

**RAPPRESENTAZIONE DELL'INFORMAZIONE****Sistemi di numerazione**

Qualunque numero intero positivo decimale è rappresentato da un insieme di cifre e simboli, che formano una stringa. La prima cifra a partire da destra rappresenta le unità, la seconda le decine, la terza le centinaia e così via. Per esempio, la stringa 7324 indica la *somma* di 4 unità, 2 decine, 3 centinaia e 7 migliaia.

Ogni cifra, pertanto, ha una influenza (peso) diversa in relazione alla posizione che essa occupa all'interno della stringa.

Esistono vari sistemi di numerazione: il sistema decimale usato abitualmente nell'aritmetica; il sistema binario, il sistema ottale e il sistema esadecimale, comunemente impiegati nei circuiti e nei calcolatori elettronici.

A prescindere dal sistema utilizzato un intero positivo può essere scritto in forma polinomiale. Ad esempio, l'intero positivo  $X$ , rappresentato dalla stringa 425, nell'aritmetica usuale vale

$$X = (4 \cdot \text{dieci}^2 + 2 \cdot \text{dieci}^1 + 5 \cdot \text{dieci}^0).$$

Questo perché abitualmente si opera in base dieci.

Operare in base dieci significa anche che ogni intero non negativo e minore di dieci va rappresentato utilizzando una sola cifra particolare. Infatti, le cifre della base dieci sono 0,1,2,3,4,5,6,7,8,9; l'intero positivo dieci viene rappresentato dalla stringa 10 formata da due delle dieci cifre viste precedentemente e disposte una di seguito all'altra.

Se non viene indicata la base, nessuno ci vieta di interpretare la stringa 425 come:

$$X = (4 \cdot \text{otto}^2 + 2 \cdot \text{otto}^1 + 5 \cdot \text{otto}^0).$$

In tal modo si opera in base otto. Operare in base otto significa anche che ogni intero non negativo e minore di otto va rappresentato utilizzando una sola cifra particolare. Infatti, le cifre della base otto sono 0,1,2,3,4,5,6,7; l'intero positivo otto viene rappresentato dalla stringa 10.

Analogamente la stessa stringa 425 può essere interpretata come

$$X = (4 \cdot \text{sedici}^2 + 2 \cdot \text{sedici}^1 + 5 \cdot \text{sedici}^0).$$

La base adoperata è la base sedici. In tale base le cifre sono 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F; mentre sedici è rappresentato dalla stringa 10.

**RAPPRESENTAZIONE DELL'INFORMAZIONE**

Però, la base per eccellenza dei circuiti e dei calcolatori elettronici è la base due. Tale base prevede l'utilizzo di due sole cifre o bit (binary digit): il bit 0 ed il bit 1.

Un raggruppamento di bit è chiamato byte (generalmente 8 bit) e word (16 bit).

Per evitare confusioni delle basi, si conviene indicare la stessa base in cui si opera scrivendola nel modo dell'esempio seguente:  $(425)_{\text{dieci}}$ , ciò significa che si vuole operare in base dieci.

Anche se non molto preciso e corretto, si usa indicare in cifre e no in lettere la base:  $(425)_{10}$  e tale forma sarà da noi seguita.

Per “base o radice del sistema” si indica il numero di cifre utilizzate ed è indicato con la lettera “b”.

Esiste la seguente regola: detta b la base di un numero intero M di m cifre, tale numero può essere scritto nella seguente forma polinomiale

$$(M)_b = \sum_{i=0}^{m-1} (A_i \cdot b^i) = A_0 \cdot b^0 + A_1 \cdot b^1 + \dots + A_{m-2} \cdot b^{m-2} + A_{m-1} \cdot b^{m-1}$$

dove con  $A_i$  si intende la generica cifra del numero.

