

Unità Didattica 1

Sistemi di Numerazione

Sistemi di Numerazione Posizionali (1)

- Criterio per la rappresentazione di un insieme infinito di numeri mediante un insieme limitato di simboli.
- Un sistema di numerazione posizionale è costituito da:
 - 1) Una base b ;
 - 2) Un insieme ordinato di cifre;
 - 3) Un codice di interpretazione (*interpretazione posizionale*);
 - 4) Un insieme di algoritmi per le 4 operazioni aritmetiche fondamentali.

Sistemi di Numerazione in Base “b” (1)

- Il codice di interpretazione specifica che ad ogni posizione è associato un peso;
- Ogni cifra associata ad una posizione indica il numero delle volte che deve essere considerato il peso corrispondente alla quella posizione;
- Sistema di numerazione in base “b”:

sistema di numerazione posizionale la cui base per i pesi è “b”;

ovvero:

sistema di numerazione posizionale in cui i pesi sono espressi come potenze della base “b”.

Sistemi di Numerazione in Base “b” (2)

- Numero in base “b”:

Numero rappresentato utilizzando il sistema di numerazione in base “b” ed una sequenza di cifre del tipo: $(c_{n-1}c_{n-2} \dots c_0 \cdot c_{-1} \dots c_{-m})_b$;

- Il simbolo “.” (*punto di separazione*) separa la parte intera da quella frazionaria;

- c_0 ha peso b^0
- c_1 “ b^1
- c_{-1} “ b^{-1} .

Forma Polinomia

- Vale la relazione (conseguenza della definizione di numero in base “b”):

$$(c_{n-1}c_{n-2} \dots c_0 \cdot c_{-1} \dots c_{-m})_b = c_{n-1} b^{n-1} + c_{n-2} b^{n-2} + \dots + c_0 b^0 + c_{-1} b^{-1} \dots c_{-m} b^{-m}$$

- Tale somma di prodotti è detta

Forma Polinomia

Sistema di Numerazione in Base 10

- Base 10 ($b = 10$):
 - Cifre $c_i = \{0, 1, \dots, 9\}$
 - Esempio:
 $(127.3)_{10} = 1 \cdot 10^2 + 2 \cdot 10^1 + 7 \cdot 10^0 + 3 \cdot 10^{-1}$

Sistema di Numerazione in Base 2

- Base 2 ($b = 2$):
 - Cifre $c_i = \{0, 1\}$ (Binary digit = bit)
 - Esempio:
 $(101.01)_2 = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} = (5.25)_{10}$
- **Conseguenza:**
 - La forma polinomiale rappresenta l'algoritmo di conversione binario – decimale;
- **In generale:**

Forma Polinomiale

sistema in base b \Rightarrow **sistema decimale**

Algoritmi di Conversione Decimale - Binario

- Si distinguono due casi:
 - Numeri interi (Algoritmo delle Divisioni Successive);
 - Numeri frazionari (Algoritmo delle Moltiplicazioni Successive);
- In caso di numeri reali con parte intera e frazionaria i due algoritmi vengono applicati separatamente e combinati i risultati.

Metodo delle Divisioni Successive (1)

- Dato $(N)_{10} \in \mathcal{N}$ determinare la successione di cifre binarie $c_{n-1} c_{n-2} \dots c_0$ tale che:

$$N = c_{n-1} 2^{n-1} + c_{n-2} 2^{n-2} + \dots + c_0 2^0$$

- Dividendo per 2 entrambi i membri:

$$\text{Essendo } N = 2Q + R \quad \Rightarrow \quad N/2 = Q + R/2 \quad (R=0 \text{ o } 1)$$

- Si ha:

$$N/2 = Q + R/2 = \underbrace{c_{n-1} 2^{n-2} + c_{n-2} 2^{n-3} + \dots + c_1 2^0 + c_0 2^{-1}}_Q \quad R/2$$

- Quindi $R = c_0$ ovvero c_0 è il resto della divisione di $(N)_{10}$ per 2.

