Vittorio Cortosi

Elaborazione digitale audio e musica

Index

La Natura fisica del suono

Il suono è un fenomeno affascinante che permea la nostra vita quotidiana in modi molteplici. Sebbene spesso lo percepiamo come un'esperienza sensoriale, è importante sottolineare che il suono ha una radice profondamente radicata nella fisica. Infatti, la comprensione della natura fisica del suono ci offre una prospettiva più approfondita su come si origina, si propaga e interagisce con l'ambiente che ci circonda.

Le Onde

Per comprendere appieno la natura del suono, è fondamentale acquisire una conoscenza di base sul concetto di onde. Le onde sono perturbazioni che si propagano attraverso un mezzo o uno spazio. Si definisce sorgende dell'onda ciò che produce la perturbazione dello spazio che la circonda.

Più nel dettaglio il suono (ma in realtà tutti i tipi di onde) è un trasporto di energia senza che avvenga trasporto di materia. Inoltre, le onde si propagano ad una ben definita velocità, che però dipende dal mezzo in cui viaggiano.

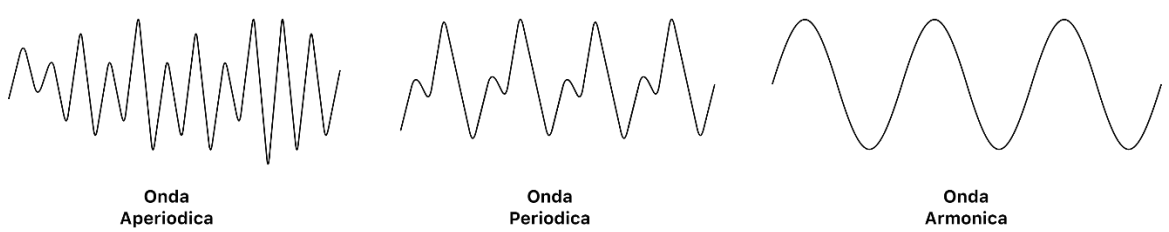
Questa GIF, mostra come la perturbazione (o vibrazione) sia provocata dall'oggetto rosso a sinistra (supponiamo sia la cassa di un altoparlante). I puntini neri che si notano sono molecole del mezzo (aria, gas o fluido). La membrana, vibrando, addensa le molecole più vicino a lei, spingendole verso una direzione (che è la direzione di propagazione dell'onda) creando zone di area compressa che si propaga e lasciando dietro di se una zona di rarefazione, seguita da una zona di compressione, e così via. La compressione si riferisce alla regione di un'onda in cui le particelle sono più vicine l'una all'altra, mentre la rarefazione si riferisce alla regione di un'onda longitudinale in cui le particelle sono più distanti l'una dall'altra.

I punti rossi evidenziano come le onde sono "un trasporto di energia senza che avvenga trasporto di materia". Infatti le molecole effettuano solo delle piccole oscillazioni seguendo la direzione di propagazione dell'onda. le molecole dendono a tornare al loro stato iniziale, non c'è trasporto di materia, ma l'energia si propaga verso la direzione dell'onda.

L'onda sonora, è definita un **onda longitudinale** che, a differenza di quelle trasversali, la direzione di propagazione dell'onda, è uguale alla direzione di spostamendo delle particelle. Nelle **onde trasversali**, come spiegato da questa gif, i punti rossi oscillano in una direzione perpendicolare (alto – basso) rispetto alla direzione di propagazione dell'onda (questo è un esempio di onde marine).

Profili d'onda

Il profilo di un onda (o la forma d'onda) specifica la forma che l'onda assume nel tempo. Tra i profili d'onda più comuni si distinguono quelli aperiodici, periodici e armonici, ognuno dei quali possiede caratteristiche uniche.



Un suono aperiodico è caratterizzato da un'onda sonora che non si ripete regolarmente nel tempo. Questo tipo di onda è spesso associato a suoni complessi (o rumori), come il fruscio del vento o il suono della pioggia. Gli eventi che generano onde sonore aperiodiche sono tipicamente casuali e irregolari, producendo una serie di picchi e valli che non seguono un pattern regolare.

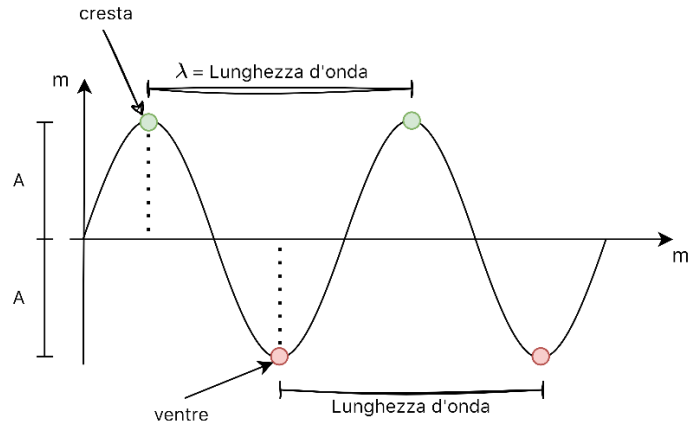
Passando ai suoni periodici, siamo di fronte a un profilo d'onda che si ripete regolarmente nel tempo. Questo significa che l'onda presenta una forma ciclica, con un periodo ben definito in cui si verifica un'intera sequenza di ripetizioni dell'onda. I suoni periodici sono spesso associati a suoni musicali, come le note suonate da un violino o una chitarra.

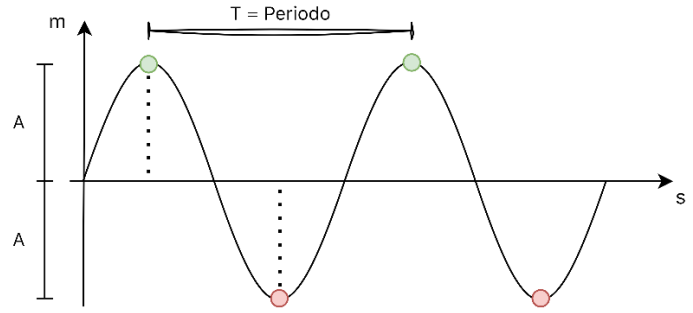
Infine, arriviamo ai suoni armonici. Un suono armonico è caratterizzato da un profilo d'onda che segue una forma sinusoidale. Non tutte le onde periodiche sono quindi sinusoidali.

A livello percettivo, la forma d'onda è il parametro percettivo del timbro.

Grandezze caratteristiche delle onde periodiche

Si parla di **creste e di ventri** quando si parla di massimo e minimo spostamento, in particolare si identificano come creste, i punti dell'onda corrispondenti al massimo spostamento verso l'alto. I ventri corrispondono al massimo spostamento verso il basso:





Ampiezza

L'ampiezza è l'altezza di un'onda rispetto alla sua posizione di equilibrio. Rappresenta l'intensità dell'onda e indica quanto è "grande" l'onda nella sua variazione rispetto al punto di riposo.

Nel contesto delle onde sonore, l'ampiezza è correlata alla percezione soggettiva di volume o intensità del suono. Un'onda sonora con un'ampiezza maggiore avrà un suono percepito come più forte o più intenso, mentre un'ampiezza minore produrrà un suono percepito come più debole.

L'ampiezza viene solitamente rappresentata come un valore positivo (nell'immagine indicato con 'A'), indicando la distanza massima che l'onda si discosta dalla sua posizione di equilibrio. Tuttavia, può anche essere negativa, indicando una deviazione nella direzione opposta.

Lunghezza d’onda

La lunghezza d’onda rappresenta la distanza tra due punti equivalenti su un'onda, cioè la distanza minima tra due punti che oscillano in fase.

Nel caso delle onde sonore, la lunghezza d'onda corrisponde alla distanza tra due creste (punti di massima escursione positiva) consecutive o tra due valli (punti di massima escursione negativa) consecutive. È importante sottolineare che la lunghezza d'onda è inversamente proporzionale alla frequenza dell'onda. In altre parole, onde con frequenze più alte hanno lunghezze d'onda più corte, mentre onde con frequenze più basse hanno lunghezze d'onda più lunghe.

La lunghezza d'onda è rappresentata dal simbolo lambda (λ) e viene misurata generalmente in metri (m) o nelle sue sottomultiple come il millimetro (mm) o il micrometro (µm). Ad esempio, nel campo delle onde sonore udibili, la lunghezza d'onda varia da circa 17 metri (per onde con una frequenza di 20 Hz) a pochi millimetri (per onde con una frequenza di 20.000 Hz o 20 kHz).

Periodo e frequenza

Se ci spostiamo nel dominio del tempo (il secondo grafico) si può definire il periodo, come il tempo impiegato dalla perturbazione per percorrere la lunghezza d’onda. La frequenza invece rappresenta il numero di cicli completi che un'onda compie in un secondo (misurata in Hz).

Il periodo di un'onda è inversamente proporzionale alla sua frequenza, ovvero un'onda con una frequenza alta avrà un periodo più breve, mentre un'onda con una frequenza bassa avrà un periodo più lungo. La relazione matematica tra periodo (T) e frequenza (f) è data dalla formula $T = 1/f$.

Pressione sonora ed Intensità sonora

La pressione sonora e l’intensità sonora, sono misure dell'energia acustica trasportata dalle onde sonore e rappresenta l'intensità del suono percepito dall'orecchio umano.

La pressione sonora è una grandezza fisica che descrive la forza esercitata dalle particelle dell'aria (rispetto alla pressione atmosferica) o di un altro mezzo di propagazione del suono durante la sua trasmissione ed è misurata in Pascal (Pa). Mentre l’intensità sonora è una misura della potenza o dell'energia trasportata da un'onda sonora che attraversa una determinata area e l’unità di misura del SI è il W/m^2

Tuttavia, queste grandezze non forniscono un'indicazione diretta dell'intensità del suono percepita dall'orecchio umano. Poiché l'orecchio umano è più sensibile a determinate frequenze rispetto ad altre, è necessario utilizzare una scala di misura che tenga conto di questa caratteristica.

Può essere quindi utile esprimere la pressione sonora in termini di decibel (dB) sonori quando si ha a che fare con l’udito degli esseri umani, dal momento che l'intensità percepita dall'orecchio è circa proporzionale al logaritmo della pressione sonora (Legge di Weber-Fechner).

Legge di Weber-Fechner

La legge di Weber-Fechner (1860) fu uno tra i primi tentativi di descrivere la relazione tra la portata fisica di uno stimolo e la percezione umana dell'intensità di tale stimolo.

La scoperta della relazione esistente tra stimolo e percezione è stata fatta da Weber in seguito all'esperimento consistente nell'incrementare di una certa quantità il peso di un oggetto sostenuto da un uomo. La percezione di tale stimolo (l'incremento di peso) risultò tanto meno accentuata quanto più pesante era l'oggetto. In altre parole aggiungere 1 kg ad un oggetto pesante 5 kg risultava essere percepito in maniera differente rispetto ad aggiungere 1 kg ad un oggetto il cui peso è di 30 kg. L'entità dell'incremento necessario perché la variazione venisse percepita dal soggetto non risultò costante.

La stessa cosa accade appunto per l'udito: un aumento di pressione sonora "X" ad ampiezze basse, risulterà in una percezione differente rispetto ad un aumento di pressione sonora (sempre di intensità X) ad ampiezze elevate.

SPL e SIL

Il Sound Pressure Level (SPL) e il Sound Intensity Level (SIL) sono due misure comuni utilizzate per quantificare l'intensità o la potenza del suono in decibel, ma differiscono nel modo in cui vengono calcolate e nelle informazioni che forniscono.

Il livello di pressione sonora (Sound Pressure Level - SPL) è una scala logaritmica utilizzata per quantificare e rappresentare la pressione sonora (o ampiezza) in modo più pratico ed efficace. Il livello di pressione sonora viene espresso in decibel (dB) ed è calcolato come il logaritmo del rapporto tra la pressione sonora misurata e una pressione di riferimento specifica, spesso definita come il livello di pressione sonora di riferimento uditivo umano (20 µPa):

$$L_{\mathrm{p}} = 10\ \log_{10}\left(\frac{p}{p_0}\right)^2 = 20\ \log_{10}\left(\frac{p}{p_0}\right)\ \mathrm{dB}$$

Dove '$\mathbf{p_0}$' è la soglia minima di udibilità (è circa la soglia uditiva a 1000 Hz) e '**p**' è il valore effettivo della pressione sonora che si vuole misurare.

La pressione di riferimento più comunemente utilizzata (in aria) è $\mathbf{p_0 = 20\ \mu Pa}$.

Sound Intensity Level (SIL) invece, è sempre il livello (una quantità logaritmica) dell'intensità di un suono rispetto a un valore di riferimento come SPL, ma la formula è leggermente diversa:

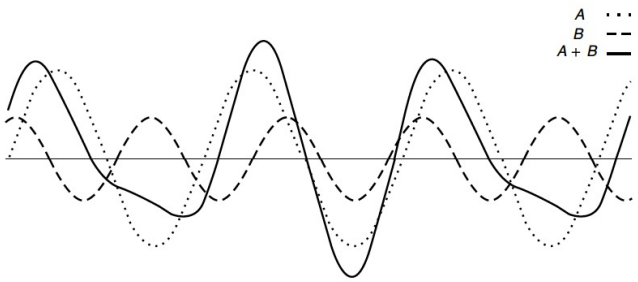
$$L_I = \frac{1}{2}\ln\left(\frac{I}{I_0}\right)\mathrm{Np} = \log_{10}\left(\frac{I}{I_0}\right)\mathrm{B} = 10\log_{10}\left(\frac{I}{I_0}\right)\mathrm{dB}$$

Dove '$\mathbf{I_0}$' è 1pW/m^2 ed '**I**' è il valore effettivo di intensità sonora che si vuole misurare.

Analisi e Trasformata di Fourier

L'analisi di Fourier è una tecnica matematica utilizzata per scomporre un segnale complesso in una serie di componenti più semplici, definite come componenti spettrali. Questa analisi si basa sul principio che qualsiasi forma d'onda periodica può essere rappresentata come una combinazione di onde sinusoidali di diverse frequenze, ampiezze e fasi.

Nel dettaglio stabilisce che qualunque forma d'onda periodica può essere approssimata quanto più si desideri sommando alla sinusoide avente la stessa frequenza dell'onda periodica (frequenza fondamentale o prima armonica) una serie di sinusoidi aventi frequenze multiple (chiamate armoniche) di quella dell'onda periodica, di ampiezza opportuna e in specifici rapporti di fase fra loro. La frequenza risultante del segnale, nel caso di un segnale periodico, sarà la frequenza della fondamentale (che è la minima frequenza presente). Ad esempio:

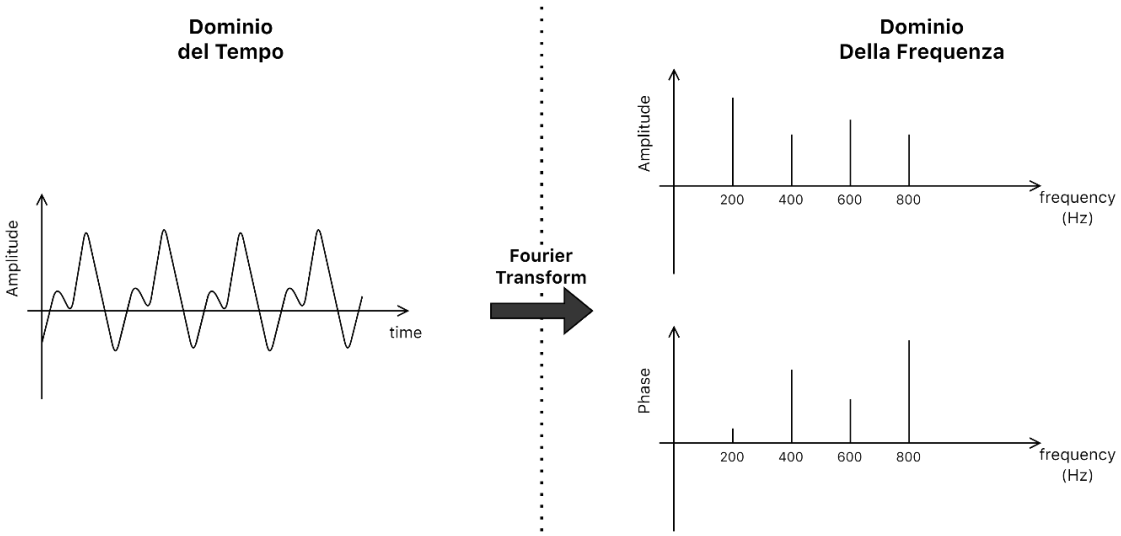


È possibile ricavare la curva continua (A + B) derivandola da due toni puri (A e B). Il teorema è valido anche per i suoni non periodici. In questo caso però, non esiste una relazione armonica tra le componenti, ma una relazione inarmonica.

La trasformata di Fourier è un algoritmo che permette di ottenere lo spettro di frequenza di un segnale, rivelando le diverse componenti che lo compongono. Questa rappresentazione spettrale fornisce informazioni dettagliate sulla distribuzione di energia delle diverse frequenze nel segnale, consentendo l'identificazione e l'analisi delle caratteristiche tonali e timbriche del suono.

Fin'ora si è visto solo la rappresentazione della forma d'onda, che è una rappresentazione del segnale in funzione del tempo (dominio del tempo).

Grazie a Fourier e alla sua traformata, è possibile rappresentare un suono anche nel **dominio della frequenza**.



In questo esempio, si è applicato la trasformata di Fourier al suono nel dominio del tempo. La trasformata ha prodotto la rappresentazione dello stesso suono ma nel dominio della frequenza. In particolare, la trasformata produrrà quelli che sono gli spettri del segnale (uno è lo spettro di ampiezza, e l'altro è lo spettro di fase): nello spettro di ampiezza, si notano tutte le componenti del segnale con i propri contributi in ampiezza mentre nello spettro di fase è indicato (di solito in gradi o radianti) la fase di ogni componente.

Quindi si può dire che esiste una sorta di rappresentazione duale del segnale: nel dominio del tempo e nel dominio della frequenza. Perché però ci interessa lavorare su un dominio rispetto all'altro? La risposta è: si utilizza una rappresentazione rispetto all'altra in base a che modifiche si vogliono effettuare. Se ad esempio si vuole applicare un filtro o modificare i contributi di ampiezza per banda (equalizzatore) si lavora sul dominio della frequenza. Se invece si vuole ridurre la durata di un brano si lavora nel dominio del tempo, non avrebbe senso lavorare in frequenza.

L'operazione di trasformata è invertibile: l'anti trasformata infatti, permette di tornare nel dominio del tempo avente gli spettri di ampiezza e fase.

Onda quadra

Per quanto riguarda le caratteristiche nel dominio della frequenza, per le onde quadre sono rilevanti le armoniche successive alla fondamentale; nel caso particolare di un'onda quadra avente duty cycle pari al 50% (il duty cycle è il rapporto tra il tempo in cui l'onda è positiva rispetto al tempo in cui l'onda è negativa) si può facilmente conoscerne la consistenza e l'ampiezza: sono presenti infatti esclusivamente le armoniche dispari, e in particolare la terza con ampiezza pari a

un terzo della fondamentale, la quinta armonica con ampiezza pari ad un quinto della fondamentale, e così via.

Le armoniche dispari presenti sono infinite, infatti, l'onda quadra è un onda con frequenze componenti molto alte.

Timbro

Il Timbro è quella caratteristica del suono per la quale, a parità di frequenza, è possibile distinguere due suoni prodotti da sorgenti diverse come ad es il suono di uno strumento da quello di un altro (vedi un flauto e una chitarra). Il timbro dipende dal contenuto spettrale del suono costituito dalla somma di componenti sinusoidali che si può studiare attraverso l'analisi di Fourier.

Battimenti

In acustica, un battimento è un pattern di interferenza tra due suoni di frequenze leggermente diverse, percepito come una variazione periodica di volume la cui velocità è la differenza delle due frequenze.

Spettrogramma

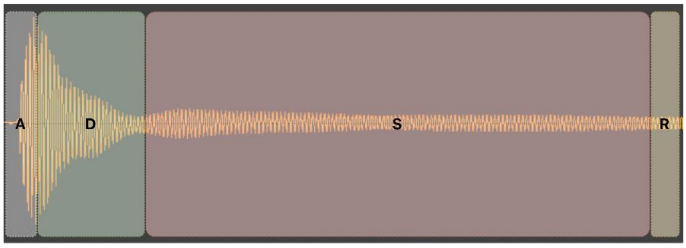
Lo spettrogramma, nell'ambito dell'analisi audio e della fonetica acustica, è la rappresentazione grafica - fornita da uno spettrografo - dell'intensità di un suono in funzione del tempo.

Dal punto di vista visivo, si configura come una heat map (o mappa di calore).

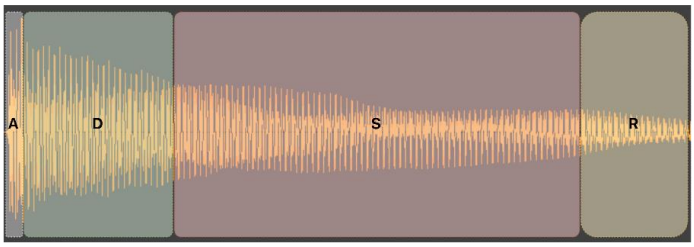
Inviluppo e ADSR

L'inviluppo descrive come la dinamica di un suono cambia nel tempo (come questo si evolve nel tempo), in particolare il modo in cui la sua ampiezza cambia nel tempo:

Inviluppo di un Violino

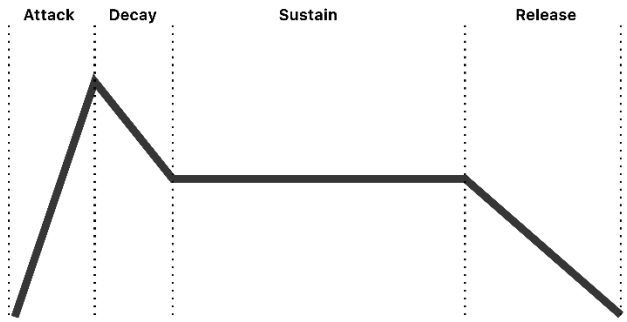


. . . . . .

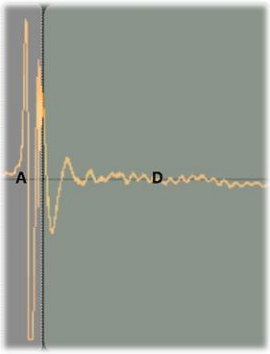


Inviluppo di una Chitarra

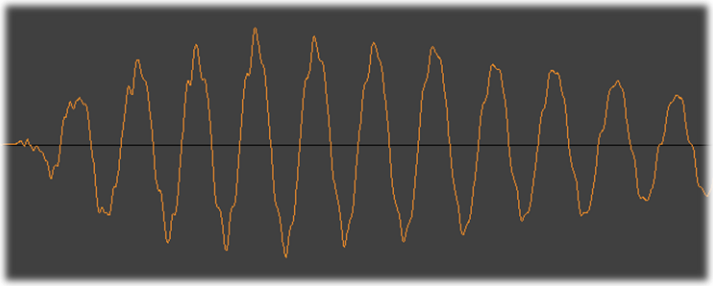
Come si comporta l'inviluppo di un violino (E chord) rispetto alla stessa corda pizzicata ma della chitarra. Riusciamo ad identificare 4 fasi in entrambi i suoni, queste fasi (o transitori) vengono identificati con la sigla **ADSR (Attack, Decay, Sustain, Release)**. Si può notare che i transitori per la chitarra sono differenti dal violino, La corda di violino ha una fase di sostegno più lunga ad esempio. Lo schema generale di un inviluppo ADSR:



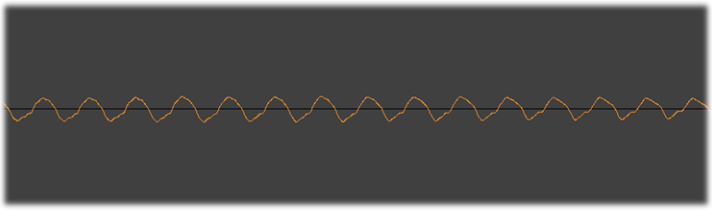
Non tutti i suoni però, posseggono tutte le fasi, ad esempio, uno schiocco di dita non possiede la fase di sostegno e rilascio, ma c'è un decadimento istantaneo:



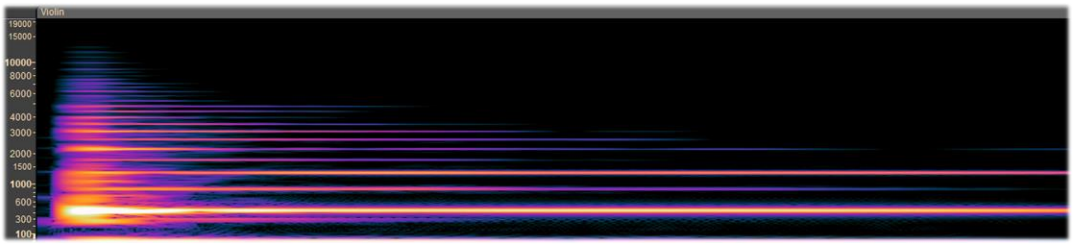
Inoltre se osserviamo le parti di attacco, ad esempio del violino, si nota che non si ha un'armonicità del suono, infatti le parti di attacco sono ricche di componenti spettrali inarmoniche.



Mentre nella fase di sostegno si nota un segnale più armonico:



Lo spettrogramma infatti mostra come l'attacco sia ricco di componenti spettrali (fino a 15KHz circa) mentre la fase di sostegno mostra un contributo di frequenze più armonico (in particolare la 440Hz è presente per tutta la durata del suono più tutte le armoniche):



Generatori di inviluppo

Il pianoforte è dotato di un pedale che permette di mantenere più o meno (a discrezione dell'artista) la fase di sostegno di una nota, questo è possibile grazie ad un generatore di inviluppo.