Analogamente, per un numero frazionario k in base b con n cifre dopo la virgola decimale, si ha:

$$(k)_b = \sum_{i=-n}^{-1} (A_i \cdot b^i) = A_{-1} \cdot b^{-1} + A_{-2} \cdot b^{-2} + \dots + A_{-n+1} \cdot b^{-n+1} + A_{-n} \cdot b^{-n}.$$

Pertanto, in generale, per un numero intero con parte frazionaria in base b generica e con m cifre per la parte intera e con n cifre per la parte frazionaria si ha:

$$(M)_b = \sum_{i=-n}^{m-1} (A_i \cdot b^i) = A_{m-1} \cdot b^{m-1} + \dots + A_0 \cdot b^0 + A_{-1} \cdot b^{-1} + \dots + A_{-n+1} \cdot b^{-n+1} + A_{-n} \cdot b^{-n}.$$

Tutti i sistemi presentano la stessa struttura e pertanto tale regola viene seguita da tutti i sistemi numerici.

Da quanto detto, si deduce che il numero intero massimo scritto con m cifre ed in base b vale  $(M_{\text{max}})_{(b)} = b^m - 1$ ; quindi, per esempio, con 5 bit in binario si possono scrivere tutti gli interi positivi da 0 a 31.

Una caratteristica molto importante dei sistemi di numerazione è lo spostamento di tutte le cifre a destra o a sinistra di una posizione (*shift right* e *shift left*). Tale spostamento comporta rispettivamente una divisione o una moltiplicazione del numero intero per la sua base. Se la

**RAPPRESENTAZIONE DELL'INFORMAZIONE**

traslazione avviene verso destra si perde la cifra meno significativa e si effettua una divisione intera per la base, per cui la cifra persa è proprio il resto della divisione. Invece, se la traslazione avviene verso sinistra si deve aggiungere uno 0 a destra e si effettua una moltiplicazione del numero per la base.

Nella fig. 1 sono riportati alcuni esempi di numeri interi positivi nei sistemi di numerazione più importanti: binario, esadecimale, ottale, decimale.

Decimale	Binario	Ottale	Esadecimale
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14
21	10101	25	15
32	100000	40	20
60	111100	74	3C
64	1000000	100	40
100	1100100	144	64
255	11111111	377	FF
500	111110100	764	1F4
1000	1111101000	1750	3E8
1023	1111111111	1777	3FF
2000	11111010000	3720	7D0

Fig. 1 – Rappresentazione di numeri interi per diversi sistemi di numerazione.

## **Regole di conversione fra sistemi numerici diversi**

E' fondamentale conoscere ed applicare correttamente le regole che permettono di rappresentare uno stesso numero nei vari sistemi di numerazione.

**RAPPRESENTAZIONE DELL'INFORMAZIONE**

Per convertire un numero binario, ottale, esadecimale o in qualunque altra base nel corrispondente numero decimale, basta applicare alla lettera l'espressione polinomiale, sia per numeri interi sia per numeri frazionari.

Per convertire un numero intero da decimale a una qualunque altra base, bisogna dividere tale numero per la base conservando il resto e dividendo ancora per la base fino ad avere il risultato nullo. I resti ottenuti a partire dall'ultimo sono proprio le cifre che rappresentano nella nuova base il numero decimale di partenza. Nel caso di sistema binario, il primo resto è il bit meno significativo **LSB** (*least significant bit*), mentre l'ultimo resto è il bit più significativo **MSB** (*most significant bit*).

Invece, per la conversione della parte frazionario del numero decimale di partenza, si deve moltiplicare (non dividere) per la base e conservare la parte intera. La nuova parte frazionaria viene moltiplicata per la base conservando sempre la parte intera. Si ripete tale operazione per un numero adeguato di volte e poi le cifre conservate dalla prima all'ultima trovata saranno le nuove cifre frazionarie. E' da notare che passando da una base ad un'altra, non sempre una frazione finita si riesce a convertire in un'altra frazione finita.

