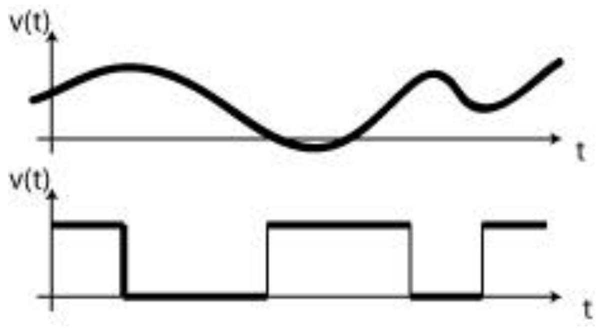
Rappresentazione digitale dell’informazione

Indice

I sistemi digitali

Segnale analogico: Un segnale è analogico quando i valori utili che lo rappresentano sono continui (infiniti) in un intervallo e non numerabili.



Segnale digitale: Un segnale è digitale quando i valori utili che lo rappresentano sono discreti e finiti.

I calcolatori moderni utilizzano due stati logici (binari), ma è possibile progettare sistemi digitali con logica a più stati (detta anche logica multi valori).

L'aritmetica dei calcolatori

L'aritmetica usata dai calcolatori è diversa da quella comunemente utilizzata dalle persone

La precisione con cui i numeri possono essere espressi è finita e predeterminata poiché questi devono essere memorizzati entro un limitato spazio di memoria

$$\sqrt{2} = \boxed{1}\boxed{.}\boxed{4}\boxed{1}\boxed{4}\boxed{2}\boxed{1}\ 3\ \ 5\ \ 6\ \ 2$$

La rappresentazione è normalmente ottenuta utilizzando il sistema binario poiché più adatto a essere maneggiato dal calcolatore

$$124 \Rightarrow 01111100$$

Numeri a precisione finita (1)

I **numeri a precisione finita** sono quelli rappresentati con un numero finito di cifre.

Fissate le caratteristiche del numero è determinato anche l'insieme di valori rappresentabili

Esempio: *Numeri non rappresentabili con 3 cifre senza virgola e senza segno*
Numeri superiori a 999
Numeri negativi
Frazioni

1	5	9

Le operazioni con i numeri a precisione finita causano errori ogniqualvolta il loro risultato non appartiene all'insieme dei valori rappresentabili:

- **Underflow**: si verifica quando il risultato dell'operazione è minore del più piccolo valore rappresentabile
- **Overflow**: si verifica quando il risultato dell'operazione è maggiore del più grande valore rappresentabile
- **Non appartenenza all'insieme**: si verifica quando il risultato dell'operazione, pur non essendo troppo grande o troppo piccolo, non appartiene all'insieme dei valori rappresentabili

Esempio: *Numeri a precisione finita con 3 cifre senza virgola e senza segno*

600+600 = 1200 ⇒ Overflow
300-600 = -300 ⇒ Underflow
007/002 = 3.5 ⇒ Non appartenenza all'insieme

Numeri a precisione finita (2)

Si noti che, a differenza dei numeri interi, i numeri a precisione finita non rispettano la chiusura rispetto alle operazioni di somma, sottrazione e prodotto.

Esempio: *Risultati non rappresentabili con 3 cifre senza virgola e senza segno*

Operazioni	Interi	Precisione finita
Somma: 600+600	1200	Overflow
Sottrazione:300-600	-300	Underflow
Prodotto: 050×050	2500	Overflow

Anche l'algebra dei numeri a precisione finita è diversa da quella convenzionale. Poiché alcune delle proprietà non vengono rispettate:

Proprietà associativa: $a + (b - c) = (a + b) - c$

Proprietà distributiva: $a \times (b - c) = a \times b - a \times c$

Non sono rispettate poiché in base all'ordine con cui vengono eseguite le operazioni si può verificare o meno un errore

