **sistema di numerazione** si intende un insieme di **simboli** di rappresentazione dei numeri e di **regole** per contare ed eseguire operazioni

- I simboli possono essere **cifre** (Es., 0,1,2,…,9) e/o **lettere** (Es., a, b, c, …, z)
  - Sistema alfabetico, numerico o alfanumerico
  - esempi di codici possono essere il sistema di numerazione arabo, quello romano, la segnaletica stradale, etc.

- Con apposite convenzioni è possibile tradurre un codice numerico in codice alfabetico e viceversa
  - Convenzione: ordine alfabetico A, B, C, D, …, Z
  - C-I-A-O => 3-9-1-13

Il sistema di numerazione decimale

- Nella vita quotidiana utilizziamo solitamente un **sistema di numerazione decimale**, che utilizza un nome individuale per
  - i numeri inferiori o uguali a dieci: zero, uno, due, tre, …, dieci
  - i multipli di 10: venti, trenta, quaranta, …
  - certe potenze di 10: cento, mille, un milione, …

  mentre i nomi dei numeri intermedi sono parole composte a partire dalle precedenti, seguendo un principio additivo o moltiplicativo

- Il nostri sistema necessita di 10 diversi segni (cifre):

0 1 2 3 4 5 6 7 8 9

- L'adozione di 10 simboli è stata verosimilmente imposta dall'anatomia delle mani, perché sulle dieci dita l'uomo ha imparato a contare

Il sistema di numerazione binario

- Il **sistema di numerazione binario** utilizza solo due simboli, invece delle dieci cifre utilizzate dal sistema di numerazione decimale

0 1

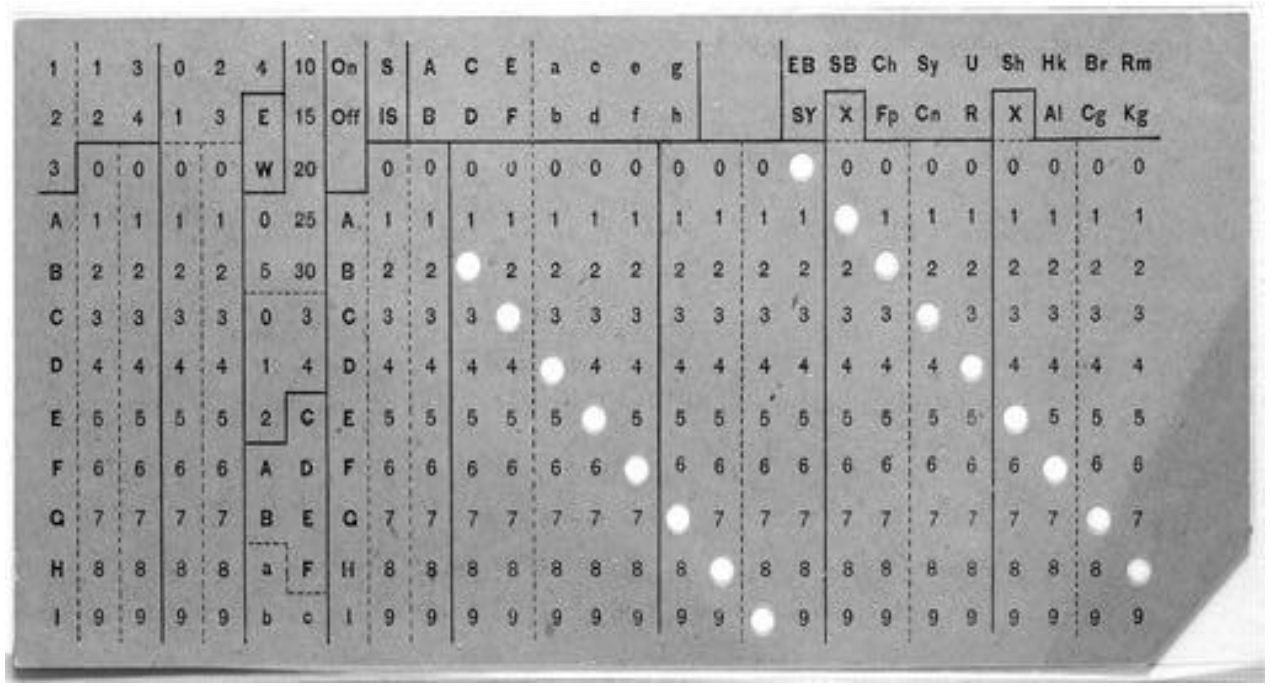
- Ciascuno dei numeri espressi nel sistema numerico binario è definito «numero binario»