Formanti

Nella scienza del linguaggio e nella fonetica , una formante è l'ampio massimo spettrale che risulta da una risonanza acustica del tratto vocale umano. In acustica, una formante è generalmente definita come un ampio picco, o massimo locale, nello spettro. Per i suoni armonici, con questa definizione, la frequenza formante è talvolta presa come quella dell'armonicoche è maggiormente accresciuta da una risonanza. La differenza tra queste due definizioni risiede nel fatto che "formanti" caratterizzino i meccanismi di produzione di un suono o il suono prodotto stesso. In pratica, la frequenza di un picco spettrale differisce leggermente dalla frequenza di risonanza associata, tranne quando, per fortuna, le armoniche sono allineate con la frequenza di risonanza.

Si può dire che una stanza ha formanti caratteristici di quella particolare stanza, a causa delle sue risonanze, cioè, al modo in cui il suono si riflette dalle sue pareti e dagli oggetti. I formanti della stanza di questa natura si rafforzano enfatizzando frequenze specifiche e assorbendone altre.

Sia nel discorso che nelle stanze, le formanti sono tratti caratteristici delle risonanze dello spazio. Si dice che siano eccitati da sorgenti acustiche come la voce, e modellano (filtrano) i suoni delle sorgenti, ma non sono esse stesse sorgenti.

In sintesi, le formanti sono frequenze specifiche che emergono da risonanze acustiche all'interno del tratto vocale umano o di ambienti acustici. Esse riflettono come il suono si modifica e si adatta a causa delle caratteristiche dell'emissione sonora e dell'ambiente in cui si propaga.

Percezione

Rappresentazione duale del segnale (tempo lungo il nervo uditivo ce la successione delle ampiezze) invece membrana basilare le frequenze si distribuiscono (dominio della frequenza)

Le curve isofoniche

Non sono altro che un grafico ricavato da .. Hanno preso un campione di persone e le hanno messe in una camera anecoica. I suoni prodotti erano toni puri e venivano prodotti alla stessa ampiezza (pressione sonora) e veniva chiesto alle persone a che volume percepissero questi toni (se fossero volumi più alti o più bassi) nonostante venissero prodotti con la stessa ampiezza. I risultati furono che le persone percepivano (mantenendo la stessa ampiezza) a volume più basso le frequenze basse e a volume più alto le frequenze alte. Questo vuol dire che per percepire allo stesso volume delle frequenze basse/medio/alte, avrai bisogno di più pressione sonora sulle freauzene basse rispetto alle medio/alte. Ed è per questo che il SubWoofer ha un cono enorme perché ha bisogno di muovere una grossa pressione sonora per riprodurre frequenze basse ad un volume alto.

Di solito si mixa a 85dBSPL. Poi si cambia la pressione sonora per isolare meglio gli estremi di banda oppure le medio frequenze per capire come vanno.

Missing fondamental

Si dice che un suono armonico abbia una fondamentale mancante, una fondamentale soppressa o una fondamentale fantasma quando i suoi armonici suggeriscono una frequenza fondamentale ma il suono manca di una componente alla frequenza fondamentale stessa. Il cervello percepisce l' altezza di un tono non solo dalla sua frequenza fondamentale, ma anche dalla periodicità implicita nel rapporto tra le armoniche superiori; possiamo percepire la stessa altezza (magari con un timbro diverso) anche se in un tono manca la frequenza fondamentale.

Ad esempio, quando una nota (che non è un tono puro) ha un'altezza di 100 Hz , sarà costituita da componenti di frequenza che sono multipli interi di quel valore (ad esempio 100, 200, 300, 400, 500.... Hz). Tuttavia, gli altoparlanti più piccoli potrebbero non produrre basse frequenze, quindi nel nostro esempio potrebbe mancare la componente da 100 Hz. Tuttavia, si può ancora sentire un tono corrispondente alla fondamentale.

Digitalizzazione

L’audio può esistere in tre grandi domini:

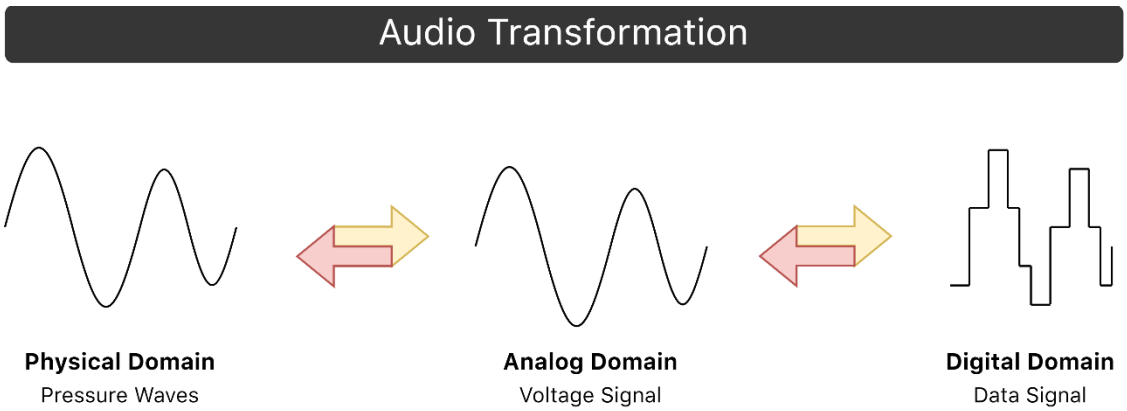
- **Dominio fisico**: l’audio esiste sottoforma di pressione sonora, perturbazione del mezzo.
- **Dominio Analogico**: l’audio esiste sottoforma di informazione, nello specifico, in un contesto analogico si parla di volts
- **Dominio Digitale**: l’audio esiste sottoforma di informazione, nello specifico, in un contesto digitale si parla di bits

Per gli ultimi due, il segnale deve poi essere riconvertito nel dominio fisico, per essere udito. Fino ad allora, è solo una rappresentazione di informazioni (analogiche o digitali) e nient’altro.

Audio Processing & Audio Transformation

Quando si tratta di manipolare e migliorare il suono, due concetti chiave emergono: l'audio transformation e l'audio processing.

Sebbene spesso utilizzati in modo intercambiabile, questi termini hanno sottili ma importanti differenze. **L'audio transformation** è un concetto che coinvolge la trasformazione del segnale da un dominio all'altro, manipolazioni spettrali, sintesi sonora e la creazione di effetti audio.



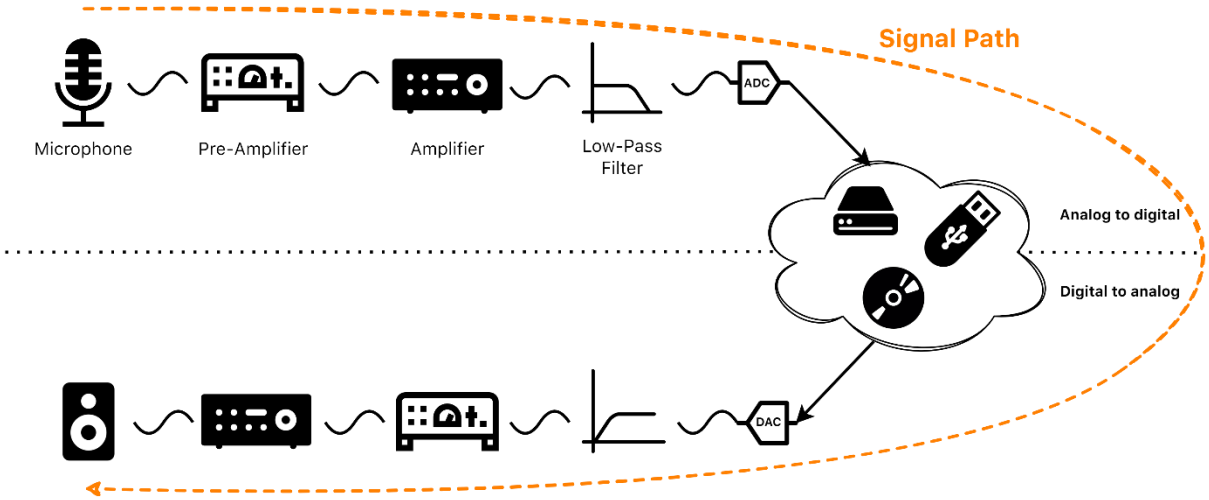
Ogni step di trasformazione ha un costo associato ad esso, e nessuno di questi step è in grado di offrire una trasformazione lossless. Ciò significa che si avrà un degrado della qualità audio e quindi una perdita di risoluzione ad ogni stage. Il “quanto” viene degradato, dipende invece da diversi fattori. Il passaggio da un dominio all’altro è chiamato processo di trasduzione.

D’altra parte, **l'audio processing** riguarda l'elaborazione dei segnali all’interno dello stesso dominio mediante una serie di tecniche mirate a modificare, filtrare o migliorare specifici aspetti del suono, come la riduzione del rumore, l'equalizzazione o la compressione. In molti casi, l’audio processing è lossless e reversibile.

Signal Path

Il signal path è il percorso che effettua il segnale prima di raggiungere il target voluto, che per il suono, sono le nostre orecchie. Il segnale più semplice è quello che non contiene nessuna forma di trasformazione o processing del segnale stesso, ad esempio un suono che viene prodotto in natura da una perturbazione ed arriva direttamente alle nostre orecchie. Il percorso del segnale che viene trattato per essere trasportato nel dominio digitale però, acquisisce un trattamento e segue un percorso ben differente.

Questo è il classico percorso che il segnale audio percorre, per essere archiviato digitalmente e poi riprodotto per arrivare infine alle nostre orecchie:

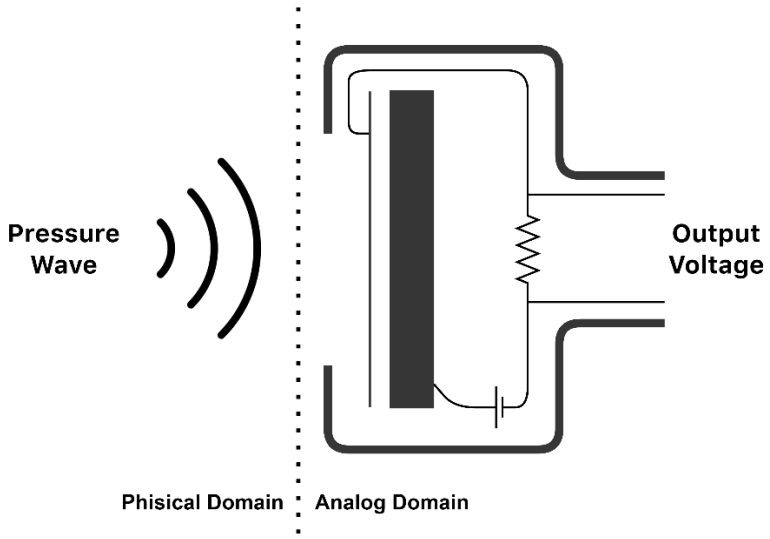


Microfono

Per arrivare a rappresentare il suono sottoforma di segnale digitale, è necessario convertire il segnale provienente dal dominio fisico (pressione sonora) in un segnale audio analogico (differenza di potenziale).

Con segnale audio analogico quindi si intende una rappresentazione di un suono, che normalmente è costituito dalla modulazione di una tensione elettrica. Un segnale audio analogico può essere originato o sintetizzato direttamente da un trasduttore (microfonico o testina).

Il microfono, non fa altro che trasdurre (convertendo il suo dominio) il segnale:



In sostanza le onde di pressione sonora arrivano alla membrana frontale, che è molto sensibile. Vibrando, genera una differenza di potenziale che viene inoltrata in output.

Pre-Amplifier

Lo scopo del preamp è quello di aumentare il livello di tensione del segnale audio proveniente da una sorgente (come un microfono, una chitarra elettrica, una tastiera, ecc.) prima di inviarlo all'amplificatore finale. Questo è necessario perché alcuni sensori producono segnali molto deboli, che potrebbero essere difficili da utilizzare direttamente o potrebbero essere soggetti a una maggiore degradazione durante il trasporto del segnale attraverso lunghe distanze.

L’obbiettivo del preamp è anche quello di aumentare la potenza di un segnale senza degradarne significativamente il rapporto segnale/rumore (SNR). Ciò significa che i preamplificatori rendono il segnale più forte mentre fanno del loro meglio per non rafforzare anche il rumore.

Il rapporto segnale-rumore (SNR o S/N) è una misura che confronta il livello di un segnale desiderato con il livello del rumore di fondo. SNR è definito come il rapporto tra la potenza del segnale e la potenza del rumore, spesso espresso in decibel. Un rapporto superiore a 1:1 (superiore a 0 dB) indica più segnale che rumore.

Un SNR alto significa che il segnale è chiaro e facile da rilevare o interpretare, mentre un SNR basso significa che il segnale è corrotto o oscurato dal rumore e può essere difficile da distinguere o recuperare. L'SNR può essere migliorato con vari metodi, come aumentare la potenza del segnale, ridurre il livello di rumore, filtrare il rumore indesiderato o utilizzare tecniche di correzione degli errori.

L'SNR può essere calcolato utilizzando formule diverse a seconda di come il segnale e il rumore vengono misurati e definiti. Il modo più comune per esprimere SNR è in decibel. Altre definizioni di SNR possono utilizzare diversi fattori o basi per il logaritmo, a seconda del contesto e dell'applicazione.

Il rapporto segnale-rumore è definito come il rapporto tra la potenza di un segnale (input significativo) e la potenza del rumore di fondo (input privo di significato o indesiderato):

$$\mathrm{SNR} = \frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}}$$

P è la potenza media misurata in unità di potenza, come watt (W) o milliwatt (mW), e il rapporto segnale/rumore è un numero puro. Sia la potenza del segnale che quella del rumore devono essere misurate negli stessi punti o in punti equivalenti in un sistema e all'interno della stessa larghezza di banda del sistema.

Se si vuole misurare in decibel invece:

$$\mathrm{SNR_{dB}} = 10 \log_{10} (\mathrm{SNR})$$

$$\mathrm{SNR_{dB}} = 10 \log_{10} \left(\frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}} \right)$$

Usando le proprietà dei logaritmi:

$$10 \log_{10} \left(\frac{P_{\mathrm{signal}}}{P_{\mathrm{noise}}} \right) = 10 \log_{10} (P_{\mathrm{signal}}) - 10 \log_{10} (P_{\mathrm{noise}})$$

Infine:

$$\mathrm{SNR_{dB}} = P_{\mathrm{signal,dB}} - P_{\mathrm{noise,dB}}$$

Quando il segnale e il rumore sono misurati in volt (V) o ampere (A), che sono misure di ampiezza, devono prima essere elevati al quadrato per ottenere una quantità proporzionale alla potenza, come mostrato di seguito:

$$\mathrm{SNR_{dB}} = 10 \log_{10} \left[\left(\frac{A_{\mathrm{signal}}}{A_{\mathrm{noise}}} \right)^2 \right] = 20 \log_{10} \left(\frac{A_{\mathrm{signal}}}{A_{\mathrm{noise}}} \right)$$

Poiché molti segnali hanno una gamma dinamica molto ampia, i segnali sono spesso espressi utilizzando la scala logaritmica dei decibel. Sulla base della definizione di decibel, segnale e rumore possono essere espressi in decibel (dB)

Un segnale che passa attraverso un preamplificatore subisce quindi un guadagno di tensione. Ad esempio, un segnale da 10 mV che entra in un preamplificatore può uscire come segnale da 1 V.

Gamma dinamica

Amplifier

Seguendo il path, dopo che il segnale viene pre-amplficato (in tensione) è necessario amplificato anche in potenza (corrente elettrica), questo è l'obbiettivo dell'amplificatore.

Un amplificatore, per mezzo di componenti attivi, amplifica il segnale in entrata di un fattore moltiplicativo comunemente indicato come guadagno (A), rendendolo disponibile in uscita con una tensione e/o una corrente disponibile maggiori che all'ingresso. Nel dettaglio, i componenti attivi sono ad es. le valvole, i transistor, etc. Ci sono amplificatori a basso guadagno, medio guadagno, alto guadagno: ciò che li differenzia è il numero di componenti attivi presenti nell'amplificatore stesso ovvero degli Stadi di amplificazione e come siano configurati.

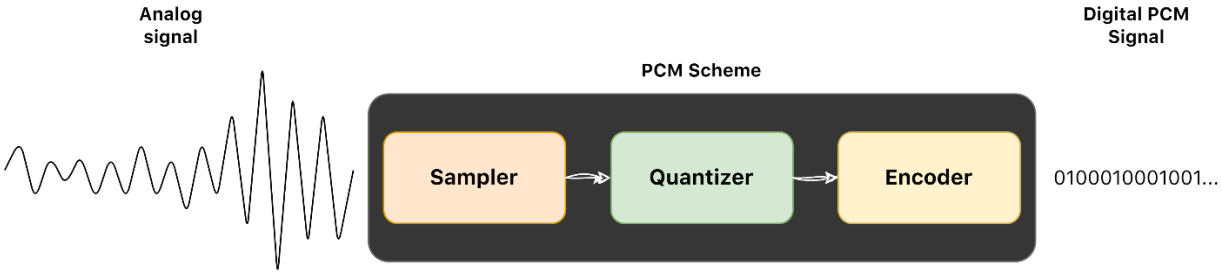
Il guadagno di un amplificatore audio si esprime in dB (decibel) ed è pari a 20 volte il logaritmo in base 10 dell'amplificazione in tensione. In altri amplificatori si considera, invece, l'amplificazione in potenza e, in tal caso, il guadagno è 10 volte logaritmo in base 10 del rapporto fra la potenza in uscita e quella in ingresso. È interessante notare che a un rapporto unitario (A=1) corrispondono 0dB, ovvero un guadagno nullo: un dispositivo simile non guadagna nulla rispetto all'ingresso.

Più specificamente, un generico amplificatore (lineare) ha un ingresso a cui è applicato il segnale da amplificare ed una uscita da cui viene prelevato il segnale amplificato, di ampiezza pari ad A volte il segnale originale. Se il guadagno dell'amplificatore non è costante, ma varia in base a determinate caratteristiche del segnale, si parla di amplificatore non lineare.

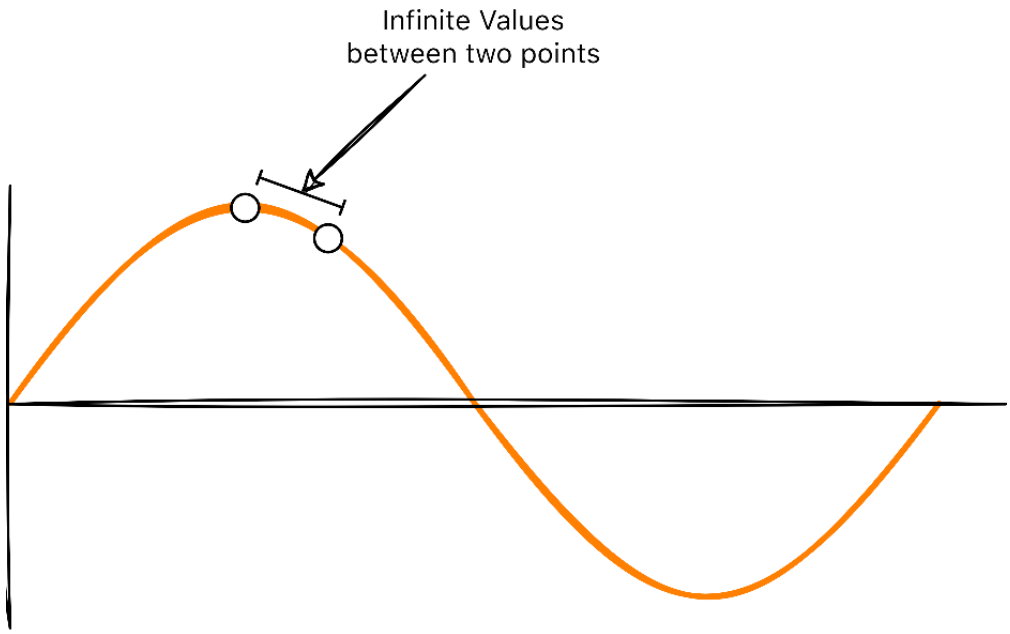
Pulse Code Modulation

A questo punto, è necessario capire come ottenere una rappresentazione digitale di un segnale audio analogico. La pulse code modulation (PCM) è un metodo di rappresentazione digitale di un segnale analogico.

I passaggi principali coinvolti nel PCM sono il **campionamento**, la **quantizzazione** e la **codifica:**



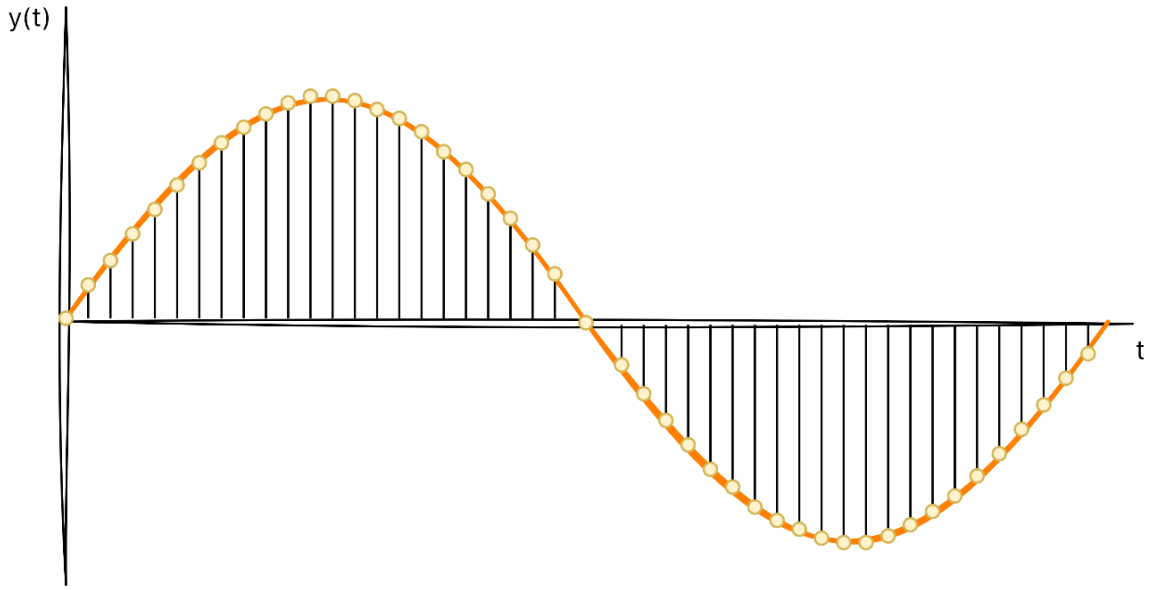
Campionamento (Sampling)



La prima cosa da notare è che un qualsiasi segnale nel dominio analogico è una funziona continua nel tempo, qualsiasi intervallo si possa considerare, non ci saranno punti in cui la funzione non è definita. Questo significa anche che tra due punti distinti esistono infiniti valori.

Ma come si immagazina un segnale continuo, su supporti digitali che non hanno capacità infinita? Nessun dispositivo di storage al mondo è capace di immagazinare anche solo un singolo periodo di una sinusoide, per ogni suo punto, nel tempo.

Quindi, per ottenere una rappresentazione digitale, è necessario immagazinare una quantità di valori della funzione che sia sufficiente per ricostruire la funzione che stiamo convertendo. È necessario "rompere" la funzione continua rendendola discontinua, prelevando solo alcuni valori nel tempo ottenendo così un segnale discreto (per segnale discreto o segnale tempo-discreto si intende una successione di valori di una certa grandezza in corrispondenza di una serie di valori discreti nel tempo):



Quando un segnale discreto è composto da una serie di valori ottenuti in corrispondenza di istanti spaziati uniformemente nel tempo, si dice che è associato ad una particolare **frequenza di campionamento.** Il processo di trasformazione da analogico a digitale, inizia proprio qui.

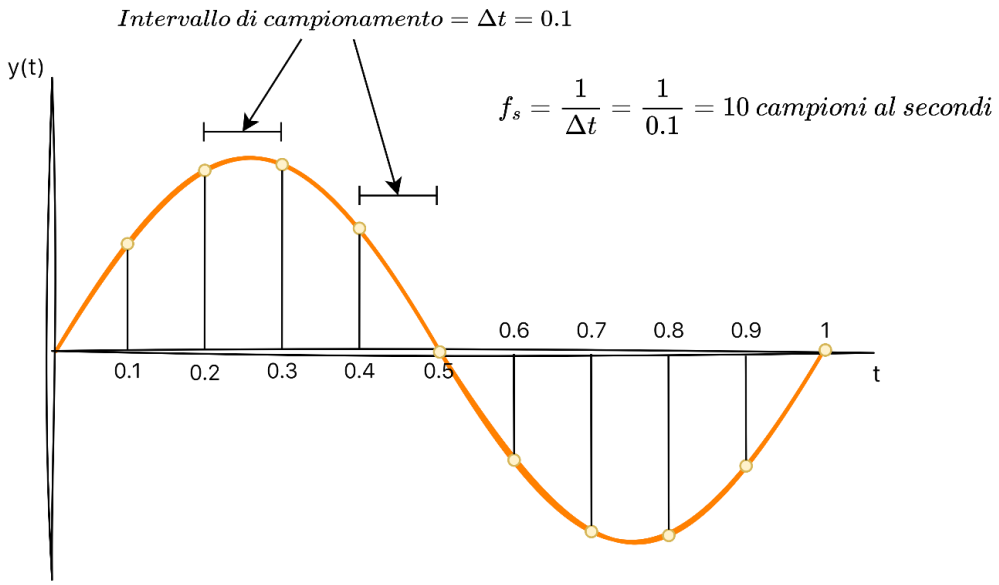
La prima domanda da porsi è: "quanti campioni prendiamo in un intervallo di tempo?". Oppure "quanti campioni sono sufficienti per rappresentare correttamente un segnale?". Fortunatamente la risposta non è così arbitraria come si crede, infatti, la conversione da analogico a segnale discreto è governata dal teorema di Nyquist Shannon (Nyquist Shennon sampling theorem).