Esempio: *Operazioni con numeri a precisione finita di 3 cifre senza virgola e senza segno*

$$400 + (300 - 500) = (400 + 300) - 500$$

$$400 + (\underbrace{-200}_{\textit{Underflow}}) = 700 - 500$$

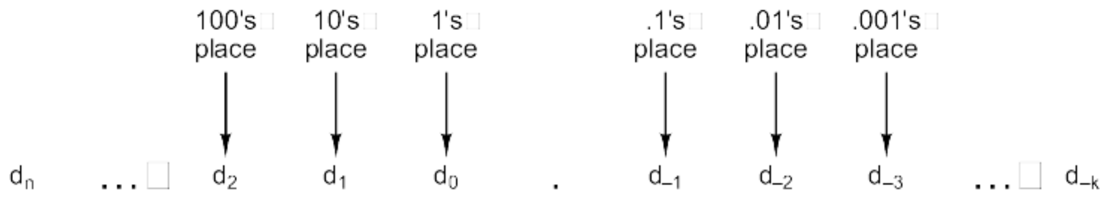
$$50 \times (50 - 40) = 50 \times 50 - 50 \times 40$$

$$50 \times 10 = \underbrace{2500}_{\textit{Overflow}} - \underbrace{2000}_{\textit{Overflow}}$$

ATTENZIONE: non confondere i numeri negativi con le operazioni di sottrazione

Notazione posizionale (1)

I sistemi di numerazione posizionale associano alle cifre un diverso valore in base alla posizione che occupano nella stringa che compone il numero.



$$Valore = \sum_{i=-k}^{n} d_i \times 10^i$$

Esempio: *Rappresentazione posizionale di 5798.46*

$$5 \times 10^3 + 7 \times 10^2 + 9 \times 10^1 + 8 \times 10^0 + 4 \times 10^{-1} + 6 \times 10^{-2}$$
$$= 5000 + 700 + 90 + 8 + 0.4 + 0.06$$

- Un sistema di numerazione posizionale è definito dalla **base (o radice)** utilizzata per la rappresentazione.

- Un sistema posizionale in base *b* richiede *b* simboli per rappresentare i diversi valori tra 0 e (*b*-1)

Sistema decimale (*b*=10) 0 1 2 3 4 5 6 7 8 9

Sistema binario (*b*=2) 0 1: ogni cifra, detta *bit* (**B**inary dig**IT**), può essere rappresentata direttamente tramite un livello elettrico di tensione

Sistema ottale (*b*=8) 0 1 2 3 4 5 6 7

Sistema esadecimale (*b*=16) 0 1 2 3 4 5 6 7 8 9 A B C D E F: è utilizzato nel linguaggio dell'assemblatore poiché è molto compatto e semplice da scandire. Inoltre ben si presta alla traduzione in valori binari poiché ogni cifra corrisponde esattamente a 4 cifre binarie.

Notazione posizionale (2)

A ogni numero corrisponderanno rappresentazioni diverse in basi diverse

Binary	1	1	1	1	1	0	1	0	0	0	1
	$1\ 2^{10}$ +	$1\ 2^9$ +	$1\ 2^8$ +	$1\ 2^7$ +	$1\ 2^6$ +	$0\ 2^5$ +	$1\ 2^4$ +	$0\ 2^3$ +	$0\ 2^2$ +	$0\ 2^1$ +	$1\ 2^0$
	1024	+ 512	+ 256	+ 128	+ 64	+ 0	+16	+ 0	+ 0	+ 0	+ 1
Octal	3	7	2	1							
	$3\ 8^3$ +	$7\ 8^2$ +	$2\ 8^1$ +	$1\ 8^0$							
	1536	+ 448	+ 16	+ 1							
Decimal	2	0	0	1							
	$2\ 10^3$ +	$0\ 10^2$ +	$0\ 10^1$ +	$1\ 10^0$							
	2000	+ 0	+ 0	+ 1							
Hexadecimal	7	D	1								
	$7\ 16^2$ +	$13\ 16^1$ +	$1\ 16^0$								
	1792	+ 208	+ 1								

Dato che l'insieme dei simboli utilizzati dalle varie basi non è disgiunto è necessario aggiungere al numero un pedice che indichi la radice utilizzata.

$$11111010001_2 = 3721_8 = 2001_{10} = 7D1_{16}$$

Conversione tra basi (1)

Binario ⇔ Ottale: dato che una cifra del sistema ottale è rappresentabile esattamente con tre cifre del sistema binario, la conversione può essere ottenuta raggruppando le cifre binarie a 3 a 3 a partire dalla virgola binaria. L'operazione contraria è ugualmente semplice, ogni cifra ottale viene convertita in esattamente tre cifre binarie.

Esadecimale ⇔ binario: il processo di conversione è equivalente a quello binario-ottale ma le cifre binarie devono essere raggruppate a 4 a 4.