- I circuiti elettronici possono produrre un solo tipo di segnale: la tensione. Le informazioni che il segnale può produrre sono quindi 2:
  - **segnale presente**, cioè presenza di tensione (per convenzione indicato con **1**)
  - **segnale assente** (per convenzione indicato con **0**).

- Basterà un impulso elettrico per trasmettere un'informazione. 0 e 1 sono le uniche due «lettere» dell'alfabeto che un computer può comprendere. Questo giustifica l'utilizzo del codice binario da parte dei dispositivi elettronici

Una parentesi storica: la scheda perforata

- L'informazione era rappresentata attraverso la presenza o l'assenza di fori in posizioni predefinite della scheda (**due possibili stati**: **1** e **0**)





Sistema di numerazione posizionale

- Un sistema di numerazione si dice **posizionale** se i simboli (cifre) usati per scrivere i numeri assumono valori diversi a seconda della posizione che occupano nella notazione

- Ad esempio nel sistema decimale: spostandosi a sinistra il peso della cifra aumenta di 10 volte:
  - Esempio: 123 = 100 +20 +3

- Un sistema non posizionale è, ad esempio, quello degli antichi numeri romani:
  - Esempio: XVIII = 18, XIX = 19, …

La base di un sistema numerico

- La **base** di un sistema di numerazione posizionale è il numero di cifre distinte, inclusa quella per lo 0, che il sistema usa per rappresentare i numeri

- Il sistema numerico decimale è un sistema di numerazione posizionale **a base 10** che, per rappresentare i numeri utilizza dieci cifre da 0 a 9
  - 0 1 2 3 4 5 6 7 8 9
  - Per indicare la base si può mettere un **pedice** in fondo al numero
    - Es., 123_{10}

Rappresentazione dei numeri in base *p*

Un numero generico di $\boldsymbol{m}$ cifre è rappresentato dalla sequenza:

- $a_n, a_{n-1}, a_{n-2}, \dots, a_0$

- $n = \boldsymbol{m} - 1$
- a_n: cifra più significativa
- a_0: cifra meno significativa
- $a_i \in \{0,1, \dots, p-1\}$

- 315689 $m = 6$ cifre, base $p = 10$

- $n = m - 1 = 6 - 1 = 5$
- $a_5 = 3$: cifra più significativa
- $a_0 = 9$: cifra meno significativa
- $a_i \in \{0,1, \dots, 9\}$

ATTENZIONE! Si parte dalla cifra a_0!

Rappresentazioni in base *p* in informatica

- Nel campo dell'informatica, con l'avvento dei calcolatori digitali, si usano spesso sistemi di numerazione posizionale in basi che sono potenze di 2:
  - $p = 2$: sistema numerico **binario**
  - $p = 8$: sistema numerico **ottale**
  - $p = 16$: sistema numerico **esadecimale**

- Nel sistema esadecimale, come cifre corrispondenti ai numeri da 10 a 15 si usano le **lettere dell'alfabeto dalla A alla F**

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

Rappresentazione in base 2: i numeri binari

- Base binaria: $p = 2$
  - cifre $a_i \in \{0,1\}$ chiamate **bit** (***binary digit***)

- **8 bit** sono chiamati ***byte***

- Con $\boldsymbol{m}$ cifre in base 2 (cioè con ***m* bit**), posso rappresentare $\boldsymbol{2^m}$ numeri nell'intervallo discreto:

$$[\mathbf{0}, \mathbf{2^m - 1}]$$

- Ad esempio, con $m = 5$ → $[0,2^5 - 1] = [0,31] = 32$ numeri
  - Ricordate che il primo elemento è SEMPRE lo 0!