Nyquist-Shennon Theorem

Il teorema del campionamento di Nyquist-Shannon è un teorema nel campo dell'elaborazione del segnale che funge da ponte fondamentale tra segnali a tempo continuo e segnali a tempo discreto.

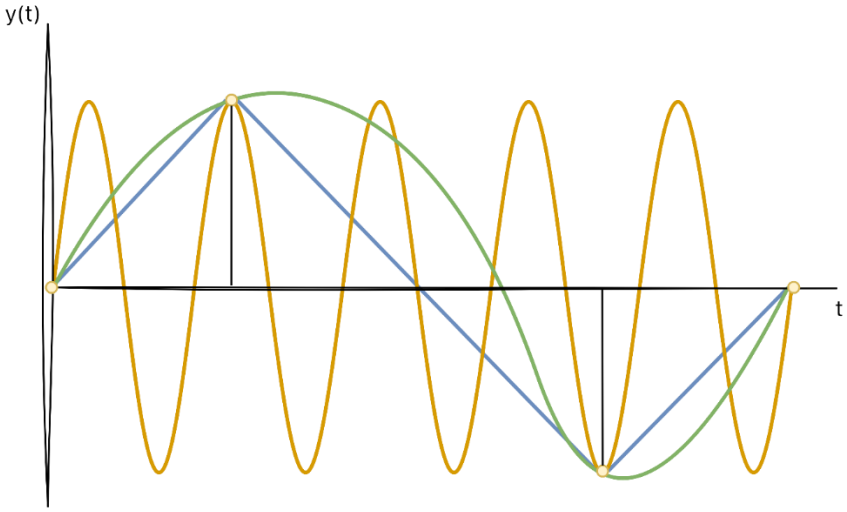
Il teorema di Nyquist-Shannon, noto anche come teorema di campionamento o teorema di campionamento di Nyquist, è uno dei principi fondamentali nella teoria del campionamento dei segnali e nella teoria dell'informazione. Esso stabilisce la minima frequenza, detta frequenza di Nyquist (o anche cadenza di Nyquist), necessaria per campionare un segnale analogico senza perdere informazioni, e per poter quindi ricostruire il segnale analogico a tempo continuo originario.

Il campionamento è il primo passo del processo di conversione analogico-digitale di un segnale. Consiste nel prelievo di campioni (samples) da un segnale analogico e continuo nel tempo ogni $\boldsymbol{\Delta t}$ secondi. Il valore $\boldsymbol{\Delta t}$ è detto intervallo di campionamento (o periodo di campionamento), mentre $\boldsymbol{f_s = \frac{1}{\Delta t}}$ è la frequenza di campionamento.



Il teorema di Nyquist-Shannon stabilisce che, dato un segnale analogico la cui banda di frequenze sia limitata dalla frequenza $\boldsymbol{FM}$, il segnale può essere univocamente ricostruito a partire dai suoi campioni presi a frequenza di campionamento $\boldsymbol{f_s = \frac{1}{\Delta t}}$ se $\boldsymbol{f_s > 2f_M}$. In altre parole, la frequenza di campionamento deve essere almeno il doppio della frequenza più alta presente nel segnale analogico. (2FM è anche chiamato Tasso di Nyquist).

Ora osserviamo questo scenario:



Qui abbiamo un suono da 1HZ (come quello di prima) ma campionato a 4HZ (4 campioni al secondo). Rispetto ai 10 campioni di prima la forma d'onda potrebbe risultare ambigua.

Il fatto è che non è così. È possibile dimostrare che, per un suono da 1HZ, sia che si prendano 10 campioni, sia che se ne prendano 4, quando si riconvertirà il segnale in analogico, il risultato sarà sempre la stessa forma d'onda. Da questo ne consegue che all'aumento del numero dei campioni non si ha un aumento della qualità. Così come una traccia musicale standard (campionata a 44.100Hz) non avrà un incremento di qualità dell'audio nel caso in cui si aumenti da 44.100 a 96KHz o addirittura 192KHz.

Cosa succede però se il criterio di Nyquist non viene rispettato è non si campiona ad almeno il doppio della frequenza di Nyquist? Supponiamo di avere un segnale dove la frequenza di Nyquist è di 4000Hz, secondo il teorema del campionamento di Nyquist si dovrebbe campionare ad almeno il doppio della frequenza di Nyquist, quindi 8000Hz. Cosa succede se si campiona a 7000Hz? Succede che si andrà in contro ad un fenomeno chiamato Aliasing.

Aliasing

Nell'elaborazione del segnale, l'aliasing si verifica quando la frequenza di campionamento di un segnale è inferiore alla frequenza di Nyquist, causando sovrapposizione di componenti di frequenza. Questo fenomeno provoca distorsioni o artefatti nel segnale ricostruito dai campioni, portando a differenze rispetto al segnale continuo originale.

L'aliasing temporale si verifica nei segnali campionati nel tempo, come nell'audio digitale. L'aliasing spaziale, invece, si manifesta nei segnali campionati nello spazio, ad esempio, nelle immagini digitali. In pratica, nel fenomeno dell'aliasing temporale, componenti ad alta frequenza del segnale saranno erroneamente rappresentati come componenti a frequenze più basse durante la ricostruzione del segnale analogico, portando a sovrapposizioni e distorsioni indesiderate nel segnale ricostruito.

Se il segnale analogico contiene frequenze superiori alla frequenza di Nyquist e queste frequenze non vengono adeguatamente filtrate prima del campionamento, si verificherà l'aliasing, e queste frequenze verranno ricostruite in modo errato nel segnale digitale risultante.

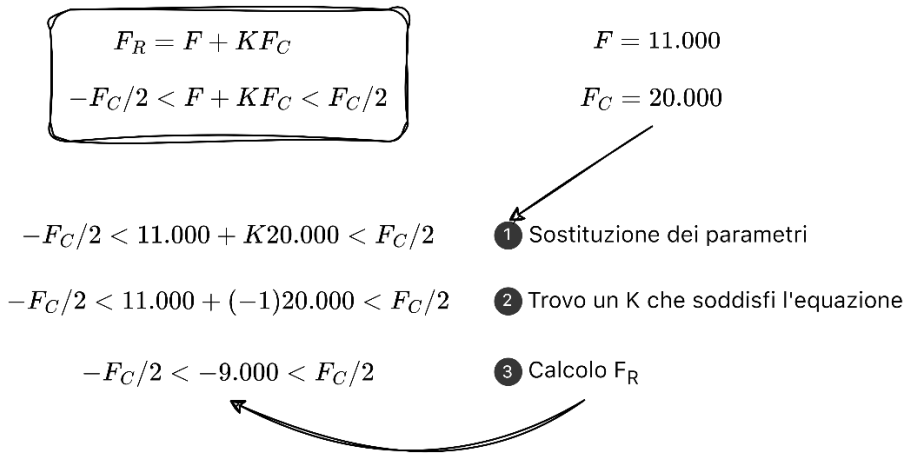
Quando un segnale analogico viene campionato a una frequenza inferiore alla frequenza di Nyquist, le frequenze al di sopra della frequenza di Nyquist (metà della frequenza di campionamento) vengono riflettute all'indietro, cioè ripiegate o piegate, nella banda di frequenza al di sotto della frequenza di Nyquist. In dettaglio, data una frequenza in ingresso '**F**', la frequenza ricostruita '$\mathbf{F_R}$' sarà:

$$F_R = F + KF_C$$

Dove '**K**' è un intero tale che F_R si trova all'interno dell'intervallo:

$$-F_C/2 < F + KF_C < FC/2$$

Se ad esempio si ha una frequenza di campionamento $F_C = 20.000$ e come segnale analogico in ingresso si riceve $F = 11.000$, allora la frequenza ricostruita sarà:



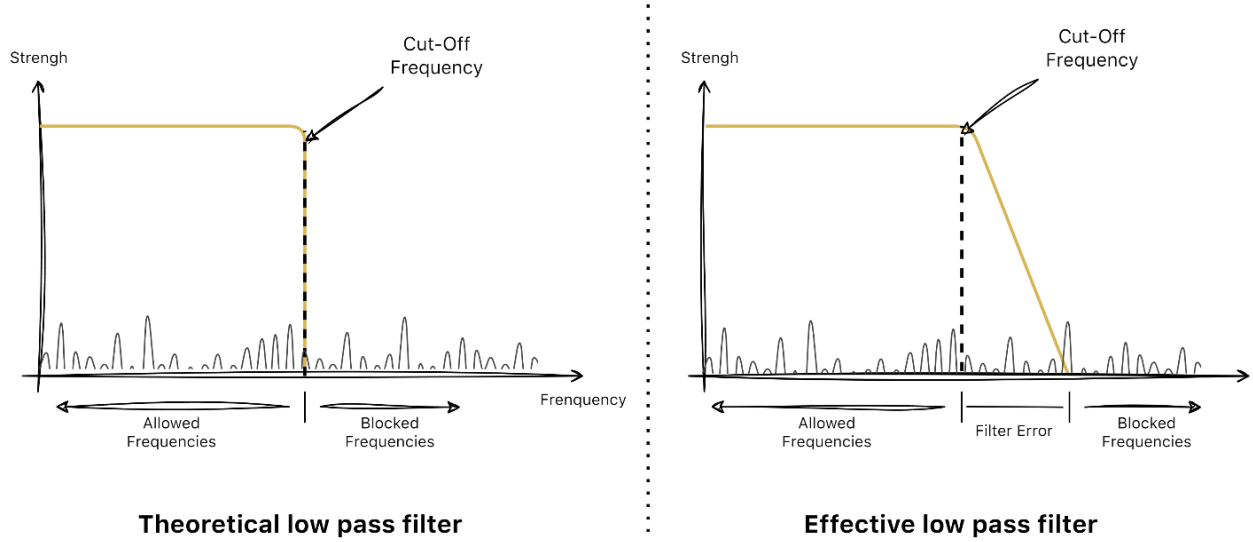
Di conseguenza, queste frequenze "alias" vengono interpretate come se fossero parte del segnale originale, anche se in realtà non lo sono.

L’aliasing comporta due principali problemi:

- **Perdita di informazioni**: Le frequenze che vengono ripiegate nell'intervallo di frequenza corretto vengono sostituite con le frequenze aliased. Questa sostituzione causerà una perdita di informazioni e può alterare significativamente il suono originale.
- **Distorsione**: Le frequenze aliased che vengono ricostruite possono causare distorsioni indesiderate nel segnale audio, creando suoni che non erano presenti nel segnale originale.

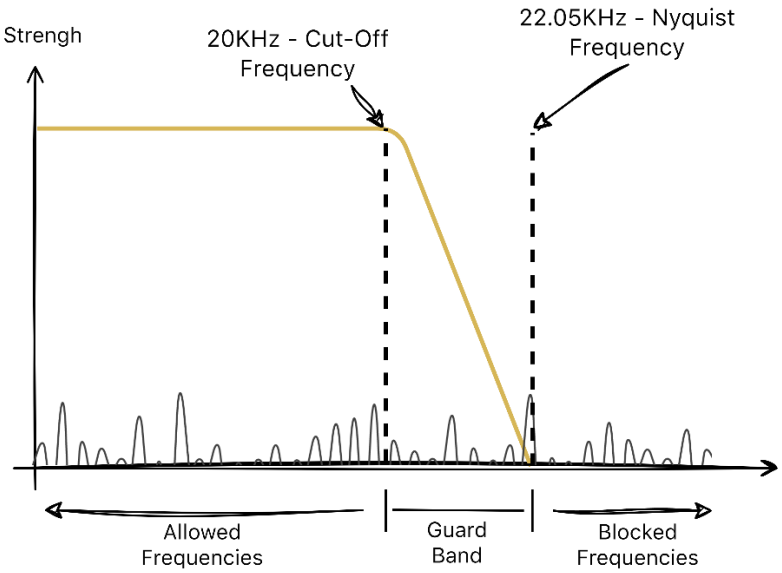
Per evitare questi problemi, è essenziale utilizzare un **filtro anti-aliasing** prima del processo di campionamento. Il filtro anti-aliasing attenua le frequenze al di sopra della frequenza di Nyquist, assicurandosi che solo le frequenze corrette vengano campionate. In questo modo, si può ottenere una rappresentazione digitale accurata del segnale originale senza aliasing o foldover.

Conviene inoltre non campionare esattamente alla frequenza di nyquist, ma avere un buffer/margine di frequenze. Questo perché i bandlimiter (filtri) non sono perfetti:

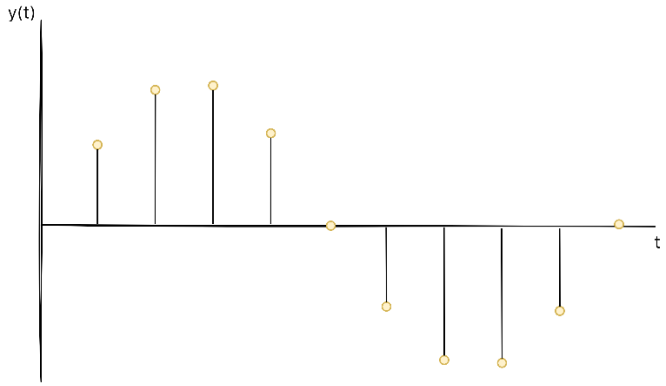


Theoretical low pass filter **Effective low pass filter**

Il grafico di sinistra mostra come sarebbe un filtro teorico perfetto. Questo però è fisicamente impossibile, l’immagine di destra mostra come un filtro reale si comporta. Si nota come permette ad alcune frequenze oltre la frequenza di taglio (supponiamo sia quella di Nyquist). Infatti per l’audio non si tende a campionare a 40KHz ma a 44.1KHz oppure 48KHz.



Una volta campionato il segnale analogico, si avrà quindi un segnale discreto nel tempo in cui ogni campione sarà preso ogni Δt:

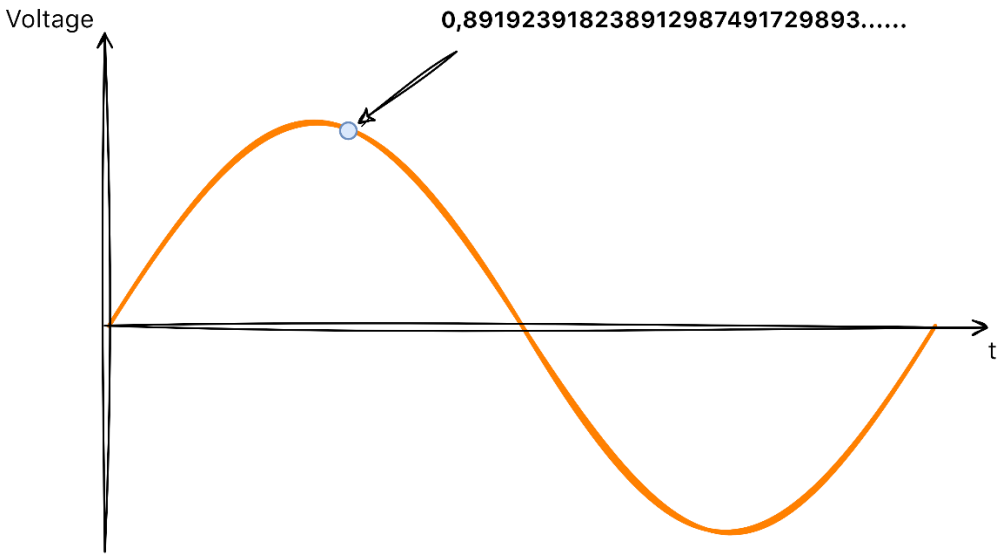


Insieme al campionamento però avviene anche un altro processo, chiamato **quantizzazione**.

Quantizzazione

Ciò che abbiamo rappresentato fino ad ora è soltato un numero di campioni per secondo del segnale analogico in ingresso. Quello che non abbiamo ancora rappresentato è l'ampiezza per ogni campione acquisito.

La quantizzazione ha il compito di assegnare ad ogni campione un suo livello di ampiezza ma nel mondo digitale: nel dominio fisico, l'ampiezza è rappresentata dal numero di particelle addensate, nel mondo analogico è rappresentata dal valore di tensione, nel mondo digitale è necessario rappresentarlo sottoforma di bit.



Nel dominio analogico, anche il valore di tensione è continuo. Ciò significa che ogni livello di tensione è rappresentato da infiniti numeri decimali. Ovviamente nella conversione da analogico a digitale, come non è possibile rappresentare ogni possibile campione nel tempo, non è altrettanto possibile rappresentare tutte le infinite cifre del valore di tensione per campione (essendo un valore anch'esso continuo). C'è una differenza però tra campionamento e quantizzazione sotto questo punti di vista: se il campionamento è lossless (è possibile ricostruire esattamente la forma d'onda originale) la quantizzazione non lo è (si dice che è lossy).

Ad ogni modo, è necessario interpretare i valori di tensione dei campioni, in sequenze di bit. Ogni campione avrà la sua sequenza di bit associata. Per fare questo, si discretizza l'ampiezza (asse y) in livelli (chiamati livelli di quantizzazione), questo processo prende appunto il nome di **quantizzazione**.

Prima di tutto è necessario individuare l'accuracy con la quale si vuole quantizzare il segnale, questo parametro è chiamato **bit depth** ed indica il numero di bit con il quale si rappresenta un campione di ampiezza.

In secondo luogo è necessario individuare il range di ampiezze del segnale analogico che si sta quantizzando, in particolare serve recuperare $\boldsymbol{V_{Max}}$ e $\boldsymbol{V_{Min}}$ che rappresentano rispettivamente il valore più alto e più basso di tensione presente nel segnale analogico.

Una volta ottenuti, il resto è calcolabile utilizzando questi due parametri appena ottenuti, in particolare troveremo:

- Il numero di livelli con il quale il segnale sarà suddiviso, chiamate anche Regioni di quantizzazione
- La dimensione di una regione

Il numero di regioni di quantizzazione è dato da:

$$L = 2^{bit\_depth}$$

Se ad esempio avremo 4 bit, il numero di regioni sarà $2^4 = 8$ regioni.

La dimensione di una regione Δ è dato da:

$$\Delta = \frac{V_{Max} - V_{Min}}{L}$$

Il rapporto tra la differenza di tensione massima e minima con il numero di regioni di quantizzazione, indica sostanzialmente quanto è grande una regione.

Un esempio:

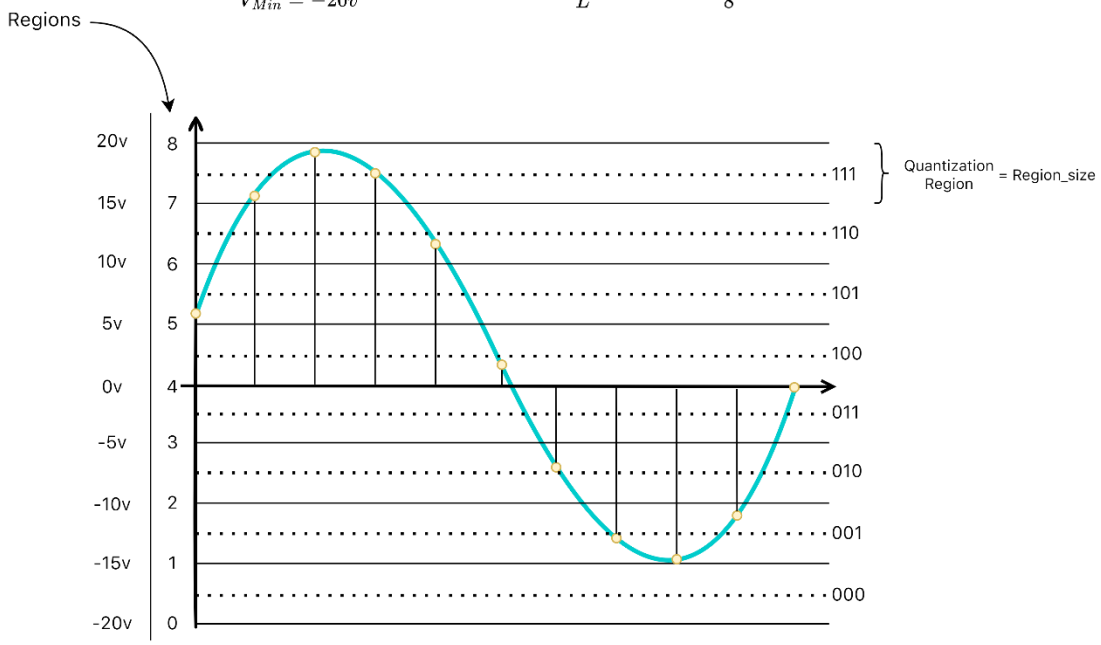
$bit\_depth = 3$

$V_{Max} = 20v$

$V_{Min} = -20v$

$L = 2^{bit\_depth} = 8\ Quantization\ Region$

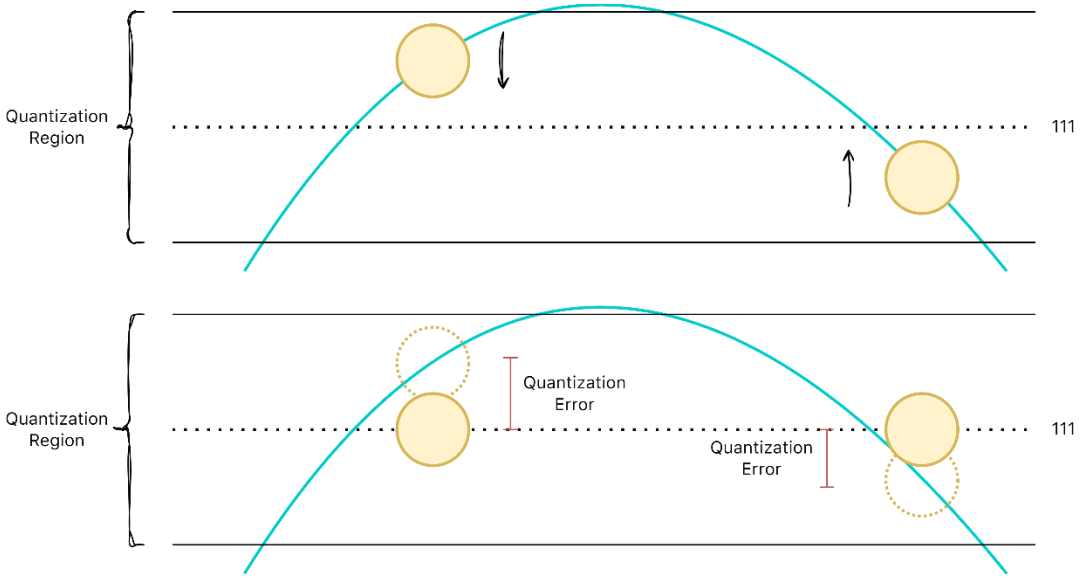
$\Delta = \frac{V_{Max} - V_{Min}}{L} = \frac{20 - (-20)}{8} = 5\ Region\_size$



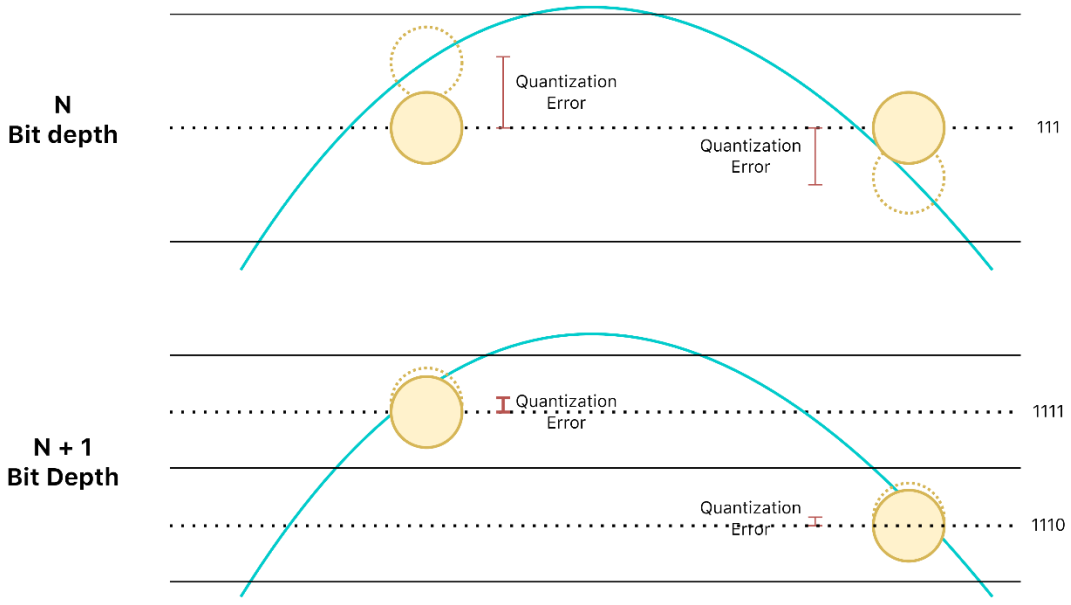
Nell'esempio il segnale ha un valore di tensione massimo V_{Max} che è +20v ed un valore di tensione minima che è V_{Min} che è -20v. il segnale verrà quantizzato con una bit depth di 3, ciò significa che il numero di regioni di quantizzazione saranno $2^3 = 8$. La dimensione di una regione di conseguenza è 5.