Example 1

Hexadecimal	1 9 4 8 . B 6
Binary	0 0 0 1 1 0 0 1 0 1 0 0 1 0 0 0 . 1 0 1 1 0 1 1 0 0
Octal	1 4 5 1 0 . 5 5 4

Example 2

Hexadecimal	7 B A 3 . B C 4
Binary	0 1 1 1 1 0 1 1 1 0 1 0 0 0 1 1 . 1 0 1 1 1 1 0 0 0 1 0 0
Octal	7 5 6 4 3 . 5 7 0 4

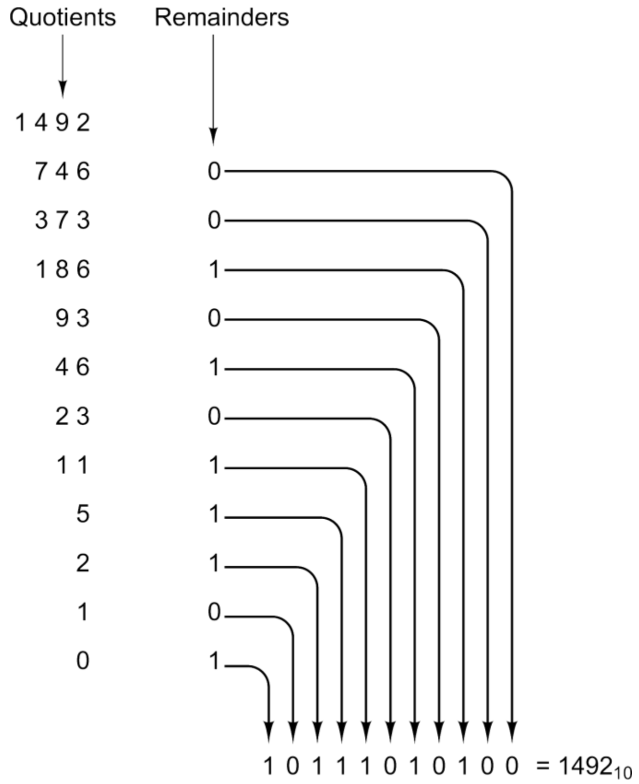
Conversione tra basi (2)

Decimale ⇒ Binario (Metodo generale): si procede sottraendo al numero da decomporre la più grande potenza di 2 minore del numero da decomporre. Il processo viene applicato ricorsivamente al resto della sottrazione. Il risultato binario si ottiene ponendo a uno le cifre corrispondenti alle potenze che sono state utilizzate nella decomposizione.

$1492.25 = 2^{10} + 468.25$
$468.25 = 2^{8} + 212.25$
$212.25 = 2^{7} + 84.25$
$84.25 = 2^{6} + 20.25$
$20.25 = 2^{4} + 4.25$
$4.25 = 2^{2} + 0.25$
$0.25 = 2^{-2}$

10111010100.01

Decimale ⇒ Binario (Solo per interi): la codifica viene ottenuta direttamente procedendo in modo ricorsivo. Si divide il numero per 2: il resto (0 o 1) farà parte del risultato mentre il quoziente verrà utilizzato come input al seguente passo di ricorsione.



Conversione tra basi (3)

Binario ⇒ Decimale (Primo metodo): si procede traducendo le singole cifre binarie alle corrispondenti potenze di due in base decimale e sommando i risultati parziali:

