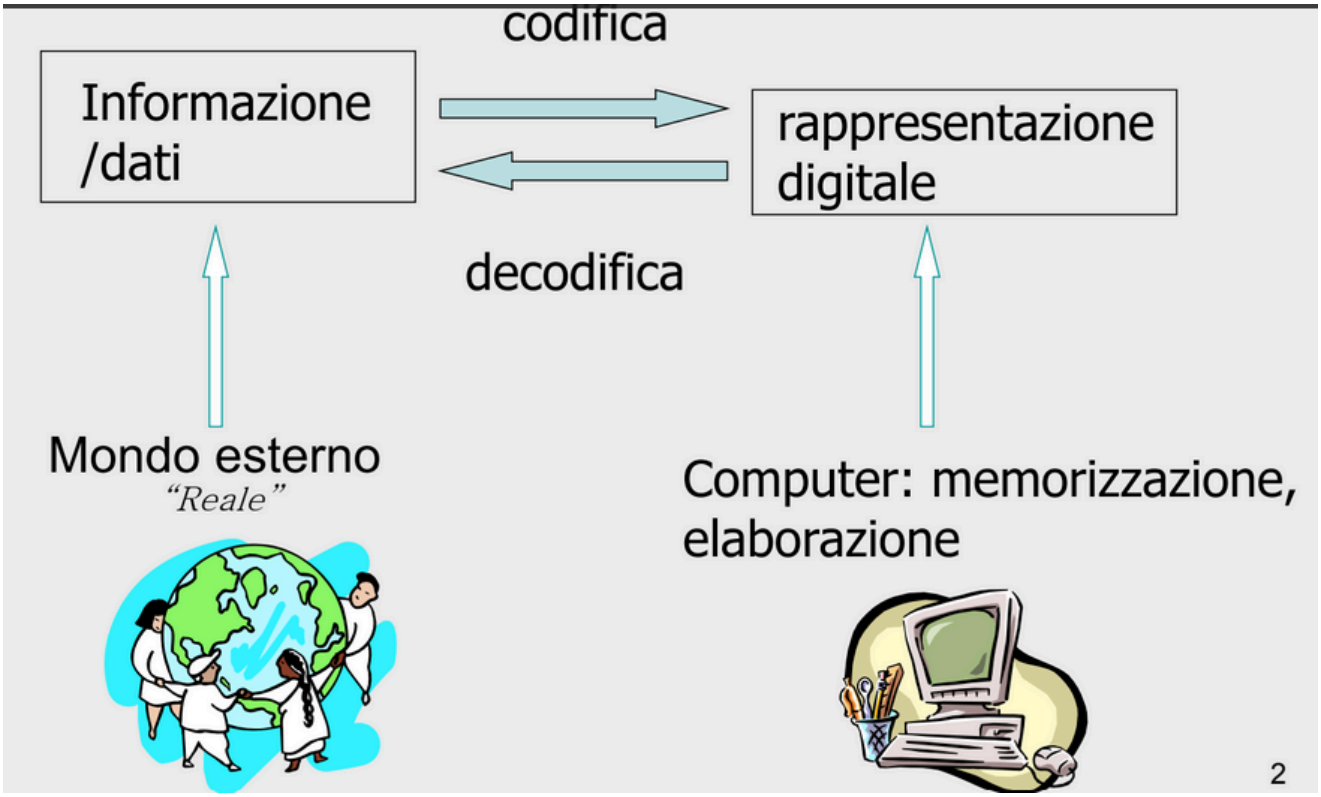
2 - Rappresentazione dell'informazione

Sull'hardware corrisponde un segnale fisico che si trasforma in 0 e 1, ma come fa un informazione ad essere tradotta?
Nella trasformazione dal mondo reale a quello digitale troviamo questa similitudine rappresentata in questo schema



Bit, byte e word

La differenza tra il mondo reale e il mondo digitalizzato è la presenza dei numeri reali, nel calcolatore ci sono dei segnali discreti a tempi discreti, che si trasformano in valori quantizzati e campionati.
L'informatico per semplicità ha deciso di passare dai 0 ai 5V ai 0 e 1,battezandolo come **Bit**(Binary digit, Cifra Binaria),gli si associa a 1 il significato di vero e 0 il significato di falso

Per poter rappresentare un numero maggiore di informazione si usano le sequenze di bit, il processo che fa corrispondere ad un dato reale una sequenza di bit è la **codifica di informazione**.
La codifica di informazione non è altro che un modo per tradurre le sequenze di N bit, in modo esponenziale(2^n),infatti i sistemi moderni memorizzano e manipolano miliardi di questi bit, memorizzati in stringhe.