Si parla di quantizzaione uniforme quando le regioni sono di dimensioni fissa, altrimenti si parla di quantizzazione non uniforme. La LPCM (Linear-Pulse code modulation) è un tipo specifico di PCM in cui i livelli di quantizzazione sono linearmente uniformi.

Nell'esempio ci sono anche linee tratteggiate. Quelle linee rappresentano il valore binario con il quale i campioni saranno rappresentati e sono in corrispondenza della metà di ogni regione di quantizzazione. Ogni campione sarà "attratto" dal centro della regione.

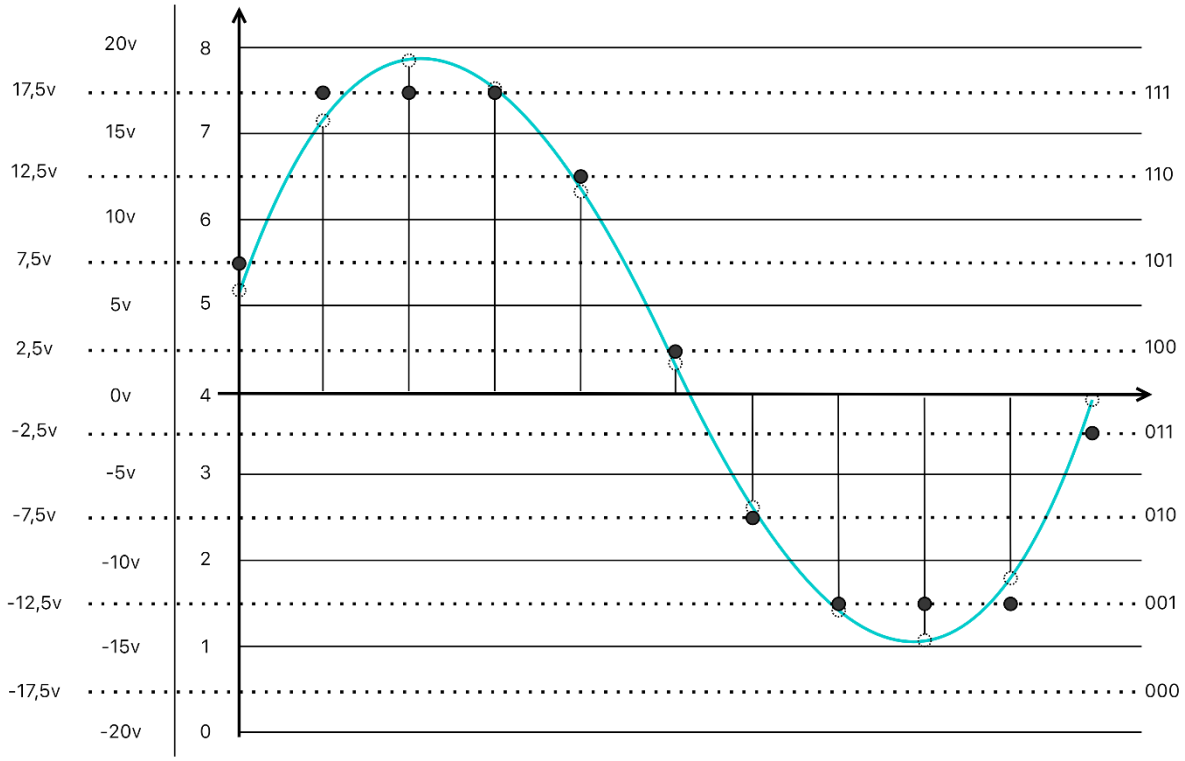


Qui una versione zoommata. La differenza tra l'origine vera del punto e la metà della regione è chiamato errore di quantizzazione. Si può intuire che il valore massimo di errore di quantizzazione è dato dalla lunghezza della metà di una regione. L'errore di quantizzazione si attenua all'aumentare del bit depth con cui si quantizza e di conseguenza del numero di regioni di quantizzazione:



Questo perche all’aumento del numero di bit di quantizzazione, aumenta il numero di regioni e di conseguenza la dimensione di una regione, riducendo le approssimazioni.

Per l’esempio di prima avremo quindi:



Original Voltage	5,5	15,5	19,5	18	11,5	1,5	-6,5	-13	-14,5	-11	-0,5
Quantized Voltage	7.5	17,5	17,5	17,5	12,5	2,5	-7,5	-12,5	-12,5	-12,5	-2,5
Quantization Error	2	2	2	0,5	1	1	-1	-0,5	-2	-1,5	-2
Binary Encoding	101	111	111	111	110	100	010	001	001	001	011

Errore di quantizzazione e Rumore di quantizzazione

Quindi l'errore di quantizzazione è l'errore introdotto quando si approssima un segnale analogico con un valore quantizzato, cioè un valore discreto. Questo errore deriva dalla natura discreta dei valori rappresentabili nei sistemi digitali, che non possono rappresentare infiniti valori possibili presenti in un segnale analogico. Quando si arrotonda o si tronca un valore analogico per rappresentarlo digitalmente, si introduce un errore tra il valore reale e il valore rappresentato.

Il rumore di quantizzazione è la componente di errore casuale associata all'errore di quantizzazione. Poiché il processo di approssimazione da segnale analogico a segnale digitale non è perfetto, c'è un livello di incertezza introdotto nel valore quantizzato. Questa incertezza si traduce in una componente di errore casuale che è presente nel segnale digitale rappresentante l'originale segnale analogico. Questo rumore di quantizzazione è fondamentalmente una forma di rumore aggiunto al segnale a causa del processo di conversione.

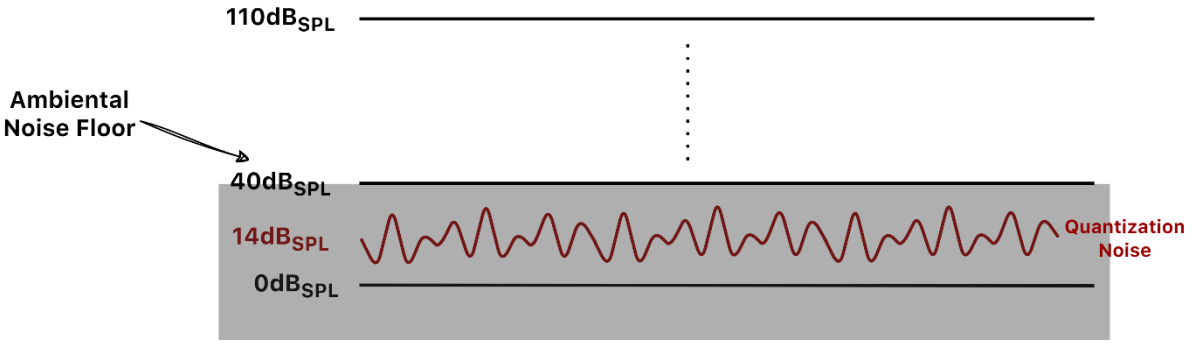
L'SQNR è una misura che quantifica quanto il segnale desiderato (l'originale segnale analogico) sia predominante rispetto al rumore di quantizzazione. In altre parole, l'SQNR rappresenta quanto il segnale è forte rispetto al rumore introdotto dal processo di quantizzazione. Un alto valore di SQNR indica che il segnale è ben rappresentato e che l'effetto del rumore di quantizzazione è ridotto.

L'SQNR è solitamente espressa in decibel (dB) ed è calcolata con la seguente formula:

$$\mathrm{SQNR} = 20\log_{10}(2^n) \approx 6.02 \cdot n \ \mathrm{dB}$$

dove 'n' è il numero di bit di quantizzazione (la bit depth). Questo ci suggerisce che ogni bit nel processo di quantizzazione ci garantirà 6dB di SQNR.

Ad esempio, un ADC a 16 bit ha un SQNR massimo di 6,02 × 16 = 96 dB. Cosa significa? Significa che se entrassimo in una camera anecoica e riproducessimo un brano a 96dB quantizzato utilizzando questo ADC, essendo che il nostro orecchio ha una gamma dinamica di 130dB, nel punto in cui il brano è in silenzio, inizieremmo a sentire il rumore di quantizzazione. Se alzassimo il volume a 110db ad esempio lo sentiremmo ancora di più. Nei contesti reali è possibile avere più gamma dinamica per il brano, in quanto il rumore ambientale maschera il rumore di quantizzazione. Qui viene spiegato meglio il tutto.



Se utilizzassimo invece un ADC a 24bit, avremmo un SQNR massimo di 6,02 × 24 = 144dB. In questo caso, dato che il nostro orecchio ha una soglia del dolore di 130db, anche in una camera anecoica alzando il volume ad un massimo sopportabile, non sentiremmo il rumore di quantizzazione.

Quantizzazione non uniforme

Nella quantizzazione uniforme, oltre ad avere regioni di dimensione fissa, anche l'ampiezza media del rumore di quantizzazione è uniforme. Questo perché le regioni essendo della stessa dimensione il rumore medio sarà anch'esso uniforme. La quantizzazione uniforme quindi non rispetta l'andamento della nostra percezione.

Le codifiche PCM in cui i livelli di quantizzazione variano in funzione dell'ampiezza sono l'algoritmo A-law e l'algoritmo μ-law. Ad ogni modo, sebbene PCM sia un termine più generale, viene spesso utilizzato per descrivere i dati codificati come LPCM.

È possibile invece, utilizzando la quantizzazione non uniforme, distribuire l'errore sulle zone del segnale in cui la nostra percezione fa più fatica a riconoscere il rumore. Ad esempio, quando c'è silenzio o comunque il brano ha ampiezze deboli, sentiamo molto di più il rumore di quantizzazione rispetto a quando ci sono parti in cui si hanno ampiezze alte e c'è tanta musica.

Con la quantizzaione non uniforme, possiamo progettare le regioni di quantizzazione per corrispondere alle caratteristiche del segnale, come la sua distribuzione di probabilità oppure, come ho detto, alla sua importanza percettiva.

Il vantaggio della quantizzazione non uniforme è che può ottenere un rumore di quantizzazione inferiore e un SQNR più elevato rispetto alla quantizzazione uniforme, per lo stesso numero di livelli di quantizzazione e intervallo di segnale.

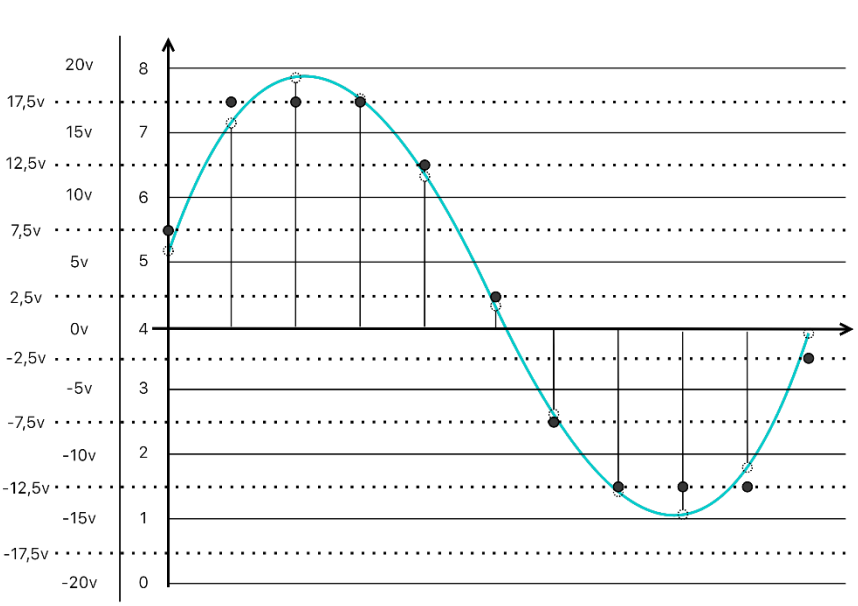
Encoding

L'ultima fase dello schema PCM è l'encoding. Di solito rappresentata insieme alla quantizzazione, ma logicamente può anche essere separata.

In sostanza, una volta discretizzato anche l'ampiezza dei campioni, è necessario identificare dei metodi di rappresentazione binari per quest'ultimi.

I metodi di codifica più diffusi sono:

- **Unipolare**: in cui il valore di ampiezza cresce da 0 a 2^n-1 ed incui lo zero rappresenta il valore di tensione più basso.
- **Bipolare**:
    - **Complemento a due**: qui il valore di ampiezza è relativo allo zero, quindi per tensioni negative avremo valori negativi che crescono verso il basso e tensioni positive, valori che crescono verso l'alto
    - **Segno e magnitudo**: Qui i valori di ampiezza sono come in complemento a due, quindi gli estremi sono massimi negativi e positivi, solo che ci sono due zeri (000, 100 nell'esempio).



Unipolare	Complemento a Due	Segno e Magnitudo
111	011	011
110	010	010
101	001	001
100	000	000
011	111	100
010	110	101
001	101	110
000	100	111

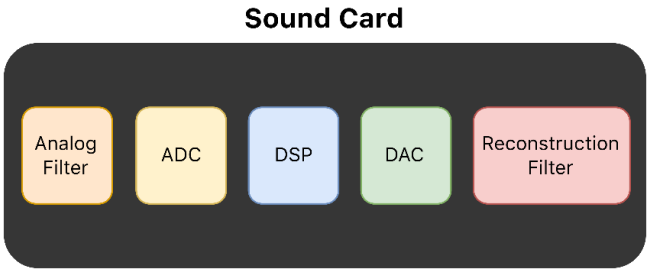
È utile rappresentare i campioni in codifiche distinte in base a ciò che si sta facendo. Se ad esempio si ha un suono e si vuole fare mixing, è più naturale rappresentare i dati in complemento a due: basta osservare ad esempio che il silenzio è rappresentato dallo zero, mentre non è così nell'Unipolare.

La codifica unipolare è usata in uscita dalle schede audio, ad esempio, quando si digitalizza un segnale (quello che abbiamo fatto fino ad ora con lo schema PCM), in cui i valori binari vengono rappresentati dal basso verso l'alto a crescere.

Scheda Audio

Tutti i concetti di filtri, sampling, quantizzazione ed elaborazione generale dell'audio, è effettuata dalla scheda audio. Una scheda audio è una scheda di espansione interna che fornisce input e output di segnali audio da e verso un computer. La scheda audio può essere integrata sulla scheda madre, oppure può anche essere un dispositivo esterno che si collega via USB o altre interfacce e che effettua le stesse operazioni della scheda audio interna.

Lo schema di una scheda audio può essere rappresentato così:



La base di una scheda audio consiste in un filtro analogico, un'unità di conversione analogico-digitale (ADC), un digital signal processor (DSP), un convertitore digitale-analogico (DAC) e un filtro di ricostruzione (anti-imaging).

Il segnale analogico in ingresso viene inviato ad un filtro analogico (quello che abbiamo visto essere un filtro anti-aliasing), che viene applicato per limitare la gamma di frequenza dei segnali analogici prima del processo di campionamento. Il segnale band-limited, all'uscita del filtro anti-alasing viene quindi campionato e convertito tramite l'unità ADC nel segnale digitale, che è discreto sia nel tempo che nell'ampiezza. Il DSP accetta quindi il segnale digitale ed elabora i dati digitali in base al contesto (magari bisogna applicare una compressione, algoritmi MPEG o altro). Tutto questo il DSP lo fa insieme a degli algoritmi, quindi c'è anche del software dietro.

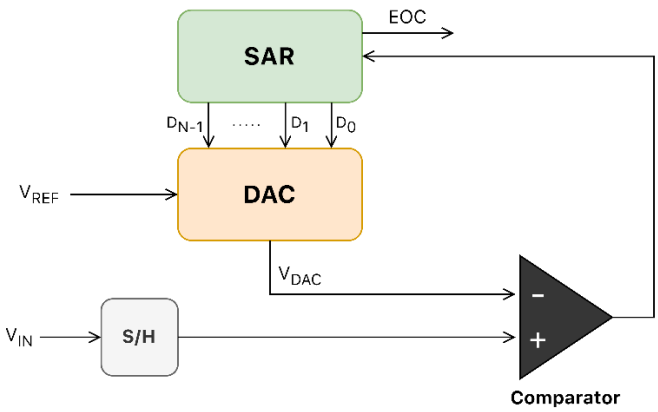
L'unità DSP è un tipo speciale di microprocessore digitale e può essere anche un dispositivo esterno. Di solito nelle schede audio di bassa qualità il DSP non c'è e le elaborazioni digitali vengono fatte dalla CPU.

Il blocco successivo è l'unità DAC, che converte un segnale digitale elaborato in un segnale di uscita analogico. Il blocco finale è un filtro anti-imaging (o smoothing) che serve a rimuovere le alte alte frequenze che si sono create durante la digitalizzazione.

ADC

L'analog to digital converter è il componente che si occupa di campionare e quantizzare il segnale analogico in ingresso (già filtrato per impedire aliasing). Abbiamo visto come viene effettuato tutto ciò a livello concettuale, ma a livello pratico? come avviene la conversione? In sostanza esistono più tipologie di ADC come il Flash ADC, l'integrating ADC, Successive-approximation ADC. Noi guarderemo solo l'ultimo: **Successive-approximation ADC.**

Il Successive-approximation ADC è un tipo di convertitore analogico-digitale che converte una forma d'onda analogica continua in una rappresentazione digitale discreta utilizzando una ricerca binaria attraverso tutti i possibili livelli di quantizzazione prima di convergere su un'uscita digitale per ciascuna conversione. Questo è lo schema di un SA-ADC:



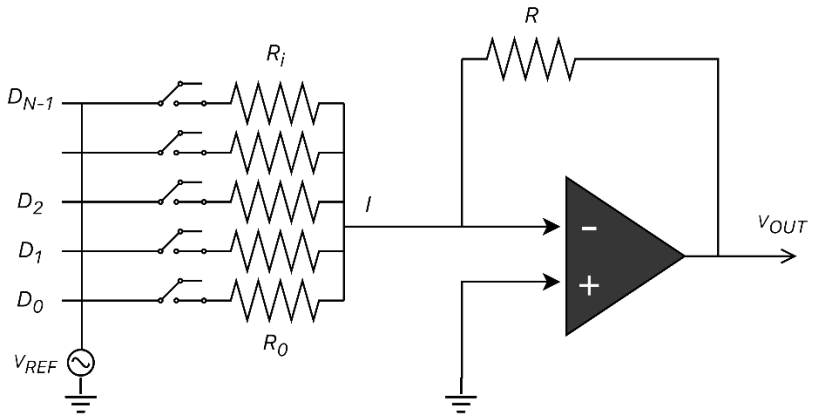
Quello che succede in un SA-ADC, è che il **S/H** (sample and hold) estrae un campione del segnale di ingresso, in particolare leggendo il valore di tensione. Il sample and hold mantiene il valore di tensione per il tempo necessario a permettere all'ADC di approssimare al meglio il valore di V_{DAC} a quello di V_{IN}. Il tempo che impiega l'ADC per produrre la sequenza è detto tempo di assestamento (settling time)

Il valore letto sarà il primo input del comparator. Il secondo input del comparator è il valore di tensione approssimato generato dal DAC (V_{DAC}).

Il successive approximation register (SAR) genera una sequenza di bit (all'inizio è il valore centrale 1000...) che ad ogni iterazione si approssima alla sequenza di bit finale. Ad ogni iterazione, la sequenza estratta dal SAR viene data in pasto al DAC. Quindi il DAC genera un valore di tensione che ogni volta si approssima sempre di più al valore V_{IN}. Il modo in cui funzionano le approssimazioni successive è semplice, ed è spiegato qui. Oppure qui. Non lo scrivo perché mi servirebbero 30 immagini.

DAC

L'altro componente fondamentale è il DAC. che come abbiamo visto opera anche insieme all'ADC per la conversione Analog-to-digital. Il DAC è molto più semplice:

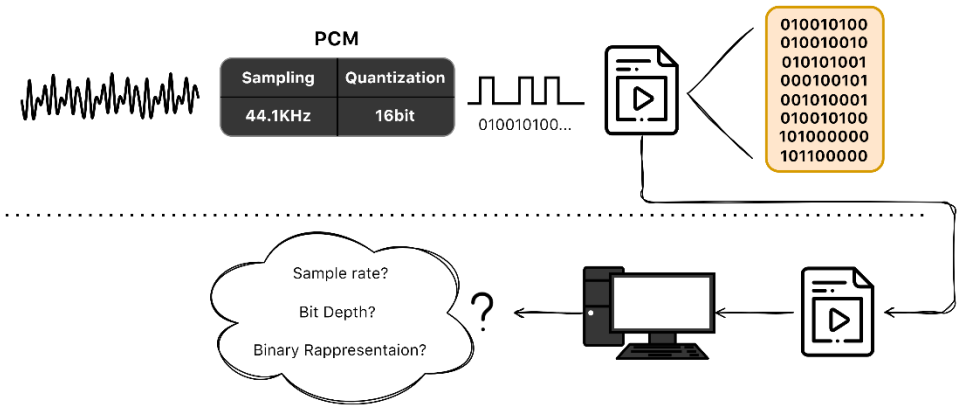


In sostanza ogni bit è rappresentato da una propria resistenza. Più il bit pesa in termini di potenze di due, più sarà bassa la resistenza. La somma delle correnti produrrà una I totale.

Containers & File Formats

Abbiamo visto come trasformare un segnale analogico in un segnale digitale, attraverso la codifica PCM. Nel mondo reale però la rappresentazione digitale del segnale è conservata in un supporto digitale permanente che è il file.

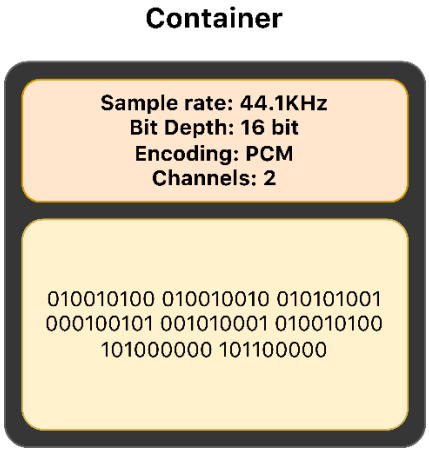
Ora che sappiamo che il file è il mezzo di scambio per i contenuti audio nel mondo digitale, come definiamo il file?



Se produciamo un file con i dati grezzi (raw) senza indicare ad esempio che sample rate abbiamo usato e che bit depth è stata utilizzata, chi riceverà quel file non saprà interpretarlo in modo corretto.

Container Format

Per questo, insieme ai dati grezzi si affiancano delle informazioni che descrivono il file (file autodescriventi).

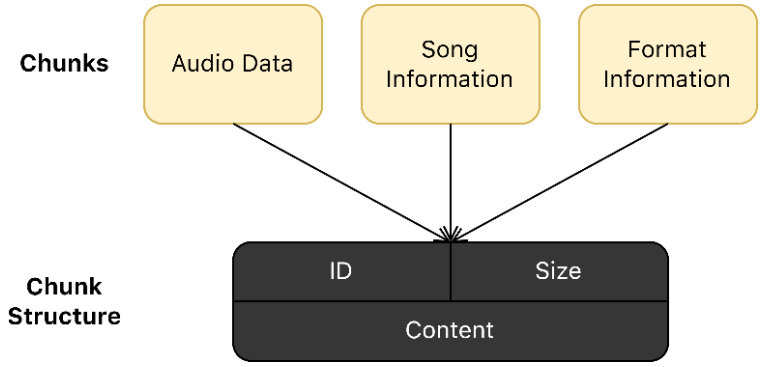


Questo box viene chiamato container (informalmente wrapper). Il container format è quindi un file che consente di incorporare più flussi di dati in un singolo file, insieme ai metadati per identificare e dettagliare tali flussi (nell'esempio sopra c'è solo un flusso).

Formato .WAV

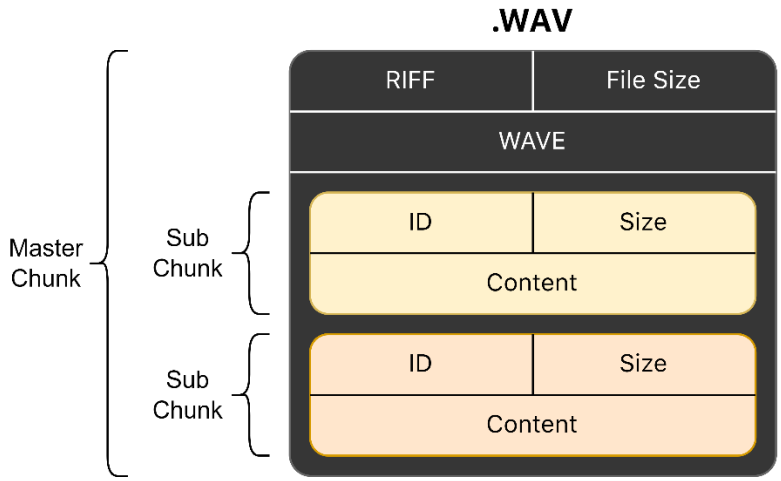
Waveform Audio File Format (WAVE) è uno standard di formato di file audio, sviluppato da IBM e Microsoft, per memorizzazione di un bitstream audio su personal computer. È il formato principale utilizzato sui sistemi Microsoft Windows per l'audio non compresso, molto versatile che supporta diversi tipi di quantizzazione, sample rate, canali, etc.. La codifica più usuale è il formato LPCM

(Linear Pulse-Code Modulation), dove le regioni di quantizzazione sono della stessa dimensione (uniforme).