Per convertire un numero da binario ad una base potenza del due come base 4, ottale ed esadecimale, si divide il numero binario rispettivamente in gruppi di 2 cifre, 3 cifre o 4 cifre partendo dall'eventuale punto frazionario ed andando verso sinistra per ottenere la parte intera e verso destra per la parte frazionaria. A ciascun gruppo di cifre binarie si sostituisce la corrispondente singola cifra della nuova base in questione. Eventualmente si possono aggiungere degli zeri a sinistra del numero intero e a destra della parte frazionaria perché non si varia il numero di partenza e questo in qualunque base.

Viceversa, per convertire un numero in base potenza del due nel binario corrispondente è sufficiente convertire ciascuna cifra nella corrispondente sequenza binaria.

**RAPPRESENTAZIONE DELL'INFORMAZIONE****Aritmetica binaria**

In questo paragrafo saranno prese in considerazione le regole che permettono di effettuare le quattro operazioni dell'aritmetica binaria nel caso di numeri positivi. In un secondo tempo, saranno trattati e rappresentati in forma binaria anche i numeri negativi, mediante il criterio del modulo e del segno ed il criterio del complemento a 1 e del complemento a 2.

L'**addizione binaria** è l'operazione di somma fra due numeri binari, che è eseguita come per il decimale. Le regole per l'addizione valide per il sistema binario sono:

$$0 + 0 = 0 \quad 0 + 1 = 1 \quad 1 + 0 = 1 \quad 1 + 1 = 0 \text{ (con riporto di 1).}$$

Nei sistemi digitali i numeri sono generalmente rappresentati mediante una stringa di un numero di cifre ben fissato e limitato. Pertanto, se nell'effettuare una addizione si ha un eventuale riporto nella somma dei bit più significativi, si ottiene un trabocco (*overflow*), di cui i circuiti elettronici devono tenerne conto. Infatti, trascurare questo ultimo bit porta a non considerare il bit più significativo e quindi quello più importante.

La **sottrazione binaria** è l'operazione di differenza fra due numeri binari, che è eseguita come nel caso decimale. Le regole della sottrazione binaria sono le seguenti:

$$0 - 0 = 0 \quad 1 - 0 = 1 \quad 1 - 1 = 0 \quad 0 - 1 = 1 \text{ (con prestito di 1 dalla cifra precedente).}$$

In questo caso, deve sempre essere verificato che il minuendo sia maggiore o uguale al sottraendo.

La **moltiplicazione binaria** è l'operazione di prodotto fra due numeri binari, che è eseguita come nel caso decimale. Le regole della moltiplicazione binaria sono le seguenti:

$$0 \times 0 = 0 \quad 1 \times 0 = 0 \quad 1 \times 1 = 1 \quad 0 \times 1 = 0.$$

Come nel caso decimale, l'operazione di moltiplicazione può essere effettuata sommando il moltiplicando per il numero di volte pari al moltiplicatore.

La **divisione binaria** è l'operazione che determina il quoziente ed il resto tra due numeri binari (dividendo e divisore). La divisione si fa, come per il decimale, per successive sottrazioni.

Si mettono a confronto gli n bit del divisore con gli n bit più significativi del dividendo. Se il numero formato dagli n bit più significativi del dividendo è maggiore o uguale al numero formato dai bit del divisore, si inserisce un 1 sul quoziente e si fa la differenza fra i due numeri; in caso

**RAPPRESENTAZIONE DELL'INFORMAZIONE**

contrario si segna uno 0 sul quoziente. In ogni caso si aggiunge il bit successivo o uno 0 se non ci sono più bit a disposizione e si ripete l'operazione di confronto fino ad ottenere un resto nullo o fino ad un certo numero di cifre frazionarie nel quoziente.

I numeri relativi (**binari con segno**) possono essere rappresentati in due modi diversi a seconda delle applicazioni: in modulo e segno ed in complemento a 2. Per le operazioni di addizione e di sottrazione fra numeri relativi è utilizzata la rappresentazione in complemento a due; invece, per le operazioni di moltiplicazione e di divisione è usata la rappresentazione in modulo e segno.