Metodo delle Divisioni Successive (2)

- $Q = c_{n-1} 2^{n-2} + c_{n-2} 2^{n-3} + \dots + c_1 2^0$
 - Dividendo per 2 entrambi i membri:
- Essendo $Q = 2Q' + R \quad \Rightarrow \quad Q/2 = Q' + R'/2 \quad (R'=0 \text{ o } 1)$
- Si ha:

$$Q/2 = \underbrace{c_{n-1} 2^{n-3} + c_{n-2} 2^{n-4} + \dots + c_2 2^0}_{Q'} + \underbrace{c_1 2^{-1}}_{R'/2}$$
 - Quindi $R' = c_1$
ovvero c_1 è il resto della divisione di $(Q)_{10}$ per 2.

Metodo delle Divisioni Successive (3)

- I coefficienti della forma polinomiale in base 2 di $(N)_{10}$ intero, sono rappresentati dai resti delle divisioni successive di $(N)_{10}$ per 2.
- L'algoritmo termina quando si ottiene un quoziente uguale a zero.

Metodo delle Divisioni Successive (es.)

- $(N)_{10} = 13$
- $N/2 = 13/2 \Rightarrow Q = 6 \ R=1 \Rightarrow c_0 = 1$
- $Q/2 = 6/2 \Rightarrow Q = 3 \ R=0 \Rightarrow c_1 = 0$
- $Q/2 = 3/2 \Rightarrow Q = 1 \ R=1 \Rightarrow c_2 = 1$
- $Q/2 = 1/2 \Rightarrow Q = 0 \ R=1 \Rightarrow c_3 = 1$

$$(N)_{10} = 13 \Rightarrow (N)_2 = (1101)_2$$

$$(N)_2 = (1101)_2 \Rightarrow (N)_{10} = (1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0) = (13)_{10}$$

Metodo delle Moltiplicazioni Successive (1)

- Dato $(F)_{10}$, parte frazionaria di un numero reale, det. la successione di cifre binarie $c_{-1} c_{-2} \dots c_{-m}$ tale che:

$$F = c_{-1} 2^{-1} + c_{-2} 2^{-2} + \dots + c_{-m} 2^{-m}$$

- Moltiplicando entrambi i membri per 2:

$$F \cdot 2 = \underbrace{c_{-1} + c_{-2} 2^{-1} + \dots + c_{-m} 2^{-m+1}}_P = P + M$$

Parte Intera Parte Frazionaria

- $P = c_{-1}$, ovvero la parte intera del prodotto $F \cdot 2$ rappresenta la cifra binaria di $(F)_2$ immediatamente successiva al punto

Metodo delle Moltiplicazioni Successive (2)

$$M = c_{-2} 2^{-1} + c_{-3} 2^{-2} + \dots + c_{-m} 2^{-m+1}$$

- Moltiplicando entrambi i membri per 2:

$$M \cdot 2 = \underbrace{c_{-2} 2^{-1} + \dots + c_{-m} 2^{-m+2}}_{M'} = \underbrace{P'}_{P'} + M'$$

- $P' = c_{-2}$ ovvero la parte intera del prodotto $M \cdot 2$ rappresenta la seconda cifra binaria di $(F)_2$ immediatamente successiva al punto

Metodo delle Moltiplicazioni Successive (3)

- I coefficienti della forma polinomiale in base 2 di $(F)_{10}$ frazionario, sono rappresentati dalle parti intere delle moltiplicazioni successive di $(F)_{10}$ per 2
 - L'algoritmo termina:
 - quando si ottiene una parte frazionaria uguale a zero (approssimazione esatta);
- Oppure:
- dando un numero massimo di moltiplicazioni da eseguire (n° massimo di bit da calcolare) (approssimazione per difetto)
- Per un'approssimazione migliore occorre iterare ulteriormente il procedimento con nuove moltiplicazioni.

Metodo delle

Moltiplicazioni Successive (es.)

$$(F)_{10} = 0.31$$

$$P = 0 \quad M = 0.31$$

$$M \cdot 2 = 0.31 \cdot 2 = 0.62$$

$$P = c_{-1} = 0 \quad M = 0.62$$

$$M \cdot 2 = 0.62 \cdot 2 = 1.24$$

$$P = c_{-2} = 1 \quad M = 0.24$$

$$M \cdot 2 = 0.24 \cdot 2 = 0.48$$

$$P = c_{-3} = 0 \quad M = 0.48$$

$$M \cdot 2 = 0.48 \cdot 2 = 0.96$$

$$P = c_{-4} = 0 \quad M = 0.96$$

$$M \cdot 2 = 0.96 \cdot 2 = 1.92$$

$$P = c_{-5} = 1 \quad M = 0.92$$

$$(F)_{10} = 0.31 \Rightarrow (F)_2 = (0.01001)_2$$

Approx per difetto

$$(F)_2 = (0.01001)_2 \Rightarrow (F)_{10} = 1 \cdot 2^{-2} + 1 \cdot 2^{-5} = 0.28125$$