1 bit = 2 combinazioni
2 bit = 4 combinazioni

3 bot = 8 combinazioni
n bit = 2^n combinazioni

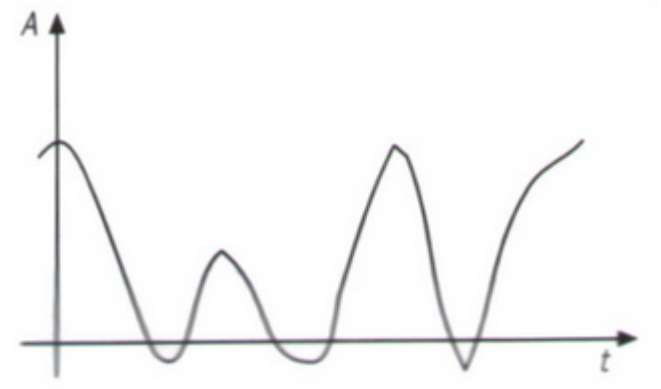
Quando si parla di **Byte**,noi parliamo di 8 bit in sequenza
I moderni sistemi ragionano in multipli

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

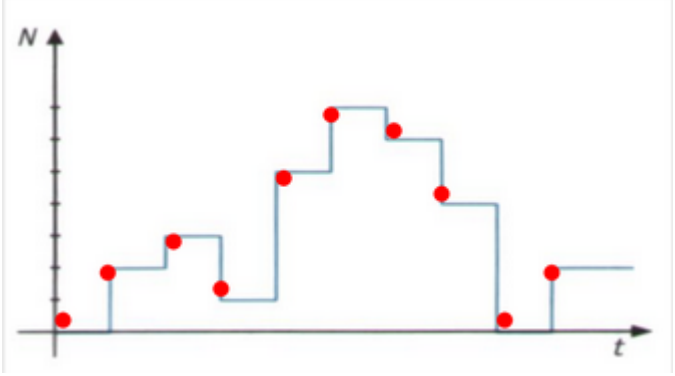
Rappresentazione digitale dei segnali

Tutto quello che viviamo nel mondo reale è un segnale, che si contraddistingue in segnale **analogico** o **digitale**:

- Il segnale analogico è un segnale tempo continuo che può contenere tutti gli infinti, valori di grandezza fisica osservabile che rappresenta contenuti all'interno di un determinato range, i valori sono continui (infiniti) in un intervallo e non numerabile



- Il segnale digitale è un segnale di tempo-discreto che all'interno di un determinato range può assumere solo un numero discreto (numerabile) di valore, i suoi vantaggi sono l'uso di un linguaggio binario, ha una costanza di qualità del tempo e la possibilità di essere compresso



Per convertire un segnale analogico in digitale è necessario fare due azioni:

- **Campionamento**: trasformare un segnale tempo continua in un segnale tempo-discreto, operando sui tempi, che son prestabiliti a intervalli regolari T
- **Quantizzazione**: trasformare un segnale a valori continui a valori discreti, operando sulle ampiezze, più è alto il numero di bit e più il margine di errore che si può commettere, cioè si riduce la distanza media tra il valore campionato e il suo valore quantizzato

Problemi legati al campionamento

Il campionamento va definito e tenuto costante per tutto il suo periodo(esempio: ogni 10 minuti)
Sbagliare il campionamento comporta:

- La perdita di informazioni nel caso sia infrequente o una distanza troppo alta
- L'eccesso di dati nel caso sia troppo frequente (comparsa di dati non presenti nell'originale,**Aliasing**)

Per determinare esattamente la frequenza di campionamento minima (la più bassa da poter prendere) si usa il teorema di Shannon

Teorema di Nyquist-Shannon: Dato un segnale con larghezza di banda B finita e nota, la frequenza minima di campionamento per poter ricostruire correttamente tale segnale deve pari ad almeno $2 \times B$
Un segnale campionato con frequenza pari a f_0 ,avrà massima frequenza riscrivibile (quantizzazione) pari a $f_0/2$

Nella quantizzazione deve essere scelto un errore di quantizzazione disposto a tollerare, in base a quanto si è disposti a perdere informazioni

Codifica dei dati: Sistema Numerico

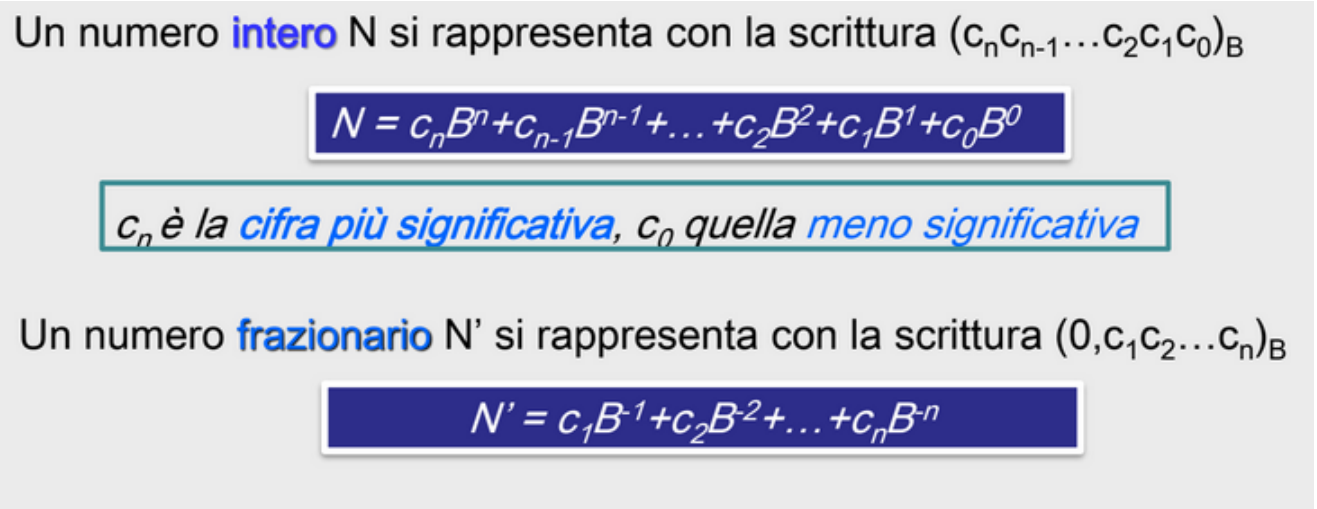
Un sistema numerico viene determinato per mezzo di