Esercizio

- Quanti numeri si possono rappresentare nel sistema binario se vengono utilizzati 10 bit?

$$\mathbf{2^m \text{ numeri} \in [0, 2^m - 1]}$$

$$\mathbf{[0, 2^{10} - 1]}$$

$$\mathbf{[0, 1024 - 1]}$$

$$\mathbf{[0, 1023] = 1024}$$

Byte e multipli di byte

- Con un byte (m = 8 bit) posso rappresentare al massimo $2^8 = 256$ numeri
  - Min: $00000000_2 = 0_{10}$
  - Max: $11111111_2 = 255_{10}$

- Alcuni **multipli del *byte*** →

Nome	Simbolo	Multiplo
kilobyte	kB	10^3
megabyte	MB	10^6
gigabyte	GB	10^9
terabyte	TB	10^{12}
petabyte	PB	10^{15}
exabyte	EB	10^{18}
zettabyte	ZB	10^{21}
yottabyte	YB	10^{24}

Utilizzo dei bit per rappresentare un numero binario

- Quando si rappresenta un numero binario è necessario stabilire a priori il numero di bit $\boldsymbol{m}$ che si vuole utilizzare

- **Tutti i bit** saranno utilizzati nella rappresentazione del numero
  - I bit «vuoti» a sinistra saranno «riempiti» da zeri

- Questo implica che **lo stesso numero binario** può avere
  - **Rappresentazioni diverse** a seconda del numero di bit usati

- Esempio: il numero 2_{10} può essere rappresentato:
  - 10_2 se $m = 2$
  - 010_2 se $m = 3$
  - 000000010_2 se $m = 9$

Corrispondenza tra numeri in basi diverse

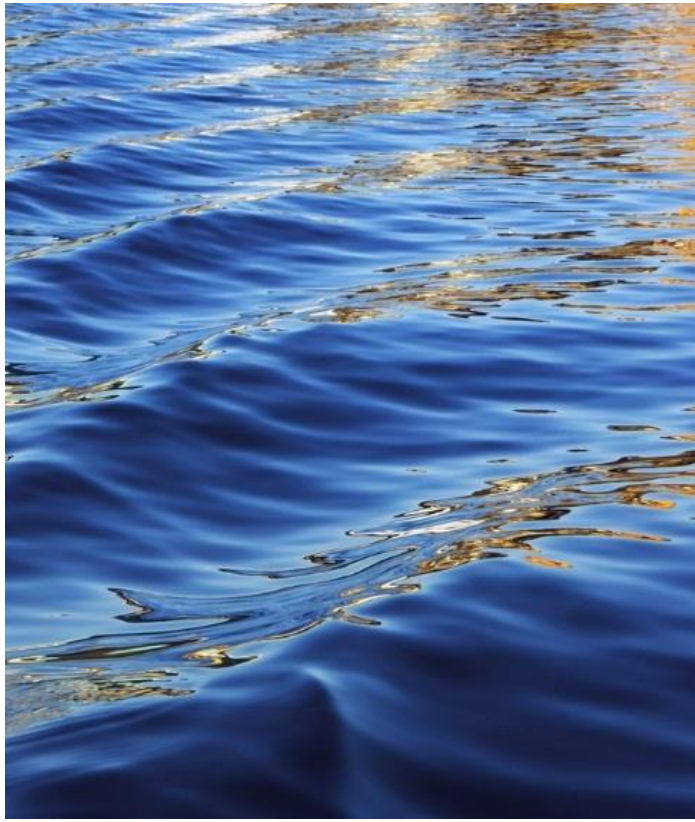
Corrispondenza con i primi 16 numeri in base 10

0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

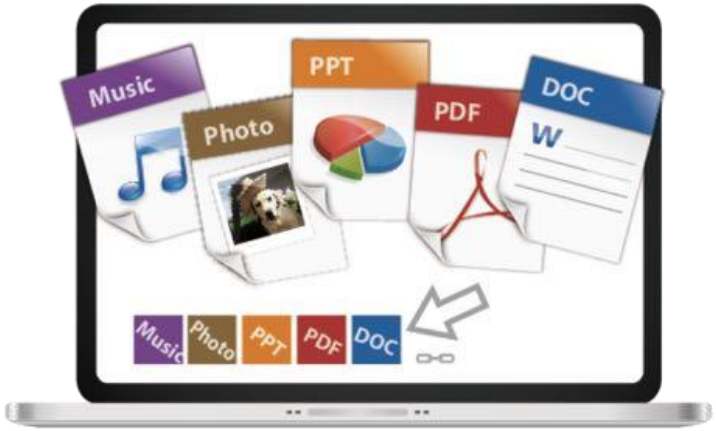
Primi 16 numeri in binario con $m = 4$ bit



Conversioni in basi diverse



Perché sono importanti le conversioni in basi diverse?