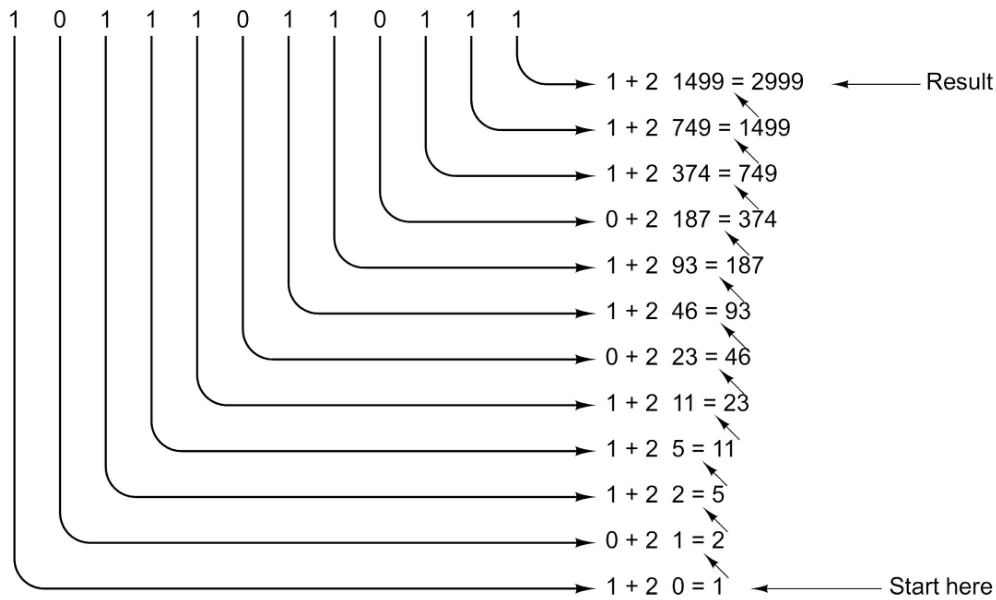
$$10111010100.01$$

$$2^{10} + 2^{8} + 2^{7} + 2^{6} + 2^{4} + 2^{2} + 2^{-2}$$

$$1024+256+128+64+16+4+0.25$$

$$1492.25$$

Binario ⇒ Decimale (Secondo metodo): la codifica viene ottenuta ricreando il valore posizionale di ogni cifra tramite successive moltiplicazioni per due a partire dalle cifre più significative verso quelle meno significative.



Altre conversioni: la conversione da decimale a ottale e da decimale a esadecimale può essere fatta passando per il sistema binario, oppure applicando i metodi precedentemente descritti e ponendo attenzione al corretto uso delle basi.

Esercizi

1. Si convertano i seguenti numeri decimali in base 2:

   371 3224 114.65625

2. Si convertano i seguenti numeri binari in base 8 e 16:

   11100110100110 1111001100011100

3. Si convertano i seguenti numeri esadecimali in base 2:

   FA31C CCCAB001

4. Si convertano i seguenti numeri esadecimali in base 10:

   AAB E0CC

Il calcolatore e i numeri binari

Prima di procedere nello studio dei numeri binari è bene ricordare che il codice e i dati di un programma vengono memorizzati, e successivamente, utilizzati da un calcolatore la cui architettura interna stabilisce il loro formato e il campo dei valori che possono assumere.

La più importante unità di misura dell'informazione manipolata dal calcolatore è il **BYTE** composto da 8 bit.

0	1	1	0	1	0	1	1

Nel byte il bit più a destra è quello meno significativo mentre quello a sinistra è quello più significativo.

Sequenze di bit più lunghe di un byte sono denominate **WORD**. La loro lunghezza dipende dalle caratteristiche del sistema, ma è sempre un multiplo del byte: 16/32/64/128 bit.

L'intervallo di valori codificabili dipende ovviamente dal numero di configurazioni possibili e dal tipo di dato da rappresentare. Con n bit sono possibili 2^n configurazioni.

Esempio: *Intervallo di interi positivi rappresentabili con n bit*

n	**Num. Configurazioni**	**Intervallo**
1	2	0 - 1
8	256	0 - 255
16	65.536	0 – 65.535
32	4.294.967.296	0 - 4.294.967.295
64	18.446.744.073.709.551.616	0 – 18.446.744.073.709.551.615

Unità di misura nel sistema binario

Il bit rappresenta la più piccola unità di misura dell'informazione memorizzabile in un calcolatore. I sistemi moderni memorizzano e manipolano miliardi di bit; per questo motivo sono stati definiti molti multipli.

Nome	Sigla	In bit	In byte	In potenze di 2
Bit	Bit	1 bit	1/8	2^1=2 stati
Byte	Byte	8	1	2^8=256 stati
KiloByte	KB	8.192	1.024	2^{10} byte
MegaByte	MB	8.388.608	1.048.576	2^{20} byte
GigaByte	GB	8.589.934.592	1.073.741.824	2^{30} byte
TeraByte	TB	8.796.093.022.208	1.099.511.627.776	2^{40} byte

I prefissi (Kilo, Mega, ecc.) sono normalmente associati a potenze di 10 mentre per i multipli del bit si opera su potenze di 2.

ATTENZIONE 1MB non corrisponde a 1000KB ma a 1024KB