- Un insieme limitato di simboli (cifre) che rappresentano quantità prestabilite
  Esempio: sistema decimale, sistema binario, sistema romano
- Regole per costruire i numeri
  1. Sistemi non posizionali, come quello Romano additivo
  2. Sistemi posizionali, come quello decimale, il valore delle cifre dipende dalla posizione all'interno del numero
- Regole che consentono di eseguire operazioni tra i numeri

Sistema in base B

- La base definisce il numero di cifre diverse dal sistema di numerazione

- La cifra di minore valore (in termine di valore assoluto) è sempre lo 0, le altre sono nell'ordine (1,2...$B-1$)
- Se $B > 10$ occorre introdurre $B-10$ simboli in aggiunta delle cifre decimali
  Esempio: Sistema esadecimale:0,1,2...,A,B,C,D,E,F.

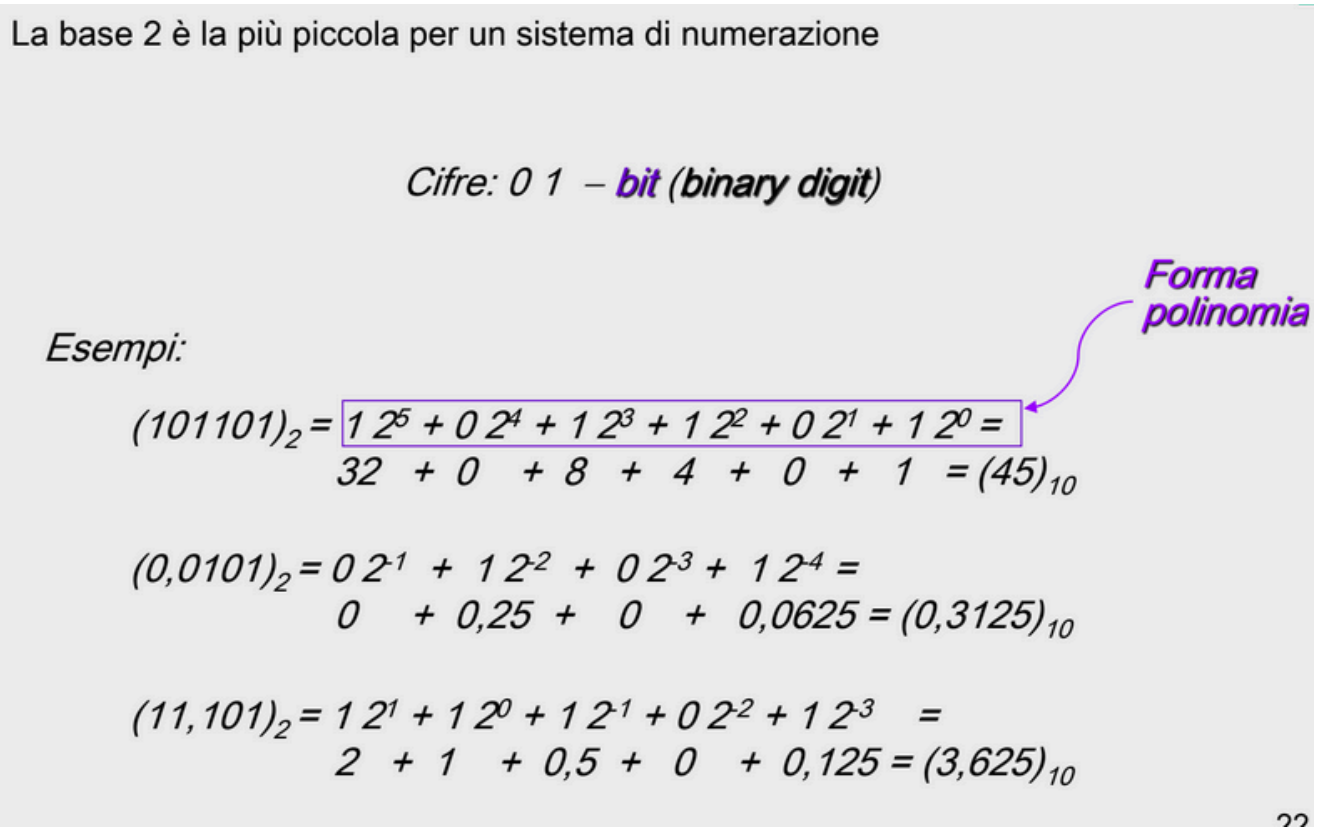


Numeri interi senza segno

Con n cifre, in base B, si rappresentano tutti i numeri interi positivi da 0 a $B^n - 1$ con B che sono numeri distinti