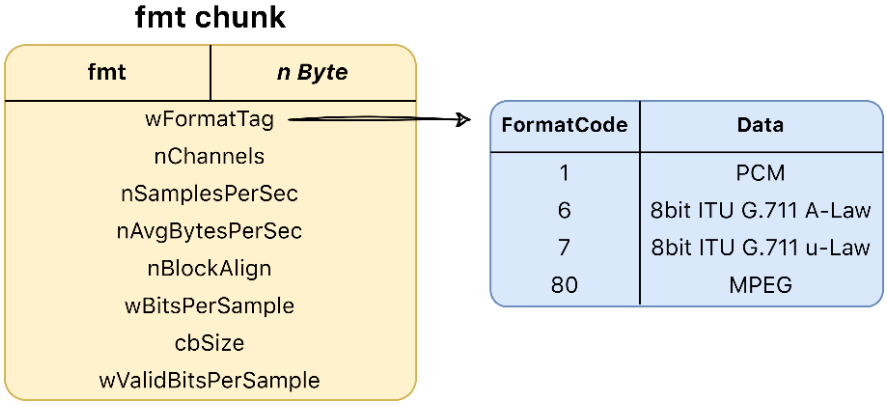
Il container format del formato WAV non è altro che un insieme di chunk. I chunk non sono altro che blocchi di dati che contiene informazioni rilevanti del file. Ogni chunk è descritto dal suo ID, la sua dimensione in byte ed il contenuto del chunk stesso.

Alla radice abbiamo il master Chunk che è il RIFF Chunk. I primi 4 Byte del RIFF Chunk contengono i caratteri "RIFF", i successivi 4 byte indicano la dimensione totale del file, successivamente altri 4 byte con i caratteri "WAVE".

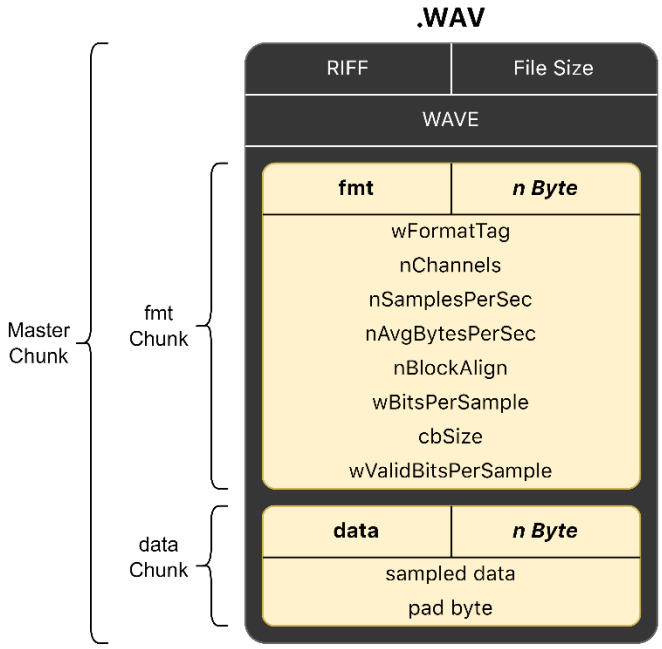


Il contenuto del Master Chunk è una sequenza di Sub-Chunk. Nel formato WAVE, esistono molti chunk, uno dei più importanti è il format chunk "fmt" che contiene informazioni sul come i dati grezzi sono stati elaborati, includendo quindi il numero di canali, il sample rate, etc. L'altro chunk importante è il "data" chunk. In questo chunk sono contenuti i dati audio.

Il format chunk è così formato:



Se mettiamo insieme lo schema generale di un file WAVE con i due chunk (fmt e data) avremo:



Tutto ciò è spiegato meglio qui.

WAVE è un formato di file del sottotipo RIFF. RIFF (Resource Interchange File Format) è una struttura di file con tag per file di risorse multimediali. A rigor di termini, RIFF non è un formato di file, ma una struttura di file che definisce una classe di formati di file più specifici (WAVE appunto, AVI, WebP).

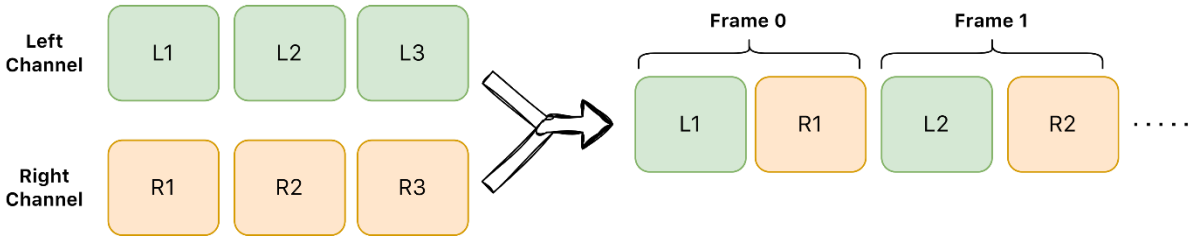
Altri formati per l'audio

Esistono molti formati alla fine. La differenza sta nella "potenza" rappresentativa, cioè, quante cose riesce a specificare rispetto ad un altro formato. Il successo di file come AIFF e WAVE è il fatto che sono formati molto versatili.

Channel Interleaving

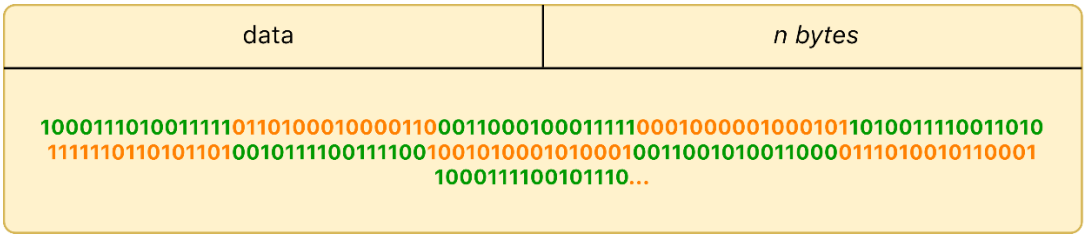
Come vengono immagazinati i campioni nel data chunk, nel caso in cui ci fossero più canali audio? Viene applicata la tecnica dell'interleaving (interfogliamento).

Questa tecnica coinvolge la disposizione sequenziale dei campioni audio provenienti da diversi canali all'interno di un unico flusso di dati. In pratica, invece di avere tutti i campioni di un singolo canale in sequenza seguiti dai campioni di altri canali, i campioni vengono "intrecciati" o mescolati tra i canali. Nel contesto dell'elaborazione dei segnali audio o delle comunicazioni digitali, l'interleaving è utilizzato per organizzare i dati in modo specifico al fine di ottimizzare la trasmissione, la memorizzazione o la riproduzione di informazioni multicanale.



Quindi, come da immagine, vengono memorizzati un campione per canale, in ordine di canale prima di passare al campione successivo. Un frame è costituito da un gruppo di campioni per

ciascun canale. Se si ha un audio stereo (quindi con canali sinistro e destro), un campione di ciascuno canale crea un frame. Ecco un esempio di come sarebbe l'interleaving nel chunk data di un file WAVE codifica LPCM 16bit:



L'esempio è stereo ovviamente.

I vantaggi di inserire i campioni interfogliati, sono molti:

- **Sincronizzazione**: L'interleaving può contribuire alla sincronizzazione di più tracce audio. Ad esempio, in produzioni musicali o nell'audio di un film, è fondamentale che le diverse tracce audio (strumenti musicali, dialoghi, effetti sonori, etc.) siano sincronizzate correttamente. L'interleaving audio può semplificare la gestione di queste tracce e garantire una sincronizzazione accurata.
- **Latenza**: Il fatto di interlacciare i campioni, ottimizza la lettura dei campioni. Se ci fossero prima tutti i campioni di un canale seguiti dai campioni dell'altro canale, per leggere un frame audio (quindi coppia sinistro-destro), bisognerebbe sposarsi in avanti nel file di molti byte per arrivare al campione destro e poi tornare indietro di altrettanti byte per leggere il prossimo campione sinistro.
- **Risparmio di banda**: se hai più fonti audio da trasmettere o registrare, invece di avere un flusso separato per ciascuna fonte, puoi combinare queste fonti in un singolo flusso audio interleaved. Questo può essere particolarmente utile in situazioni in cui la larghezza di banda è limitata, come nella trasmissione radiofonica o nelle trasmissioni in streaming.

Tra gli svantaggi invece:

- **Spreco di spazio**: L'interleaving audio potrebbe richiedere più spazio di archiviazione rispetto a tenere le tracce audio separate, poiché tutte le tracce vengono combinate in un unico flusso. Questo può essere un problema in situazioni in cui lo spazio di archiviazione è limitato, ad esempio quando si tratta di distribuire contenuti online o in ambienti in cui l'efficienza della memorizzazione è importante.
- **Lavoro extra per modifiche**: Se è necessario apportare modifiche a una singola traccia all'interno di un flusso interleave, potrebbe essere più complicato e richiedere più tempo rispetto a modifiche su tracce audio separate.

Supporti fisici per l’audio digitale

Dopo aver visto gli aspetti fondamentali della conversione analogico a digitale e aver esplorato i vari formati di file utilizzati per preservare l'audio digitale, sorge adesso un argomento altrettanto cruciale: i supporti fisici, veri e propri custodi di queste preziose rappresentazioni sonore digitali. Sottolineare l'importanza di questi mezzi di archiviazione è vitale, poiché sono loro a garantire la persistenza nel tempo di dati così rilevanti.

I supporti sono molteplici ed ognuno con caratteristiche fisiche diverse:

- Dischi magnetici
- Dischi ottici
- Memorie a stato solido

Nella scelta del supporto, si effettuano delle scelte in base ad alcuni parametri:

- **Capacità di contenere i dati**: I supporti fisici devono avere abbastanza spazio per ospitare i dati audio digitali. Una capacità ampia è essenziale per immagazzinare registrazioni di lunga durata e ad alta qualità.
- **Velocità di scrittura/lettura per registrazione/riproduzione**: La velocità con cui i dati possono essere scritti e letti dal supporto è cruciale per la registrazione e la riproduzione dell'audio digitale. Una buona velocità garantisce un flusso costante di dati senza ritardi indesiderati.

Il calcolo per determinare quanto spazio occupa il file audio (non compresso) in byte può essere eseguito utilizzando la seguente formula:

$$\text{Size} = F_s \times B \times C \times T$$

Dove:

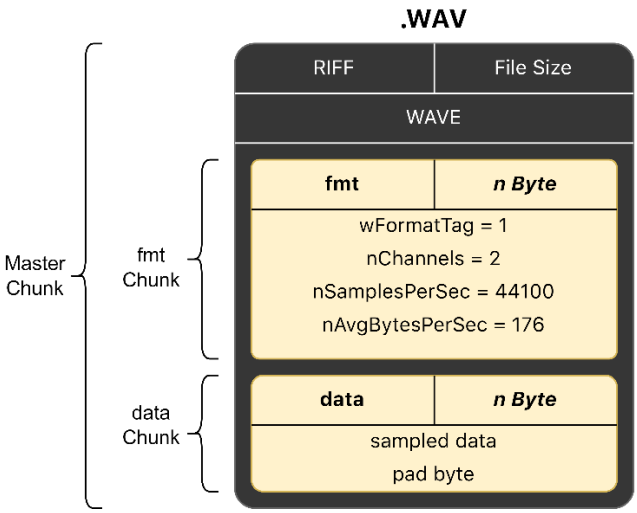
- **Fs** è la frequenza di campionamento
- **B** è il BitDepth (in byte)
- **C** è il numero di canali
- **T** è la durata del brano in secondi.

Avendo un brano campionato a 44.1KHz, una bit Depth di 16, a due canali, con una durata di 210secondi (3 minuti e mezzo), si avrebbe:

$$\text{Size} = 44.100 \times 2 \times 2 \times 210 \approx 37\text{MByte}$$

Un po tanto, infatti non è compresso. Questo potrebbe essere un file WAV o AIFF.

Anche la velocità di lettura e scrittura è cruciale. Supponete uno scenario in cui si ha un file Wave semplificato (senza tutti i parametri) così composto:



Il parametro nAvgBytesPerSec indica quanto deve essere la velocità del supporto che andrà a riprodurre questo file (in particolare quella in lettura). Per calcolare la bandwidth necessaria per seconds, è sufficiente seguire questa formula:

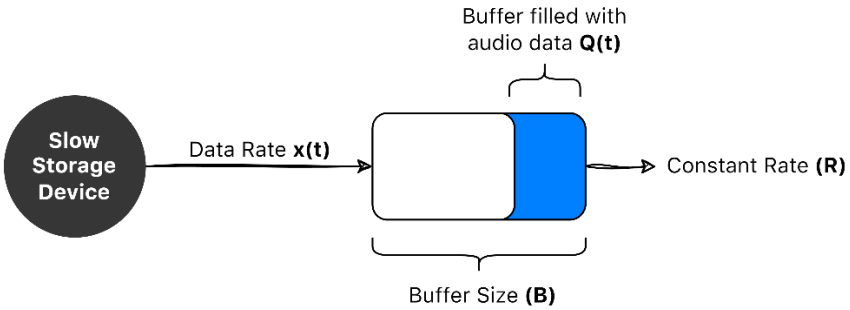
$$\mathrm{BW} = F_s \times B \times C$$

Per il file WAVE proposto sarà:

$$\mathrm{BW} = 44.100 \times 2 \times 2 \approx 176\mathrm{KByte}$$

Che è esattamente il dato indicato nel file WAVE (Per quanto riguarda la bandwith di rete, quindi bit/s basta moltiplicare per 8).

Cosa succede se il data rate del supporto in cui è memorizzato questo file WAV è inferiore a quella necessaria per riprodurre il file (in questo caso 176KByte/s)? Semplicemente non sarà possibile riprodurre il file. È necessario quindi introdurre un buffer che permetta di assorbile il collo di bottiglia dovuto alla scarsa velocità di lettura.

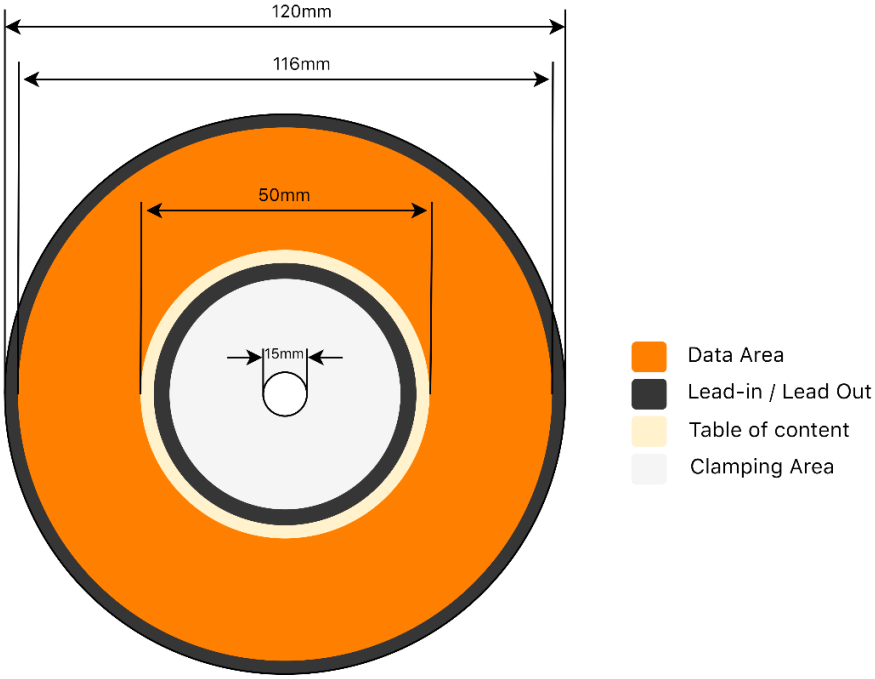


Se x(t) è minore del rate R, è chiaro che il buffer B si svuoterà istantaneamente. È necessario quindi immagazzinare prima dei dati nel buffer B. In questo modo si assorbe la scarsa velocità di lettura del supporto.

Compact Disk

Il compact disc (CD) è un supporto fisico di archiviazione dati su disco ottico digitale sviluppato congiuntamente da Philips e Sony per archiviare e riprodurre registrazioni audio digitali. Il formato è stato successivamente adattato (come CD-ROM) per l'archiviazione di dati generici. Diversi altri formati sono stati ulteriormente derivati, tra cui l'audio riscrivibile e l'archiviazione dei dati (CD-R), supporti riscrivibili (CD-RW), Video CD (VCD), Super Video CD (SVCD).

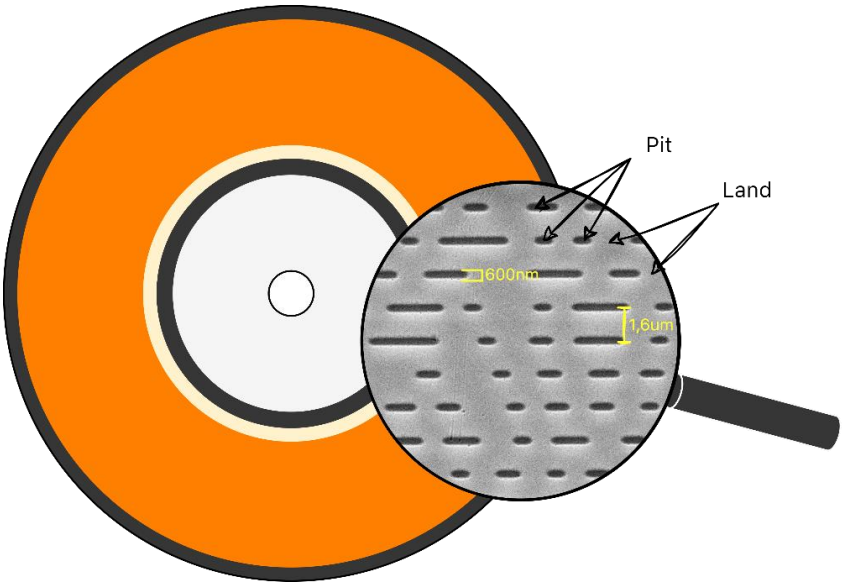
A livello fisico è così definito:



I CD standard hanno un diametro di 120 millimetri (4,7 pollici) e sono progettati per contenere fino a 74 minuti di audio digitale stereo non compresso o circa 650 MiB di dati.

Dal centro verso l'esterno, i componenti sono: il foro centrale (15 mm), l'area di prima transizione (anello di bloccaggio), l'area di seconda transizione che è la lead-in, un'area di TOC (table fo content), successivamente l'area dei dati ed infine l'area di lead-out. Si può dire che l'area dati è compresa tra le aree di lead.

I dati sono rappresentati come minuscole rientranze note come pit, codificate in una traccia a spirale. Le aree tra i pit sono conosciute come land:



Si potrebbe pensare che i pit rappresentano l'uno binario e i land lo zero, ma non è così. Infatti viene utilizzata la **rappresentazione NRZI (Non-Return-To-Zero Inverted)**. In sostanza, con questa codifica, un passaggio da pit a land o da land a pit indica un uno, mentre nessun cambiamento indica una serie di zero.

000100000001000100010000000000100010001000000001000

I dati che verranno scritti sul disco però, vengono codificati in una modulazione chiamata **Eight-To-Fourteen Modulation (modulazione da otto a quattordici)**.

Utilizzando questa modulazione, i dati che dovranno essere salvati sul disco, prima vengono raggruppati a gruppi 8 bit. Ogni blocco di 8 bit viene tradotto in una corrispondente parola di codice di quattordici bit utilizzando una lookup table. Le parole di 14 bit sono scelte in modo tale che quelle binarie siano sempre separate da un minimo di due e un massimo di dieci zeri binari. Qui la tabella.

EFM minimizza le transizioni 0-1 e 1-0 in lettura, assicurano che ci siano almeno due zeri tra ogni due uno. È così garantito che ogni pit e land sia lungo almeno tre cicli di bit-clock. Questa proprietà è molto utile poiché riduce gli errori del pickup ottico utilizzato nel meccanismo di riproduzione. Il massimo di dieci zeri consecutivi garantisce il recupero del clock nel caso peggiore nel player.

EFM richiede tre bit di fusione tra parole di codice adiacenti a quattordici bit. Sebbene non siano necessari per la decodifica, assicurano che le parole di codice consecutive possano essere concatenate senza violare il vincolo di lunghezza minima e massima specificata. Sono anche selezionati per mantenere l'equilibrio DC della sequenza codificata. Pertanto, in ultima analisi, sono necessari diciassette bit di spazio su disco per codificare otto bit di dati.

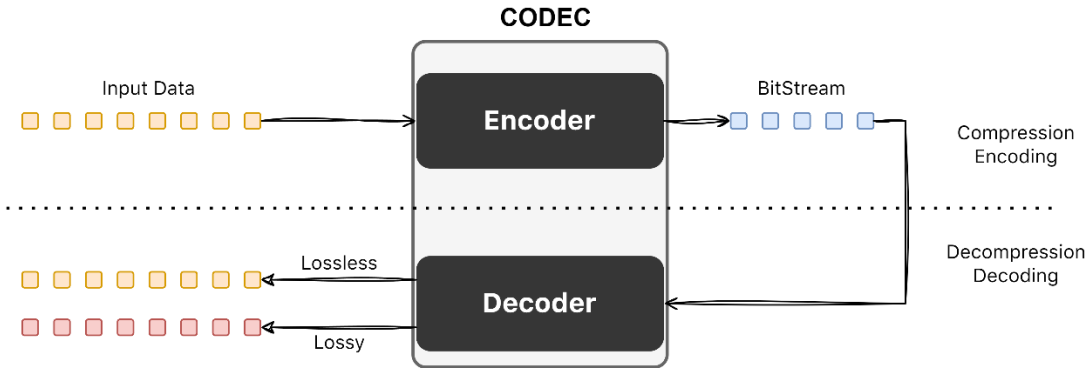
Prima di venire effettivamente scritti, i dati vengono dati in pasto ad un algoritmo che aggiunge ridondanza ed Interleaving in modo da renderlo resiliente ad errori. Questo algoritmo è chiamato Cross-interleaved Reed–Solomon.

Compressione

Fin ora abbiamo visto come avviene la trasformazione da analogico a digitale, i file che sono il contenitore delle informazioni digitali ed i supporti fisici che permettono la permanenza di quest'ultimi.

In passato, i dispositivi di memoria non erano un granchè, a malapena si riusciva ad arrivare al Gigabyte. Questo ha portato gli ingegneri del software e matematici a cercare metodi per rendere le codifiche audio più leggere e di conseguenza i file che immagazzinavano queste codifiche audio, da qui è nata la compressione Audio. Anche se ora si è un po risolto il problema dello spazio, c'è sempre un fattore di trasmissione (sulla rete internet ad esempio) che spinge ad utilizzare la compressione ancora in modo rilevante. Se si suppone ad esempio che ognuno di noi ascoltasse Spotify senza compressione, avremmo gli apparati di rete dei nostri Provider saturi di traffico da smaltire (e sarebbe soltanto audio, mischiato ad altri flussi di traffico sarebbe ancora peggio).

Schema di compressione



Questo è un classico schema di compressione. Il cuore dello schema è il codec. Un codec è un dispositivo o software che codifica o decodifica un flusso. Esso stesso è una combinazione di encoder/decoder: l'Encoder (codificatore) codifica un flusso di dati o un segnale per la trasmissione o l'archiviazione, mentre il Decoder (decodificatore) inverte la codifica per la riproduzione o la modifica.

Storia: A metà del XX secolo, un codec era un dispositivo che codificava i segnali analogici in forma digitale utilizzando la modulazione del codice a impulsi (PCM). Successivamente, il nome è stato applicato anche al software per la conversione tra formati di segnali digitali, comprese le funzioni di compressione, come MPEG.

Se si passa da un formato non compresso ad uno compresso, i dati audio in ingresso sono una successione di campioni, ed in uscita si ha un bitstream che non è identificabile (non si hanno campini audio distinguibili, in quanto è stato applicato uno schema di encoding). In fase di decoding invece quel bitstream non identificabile sarà ricostruito nei campioni audio e riprodotto.

Lossless e lossy

Due approcci comuni alla compressione audio sono la compressione "lossless" e la compressione "lossy". Queste due tecniche si differenziano principalmente per il modo in cui trattano i dati durante il processo di compressione e decompressione.

Per **compressione Lossless** si intende che l'audio compresso e poi decompresso sarà identico all'originale, senza alcuna perdita di qualità. In altre parole, tutti i dati vengono mantenuti intatti durante il processo. Questo è possibile attraverso algoritmi di compressione che cercano modelli e ridondanze nei dati audio e li rappresentano in modo più efficiente. I formati di compressione lossless comuni includono FLAC, ALAC (Apple Lossless), e WAV (senza compressione effettiva ma

con alcune informazioni di metadati compressi). La **compressione Lossy** d'altro canto, si basa sulla rimozione selettiva di informazioni considerate meno udibili o rilevanti per l'ascolto umano. Questo processo comporta una perdita di qualità audio, che può variare in base alla quantità di dati rimossi. Tuttavia, una buona compressione lossy cerca di minimizzare la perdita di qualità mantenendo le parti più cruciali dell'audio. Il vantaggio principale della compressione lossy è che riduce significativamente le dimensioni dei file, consentendo di memorizzare o trasferire più file audio in spazi limitati. Formati di compressione lossy noti includono MP3, AAC, e OGG.

La compressione audio lossy ha avuto quindi un impatto rivoluzionario nel mondo digitale per diverse ragioni come

- **Efficienza di spazio**: La compressione lossy ha reso possibile la diffusione e lo scambio di file audio attraverso internet, quando le velocità di connessione erano più lente e lo spazio di archiviazione era più limitato. Questo ha aperto la strada al download e alla condivisione di file audio su scala globale.
- **Streaming**: La compressione lossy è fondamentale per la diffusione di contenuti audio in streaming, poiché consente di trasmettere audio di qualità accettabile su reti con larghezza di banda limitata. Questo ha portato all'esplosione dei servizi di streaming musicale.
- **Portabilità**: La compressione lossy ha reso possibile l'ascolto di musica in movimento attraverso dispositivi mobili come lettori MP3 e smartphone. Questo ha cambiato radicalmente le abitudini di ascolto delle persone.