- Un calcolatore ha bisogno di rappresentare numeri, lettere, immagini, filmati e suoni attraverso le sole cifre 0 e 1
- Il computer converte ogni numero, simbolo o parola in sistema binario, esegue i calcoli, e riconverte i risultati dal sistema binario a quello decimale per far si che siano comprensibili dall'utente
- In questo modo restituisce immagini, video, audio, e qualsiasi tipo di dato

Scrittura in forma polinomiale e scrittura in una base

- La scrittura posizionale si basa sulla possibilità di scrivere i numeri in **forma polinomiale**

- Ad esempio, nella scrittura decimale (cioè in base 10) il numero 1234 corrisponde a

$$1234 = 1 \text{ migliaio} + 2 \text{ centinaia} + 3 \text{ decine} + 4 \text{ unità} =$$

$$= 1 \times 1000 + 2 \times 100 + 3 \times 10 + 4$$

- Utilizzando le potenze di 10, posso riformularlo come segue:

$$1234 = 1 \times 10^3 + 2 \times 10^2 + 3 \times 10^1 + 4 \times 10^0$$

Forma polinomiale di un numero naturale

- Un numero naturale N, composto da m cifre ($n = m - 1$), in base p, si esprime come

$$N_p = a_n \cdot p^n + a_{n-1} \cdot p^{n-1} + \cdots + a_1 \cdot p^1 + a^0 \cdot p^0 = \sum_{i=0}^{n} a_i \cdot p^i \qquad (1)$$

- Ad esempio:
  - $8587_{10} = 8 \cdot 10^3 + 5 \cdot 10^2 + 8 \cdot 10^1 + 7 \cdot 10^0$

Conversioni: da base 2 a base 10

- Utilizzando la formula (1) nelle slide precedenti, ed utilizzando la base $p = 2$, è possibile trasformare un qualsiasi numero binario in un numero in base 10

- Ad esempio: convertire il numero binario 101011_2 in base 10

$$1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$$

- Vale a dire:
  - $1 \cdot 2^5 = 32$
  - $0 \cdot 2^4 = 0$
  - $1 \cdot 2^3 = 8$
  - $0 \cdot 2^2 = 0$
  - $1 \cdot 2^1 = 2$
  - $1 \cdot 2^0 = 1$

Somma = 43_{10}

$$N_p = a_n \cdot p^n + a_{n-1} \cdot p^{n-1} + \cdots + a_1 \cdot p^1 + a^0 \cdot p^0 = \sum_{i=0}^{n} a_i \cdot p^i$$