Relazioni decimale-binario

- Qual è il numero massimo rappresentabile con n bit?
 - Tale valore si ottiene dalla forma polinomiale su n bit, ponendo $c_i=1$:

$$1 \cdot 2^{n-1} + 1 \cdot 2^{n-2} + \dots + 1 \cdot 2^0 = \sum_{k=0}^{n-1} 2^k = \frac{1 - 2^{(n-1)+1}}{1 - 2} = 2^n - 1$$

Relazioni decimale-binario

- Dato $(N)_{10}$ intero qual è il numero n di bit necessario per la corrispondente rappresentazione binaria?

– È il più piccolo n tale che: $2^n - 1 \geq N$

essendo 2^{n-1} il massimo valore rappresentabile con n bit, ed essendo $2^{n-1} - 1 < N$ per definizione di n .

$$2^n \geq N + 1 \Rightarrow n \geq \log_2(N + 1) \Rightarrow n = \lceil \log_2(N + 1) \rceil$$

Operazioni elementari in base b

- Qualunque sia la base b è possibile definire le regole per le operazioni elementari mediante una tabella in cui è riportato il risultato di tutte le possibili combinazioni tra due cifre del sistema di numerazione rispetto all'operazione considerata.
- In generale (per esempio per l'addizione), essendo c_i, c_j, \dots, c_k ... cifre del sistema di numerazione in base b

$$\begin{cases} c_i + c_j = c_k & \text{se } c_i + c_j \leq b - 1 \\ c_i + c_j = 1c_l & \text{se } c_i + c_j > b - 1 \end{cases}$$

Riporto 

Somma e sottrazione in base 2

- Regole per la somma • Regole per la sottrazione

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 10$$

↑
Riporto

$$0 - 0 = 0$$

$$1 - 0 = 1$$

$$1 - 1 = 0$$

$$10 - 1 = 1$$

↑
Prestito

Moltiplicazione e divisione in base 2

- Regole per la moltiplicazione

$$0 \cdot 0 = 0$$

$$0 \cdot 1 = 0$$

$$1 \cdot 0 = 0$$

$$1 \cdot 1 = 1$$

- Divisione:

– La divisione binaria è calcolata in maniera analoga alla divisione decimale.

Sistemi di numerazione in base 8 e 16

- Sistema ottale (base 8):
 - cifre: $c_i = \{0, \dots, 7\}$
 - es. forma polinomiale: $(53.76)_8 = 5 \cdot 8^1 + 3 \cdot 8^0 + 7 \cdot 8^{-1} + 6 \cdot 8^{-2} = 43.96875$
- Sistema esadecimale (base 16):
 - cifre: $c_i = \{0, \dots, 9, A, B, C, D, E, F\}$
 - es. forma polinomiale: $(D3.B)_8 = 13 \cdot 16^1 + 3 \cdot 16^0 + 11 \cdot 16^{-1} = 211.6875$
- Le regole per le operazioni fondamentali sono analoghe a quelle binarie

Conversione Binario-Ottale (1)

- Osservazione: se $c_i = \{0,1\}$

$$0 \leq c_2 \cdot 2^2 + c_1 \cdot 2^1 + c_0 \cdot 2^0 \leq 7$$

- Al variare di c_2, c_1, c_0 in $\{0, 1\}$, il trinomio precedente rappresenta una cifra $d_j = \{0, 1, \dots, 7\}$ del sistema di numerazione di base 8

Conversione Binario-Ottale (2)