Sistema in base binario

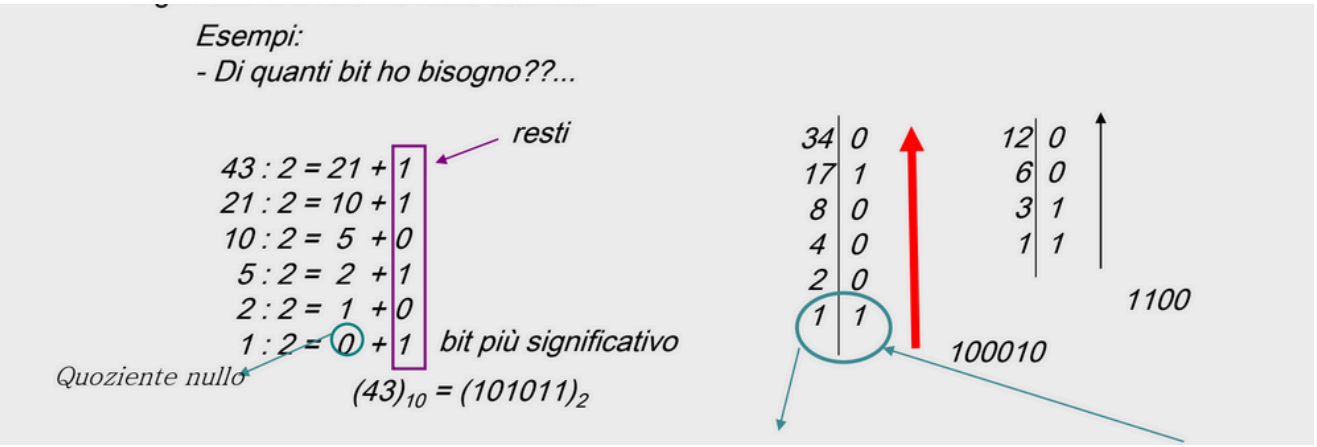
La base 2 è la più piccola per un sistema di numerazione



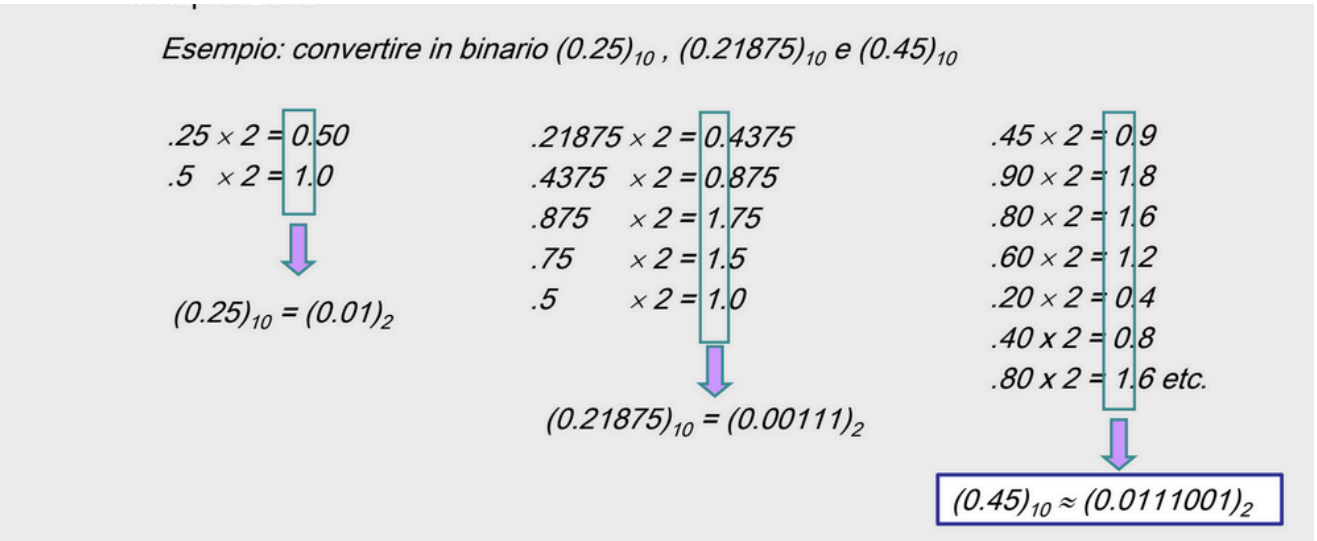
Conversione di base

Dec2bin

Libreria presente nei linguaggi di programmazione per convertire un numero intero decimale in binario, si ottiene questo procedimento tramite la divisione per 2 fino ad ottenere un quoziente nullo, le cifre del resto sono le cifre del numero, la cifra più significativa è sempre l'ultima



Nel caso ci sia un numero binario si moltiplica il numero frazionario per 2,leggendolo dall'alto verso il basso



Un numero come il 0.45 non è possibile rappresentarlo in una macchina, poiché è un numero infinito nel suo essere decimale con la virgola
Chi decide quanta dimensione può essere un eventuale frazionario numero è la macchina e la sua architettura (32 o 64 bit)

Sistema esadecimale

Nel sistema esadecimale, dove troviamo le lettere, possiamo utilizzare sempre la formula polinomiale, con un numero di bit minore