I numeri positivi ed i numeri negativi decimali si distinguono tra loro grazie ai segni + e -. La stessa cosa è fatta per i numeri binari dove viene aggiunto a sinistra della cifra più significativa il bit di segno: 0 per un numero positivo, ed 1 per un numero negativo.

In definitiva, il bit aggiunto a sinistra rappresenta il segno, mentre i rimanenti bit rappresentano il modulo del numero (**criterio del modulo e del segno**).

Il complemento ad 1 ( $C_1$ ) di un numero binario si ottiene invertendo ogni bit, se è 1 diventa 0 e se è 0 diventa 1. Da notare che la somma del numero dato e del suo complemento ad 1 produrrà una serie di 1, tanti quanti sono i bit del numero dato.

Il **complemento a 2** di un numero si ottiene dal complemento a 1 sommando un ulteriore 1 al bit meno significativo e propagandone l'eventuale riporto.

In generale, sommando un numero a  $n$  cifre con il suo complemento a 2 si ottiene un numero a ( $n + 1$ ) cifre, con la cifra più significativa uguale a 1 e le altre  $n$  cifre pari a 0.

Un metodo veloce per effettuare il complemento a 2 di un numero binario è quello di partire da destra verso sinistra lasciando inalterati tutti i bit fino al primo 1 compreso, ed invertendo tutti gli altri.

Per la moltiplicazione e la divisione binaria, i numeri relativi sono rappresentati facilmente in modulo e segno. Infatti, basta considerare separatamente il bit del segno dai bit del modulo. Il segno del risultato è: 0 se i due bit dei segni sono uguali, 1 se i due bit sono diversi. Poi, si applicano ai moduli le regole viste.

Per l'addizione e la sottrazione binaria, si utilizza il complemento a 2. In particolare, la sottrazione deve essere eseguita sommando al minuendo il complemento a 2 del sottraendo, trascurando l'eventuale riporto dell'ultima cifra a sinistra. Il risultato, se è positivo, è espresso in

**RAPPRESENTAZIONE DELL'INFORMAZIONE**

binario; se è negativo è espresso in complemento a 2 e pertanto deve essere nuovamente complementato a 2.

**Operazioni aritmetiche in base generica**

Tutti i sistemi di numerazione hanno la stessa struttura e seguono le stesse regole. Pertanto, le operazioni viste in binario hanno le stesse caratteristiche delle operazioni decimali e delle operazioni in qualunque base “b”. È importante che si conoscano le cifre del sistema numerico che si sta utilizzando, ricordando che si parte dallo 0 e si cresce fino alla cifra subito precedente al valore della base.

Per operare in qualsiasi tipo di base, è necessario conoscere il complemento alla base ed alla base-1.

Esiste un metodo veloce per il calcolo del complemento alla base. Consiste nel partire dalla cifra meno significativa del numero dato e sommarci la cifra che produce il valore della base. Per le altre cifre, invece, è necessario sommare quella cifra che fornisce il valore della base-1.

Nella eventualità della presenza di uno o più 0 a sinistra del numero, la corrispondente cifra del complemento vale sempre 0.

**Rappresentazione dei numeri in virgola fissa ed in virgola mobile**

Nei calcolatori e nei circuiti elettronici i numeri possono essere rappresentati in due modi distinti: in virgola fissa ed in virgola mobile.

Nel primo caso la rappresentazione del numero avviene con una stringa di cifre di lunghezza costante e supponendo di avere una stringa di 32 bit, la virgola ha una posizione ben definita fra due bit, dividendo la parte intera da quella frazionaria. Inoltre, il bit più significativo fornisce il segno del numero.