Simmetria tra coding e decoding

Questo concetto si riferisce alla simmetria del processo di compressione e decompressione: se entrambi i processi richiedono la stessa quantità di risorse computazionali e tempo, l'algoritmo è considerato simmetrico; se richiedono quantità diverse, l'algoritmo è considerato asimmetrico.

Gli algoritmi di **compressione simmetrici** richiedono lo stesso livello di risorse e tempo sia per il processo di compressione che per quello di decompressione. Ciò significa che il tempo e le risorse richieste per comprimere un dato sono approssimativamente equivalenti a quelli richiesti per decomprimerlo. Questi algoritmi sono particolarmente utili in applicazioni in cui la simmetria è importante, ad esempio in sistemi in cui i dati devono essere compressi su un lato e decompressi su un altro lato con risorse bilanciate.

Gli algoritmi di **compressione asimmetrici** invece richiedono quantità diverse di risorse e tempo per la compressione e la decompressione. Spesso, l'obiettivo principale di tali algoritmi è la compressione, mentre la decompressione può richiedere meno risorse. Questo può essere vantaggioso in scenari in cui la compressione viene eseguita una sola volta e la decompressione avviene molte volte. Gli esempi includono molti algoritmi di compressione con perdita, come JPEG (immagini) e MP3 (audio).

La maggior parte degli algoritmi che usiamo sono asimmetrici, perché la decompressione deve essere molto più semplice della compressione in quanto è operata dai consumer, che hanno strumenti molto più deboli, molto più poveri dal punto di vista computazionale. Deve essere posibile decomprimere in modo efficiente da qualsiasi lettore MP3, computer, smartphone o dispositivo in generalle.

Complessita del codec

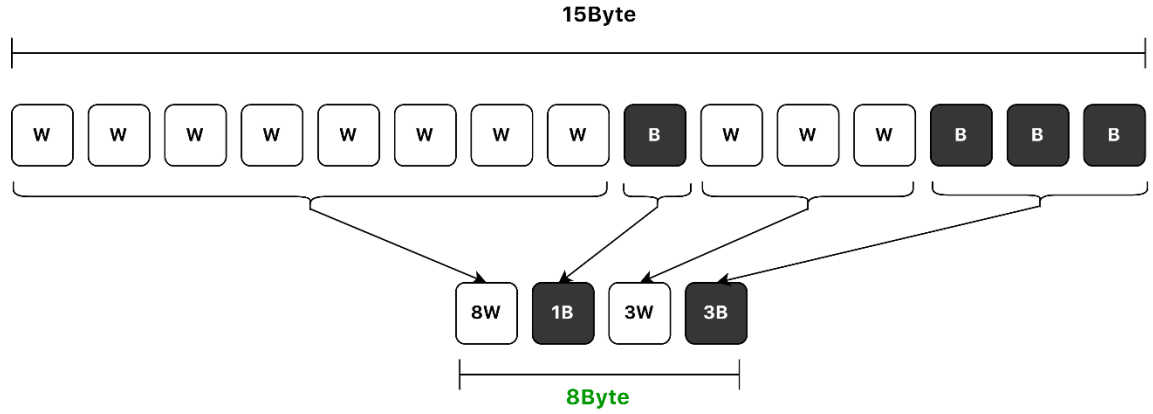
Può essere semplice o complesso quando si basa su pochi principi e semplice cose molto banali, come vedremo tra poco, e diventa complesso quando invece usa pesantemente la la psicoacustica, cioè i principi percettivi per poter studiare bene il segnale. Infatti, quello che avviene è che la ridondanza che noi possiamo recuperare la recuperiamo quando applichiamo i principi della psicoacustica prima non riusciamo a recuperarlo.

Run-Length Encoding

La codifica Run Length Encoding (RLE) è una forma di compressione dei dati senza perdita (lossless) in cui le **run** di dati (sequenze in cui lo stesso valore di dati si verifica in molti elementi di dati consecutivi) vengono archiviate come un singolo valore di dati e vengono conteggiate, anziché come l'esecuzione originale.

Molti algoritmi di compressione (come RLE) funzionano grazie ad un principio generale della compressione che è la ridondanza. La ridondanza si riferisce alla presenza di informazioni ripetitive o prevedibili in un insieme di dati. In particolare, non tutti i dati sono necessari per ricostruire l'informazione.

Questa codifica è stata inventata in principio per la compressione lossless di immagini, il funzionamento è abbastanza semplice ed è il seguente:



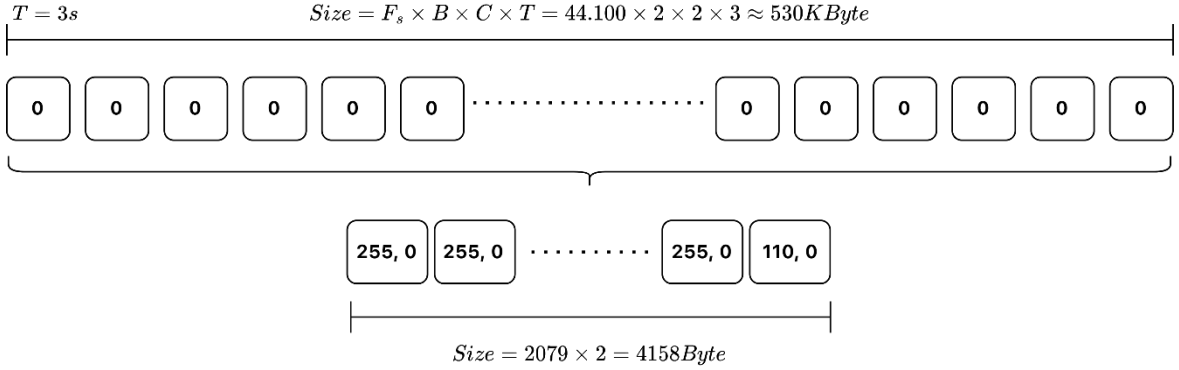
La sequenza originaria era composta da 15Byte in cui ogni byte indicava White or Black. Applicando RLE, si sfrutterà la ridondanza di dati contigui (chiamati appunto **run**) indicando il numero di occorrenze per ogni run.

Se ci spostiamo nel mondo dell'audio, supponiamo di avere 3 secondi di silenzio di una traccia CD Audio. Quindi codifica LPCM 16bit 44.1KHz, avremo:

$F_s = 44.100Hz$

$B = 16bit = 2Byte$

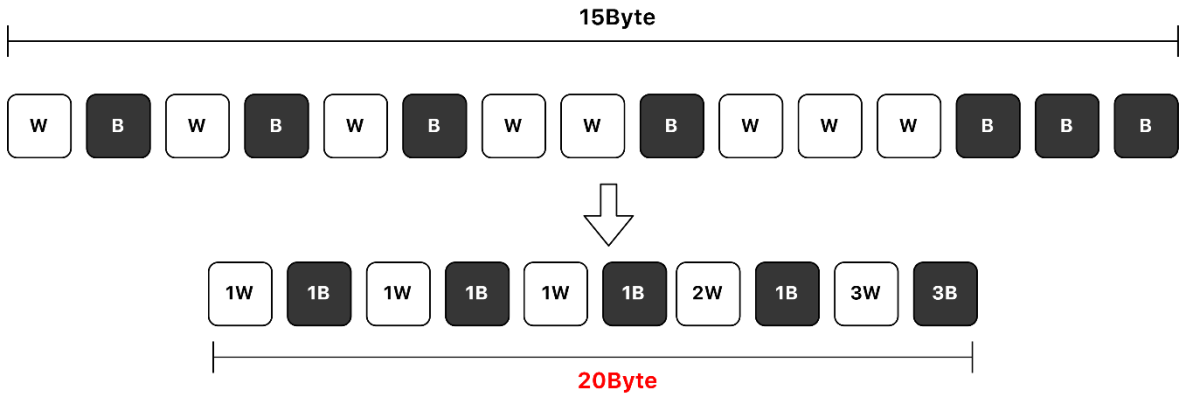
$T = 3s$ $\qquad Size = F_s \times B \times C \times T = 44.100 \times 2 \times 2 \times 3 \approx 530KByte$



$Size = 2079 \times 2 = 4158Byte$

Si è passati da 530 KByte a 4,1 KByte. Ogni run è scritta su un byte (quindi massimo 255 per run) più il valore del run (sempre 1 byte). Come nel caso di 530.000 zeri contigui, se un run supera il byte, si spezza su più run contigui, qui una risorsa.

Questa tecnica è più efficiente sui dati che contengono molte ripetizioni. Per i file che non hanno molte ripetizioni invece, RLE potrebbe aumentare la dimensione del file. Ad esempio:



Complessità RLE

Per quanto riguarda l'asimmetria tra encoding e decoding, i due schemi sono abbastanza simmetrici, se non perfettamente simmetrici. Sia encoder che decoder hanno una complessità temporale nell'ordine di $\mathcal{O}(N)$. In termini di spazio, nel caso peggiore la complessità raddoppia $\mathcal{O}(N^2)$, altrimenti $\mathcal{O}(1)$ in quanto nessuno spazio extra è utilizzato.

RLE con gate

Esiste una versione di RLE che comprime il silenzio. In sostanza si ha una soglia entro il quale tutto il segnale viene portato a zero, successviamente si effettua RLE sulle porzioni di segnale che sono state ridotte a zero. Questa applicazione è lossy.

µ-Law

Questo pezzo è estratto dal libro "Digital Signal Processing: Fundamentals and Applications" di Li Tan (Purdue University Northwest · Department of Electrical & Computer Engineering Ph.D EE).

L'algoritmo µ-law (mu-law) è un algoritmo companding che accetta un segnale di ingresso analogico utilizzato principalmente nei sistemi di telecomunicazione in Nord America e Giappone ma ha anche un applicazione di compressione digitale. È uno dei due algoritmi di companding nello standard G.711 di ITU-T, l'altro è A-law. L'algoritmo di companding µ-law, è un metodo utilizzato per la compressione non lineare dei segnali audio analogici. Il vantaggio della codifica mu-law è che preserva parte della gamma dinamica che andrebbe persa se venisse utilizzato un metodo lineare per ridurre il bit depth per campione.

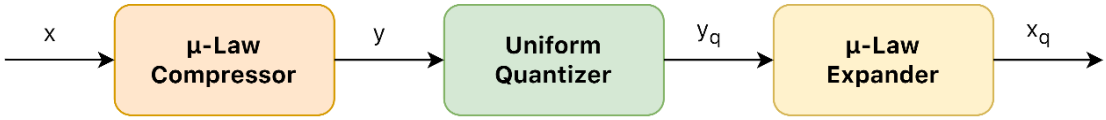
Gli algoritmi di companding riducono la gamma dinamica di un segnale audio. Nei sistemi analogici, ciò può aumentare il rapporto segnale/rumore (SNR) ottenuto durante la trasmissione; nel dominio digitale, può ridurre l'errore di quantizzazione (quindi aumentando l'SQNR).

Si può dimostrare matematicamente che la quantizzazione non lineare di µ-law aumenta effettivamente la gamma dinamica di 33 dB o 5/2 bit su un segnale quantizzato linearmente, quindi 13,5 bit (che arrotonda per eccesso a 14 bit) è la risoluzione massima richiesta per un segnale digitale in ingresso da comprimere per µ-law a 8 bit.

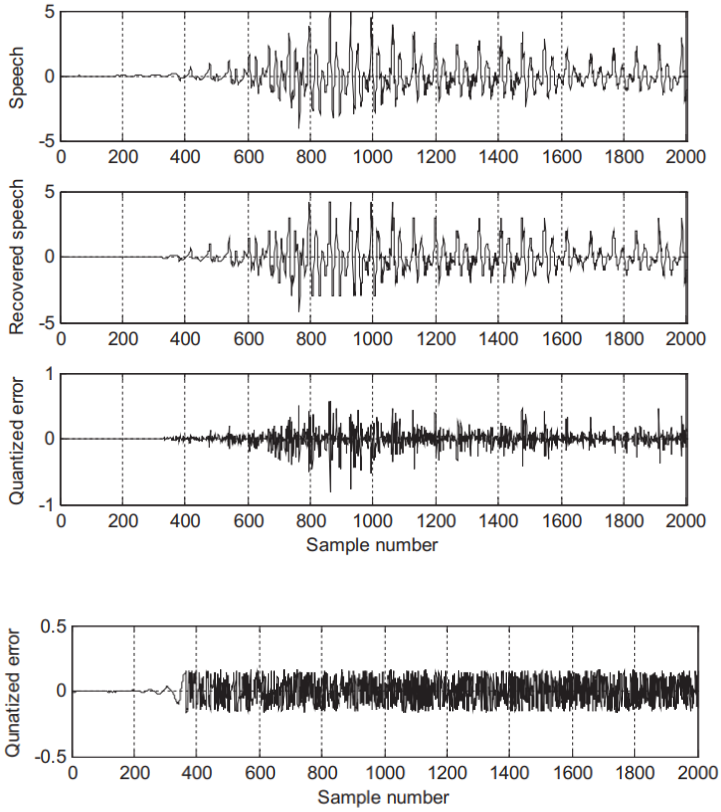
Analog µ-Law companding

Per ridurre il numero di bit richiesti per codificare ciascun dato vocale, viene applicata la compressione µ-Law, chiamata anche codifica log-PCM. La compressione µ-Law è un processo di compressione ed esplora il principio secondo cui le ampiezze più elevate dei segnali analogici vengono compresse prima dell'ADC ed espanse dopo la conversione da digitale ad analogico (DAC). Come studiato nel quantizzatore lineare, l'errore di quantizzazione è distribuito

uniformemente. Ciò significa che l'errore di quantizzazione massimo rimane lo stesso, indipendentemente da quanto grandi o piccoli siano i campioni vocali. La compressione μ-Law può essere utilizzata per ridurre l'errore di quantizzazione quando l'ampiezza del campione è minore e per aumentare l'errore di quantizzazione quando l'ampiezza del campione è maggiore, utilizzando lo stesso numero di bit per campione.



Come mostrato sopra, x è il campione vocale originale, che è l'input del compressore, mentre y è l'output del compressore μ-Law. Quindi l'uscita y viene quantizzata uniformemente. Supponendo che il campione quantizzato y_q sia codificato e inviato all'espansore μ-Law, l'espansore eseguirà il processo inverso per ottenere il campione vocale quantizzato x_q. I processi di compressione e decompressione fanno sì che l'errore di quantizzazione massimo $|x_q - x|_{max}$ sia piccolo per le ampiezze del campione più piccole e grande per le ampiezze del campione più grandi.



La figura sopra mostra i dati vocali originali, i dati vocali quantizzati utilizzando la compressione μ-Law e l'errore di quantizzazione (l'ultimo grafico è l'errore di quantizzazione ad 8 bit LPCM). L'onda vocale quantizzata è molto vicina all'onda vocale originale. Possiamo osservare che l'ampiezza dell'errore di quantizzazione cambia in base all'ampiezza. Un errore di quantizzazione maggiore viene introdotto quando l'ampiezza dei dati vocali è maggiore; d'altro canto, un errore di quantizzazione minore si produce quando l'ampiezza dei dati vocali è minore. Invece l'ultimo grafico, mostra come l'errore di quantizzazione sia uniforme con LPCM.

L'equazione per la compressione μ-Law è la seguente:

$$F(x) = \mathrm{sgn}(x)\frac{\ln(1+\mu|x|)}{\ln(1+\mu)}, \quad -1 \le x \le 1$$

Dove **|x|** è l'input normalizzato tra -1 ed 1, mentre **μ** è un parametro positivo per controllare il grado di compressione: μ = 0 corrisponde a nessuna compressione, mentre μ = 255 è quella adottata da ormai tutti.

In espansione invece, l'equazione è:

$$F^{-1}(y) = \operatorname{sgn}(y)\frac{(1+\mu)^{|y|}-1}{\mu}, \quad -1 \le y \le 1$$

Digital u-law companding

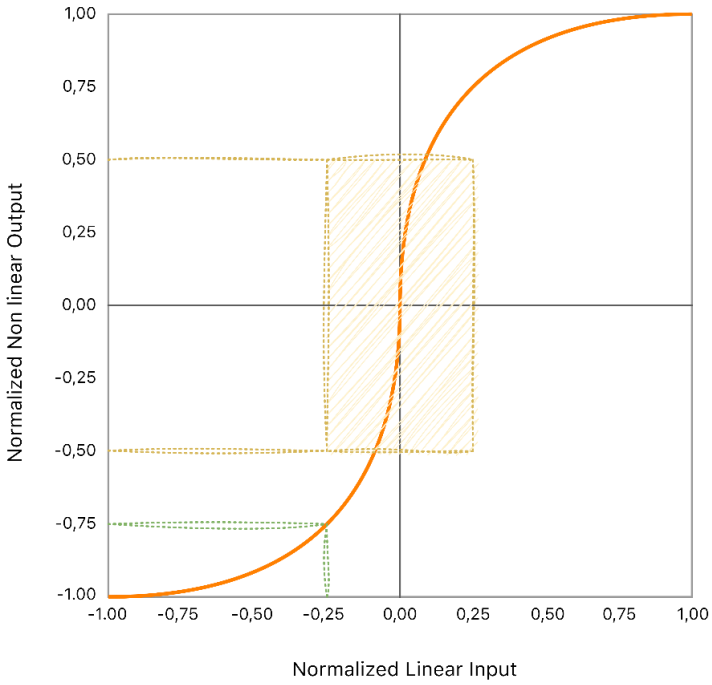
La seconda applicazione che trova μ-Law è nel digitale, dato che in molte applicazioni multimediali, il segnale analogico viene prima campionato e poi digitalizzato in un codice PCM lineare con un numero maggiore di bit per campione, se si vuole partire da un segnale già digitalizzato in LPCM, e si vuole risparmiare ulteriormente spazio/banda, la compressione digitale μ-Law comprime ulteriormente il codice PCM lineare con un numero inferiore di bit per campione (8) senza perdere la qualità del suono.

Si immagini questo scenario: deve essere trasmesso un segnale audio telefonico. Originariamente è a 16 bit. Per ridurre la quantità di dati da trasmettere, i campioni vengono ridotti a 8 bit. In una compressione lineare, riducendo la bit depth, la quantità di errore di quantizzazione a basse ampiezze è uguale alla quantità di errore ad ampiezze elevate. Tuttavia, sappiamo che il rumore di quantizzazione è maggiormente percepito per campioni di ampiezza bassa che per campioni di ampiezza alta.

L'equazione (che è, più precisamente, una funzione) ridistribuisce efficacemente i valori del campione in modo che ci siano più livelli di quantizzazione ad ampiezze inferiori e meno livelli di quantizzazione ad ampiezze più elevate. L'equazione di compressione è la stessa:

$$F(x) = \operatorname{sgn}(x)\frac{\ln(1+\mu|x|)}{\ln(1+\mu)}, \quad -1 \le x \le 1$$

Quello che succede è che il campione di input viene normalizzato tra -1 e 1. Il campione normalizzato viene elaborato dalla funzione sopra che lo rapporta su una scala logaritmica. Infatti la fase di compressione u-law non fa altro che quantizzare in modo non uniforme l'input.

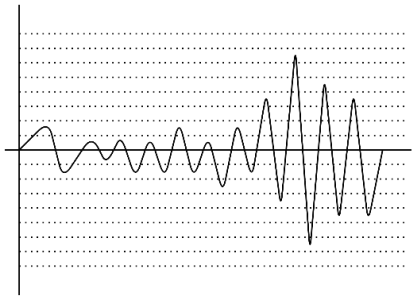


Si può notare che i valori agli estremi (ampiezza alta) occuperanno un range di valori minore rispetto ai valori a bassa ampiezza. Ad esempio i valori di input normalizzati tra -1 e -0,25 occuperanno solo il range di valori da -1 a -0,75. Mentre i valori a bassa amplitude (da -0,25 a +0,25) occuperanno un range di valori molto più ampio. In questo modo si lascia più spazio di dettaglio alle ampiezze deboli e meno spazio di dettaglio alle ampiezze estreme. L'effetto in sostanza è quello di aggiungere rumore proporzionale alla potenza del segnale.

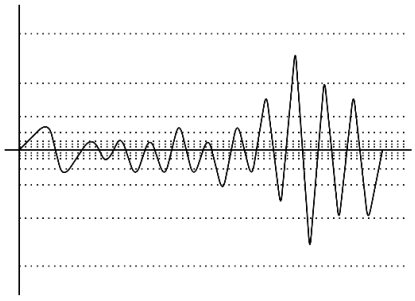
Ecco un esempio in cui si prendono 4 valori da un segnale aventi campioni con bit depth 16 (-32768, + 32767) che vengono normalizzati tra 0 e 1 (primo step), poi vengono dati in pasto alla funzione di compressione mu-law, ed infine il valore in uscita da F viene riportato su una scala a 8 bit (-128, + 127):

Input 16 bit	$\frac{input}{32768}$ ⇑ Normalized input	$F(x) = \text{sgn}(x)\frac{\ln(1+\mu\lvert x\rvert)}{\ln(1+\mu)}$ ⇑ u-law, non linear compressing	$F(x) \times 128$ ⇑ Output 8 bit	
33	0,001007	0,0412	5	4 region distance
66	0,002014	0,0747	9	
32145	0,980987	0,9966	127	Same region
32178	0,981994	0,9967	127	

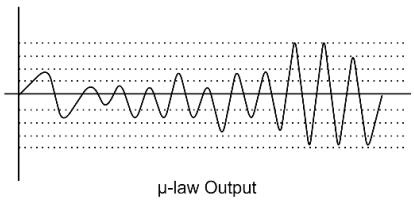
Si nota come la compressione sia non lineare, in quanto 33 e 66, così come 32145 e 32178, distino di 33 regioni, ma una volta effettuata la compressione e riportato il valore a 8 bit, si osserva che tra 33 e 66 ci sono 4 regioni di distanza, mentre 32145 e 32178 sono stati riportati addirittura nella stessa regione. L'errore di quantizzazione è più accentuato alle ampiezze alte e di conseguenza anche il rumore di quantizzazione. Qui c'è l'esempio spiegato. Una volta effettuata la compressione avremo qualcosa di simile:



Input Uniform Quantized



Non Uniform Quantized

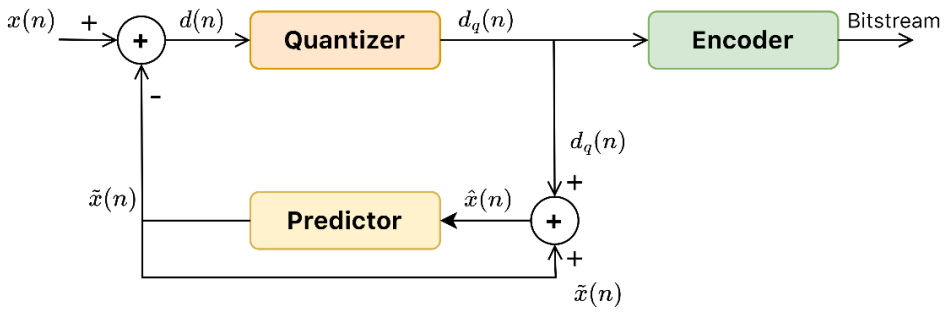


μ-law Output

DPCM

La compressione dei dati può essere ulteriormente ottenuta utilizzando la Differential Pulse Code Modulation (DPCM). L'idea generale è quella di utilizzare i valori recuperati in passato come base per prevedere i dati di input correnti e quindi codificare la differenza tra l'input corrente e l'input

previsto. Poiché la differenza ottenuta ha una gamma dinamica del segnale significativamente ridotta, può essere codificata con meno bit per campione. Pertanto, otteniamo la compressione dei dati.

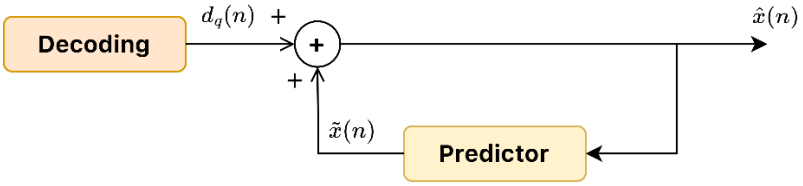


La Figura mostra un diagramma schematico per il codificatore DPCM, dove:

- $\boldsymbol{x(n)}$ è il segnale originale in ingresso
- $\boldsymbol{\tilde{x}(n)}$ è il segnale predetto
- $\boldsymbol{\hat{x}(n)}$ è il segnale quantizzato
- $\boldsymbol{d(n)}$ è il segnale differenza da quantizzare
- $\boldsymbol{d_q(n)}$ è il segnale differenza quantizzato.

Il quantizzatore può essere scelto come quantizzatore uniforme, o qualsiasi altro quantizzatore disponibile. Il blocco di codifica produce un flusso di bit binario nella codifica DPCM. Il predittore utilizza il segnale previsto in passato e il segnale di differenza quantizzata per prevedere il valore di ingresso corrente $\boldsymbol{x(n)}$ il più vicino possibile.