HEX2BIN

- Un numero binario di 4n bit corrisponde a un numero esadecimale di n cifre

Esempio: 32 bit corrispondono a 8 cifre esadecimali

1101	1001	0001	1011	0100	0011	0111	1111
D	9	1	B	4	3	7	F

$(D91B437F)_{16}$

Esempio: 16 bit corrispondono a 4 cifre esadecimali

0000	0000	1111	1111
0	0	F	F

$(00FF)_{16}$

Numeri binari negativi

Per rappresentare numeri con segno negativo in binario occorre utilizzare 1 bit per definire il segno del numero, si possono usare 3 tecniche di codifica:

- Modulo e segno
- Complemento a 1 (poco usato)
- Complemento a 2

Rappresentazione modulo e segno

Si usa un bit per rappresentare il segno del numero:
Il bit più significativo rappresenta il segno: 0 per i numeri positivi e 1 per i negativi, infatti con n cifre si possono rappresentare tutti i numeri compresi fra $-(2^{n-1}-1)$ e $+(2^{n-1}-1)$

Ad esempio: numeri a 4 cifre

0000 +0	1000 -8
0001 +1	1001 -7
0010 +2	1010 -6
0011 +3	1011 -5
0100 +4	1100 -4
0101 +5	1101 -3
0110 +6	1110 -2
0111 +7	1111 -1

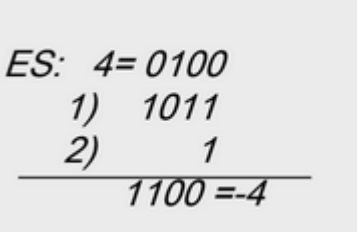
Nota: 0111 +7
1000 -8

Rappresentazione in complemento a 2

Il complemento a 2 di un numero binario $(N)_2$ a n cifre è il numero $2^n - (N)_2$, il bit più a sinistra indica sempre il segno.

Un numero positivo è individuato dal primo bit uguale a 0 (rappresentazione modulo e segno),per ottenerne il suo negativo rispetto al modulo e segno si procede in due passi:

1. Si sostituiscono tutti gli uno con degli zero e viceversa (complemento a 1),cioè si scambia il numero positivo con quello contrario
2. Si aggiunge 1 al risultato (somma binaria simile a quella decimale),si riporta 1 se la somma è maggiore di 1



Si possono rappresentare tutti i numeri da $-(2^{n-1})$ a $+(2^{n-1}-1)$

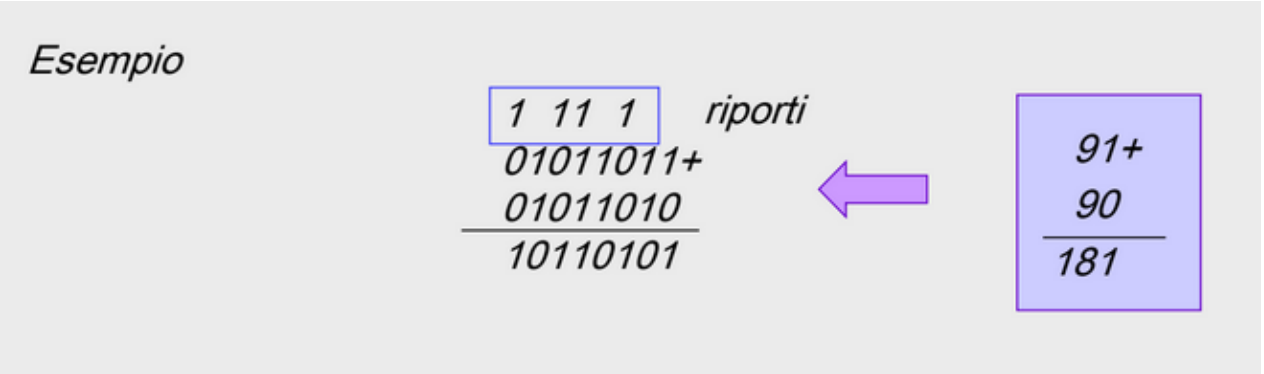
Il metodo di complemento a due quindi rende più facile trattare i numeri negativi senza trattarli separatamente rispetto alla rappresentazione modulo e segno

(Tipica domanda d'esame: il complemento a 2 ha lo stesso problema del modulo?)

Addizione e sottrazione binaria

L'addizione binaria e la sua sottrazione sono elementi che nella macchina vengono trattati nello stesso modo.
Le possibilità per l'addizione sono 4:

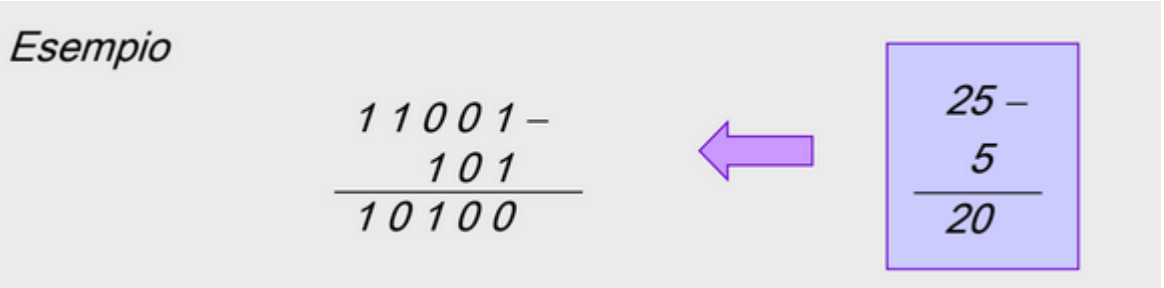
- $0+0=0$
- $1+0=1$
- $0+1=1$
- $1+1=1$ con il riporto di 1