Conversioni: da base 8 e 16 a base 10

- **Base ottale**
  - $p = 8$
  - $a_i \in \{0, 1, 2, 3, 4, 5, 6, 7\}$

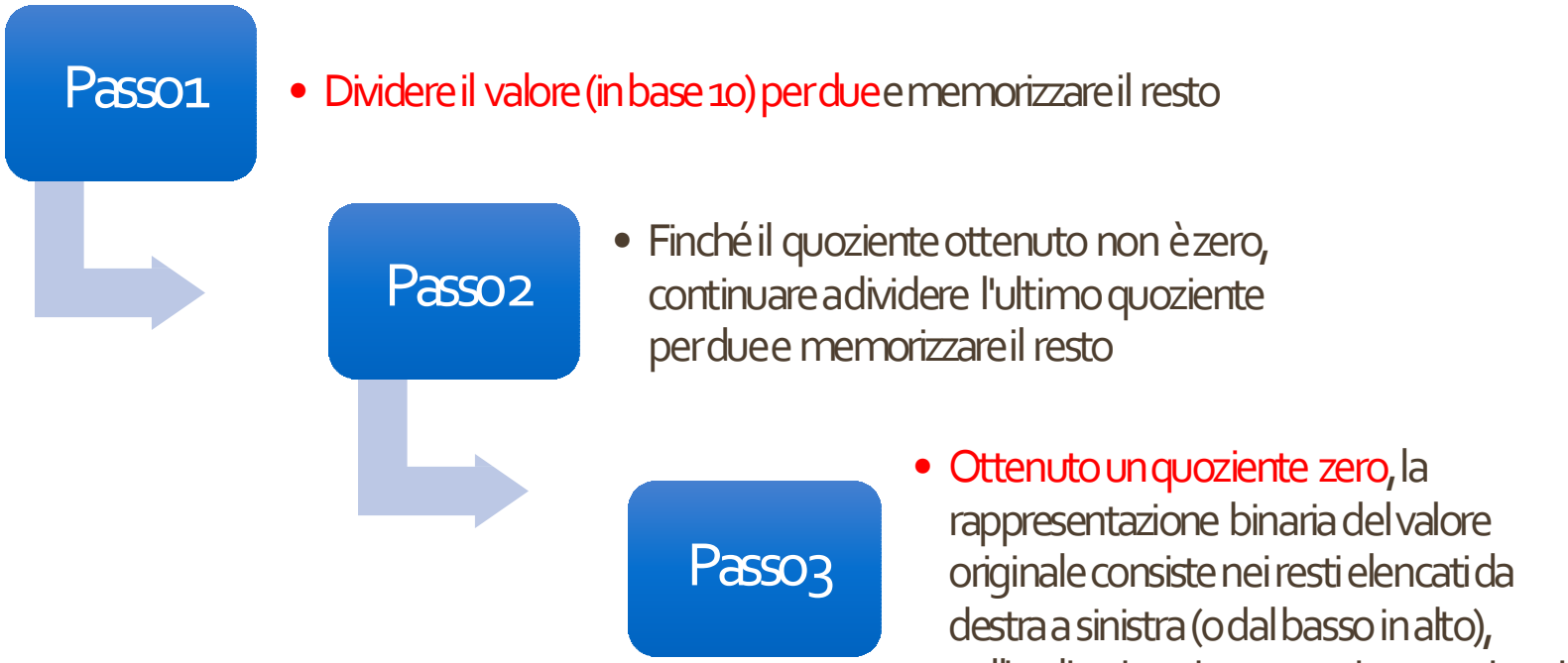
- Esempio: $234_8 = (2 \cdot 8^2 + 3 \cdot 8^1 + 4 \cdot 8^0) = 156_{10}$

- Base esadecimale
  - $p = 16$
  - $a_i \in \{0, 1, 2, 3, 4, 6, 7, 8, 9, A, B, C, D, E, F\}$

- Esempio: $B7F_{16} = (11 \cdot 16^2 + 7 \cdot 16^1 + 15 \cdot 16^0) = 2943_{10}$
  - Notare: «11» al posto di «B» e «15» al posto di «F», i loro equivalenti in base dieci