D'altra parte, il decoder recupera il segnale differenza quantizzato, che può essere aggiunto al segnale di uscita del predittore per produrre il segnale quantizzato e recuperato:



Come sempio, prendiamo questi valori di input: x(0) = 6, x(1) = 8, x(2) = 13. Avremo:

n = 0 x(0) = 6	n = 1 x(1) = 8	n = 0 x(2) = 13	
$\tilde{x}(0) = \hat{x}(-1) = 0$	$\tilde{x}(1) = \hat{x}(0) = 5$	$\tilde{x}(2) = \hat{x}(1) = 7$	
$d(0) = x(0) - \tilde{x}(0) = 6 - 0 = 6$	$d(1) = x(1) - \tilde{x}(1) = 8 - 5 = 3$	$d(2) = x(2) - \tilde{x}(2) = 13 - 7 = 6$	
$d_q(0) = Q[d(0)] = 5$	$d_q(1) = Q[d(1)] = 2$	$d_q(2) = Q[d(2)] = 5$	
$\hat{x}(0) = \tilde{x}(0) + d_q(0) = 0 + 5 = 5$	$\hat{x}(1) = \tilde{x}(1) + d_q(1) = 5 + 2 = 7$	$\hat{x}(2) = \tilde{x}(2) + d_q(2) = 7 + 5 = 12$	
binary code: 110	**binary code: 101**	**binary code: 110**	**Encoding**
binary code: 110	**binary code: 101**	**binary code: 110**	**Decoding**
$d_q(0) = 5$	$d_q(1) = 2$	$d_q(1) = 5$	
$\tilde{x}(0) = \hat{x}(-1) = 0$	$\tilde{x}(1) = \hat{x}(0) = 5$	$\tilde{x}(2) = \hat{x}(1) = 7$	
$\hat{x}(0) = \tilde{x}(0) + d_q(0) = 0 + 5 = 5$	$\hat{x}(1) = \tilde{x}(1) + d_q(1) = 5 + 2 = 7$	$\hat{x}(2) = \tilde{x}(2) + d_q(2) = 7 + 5 = 12$	

Per l'esempio è stat usata la tabella seguente:

Table 11.5 Quantization Table for the 3-bit Quantizer in Example 11.5

Binary Code	Quantization Value $d_q(n)$	Subrange in $d(n)$
0 1 1	−11	$-15 \le d(n) < -7$
0 1 0	−5	$-7 \le d(n) < -3$
0 0 1	−2	$-3 \le d(n) < -1$
0 0 0	0	$-1 \le d(n) < 0$
1 0 0	0	$0 \le d(n) \le 1$
1 0 1	2	$1 < d(n) \le 3$
1 1 0	5	$3 < d(n) \le 7$
1 1 1	11	$7 < d(n) \le 15$

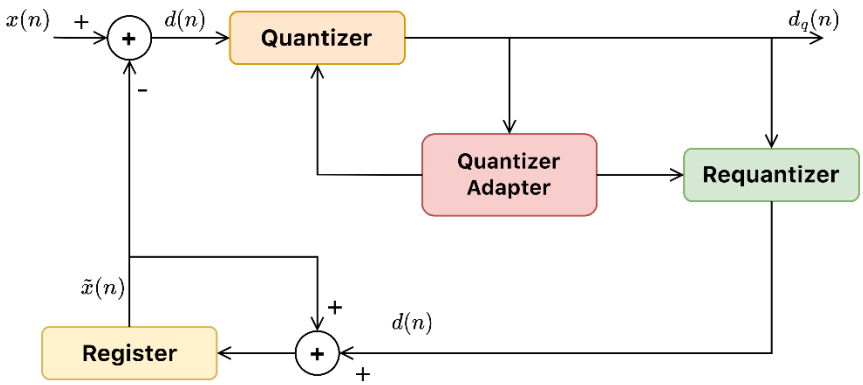
Questi esempi sono sempre presi dal libro "Digital Signal Processing: Fundamentals and Applications" di Li Tan (Purdue University Northwest · Department of Electrical & Computer Engineering Ph.D EE).

Da questo esempio possiamo verificare che il codice a 5 bit è compresso nel codice a 3 bit. Tuttavia, possiamo vedere che ogni pezzo di dati recuperato presenta un errore di quantizzazione. Pertanto, DPCM è tipo schema di compressione dei dati lossy.

Il problema è quando il segnale in ingreso possiede differenze elevate tra campioni consecutivi (problame di slope overload). Questo può essere risolto utilizzando una frequenza di campionamento molto elevata (molto maggiore della frequenza di Nyquist) e rendendo adattiva la dimensione del passo di quantizzazione (ADPCM).

ADPCM

Il problema dello slope overload, viene risolto utilizzando quindi una DPCM adattiva (Adaptive DPCM). Quello che accade è che i bit disponibili per la quantizzazione vengono adattati in base a come sta variando la forma d'onda.



Il core è appunto il Quantizer Adapter che adatta il passo di quantizzazione in base alla differenza attuale.

Se si passa da LPCM a (A)DPCM, si ha una compressione del 75%, se si parte da un algoritmo Law invece 50%.

Compressione percettiva

Le forme di compressione standard che abbiamo visto (law, DPCM, ADPCM) non producono rilevanti risparmi di spazio e tempo ed inoltre peggiorano la qualità.

L'evoluzione delle tecnologie audio ha portato alla creazione di formati di compressione che si basano sulla percezione umana e sulla psicoacustica per ottimizzare l'efficienza nella riduzione delle dimensioni dei file audio. Questo approccio alla compressione audio mira ad eliminare dati che il nostro udito umano non percepirebbe, consentendo così una considerevole riduzione delle dimensioni dei file senza una perdita significativa di qualità.

Molti algoritmi di compressione percettiva utilizzano la tecnica di transform coding. Il **transform coding** è un tipo di compressione dei dati utilizzata per dati "naturali" come segnali audio o immagini. La codifica funziona convertendo il segnale audio in un dominio diverso (da tempo a frequenza), utilizzando una trasformazione lineare reversibile come la trasformata coseno discreta (DCT) o la trasformata discreta di Fourier (DFT). Questo permette una quantizzazione più mirata e precisa rispetto al dominio del tempo. Una volta trasformati, i dati vengono analizzati e quantizzati secondo un modello psicoacustico che descrive la sensibilità dell'orecchio umano alle diverse parti del segnale. Questo processo invece aiuta a scartare le informazioni meno importanti così che le informazioni rimanenti possano essere poi compresse utilizzando vari metodi, come Run Length Encoding (RLE), Shift Coding o tecniche di codifica entropica, come l'algoritmo di Huffman (che vedremo).

In particolare, per la compressione audio, si utilizza un particolare tipo di transform coding chiamata **Sub Band Coding** (**SBC**). **SBC** è una forma di transform coding che suddivide un segnale in un numero di bande di frequenza diverse e le codifica ciascuna in modo indipendente.

Negli algoritmi basati sulla percezione, quando al ricevitore l'audio compresso viene decodificato, il risultato potrebbe non essere identico all'input originale, ma dovrebbe essere sufficientemente vicino all'originale tanto da non sentirne le differenze (si chiama in questo csao di riproduzione trasparente). Questo tipo di compressione è considerato lossy, poiché alcune informazioni vengono perse durante il processo di compressione e che arrivano addirittura a compressioni di 1/2bit per campione.

Principi Psicoacustici

I principi psicoacustici che si applicano agli algoritmi di compressione percettivi che utilizzano il transform coding per ridurre le informazioni ridondanti o non necessarie sono:

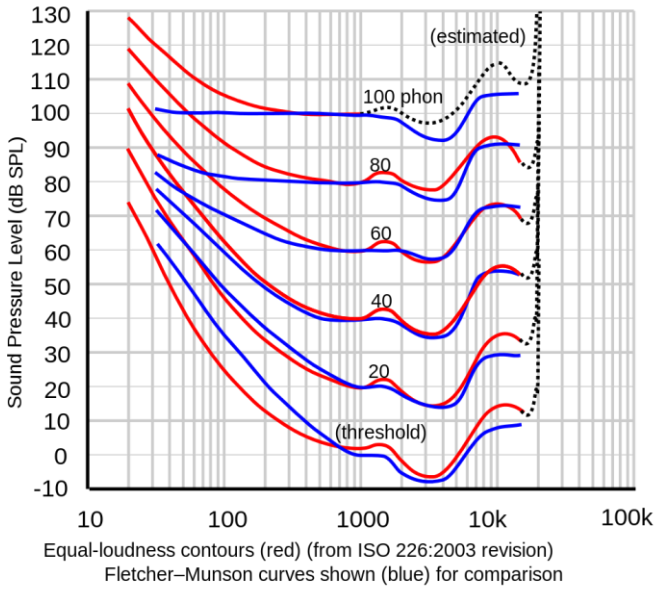
- Curve Isofoniche
- Mascheramento
- Bande Critiche

Curve Isosofoniche

Il diagramma delle curve isofoniche è un grafico della pressione sonora in dB SPL, nel dominio della frequenza, che mostra quanto forte un ascoltatore percepisce un tono puro a diverse frequenze rispetto a una frequenza di riferimento (solitamente 1000 Hz). L'unità di misura del volume percepito è il phon (fono). Per definizione, si dice che due onde sinusoidali di frequenze diverse abbiano un livello di volume uguale misurato in foni, se vengono percepite come ugualmente forti dal giovane senza problemi di udito significativi. In breve, le curve mostrano quanto forte, in dB SPL, deve essere un suono a una frequenza per essere percepito altrettanto forte di un suono a un'altra frequenza.

Le curve Fletcher-Munson sono un esperimento derivato da **Harvey Fletcher e Wilden A. Munson** e riportati in un articolo del 1933 che sono poi state sostituite e incorporate negli standard più

recenti. Le curve definitive sono quelle definite nella norma ISO 226, che si basano su una revisione di determinazioni moderne effettuate in diversi paesi:



Equal-loudness contours (red) (from ISO 226:2003 revision)
Fletcher–Munson curves shown (blue) for comparison

In blu si notano le curve isofoniche effettuate da Fletcher e Munson, mentre quelle rosse sono quelle aggiornate nello standard 226:2003.

Dal grafico si nota che, ad esempio, osservando la curva dei 40 foni, un suono a 40dB 1000Hz, è percepito con la stessa intensità sonora (volume) di un suono a 60dB 100Hz, però sono entrambi toni con intensità sonora di 40 foni (è anche per questo che le curve vengono chiamate “curve di uguale intensità sonora”).

Mascheramento

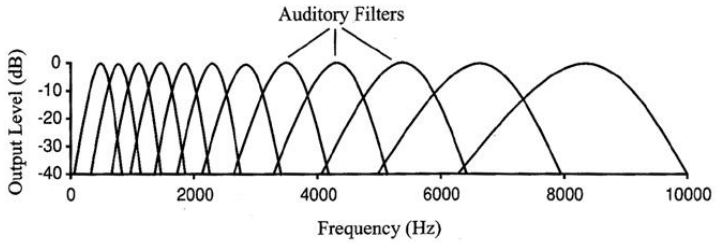
Nell'elaborazione del segnale audio, il mascheramento uditivo si verifica quando la percezione di un suono è influenzata dalla presenza di un altro suono. Il mascheramento uditivo nel dominio della frequenza è noto come mascheramento simultaneo, mascheramento di frequenza o mascheramento spettrale. Il mascheramento uditivo nel dominio del tempo è noto come mascheramento temporale o mascheramento non simultaneo.

Il **mascheramento simultaneo** si verifica quando un suono viene reso non udibile (mascherato) da un rumore o da un suono indesiderato di frequenza prossima. Un suono di frequenza prossima a quella del suono più forte è mascherato più facilmente rispetto a uno di frequenza molto diversa. Per questo motivo, il mascheramento simultaneo è anche chiamato "mascheramento di frequenza". Ad esempio, un picco potente a 1 kHz tenderà a mascherare un tono di livello inferiore a 1,1 kHz.

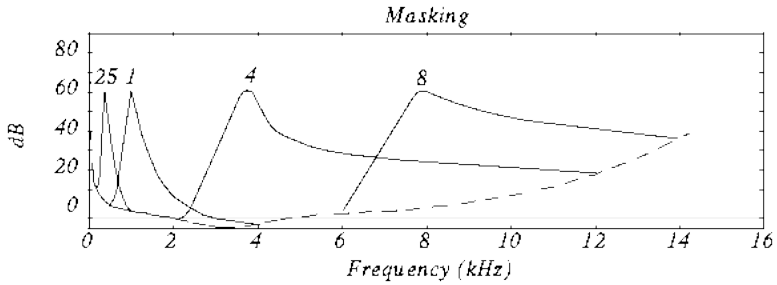
Questo effetto di mascheramento è dovuto al funzionamento della coclea, l'organo uditivo nell'orecchio interno. Un suono complesso è suddiviso in diverse componenti di frequenza e queste componenti provocano un picco nello schema di vibrazione in un punto specifico delle ciglia all'interno della membrana basilare all'interno della coclea. Questi componenti vengono poi codificati in modo indipendente sul nervo uditivo che trasmette le informazioni sonore al cervello. Questa codifica individuale si verifica solo se le componenti di frequenza sono sufficientemente diverse in frequenza, altrimenti si trovano nella stessa banda critica e sono codificate nello stesso punto e vengono percepite come un suono anziché due.

Questa sorta di “filtro uditivo” che effettua la coclea può essere sfruttato per identificare delle specifiche larghezze di banda, chiamate **bande critiche**. Il concetto di **bande critiche**, introdotto da Harvey Fletcher nel 1933 descrive la larghezza di banda di frequenza del "filtro uditivo" creato

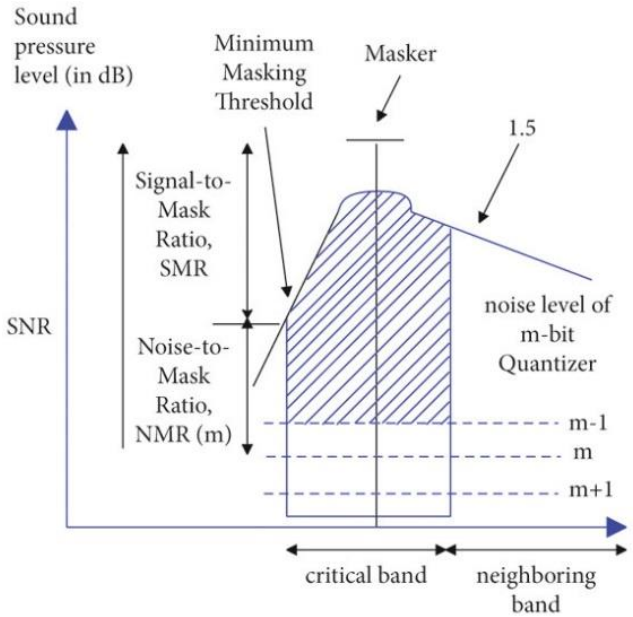
dalla coclea. Approssimativamente, la banda critica è la banda di frequenze all'interno della quale un secondo tono interferirà con la percezione del primo tono mediante mascheramento simultaneo. La bandwith per ogni banda non ha un andamento uniforme, cresce all'aumentare della frequenza centrale, seguendo in un certo senso la tonotopia della coclea:



Sono sottili nella parte bassa dello spettro, per poi diventare sempre più ampie (il nostro sistema uditivo è più preciso alle basse frequenze rispetto alle alte). Il mascheramento avviene in modo più pronunciato dentro queste bande critiche, anziché interbanda.



Mettendo insieme curve isofoniche e bande critiche, ne risulta che solo una parte di spettro sarà udibile ed utilizzabile (quello sopra le soglie di mascheramento). A questo punto, se osseriamo questa figura:



Se si parte da un SNR, che rappresenta una gamma dinamcia di m+1 bit, attuando le tecniche di psicoacustica, avremo che per ogni banda critica non ha senso rappresentare tutti gli m+1 bit, ma si può salire a m, m-2, m-3.. fino ad m-n bit. Questo fino ad incontrare quello che viene chiamato il Signal-To-Mask-Ratio (SMR) che è il segnale effettivamente utile.

Codifica di Huffman

Per codifica di Huffman si intende un algoritmo di codifica dei simboli usato per la compressione di dati senza perdita, basato sul principio di trovare il sistema ottimale per codificare stringhe basato sulla frequenza relativa di ciascun carattere.

La codifica di Huffman usa un metodo specifico per scegliere la rappresentazione di ciascun simbolo: la codifica esprime i caratteri più frequenti nella maniera più breve possibile e quelli meno frequenti nella maniera più lunga. È stato dimostrato che la codifica di Huffman è il più efficiente sistema di compressione di questo tipo: nessun'altra mappatura di simboli in stringhe binarie può produrre un risultato più breve nel caso in cui le frequenze di simboli effettive corrispondono a quelle usate per creare il codice.

Questa è la tecnica che permette di creare un albero binario di simboli:

- Ordina i simboli, in una lista, in base al conteggio delle loro occorrenze.
- Ripeti i seguenti passi finché la lista non contiene un unico simbolo:
    - Prendi dalla lista i due simboli con la frequenza di conteggio minore. Crea un albero di Huffman che ha come "figli" questi due elementi, e crea un nodo "genitore"
    - Assegna la somma del conteggio delle frequenze dei figli al genitore e ponilo nella lista in modo da mantenere l'ordine.
    - Cancella i figli dalla lista.
- Assegna una parola codice a ogni elemento basandosi sul path a partire dalla radice.

La codifica di Huffman è un tipo di **codice prefisso**: è un tipo di sistema di codice caratterizzato dal possesso della "proprietà del prefisso", che richiede che nel sistema non sia presente un'intera parola in codice che sia un prefisso di qualsiasi altra parola in codice nel sistema.

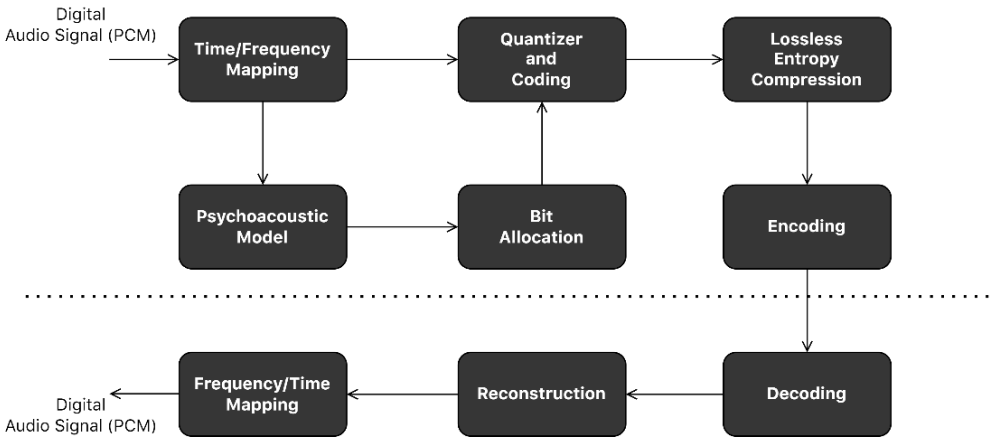
Un esempio di codice NON prefisso:

A: 0, B: 01, C: 011, D: 1

Qui, il codice per "A" è un prefisso del codice per "B" e anche del codice per "C", il che significa che potrebbe esserci ambiguità nella decodifica. Ad esempio, se ricevi la sequenza "011", non è chiaro se rappresenta "B" seguito da "C" o "C" da sola.

Schema generale di compressione percettiva

Visto come funziona la percezione, questo è lo schema generale di un algoritmo di compressione che utilizza la psicoacustica:



In input abbiamo l'audio digitale, quindi una serie di campioni PCM. Si finestra il segnale in campioni e per ogni finestra si effettua una trasformata (DCT o DFT) passando dal dominio del tempo al dominio della frequenza (transform coding). Da una parte si effettua l'analisi psicoacustica

(vengono applicate soglie di mascheramento e soglie di udibilità) grazie ai dati che arrivano dai laboratori biometrici. Questo viene dato in input al modulo di allocazione dei bit: grazie al fatto che il segnale non ha bisogno di tutto l'SNR originale e conoscendo i dati psicoacustici, possiamo ridurre i bit per campione, in quanto non abbiampo bisogno di tutto l'SNR ma solo SMR (questo per ogni banda di frequenza). Per poi finire alla quantizzazione e codifica. Tutto viene compresso attraverso compressione basata sull'entropia (tipo Huffman). Il segnale viene codificato attrvaerso algoritmi di ridondanza per correzione errori e permutazioni e spedito in rete o memorizzato su qualche supporto. Una volta arrivato a destinazione, verrà decodificato, i campioni suddivisi per bande di frequenza vengono ricostruiti e infine viene effettuata un'antitrasformata per passare dal dominio della frequenza a quello del tempo.

Ovviamente il segnale in uscita non sarà uguale a quello iniziale (lossy compression).

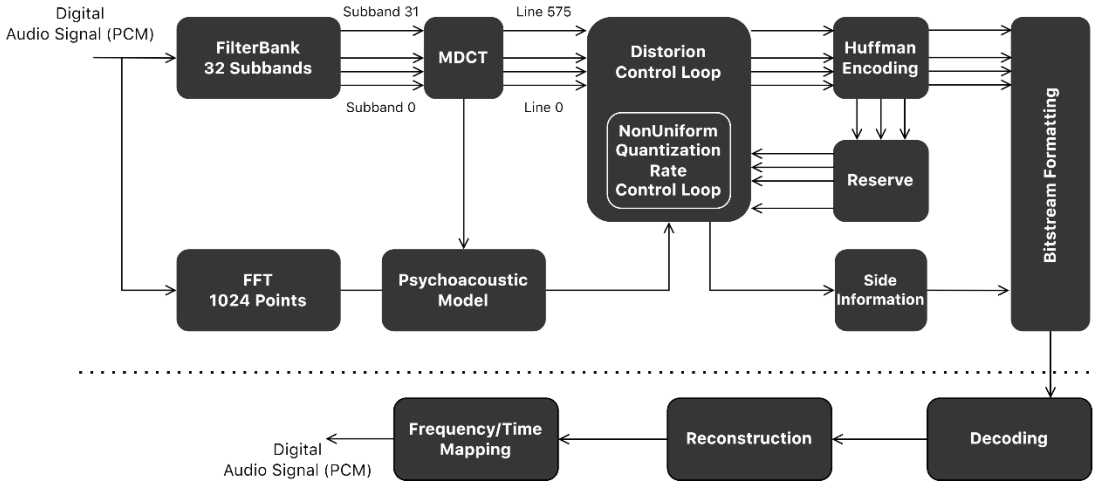
MPEG-1

La parte 3 dello standard MPEG-1 riguarda l'audio ed è definita in ISO/IEC-11172-3.

L'audio MPEG-1 è diviso in 3 livelli. Ogni livello superiore è più complesso dal punto di vista computazionale e generalmente più efficiente con bitrate inferiori rispetto al precedente. Gli strati sono semicompatibili con le versioni precedenti poiché gli strati superiori riutilizzano le tecnologie implementate dagli strati inferiori. Un decoder Layer II "completo" può anche riprodurre l'audio Layer I, ma non l'audio Layer III, sebbene non tutti i lettori di livello superiore siano "completi".

MPEG-1 Audio Layer III

Schema del layer III:



Si ha inizialmente un banco di filtri da 32 sottobande (le bande critiche di fletcher sono 25), successivamente ogni sottobanda viene trasformata nel dominio della frequenza da una trasformata chiamata MDCT che suddivide ognuna delle 32 sottobande in ultreriori 18 sottobande per arrivare ad un totale di 576 sottobande in uscita.

L'analisi psicoacustica invece avviene prima con una trasformata FFT a 1024 punti, che sarà l'input del modello psicoacustico.

Il cuore dell'algoritmo è composto dal distortion control loop e dal non-uniform quantization rate control loop. Lo scopo del **loop di controllo della distorsione** aiuta a trovare un equilibrio tra efficienza di compressione e qualità audio, garantendo che l'audio codificato rimanga il più fedele possibile all'originale. In sostanza funziona confrontando continuamente il segnale audio originale con il segnale quantizzato e codificato, monitorando la distorsione introdotta dalla quantizzazione e dalla codifica e regola i parametri per ridurre al minimo la distorsione udibile. Il circuito di controllo della distorsione tiene conto anche del modello psicoacustico, identificando le parti del

segnale audio che sono meno importanti dal punto di vista percettivo per l'orecchio umano e consente un errore di quantizzazione maggiore in quelle aree allocando meno bit. Al contrario, alloca più bit per preservare la qualità dei componenti audio critici.

Regolando dinamicamente i parametri di quantizzazione in base al modello psicoacustico, il circuito di controllo della distorsione mira a ridurre l'impatto percettivo dell'errore di quantizzazione, con conseguente maggiore efficienza di compressione senza perdita significativa della qualità audio.

Lo scopo **del loop di controllo della quantizzazione non uniforme** invece, è regolare in modo adattivo i parametri di quantizzazione, comprese le dimensioni del passo, alle caratteristiche dell'audio da codificare. Questo adattamento viene effettuato per ottimizzare l'efficienza della compressione mantenendo la qualità audio entro i limiti percettivi dell'orecchio umano.

Utilizzando un circuito di controllo quantizzatore non uniforme, MP3 può allocare più bit per quantizzare componenti audio critici con una risoluzione più fine e meno bit a componenti meno critici con una risoluzione più grossolana. Questo processo aiuta a ridurre il bit rate complessivo riducendo al minimo il degrado percettivo.

Essendo che il loop di controllo della quantizzazione, arriva a decidere il numero di bit da allocare, se per alcune sottobande il controllo sceglie di allocare meno bit (rispetto ad esempio ai 128 scelti), questi rimarranno in eccesso nella **riserva** e successivamente potranno essere usati per altre sottobande, se necessario. È una sorta di storage di gestione dell'economia dei bit disponibili. Un esempio in cui si allocano più bit, è quando il segnale è nella parte di Attack dell'inviluppo: qui sono presenti tante componenti che devono essere rappresentate altrimenti si tolgono parti di segnale.

MPEG-1 Audio Layer II

Il layer 2 aumenta il numero di campioni per sottobanda: 1152 (32 sottobande con 36 campioni ciascuno)

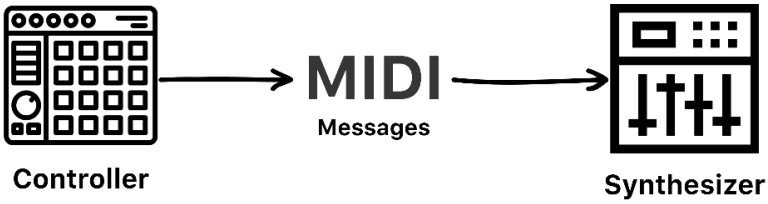
MPEG-1 Audio Layer I

MPEG-1 Layer I è una versione semplificata di MPEG-1 Layer II.