**RAPPRESENTAZIONE DELL'INFORMAZIONE**

Invece, nel secondo caso, il numero binario  $X$  può essere scritto in forma esponenziale:  $X = m \cdot 2^e$ , dove  $m$  ed  $e$  sono una coppia di numeri relativi e quindi con segno detti mantissa ed esponente, ed hanno dimensione fissa. Con tale metodo si riesce a rappresentare con un elevatissimo numero di cifre significative sia numeri molto grandi che numeri molto piccoli, ma sempre mediante una parola di un numero limitato di bit.

**Codici binari**

La definizione generale di codice binario è quella di un insieme  $\{C\}$  di configurazioni binarie che è utilizzata per rappresentare gli elementi di un certo insieme  $\{C'\}$ . Per configurazione binaria si intende una stringa di bit; il numero dei bit componenti la stringa si dice lunghezza della configurazione. Per cui, le possibili combinazioni lunghe  $n$  bit sono in tutto  $m = 2^n$ .

Ad esempio, le configurazioni lunghe 3 bit sono  $m = 2^3 = 8$  e cioè:

000; 001; 010; 011; 100; 101; 110; 111.

In generale tutti i codici possono essere seriali e paralleli. Si parla di codice seriale se è trasmesso bit dopo bit; invece, si parla di codice parallelo se tutti i bit del codice sono trasmessi contemporaneamente su più linee in parallelo. La trasmissione dei bit può essere, pertanto, seriale o parallela. In ogni modo, è necessaria la presenza di un mezzo di comunicazione o di trasmissione che permetta il collegamento fra trasmettitore e ricevitore. Tale mezzo trasmissivo può essere: un cavo elettrico, una fibra ottica e l'atmosfera.

La trasmissione parallela è, sicuramente, più veloce di quella seriale e viene utilizzata prevalentemente quando è necessaria una grande velocità di trasmissione. Però, la trasmissione parallela comporta la presenza di un maggior numero di collegamenti tra TX e RX; pertanto, la trasmissione parallela viene utilizzata solo per piccole distanze.

La trasmissione seriale, invece, è utile per le grandi distanze, per un fattore puramente economico; infatti, si utilizza un solo cavo più la massa. E', pure, conveniente perché agisce meglio nei confronti dei disturbi presenti nella trasmissione.

## **RAPPRESENTAZIONE DELL'INFORMAZIONE**

I primi codici binari di cui ci occuperemo sono i seguenti: codice binario naturale, codici numerici BCD, codice numerico Gray, codici alfanumerici.

Il primo di questi, comunque, non può essere considerato un vero e proprio codice, perché rappresenta un numero decimale convertito in binario dalla relativa regola di conversione.

### **Codice binario BCD**

I codici BCD (Binary Coded Decimal) sono usati soprattutto per l'interfacciamento dei sistemi digitali con visualizzatori, periferiche o strumenti di misura.

Ogni cifra decimale è rappresentata con una combinazione di 4 bit mentre il numero mantiene la struttura decimale. I codici numerici BCD si dicono ponderati se si riesce a risalire al valore decimale applicando opportuni pesi a ciascuna delle quattro cifre. I codici BCD più comunemente utilizzati sono tre, cioè:

- Codice BCD standard o codice 8421; ha pesi 8, 4, 2, 1, pertanto è ponderato ed utilizza, per rappresentare le cifre decimali, le stesse espressioni del binario naturale, sfruttando le prime dieci combinazioni delle sedici possibili. Il vantaggio di tale codice consiste nella facilità ed immediatezza della conversione con il sistema decimale.
- Codice BCD con peso 2, 4, 2, 1, è detto anche codice Aiken 2421 autocomplementante in quanto invertendo i bit di una cifra si ottiene il complemento a 9 della cifra stessa.
- Codice eccesso-3, non pesato, detto anche XS-3 ed autocomplementante. Infatti, la codifica si ottiene sommando 3 alla corrispondente codifica in BCD 8421. Inoltre, è autocomplementante, come per il codice precedente.