Le possibilità per la sottrazione sono 4:

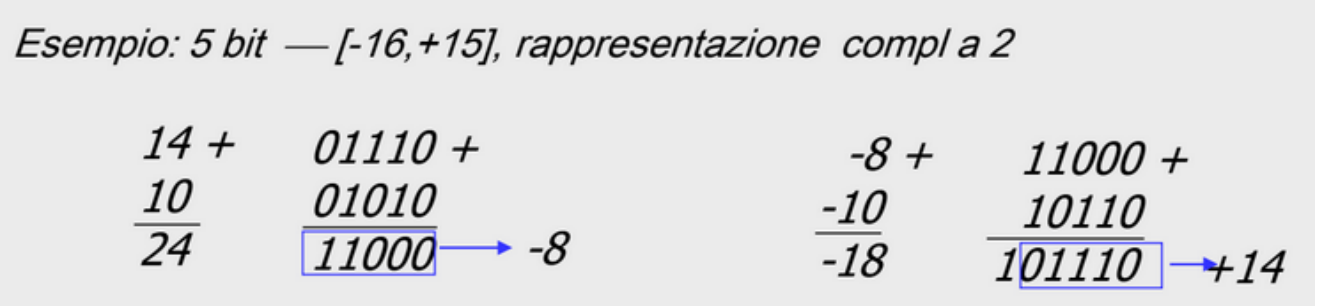
- $0-0=0$
- $1-0=1$
- $1-1=0$

- $0 - 1 = 1$ con il prestito (borrow) di 1 cifra precedente a sinistra



Overflow

Si ha un overflow quando il risultato di un operazione tra due numeri non è rappresentabile correttamente con gli N bit (limite superiore o inferiore), un programma darà sicuramente errore ,per evitarlo occorre aumentare il numero di bit utilizzati per rappresentare gli operandi (operazione non possibile se il calcolatore è già stato progettato, come 32 o 64 bit)
In caso di overflow l'hardware lo rileva e segnala un errore(bit di overflow nella PSW del processore)
In modo semplice, l'overflow accade quando un riporto(carry) va ad intaccare il segno ,cambiando completamente valore



Notazione in virgola mobile

Ogni numero può essere rappresentato come prodotto delle sue cifre significative per una potenza della base il cui valor dipende dalla posizione della virgola nel numero di partenza
Esempio:

$0.0012 = 12 * 10^{-}4$
$35.5 = 355 * 10^{-}1$

Esiste uno standard preciso (IEEE 754) per la rappresentazione in virgola mobile, come:

- **singola precisione** (32 bit = 4byte),
- **doppia precisione** (64 bit = 8 byte)
  Un qualsiasi numero può essere espresso come $N = \pm M * r^E$
  Dove:
- M=Mantissa (cifre rappresentative del numero)
- r= Base del sistema di numerazione
- E=Esponente della base
  N.B la mantissa contiene sempre solo le cifre significative

La base può essere omessa dato che nel calcolatore è sempre 2

Singola precisione

- Considerazioni sulla mantissa:
  - M intero
  - $1/r <= 1$ Mantissa normalizzata
- Considerazioni sull'esponente:
  - Numero intero relativo in complemento a 2

Un problema importante del floating point è quello di poter utilizzare numeri veramente piccoli, più si lavora con numeri grandi e più la precisione è minore
Per poter lavorare con un numero grande, si può usare il logaritmo(log) per poterlo rendere il numero più piccolo possibile nel caso la precisione sia richiesta durante delle operazioni.

Underflow

Accade quando si cerca di rappresentare un numero più piccolo del più piccolo vicino allo 0 ma a macchina non riesce per limite architetturale

Rappresentazione dei caratteri

I caratteri si possono distinguere in:

- Caratteri di comando (codice di trasmissione)
- Caratteri alfanumerici, lettere dell'alfabeto maiuscole, minuscole e cifre numeriche
- Segni come simboli di punteggiatura e operatori aritmetico-logici
  Ciascun carattere, qualunque esso sia(alfanumerico, punteggiatura, controllo) , deve essere rappresentato che compone testo deve essere rappresentato in codice binario, per farlo si utilizza il **codice ASCII**,uno standard americano per lo scambio di informazioni, utilizza 8 bit di cui 1 di parità (7 bit=128 configurazioni).
  ASCII è stato sviluppato unicamente per lingue anglosassoni e anche se si è provato ad espanderlo (IS646 o Latin-1 per altre 128 configurazioni,256 caratteri che introducono le code page) non basta per usarlo con altre lingue causa dei loro caratteri speciali(francese, tedesche etc.), per questo è stato inventato **l'UNICODE**,un sistema di codifica che assegna un numero univoco ad ogni carattere usato per scrittura dei testi in qualunque lingua
  La codifica è a 16 bit (2^{16} simboli diversi=65536 caratteri o code point), ed è universale, a discapito che pochi editor lo trattano e le dimensioni raddoppiano rispetto all'ASCII.
  Unicode ha comunque finito i code point, per questo si è deciso di sviluppare un altro

sistema di codifica, **UTF-8**, che prende i primi 128 caratteri dall'ASCII (che possono essere rappresentati in un singolo byte) e per tutti i restanti si imposta primo bit di tutti i byte a 1 per rappresentarli, la lunghezza di un carattere è a lunghezza variabile da 1 a 4 byte, ad oggi è uno standard utilizzatissimo soprattutto sul WWW ed ha ancora code point disponibili nel caso possano servire