- La forma polinomiale $(N)_2 = c_{n-1} \cdot 2^{n-1} + c_{n-2} \cdot 2^{n-2} + \dots + c_0 \cdot 2^0$

mettendo in evidenza 2^{3p} , $p=0, \dots, k$ ogni 3 elementi consecutivi a partire da destra, può essere scritta nel modo seguente:

$$\begin{aligned}
 (N)_2 &= \underbrace{(c_{3k+2} \cdot 2^2 + c_{3k+1} \cdot 2^1 + c_{3k} \cdot 2^0)}_{d_k} \cdot 2^{3k} + \dots + \underbrace{(c_{3 \cdot 2+2} \cdot 2^2 + c_{3 \cdot 2+1} \cdot 2^1 + c_{3 \cdot 2} \cdot 2^0)}_{d_2} \cdot 2^{3 \cdot 2} + \\
 &\quad \underbrace{(c_{5 \cdot 3+2} \cdot 2^2 + c_{4 \cdot 3+1+1} \cdot 2^1 + c_3 \cdot 2^0)}_{d_1} \cdot 2^3 + \underbrace{(c_2 \cdot 2^2 + c_1 \cdot 2^1 + c_0 \cdot 2^0)}_{d_0} \cdot 2^0 =
 \end{aligned}$$

$$= (d_k) \cdot 8^k + \dots + (d_2) \cdot 8^2 + (d_1) \cdot 8^1 + (d_0) \cdot 8^0 = (N)_8$$

$2^{3p} = 8^p$

Forma polinomiale
in base 8 di N

Conversione Binario \leftrightarrow Ottale (3)

Binario \Rightarrow Ottale

- La conversione binario - ottale diretta si esegue nel modo seguente:
 1. Raggruppamento della parte intera per gruppi di 3 cifre a partire da destra, aggiungendo eventualmente bit = 0 a sinistra;
 2. Raggruppamento della parte frazionaria a gruppi di 3 cifre da sinistra a destra, se necessario si aggiungono bit = 0 a destra;
 3. Conversione di ciascun gruppo di 3 cifre nella corrispondente cifra ottale;
 4. La sequenza di cifre ottali è il numero originario convertito in base 8.

Ottale \Rightarrow Binario

- Si sostituisce ogni cifra ottale nella corrispondente rappresentazione binaria

Conversione Binario-Esadecimale

- Osservazione: se $c_i = \{0,1\}$

$$0 \leq c_3 \cdot 2^3 + c_2 \cdot 2^2 + c_1 \cdot 2^1 + c_0 \cdot 2^0 \leq F = 15$$

- Al variare di $c_3 c_2 c_1 c_0$ in $\{0, 1\}$ il quadrinomio precedente rappresenta una cifra $d_j = \{0, 1, \dots, 9, A, \dots, F\}$ del sistema di numerazione di base 16;
- Si procede analogamente alla conversione binario-ottale, raggruppando le cifre binarie a gruppi di 4;
- Tali gruppi si convertono nella corrispondente cifra esadecimale.

Esempi di Conversione

Binario-Ottale Binario-Esadecimale

- Binario - Ottale

$$(1101.01)_2 = (001101.010)_2 = (15.2)_8$$

- Binario - Esadecimale

$$(1101.01)_2 = (1101.0100)_2 = (D.4)_{16}$$

Il Complemento a b

- Definizione:

Dato $N > 0 \in \mathcal{N}$ rappresentato in base b su k cifre si definisce complemento a b di N il numero $C_b(N)$ in base b tale che:

$$N + C_b(N) = b^k$$

Es: $C_{10}(68) = \underset{\text{su } k=2 \text{ cifre}}{10^2 - 68} = 100 - 68 = 32$

- Osservazione: $b^k = (\underbrace{100\dots 0}_{k \text{ cifre}})_b, \forall b$

Infatti la forma polinomiale di b^k in base b è:

$$b^k = 1 \cdot b^k + 0 \cdot b^{k-1} + \dots + 0 \cdot b^0 \quad \text{da cui:} \quad b^k = (\underbrace{100\dots 0}_{k \text{ cifre}})_b, \forall b$$

Il Complemento a $b-1$

- Definizione:

Dato $N \in \mathcal{N}$ rappresentato in base b su k cifre si definisce complemento a $b-1$ di N il numero $C_{b-1}(N)$ in base b tale che:

$$N + C_{b-1}(N) = b^k - 1$$

Es: $C_9(68) = 10^2 - 1 - 68 = 99 - 68 = 31$
su $k=2$ cifre

- Osservazione: $b^k - 1 = \underbrace{((b-1)(b-1)\dots(b-1))}_k, \forall b$

Infatti:

$$\begin{array}{r} b^k = (1 \quad 0 \quad \dots 0)_b \quad - \\ 1 = (0 \quad 0 \quad \dots 1)_b \quad = \\ \hline (0 \quad (b-1) \quad \dots (b-1))_b \end{array}$$