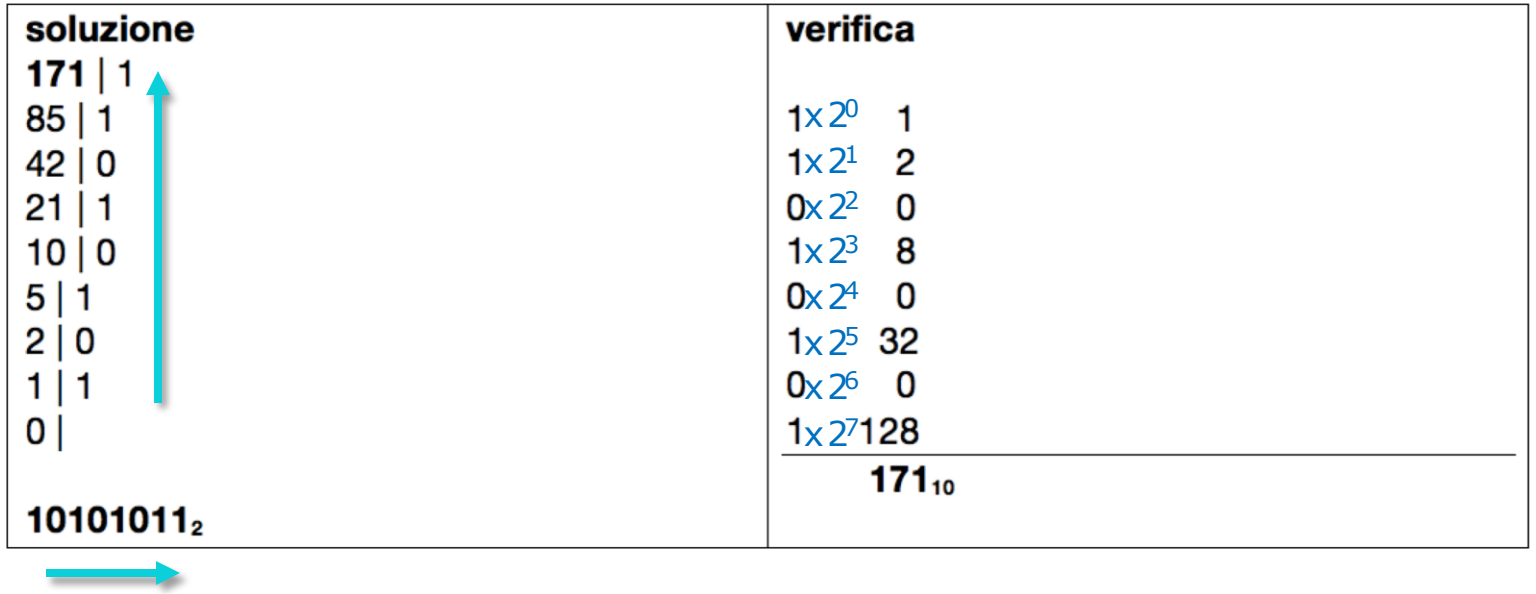
Conversioni: da base 10 a base 2

- **Algoritmo delle divisioni successive**



Conversioni: da base 10 a base 2

- **Algoritmo delle divisioni successive (esempio)**



soluzione	verifica
171 | 1	
85 | 1	1×2^0 1
42 | 0	1×2^1 2
21 | 1	0×2^2 0
10 | 0	1×2^3 8
5 | 1	0×2^4 0
2 | 0	1×2^5 32
1 | 1	0×2^6 0
0 |	1×2^7 128
10101011_2	**171_{10}**

- In informatica è spesso utile sapere solo è il resto della divisione intera, per questo esiste l'operatore **modulo** (verrà visto più in dettaglio nelle esercitazoni)

Esercizio

- Rappresentare il seguente numero in base 10 nel corrispondente numero in base 2

- $137_{10} = ?_2$

- $137 : 2 = 68$ con resto 1
- $68 : 2 = 34$ con resto 0
- $34 : 2 = 17$ con resto 0
- $17 : 2 = 8$ con resto 1
- $8 : 2 = 4$ con resto 0
- $4 : 2 = 2$ con resto 0
- $2 : 2 = 1$ con resto 0
- $1 : 2 = 0$ con resto 1