Rappresentazione digitale dei suoni

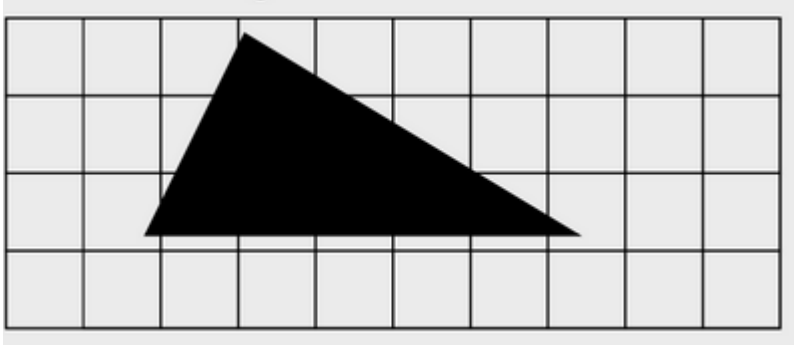
Codifica audio

Il suono è una rapida variazione di pressione, prodotta da una sorgente, in un mezzo elastico, tale variazione produce oscillazioni nelle particelle del mezzo;
Le oscillazioni sono spostamenti delle particelle intorno alla posizione di riposo e nella direzione della propagazione del suono, tali oscillazioni sono rappresentate da onde sonore
La soglia udibile da un umano è tra i 20hz e i 20khz
Un suono in versione grezza diventa di formato **Wave**,campionato a 44.1khz con 2 canali a 16 bit per ognuno. Questo formato per secondo occupa ben 0,17MB.

Algoritmi di compressione dei file audio

La musica quando ha bisogno di essere ridotta di dimensioni viene codificata con il famoso formato **MP3**,un formato lossy che sfrutta la conoscenza dell'udito umano e dei suoi limiti per ridurre le informazioni da memorizzare, esclude praticamente tutti i suoni poco forti concentrandosi su quelli più rumorosi, ha un fattore di riduzione del peso fino a 10 volte da un file grezzo come il Wav.
Nelle telecomunicazioni per codificare la voce di una chiamata viene usato il **PCM**,una codifica su 8Khz e quantizzazione su 8 bit che permette di avere un canale trasmissivo che consuma unicamente 64kbps,è stato creato unicamente per far passare la voce.
Siccome tenere un canale vocale attivo per un operatore è dispendioso si adotta anche la tecnologia del **VAD**(Voice Activation Detector),un dispositivo, prima incorporato lato operatore e ora lato hardware sui cellulari, che chiude il canale vocale quando non rileva nessuna attività vocale

Codifica delle immagini



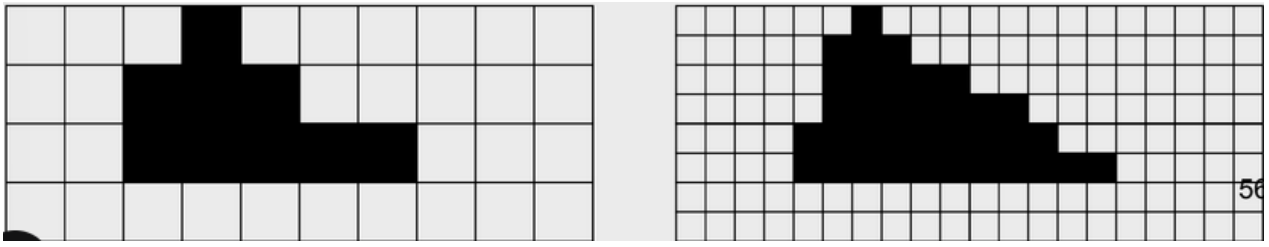
In un immagine del genere, che è bianca e nera, si può codificare in binario:

- 0 viene utilizzato per la codifica di un pixel bianco (maggiormente)
- 1 viene utilizzato per la codifica di un pixel nero(maggiormente)

L'immagine potrebbe essere codificata anche in questo modo

0	0	0	1	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0

Risoluzione e **dimensione** tra di loro sono diversi, come si vede da questa immagine:



In questo esempio la dimensione dell'oggetto non cambia, ma la risoluzione cambia, cioè la loro frequenza di campionamento aumenta.

Nell'ambito delle immagini si parla di **DPI** (Dot per Inch, numero di pixel su una linea lunga 2,54cm) per la risoluzione

Per ognuno dei pixel bisogna poi effettuare la quantizzazione.