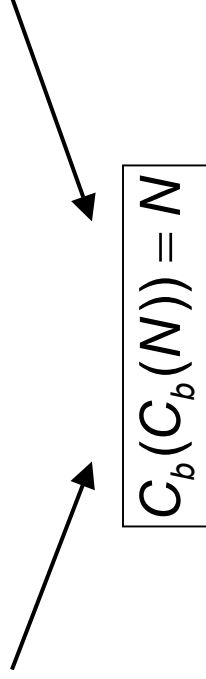
Proprietà del Complemento

1. Dalla definizione di complemento si ha:

$$\forall b \forall N, C_b(N) = C_{b-1}(N) + 1$$

2. $C_b(C_b(N)) = N$ $C_{b-1}(C_{b-1}(N)) = N$

Dim: $X = C_b(N) = b^k - N$ $C_b(X) = b^k - X = b^k - b^k + N = N$



Importante: Durante le sottrazioni per il calcolo del complemento a $b-1$ non si devono effettuare operazioni di prestito.

Complemento a 2

- Es. di complemento a 2 su $k=4$ bit:

$$C_2((1011)_2) = \underbrace{10000}_{2^4} - (1011)_2 = (0101)_2$$

- Regole pratiche per il calcolo del complemento a 2:

Regola (a):

1. Copia dei bit a partire da quello meno significativo fino al primo 1 incluso;
2. Inversione dei bit rimanenti ($0 \leftrightarrow 1$).

Regola (b):

1. Inversione di tutti i bit ($0 \leftrightarrow 1$) e somma di 1.

Complemento a 1

- Es. di complemento a 1 su $k=4$ bit:

$$C_1((1011)_2) = \underbrace{10000}_{2^4-1} - 1 - (1011)_2 = (0100)_2$$

- Regola pratica per il calcolo del complemento a 1:
 1. Inversione di tutti i bit ($0 \leftrightarrow 1$).

Teorema del Complemento

Hp) X, Y rappresentati su k cifre;

Ts) $X - Y = X + C_b(Y)$ su k cifre.

$$X \geq Y$$

Dimostrazione:

$$X - Y = X - Y + b^k - b^k = X + C_b(Y) - b^k$$

$$\left. \begin{array}{l} X \geq Y \\ Y + C_b(Y) = b^k \end{array} \right\} X + C_b(Y) \geq b^k = \underbrace{100\dots 0}_{k \text{ cifre}}$$

Quindi sulla $k+1$ -esima cifra (c_k) di $X+C_b(Y)$ c'è almeno 1 e al più 1 (essendo c_k un riporto della somma di due cifre di posto k -esimo)

$$\Rightarrow c_k = 1$$

$$\begin{array}{lcl} X + C_b(Y) & \Leftrightarrow & 1c_{k-1}c_{k-2}\dots c_0 - \\ b^k & \Leftrightarrow & \underline{100\dots 0} = \\ X + C_b(Y) \text{ su } k \text{ cifre} & \Leftrightarrow & 0c_{k-1}c_{k-2}\dots c_0 \end{array}$$

$X + C_b(Y) - b^k \Leftrightarrow c_{k-1}c_{k-2}\dots c_0 \Leftrightarrow X + C_b(Y) \text{ su } k \text{ cifre}$

Sottrazioni con il Complemento

1. Completare se necessario il minuendo e il sottraendo con lo stesso numero di cifre (k cifre);
2. Calcolare il complemento a b del sottraendo;
3. $X + C_b(Y)$;
4. Si considerano le prime k cifre della somma (equivalente a sottrarre b^k al risultato).

- Esempi:

- In base 10 su 2 cifre

$$32 - 15 \Rightarrow 32 + C_{10}(15) = 32 + 85 = 117 \Rightarrow 17 \text{ su 2 cifre}$$

- In base 2 su 4 cifre

$$(1101)_2 - (101)_2 = (1101)_2 - (0101)_2 \Rightarrow$$

$$\Rightarrow (1101)_2 + C_2(0101) = (1101)_2 + (1011)_2 = (11000)_2 \Rightarrow (1000)_2 \text{ su 4 cifre}$$