- resti affiancati invertiti (risultato in binario): 10001001_2

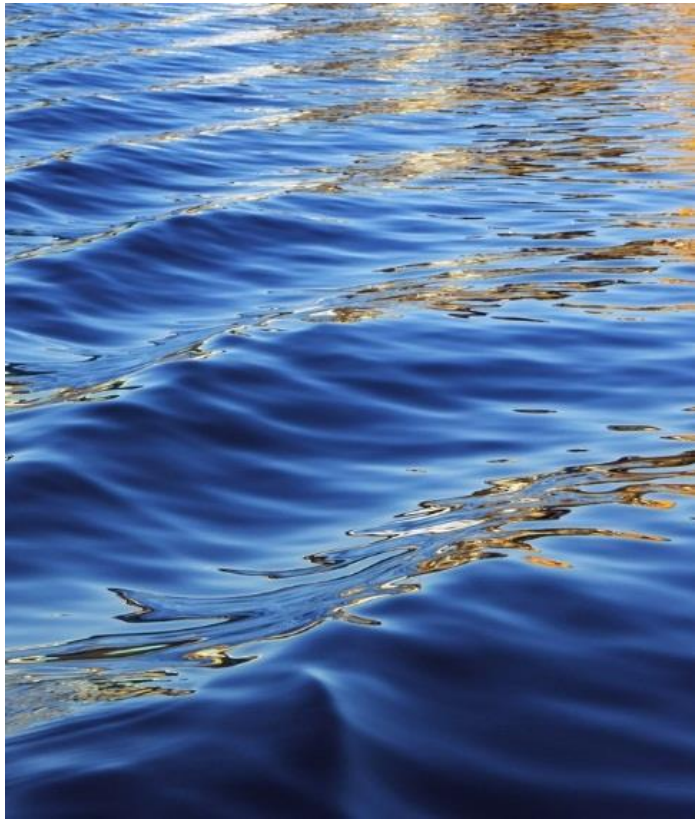
Esercizio

- Verificare che 10001001_2 sia effettivamente corrispondente al numero 137_{10}

- 10001001_2
- $1 \cdot 2^7 + 1 \cdot 2^3 + 1 \cdot 2^0$
- $128 + 8 + 1 = 137$



L’algebra di Boole



L'algebra di Boole

- Formalizzata da **George Boole** nel 1854 per la manipolazione di espressioni logiche attraverso modelli matematici
- **Struttura algebrica** che elabora matematicamente espressioni nell'ambito della logica proposizionale
- Obiettivo: «rappresentare operazioni logiche come operazioni algebriche»
- Utilizzando le **variabili booleane**, si costruiscono le funzioni booleane o logiche, definite attraverso opportune **tavole di verità**

Operatori booleani

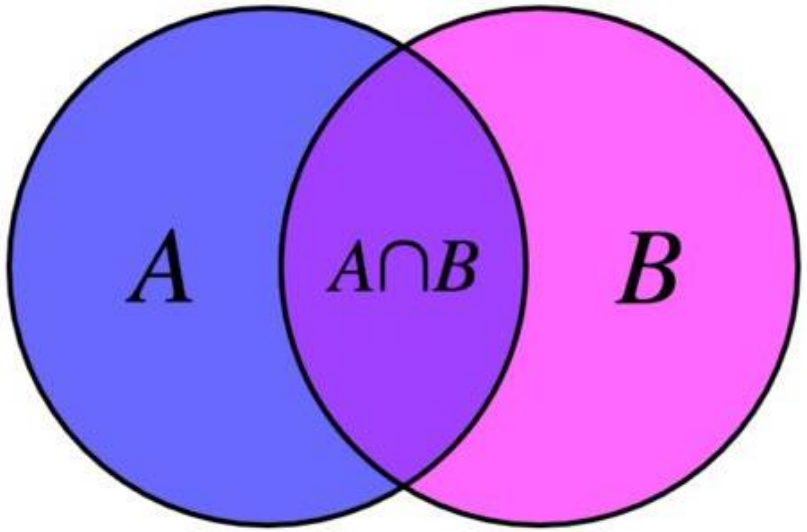
- È basata su tre operatori:

AND, **OR**, **NOT**

- Ogni formula può assumere solo due valori: «**vero**» o «**falso**». Idem per le variabili
- Rappresentiamo «vero» con «1» e «falso» con «0» (es., in Matlab)
- AND e OR sono **operatori binari** (lavorano su due operandi)
- NOT è un **operatore unario** (lavorano su un singolo operando)

Operatori booleani e rappresentazione insiemistica

- L'operatore AND corrisponde all'INTERSEZIONE
- L'operatore OR corrisponde all'UNIONE
- L'operatore NOT corrisponde all'INSIEME COMPLEMENTARE



Tavole di verità

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

A	NOT A
0	1
1	0

Esercizio

- Data la seguente tabella di verità, verificare la proprietà A AND (B OR C) = (A AND B) OR (A AND C)

A	B	C
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Regole di precedenza

- Regole di precedenza:
  - NOT ha la massima precedenza
  - poi segue AND
  - infine OR

- Se voglio alterare queste precedenze devo usare le parentesi (a volte usate solo per maggior chiarezza)

- Per valutare un'espressione booleana si usa la tabella di verità

- Due espressioni booleane sono uguali se e solo se le tabelle della verità sono identiche