Le operazioni di campionamento e quantizzazione avvengono nello spazio (2D) invece che nel tempo quando si parla di immagini.
Nel campionamento di un immagine si divide l'immagine in sottoinsiemi(**pixel**) in una griglia per cui si deve prendere un campione che poi rientra in questo sottoinsieme per

rappresentarlo.
La quantizzazione di un un immagine è la codifica del colore associato ad ogni pixel, utilizzando recenti formati si va sui 32 bit (4 byte) per pixel di cui:

- 8 bit per ogni componente fondamentale RGB
- Altri 8 per le trasparenza

Spazio dei colori

Esistono due tipi di sistemi per poter rappresentare le immagini:

- **RGB**: si mischiano i colori rosso, verde e blu, creando poi tutti i sottoinsiemi di colori. Questo sistema richiede energia per poter effettivamente dare colore all'immagine su sistemi come monitor e schermi.
- **CMY**:si sottraggono i colori ciano, magenta e giallo, hanno meno luminosità rispetto al sistema RGB, non consuma energia ed è usato prevalentemente quando bisogna mischiare sostanze colorate come inchiostri o tempere.

Per lo stesso colore (prendiamo l'esempio del Blu), se nel sistema RGB esiste direttamente, nel CMY viene ottenuto mischiando i colori

Rappresentazione dei colori

Per far capire alla macchina poi come rappresentare un colore (effettuarne quindi la quantizzazione), che la macchina non comprende veramente rispetto ad umano, bisogna far corrispondere ad ogni pixel un byte (2^8 ,256), codificando i colori in una tabella per poterli far rispondere in una matrice
La dimensione non è data da altro che il numero di pixel della base per il numero di pixel dell'altezza
La profondità dell'immagine è il numero di bit che usiamo per rappresentare il colore di un singolo pixel dell'immagine
Per calcolare il numero di bit per ogni immagine si usa la formula $dimensione * immagine$

Algoritmi di compressione delle immagini

Delle foto in formato grezzo(bitmap, .bmp) arrivano a pesare 38,29MB(esempio con una macchina da 10 Megapixel con risoluzione 3872x2592 pixel {$3872 * 2592 * 4B$}), una dimensione che è troppo grande, per una foto con così lievi cambiamenti di colore che l'occhio umano è invisibile.
Per poter quindi ridurre la dimensione dell'immagine senza che l'occhio umano se ne accorga si usano algoritmi di compressione, come il formato JPEG che sfrutta la sensibilità dell'occhio a diversi livelli di luminosità e individua per un area 8x8 unicamente i punti significativi percettibili.
Un altro formato possibile è quello delle GIF, che limiti il numero di colori che compaiono in essa scegliendo quelli più frequenti, perdendone o sostituendone alcune.

Informazione multimediale

I video sono delle sequenze di immagini(fotogrammi, frame) a 24 o 30 fps al secondo (fotogrammi per secondo) che sono accompagnate da un flusso audio.
Unendo quindi tutti i sistemi di codifica di audio e immagine grezzi separatamente verrebbe fuori per ogni minuto di filmato peserebbe 650MB, un grande problema di spazio, allora per poter ridurre questo spazio si comprime

Nei video per poter risparmiare spazio si codifica soltanto le differenze tra un fotogramma ed un altro, non codificandolo completamente ogni frame singolo, il nome di questa tecnica è il **Key frame**.

La dimensione di un filmato è dipendente da almeno 5 fattori:

- **Lunghezza del filmato**
- **Risoluzione grafica** (quanto è fitta la griglia che usiamo per digitalizzare i fotogrammi)
- **L'ampiezza della palette** di colori utilizzata (una troppo ristretta rende i colori poco realistici)
- **Numero di fotogrammi** per secondo (una frequenza limitata fa apparire la sequenza poco fluida)
- **Qualità dell'audio**

Compressione

Esistono due tipi di compressione:

- **Lossy**: con perdita di dati, minor peso generale
- **Lossless**: senza perdita di dati, minor peso ma maggiore di un lossy
  Ma come si fa ad avere una codifica lossless? La chiave sta nella riorganizzazione dei dati, un esempio vi è in questa tabella:

	1	**2**	**3**
1	a	b	
2		a	a
3		a	

  Mettiamo caso che dobbiamo memorizzare per ogni cella di questa tabella posizione e valore all'interno (con 8 bit per posizione di riga, colonna e valore),il risultato sarebbe questo: (1,1,a) (1,2,b) (2,2,a) (2,3,a) (3,2,a) Con qualche calcolo, notiamo che arriviamo a far pesare tutto questo file $(3 \cdot 8)\cdot5$ bit, arriviamo quindi a 15 byte di informazioni

Se invece decidiamo di eliminare la ridondanza di dati distinguendo per a per b otterremo questo risultato:

(1,1,2,2,2,3,3,2,a)
(1,2,b)
Con qualche calcolo, notiamo che arriviamo a far pesare tutto questo file $(9 \cdot 8) + (3 \cdot 8)$ bit, arriviamo quindi a 12 byte di informazioni, meno dei 15 byte ma senza perdere nessun tipo di informazione

Algoritmi di compressione video

Bisogna differenziare da video in download che sono memorizzati in modo standard con determinati formati (Quicktime, Avi) e i video in streaming, questi ultimi sono trasmessi man mano che arrivano in ordine n modo continuo e poi vengono buttati via per non occupare memoria.