Pertanto, in generale, supponendo di avere  $N$  cifre decimali e, quindi,  $4 \times N$  cifre binarie, è possibile nel codice BCD standard poter rappresentare solo i numeri da 0 a  $10^N - 1$ , mentre in binario naturale si possono rappresentare tutti i numeri da 0 a  $2^{4N} - 1$ .

Ad esempio con  $n = 4$  cifre decimali, si ottiene un numero in BCD con sedici cifre binarie, con le quali possiamo rappresentare tutti i numeri compresi tra 0 e 9999, mentre in binario naturale si possono rappresentare tutti i numeri compresi tra 0 e 65535.

## ***RAPPRESENTAZIONE DELL'INFORMAZIONE***

Se volessimo effettuare in BCD standard la somma di due o più numeri, dobbiamo tener conto che:

1. se il risultato è una combinazione non ammessa dal codice, per ottenere la codifica BCD esatta del risultato è necessario sommare il numero 0110 (6 decimale) alla combinazione non ammessa;
2. se il risultato è una combinazione ammessa ma con riporto, è necessario sommare sempre il numero 0110;
3. se il risultato è una combinazione ammessa dal codice, tale risultato è corretto.

### **Codice numerico Gray**

E' un codice non pesato ed utilizzato principalmente nei codificatori meccanici di posizione detti anche encoder assoluti. Ha la caratteristica che tra due configurazioni adiacenti manifesta il cambiamento di un solo bit. E' chiamato anche riflesso per il modo in cui viene costruito. Infatti, nella prima colonna da destra vi è una sequenza ripetitiva di 0110, nella seconda colonna da destra la sequenza ripetitiva è 00111100 e così via.

Esiste, infine, un metodo veloce per la determinazione del codice Gray di un qualunque numero decimale comunque grande. Si effettua, per prima cosa, la conversione in binario naturale e poi, si effettua la somma senza riporto (XOR) fra due bit adiacenti a partire da destra.

### **Codici alfanumerici**

I codici alfanumerici comprendono simboli che servono per rappresentare non solo numeri ma anche lettere alfabetiche minuscole e maiuscole, segni di punteggiatura, simboli matematici, simboli speciali, comandi per la trasmissione e per la stampa. Tali codici permettono di poter rappresentare simboli di natura non solo numerica, utilizzando semplicemente stringhe binarie. Caso molto semplice anche se ormai usato solamente nelle trasmissioni telegrafiche è il codice Morse. Esso, infatti, utilizza solo due simboli, il punto e la linea.

**RAPPRESENTAZIONE DELL'INFORMAZIONE**

I codici alfanumerici standard più impiegati nell'ambito dei calcolatori elettronici sono:

- Codice ASCII (American Standard Code for Information Interchange) a 7 bit, che codifica fino a 128 caratteri. Il codice ASCII spesso viene utilizzato con 8 bit. L'ultimo bit viene usato come bit di controllo di parità, oppure per aumentare il numero di simboli che raddoppia e diventa 256. Vari produttori hanno usato estensioni diverse provocando problemi di conversione tra file di sistemi diversi, ad esempio fra Windows e MacOS.
- Codice EBCDIC (Extended Binary Coded Decimal Interfacing Code) è un codice a 8 bit e permette di rappresentare  $2^8 = 256$  valori diversi.
- Codice Unicode è un codice che utilizza 16 bit e permette di rappresentare  $2^{16} = 65536$  valori diversi anche se finora ne sono stati assegnati solo 34168. Questo codice permette la rappresentazione dei caratteri di molti alfabeti non latini (arabo, ebraico, cinese, etc.) e di caratteri speciali (matematici, tecnici, elementi di disegno, etc.).
- Codice ISO (International Standard Organization) è nato come proposta di standard internazionale ed utilizza 7 bit.
- Codice ISO Latin 1 è una estensione a 8 bit del codice ASCII ed è molto diffuso negli usi in rete. Esso corrisponde ai caratteri del codice Unicode i cui primi 8 bit (più significativi) sono tutti zero.