Il layer I utilizza una dimensione di frame di 384 (32 sottobande con 12 campioni ciascuno) campioni con un ritardo molto basso e una risoluzione più precisa. Ciò è vantaggioso per applicazioni come teleconferenze, editing in studio, ecc. Ha una complessità inferiore rispetto al Livello II per facilitare la codifica in tempo reale. Con i sostanziali miglioramenti delle prestazioni degli elaboratori digitali però, il Livello I è diventato rapidamente superfluo e obsoleto.

MIDI

MIDI (Musical Instrument Digital Interface) è uno standard tecnico che descrive un protocollo di comunicazione per collegare dispositivi che producono e controllano il suono, come sintetizzatori, campionatori e computer, in modo che possano comunicare tra loro utilizzando messaggi MIDI

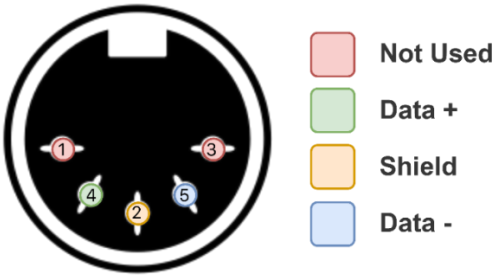


Un'applicazione MIDI comune è suonare una tastiera MIDI o un altro controller e usarlo per attivare un sintetizzatore per generare suoni, che il pubblico sente prodotti da uno speaker. I dati MIDI possono essere trasferiti tramite cavo MIDI o USB o registrati su un sequencer o una workstation audio digitale da modificare o riprodurre.

Il MIDI aveva lo scopo di consentire a qualcuno di controllare più sintetizzatori da una singola tastiera, in modo da generare, ad esempio, i massicci suoni stratificati popolari nella musica pop degli anni '80. In precedenza, tali connessioni tra strumenti non erano standardizzate, quindi le incompatibilità erano comuni. Lo standard MIDI è stato completato nel 1983 da un consorzio di produttori di apparecchiature musicali (tra cui Korg, Oberheim, Roland, Sequential Circuits e Yamaha).

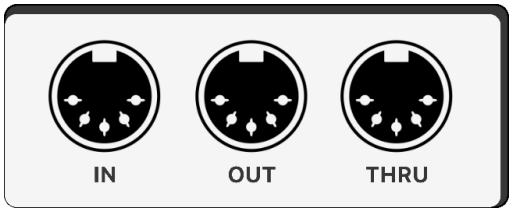
Hardware MIDI

I cavi utilizzati per la comunicazione MIDI terminano in un connettore DIN a cinque pin a 180°. Le applicazioni standard utilizzano solo tre dei cinque pin: un filo di terra (pin 2) e una coppia bilanciata di conduttori (pin 4 e 5) che trasportano un segnale dati a +5 volt. Questa configurazione del connettore può trasportare messaggi solo in una direzione, quindi è necessario un secondo cavo per la comunicazione bidirezionale.



Non esiste alcuna capacità di rilevamento degli errori in MIDI, quindi la lunghezza massima del cavo è impostata a 15 metri per limitare le interferenze.

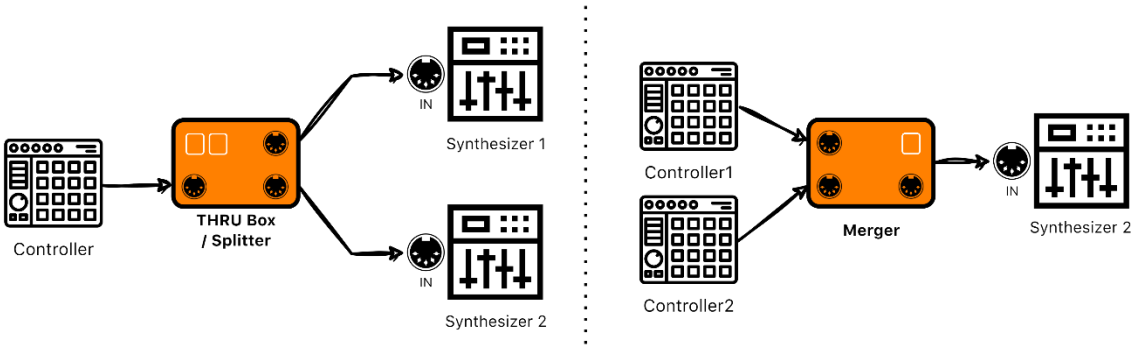
La maggior parte dei dispositivi non copia i messaggi dall'ingresso alla porta di uscita. Un terzo tipo di porta, la porta THRU, emette una copia di tutto ciò che viene ricevuto sulla porta di ingresso, consentendo l'inoltro dei dati a un altro dispositivo MIDI.



Non tutti i dispositivi sono dotati di porte thru e i dispositivi che non sono in grado di generare dati MIDI, come unità di effetti e moduli sonori, potrebbero non includere porte out.

Splitter, Merger & Router

Ogni dispositivo aggiunto al sistema e collegato in serie aggiunge ritardo al sistema stesso. Ciò può essere evitato utilizzando un MIDI thru box, che consente ad un ingresso di essere inoltrato su più porte di uscita contemporaneamente.

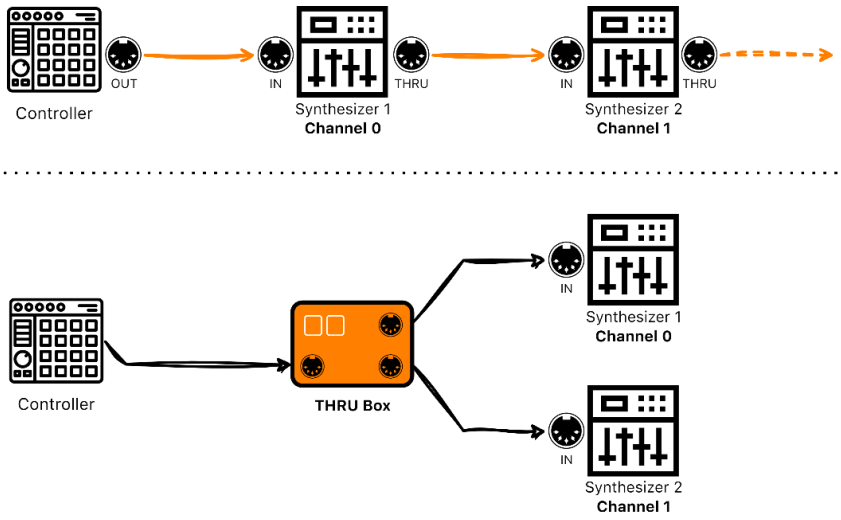


Un **Merger** MIDI è in grado di combinare l'input di più dispositivi in un singolo flusso e consente di collegare più controller a un singolo dispositivo. Uno switch software consente di passare da un dispositivo all'altro ed elimina la necessità di ricollegare fisicamente i cavi.

I router MIDI combinano tutte queste funzioni. Contengono più ingressi e uscite e consentono di indirizzare qualsiasi combinazione di canali di ingresso a qualsiasi combinazione di canali di uscita.

MIDI Channels

Il MIDI permette di inoltrare messaggi su 16 canali diversi. Ogni dispositivo midi sintonizzato su un determinato canale accetterà e decodificherà solo i messaggi destinati al canale per cui si è sintonizzato, un po come le frequenze radio.



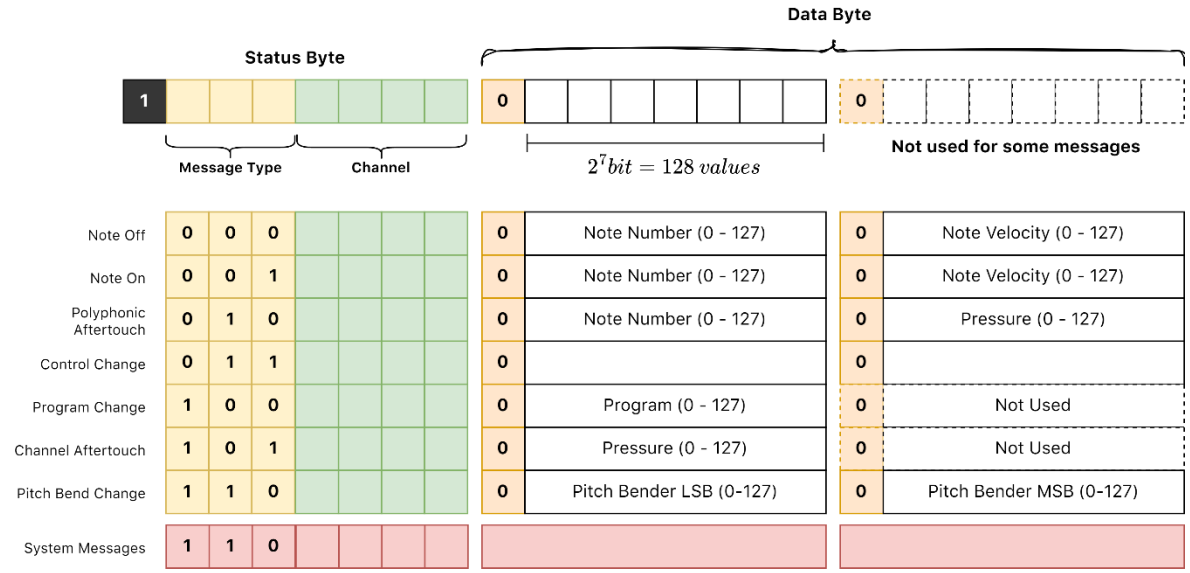
Problemi legati alla temporizzazione

La trasmissione seriale del MIDI porta a problemi di temporizzazione. Un messaggio MIDI di tre byte richiede quasi 1 millisecondo per la trasmissione. Poiché il MIDI è seriale, può inviare solo un evento alla volta. Se un evento viene inviato su due canali contemporaneamente, l'evento sul secondo canale non può essere trasmesso finché il primo non è terminato, quindi viene ritardato di 1 ms. Se un evento viene inviato su tutti i canali contemporaneamente, la trasmissione dell'ultimo canale viene ritardata fino a 16 ms. Ciò ha contribuito all'ascesa delle interfacce MIDI con più porte di ingresso e uscita, poiché il tempo migliora quando gli eventi vengono distribuiti tra più porte rispetto a più canali sulla stessa porta. Il termine slop MIDI si riferisce a errori di temporizzazione udibili che si verificano quando la trasmissione MIDI viene ritardata.

Protocollo MIDI

Il protocollo MIDI comunica (come tutti i protocolli) attraverso messaggi ben definiti. Un messaggio MIDI è semplicemente un'istruzione che controlla alcuni aspetti del dispositivo ricevente. I messaggi MIDI sono costituiti da byte che vengono trasmessi in serie a bitrate massimo di 31,25 kbit/s.

La struttura dei messaggi è la seguente:

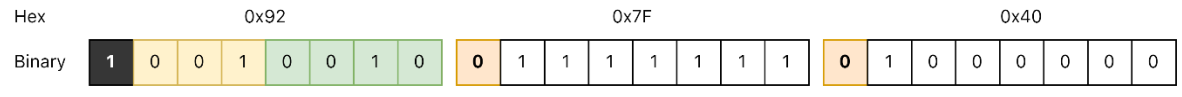


I messaggi si distinguono in status e data. Il primo bit di ogni byte indica se quel byte trasporta un messaggio di stato o di dati. Un bit di start e un bit di stop vengono aggiunti a ciascun byte per scopi di framing, quindi un byte MIDI richiede dieci bit per la trasmissione.

Un messaggio di stato viene utilizzato per indicare un'azione o un evento, come l'inizio di una nota, il cambio di un parametro o il controllo di un parametro. Ad esempio, Il messaggio “Note On” indica l'inizio di una nota (lista completa) e contiene informazioni sulla nota (numero di nota) e sulla velocità (quanto velocemente la nota viene suonata).

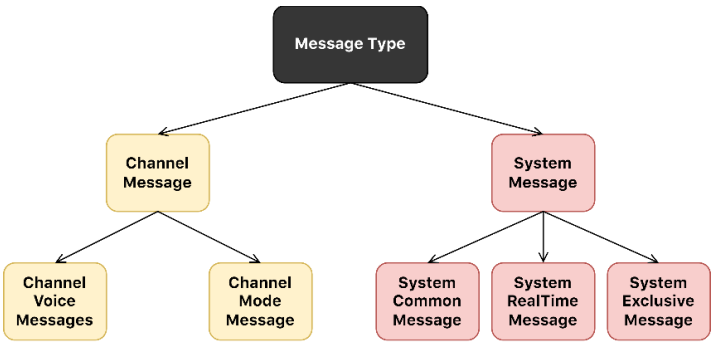
I messaggi dati invece seguono i messaggi di stato e forniscono i dati specifici relativi all'azione o all'evento indicato dal messaggio di stato.

Le note MIDI sono numerate da 0 a 127 assegnate a C_{-1} a G_9. Ciò corrisponde a un intervallo da 8,175799 a 12543,85 Hz. Ad esempio, se si vuole comunicare l’inizio di una nota G9, sul canale 2, con una velocity di 64, verrà inviata la seguente sequenza di byte:



Una cosa importante da capire di tutto ciò è che il MIDI è un protocollo di comunicazione, non c'è alcuna forma d'onda che viene comunicata, nessun segnale audio digitalizzato, solo messaggi di controllo. È un errore affermare che il MIDI trasporta audio o produce audio, trasporta solo messaggi. Poi sarà il sintetizzatore che riceve il messaggio a produrre (sintetizzare) un suono che verrà riprodotto su uno speaker.

I tipi di messaggi possono essere visualizzati attraverso un albero:

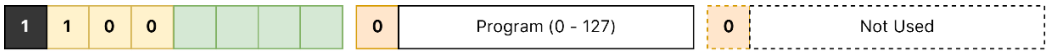


Channel Message

I channel message si dividono in **Voice** e **Mode**.

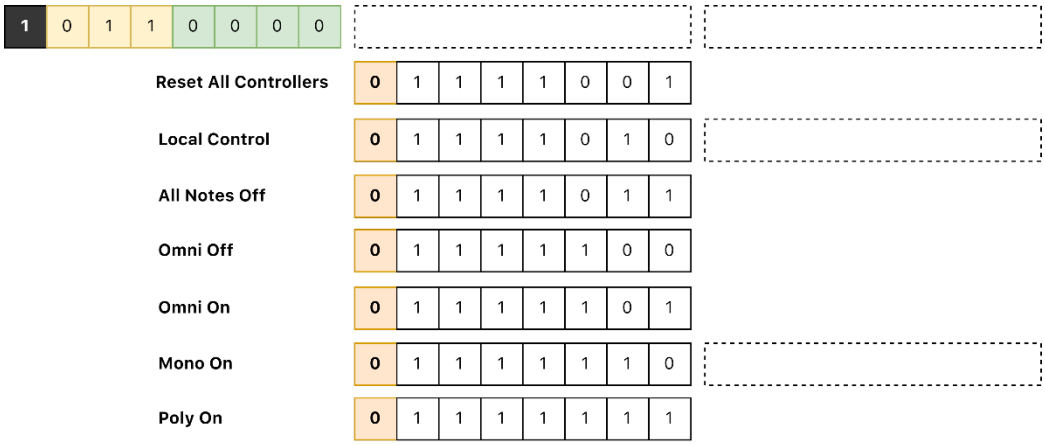
Partendo dal sottoalbero di sinistra abbiamo i **Channel voice message**: vengono utilizzati per inviare informazioni sulla performance musicale. I messaggi in questa categoria sono i messaggi Note On, Note Off, Polyphonic Aftertouch, Channel Aftertouch, Pitch Bend Change, Program Change e Control Change.

Il **program change** ad esempio, si utilizza per impostare la patch su un controller. Per patch si intede lo strumento che si vuole utilizzare (violino, chitarra, ...) e che poi verrà riprodotto dal sintetizzatore, ed è così composto:



in cui il primo data byte è il numero della patch (program 0 - 127). Essendo che ogni casa creava le proprie patch assegnando loro un valore diverso, presi dalla confusione, si è deciso di standardizzare le patch attraverso la General MIDI Patch List.

I **Channel Mode message** invece influenzano il modo in cui un sintetizzatore risponde ai voice message. Questo include solo il messaggio **Control Change** ma con il secondo byte (quello dati) che ha valori da 121 a 127. I mode messages sono i seguenti:



Ad esempio, il messaggio **All Notes Off** fornisce un metodo efficiente per disattivare tutte le note, senza creare traffico di NoteOff consecutivi.

Omni e **Poly** invece stabiliscono rispettivamente il grado di polifonia e di politimbrica. Se ad un controller viene inviato il messagio di Omni On, risponderà a tutti i canali, indipendentemente dal canale in cui era stato impostato.

Mono On invece indica che il controller suona solo una nota alla volta. Quando il controller invece è in modalità **Poly** e viene ricevuta più di una nota sui canali riconosciuti, tali note verranno riprodotte simultaneamente.

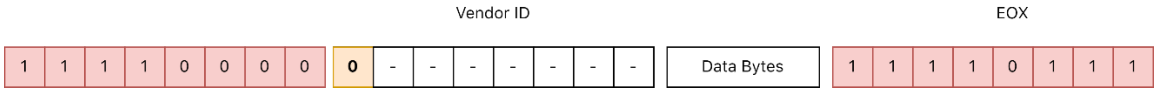
System Message

I System Message si riferiscono invece a tutti i dispositivi, non vengono inoltrati per un singolo canale e si suddividono in **common**, **real time** ed **exclusive**.

Un esempio di **common** è ad esempio il Song Position Pointer, che indica su un sequencer il numero di battute trascorse dall'inizio della canzone e viene utilizzata per iniziare la riproduzione di una sequenza da una posizione diversa dall'inizio della canzone. Normalmente è impostato su 0 per avviare la riproduzione della sequenza dall'inizio. La posizione corrente della song può essere comunicata (da un master) tramite il messaggio Song Position Pointer e può essere modificata in un ricevitore da un messaggio Song Position Pointer in arrivo. Il pointer viene incrementato finché non viene premuto STOP. Se viene premuto CONTINUA, continua ad incrementare dal valore corrente.

Un esempio di **RealTime** sono proprio i messaggi START, STOP, CONTINUE. Oppure il messaggio **Active Sensing**. L'uso dell'active sensingè facoltativo sia per i master che per gli slave e viene inviato ogni 300 ms (max) ogni volta che non vengono trasmessi altri dati MIDI. Se un dispositivo non riceve un messaggio di Active Sensing, dovrebbe funzionare normalmente. Tuttavia, una volta che um ricevitore riconosce un messaggio Active Sensing, si aspetterà di ricevere un messaggio di riscontro ogni 300 millisecondi. Se non vengono ricevuti messaggi entro questo periodo di tempo, il ricevitore presumerà che il cavo MIDI sia stato scollegato per qualche motivo e dovrebbe disattivare tutte le voci e tornare al normale funzionamento.

Invece i messaggi Exclusive non sono assegnati a nessun canale MIDI particolare. Vengono riconosciuti dai ricevitori MIDI indipendentemente dal canale base su cui sono impostati. I messaggi System Exclusive, tuttavia, rappresentano un'eccezione sul formato del messaggio stesso. I messaggi System Exclusive di ciascuno strumento hanno il proprio formato speciale in base al numero ID del produttore assegnato:



Temporizzazione

Normalmente, i musicisti esprimono il tempo come "la quantità di semiminime in ogni minuto (BPM)". Quindi un tempo di 100 BPM significa che un musicista deve essere in grado di suonare 100 semiminime fisse, una dopo l'altra, in un minuto. Questo è l'opposto del modo in cui lo esprime il formato del file MIDI.

Nel MIDI non si utilizza "note da un quarto per minuto" ma si utilizza "tempo di nota da un quarto" (Time per quarter note), quindi la durata di una nota da un quarto. La rappresentazione dei tempi come tempo per quarto invece che come quarti per mintuo consente una sincronizzazione assolutamente esatta a lungo termine con un protocollo di sincronizzazione basato sul tempo come il time code SMPTE. SMPTE è un protocollo basato sul tempo (vale a dire, si basa su secondi, minuti e ore, anziché su un tempo musicale). Pertanto è più semplice correlare il tempo del file MIDI al tempo SMPTE se lo si esprime in microsecondi.

Per indicare il time per quarter, MIDI utilizza un messaggio di sistema, un meta-event message così formattato:

FF 51 03 tt tt tt

dove *tt tt tt* è il tempo di un quarto in millisecondi. Per calcolare quanto dura un quarto avendo il BPM:

$$Time\_per\_quarter = \frac{60}{BPM}$$

Ad esempio, se avessimo 120 BPM:

$$Time\_per\_quarter = \frac{60}{120} = 0,5s = 500ms$$

Midi clock message

Una volta impostato il tempo di un quarto, il MIDI suddivide ulteriormente il quarto in parti per quarto PPQ (ticks). Il valore del tempo in microsecondi indica quanto dovrebbe essere lunga ciascuna delle "note da un quarto". Da qui, si può calcolare quanto dovrebbe essere lungo ciascuno dei clock PPQN dividendo quel valore in microsecondi per il numero di ticks:

$$tick_d = \frac{time\_per\_quarter}{PPQ}$$

Se avessimo 96 ticks per quarto:

$$tick_d = \frac{time\_per\_quarter}{PPQ} = \frac{500ms}{96} = 5,20ms$$

Verrà quindi inviato un MIDI clock message ogni 5,20ms per sincronizzare la riproduzione.

SMTPE

Il MIDI rimane in questo modo relegato solo al mondo musicale. Per collegarsi però al resto del multimedia (video ad esempio) bisogna utilizzare la rappresentazione SMTPE. Un generatore SMTPE che inoltra il segnale timecode SMTPE ad un Midi timecode generator che traduce il tutto in MIDI timecode.

Sequencer

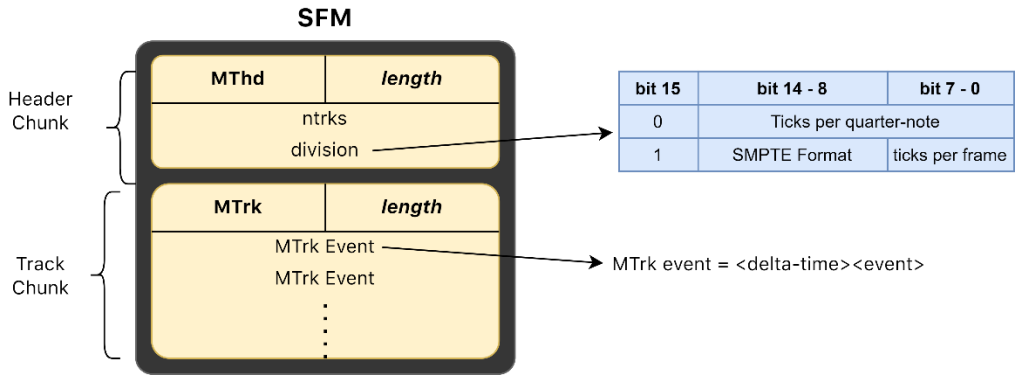
Un sequencer MIDI è un dispositivo o un software che permette di creare, modificare e riprodurre sequenze di eventi MIDI. Un sequencer MIDI consente agli utenti di registrare, organizzare e manipolare dati MIDI, come note, durate, dinamiche, espressioni e altri parametri musicali. Questi dati vengono organizzati in una sequenza temporale denominata Traccia, che è simile a una traccia musicale o a una partitura digitale.

I suoi sono organizzati in banchi (banks) ed ogni suono, tipo il piano o il suono del violino, vengono chiamati patch.

Standard MIDI File

Standard MIDI File è un formato di file utilizzato per la serializzazione di dati MIDI. I file possono contenere dati non MIDI, tra cui il nome del brano, lo strumento musicale, informazioni sul copyright, oltre che il testo della canzone.

Anche lo SMF è costituito da Chunk (Header e Track Chunk).



Il primo chunk è l'header, con ID = MThd. I campi fissi sono il numero di tracce "ntrks" e "division" che identifica in che specifica il significato dei tempi delta per gli eventi delle tracce.

Se il bit 15 di division è zero, i bit da 14 a 0 rappresentano il numero di "tick" delta time che compongono una nota da un quarto. Ad esempio, 24, 48, 72, 96...

Se il bit 15 di division è uno, i tempi delta in un file corrispondono a suddivisioni di secondo, in modo coerente con SMPTE e MIDI Time Code. I bit da 14 a 8 contengono uno dei quattro valori 24, 25, 29 o 30, corrispondenti ai quattro formati standard SMPTE e MIDI Time Code e rappresenta il numero di frame al secondo. Il secondo byte (positivo memorizzato) è la risoluzione all'interno di un frame: i valori tipici possono essere 4 (risoluzione del timecode MIDI), 8, 10, 80 (risoluzione in bit) o 100.

Il secondo chunk è una traccia, con ID = MTrk. Ovviamente ci possono essere più chunk MTrk in caso di song multi traccia. Il chunk MTrk contiene l'insieme di MIDI event che a sua volta è composto da un Delta time e dall'evento stesso. Il delta time Rappresenta la quantità di tempo prima dell'evento successivo. Se il primo evento in una traccia si verifica proprio all'inizio di una traccia, o se due eventi si verificano contemporaneamente, viene utilizzato un tempo delta pari a zero. I tempi delta sono sempre presenti. Il tempo delta è in ticks, come specificato nell'header (division tag).

Ci sono 3 tipi di SMF:

- Type 0: Una sola traccia consentita
- Type 1: Fino a 256 tracce disponibili. Il tempo è univoco per tutti.
- Type 2: Ogni traccia ha un tempo diverso specificato nell'header della traccia.

Qui tutte le specifiche.

Differenza tra MIDI ed AUDIO

Il MIDI essendo solo un insieme sequenziale di messaggi, ovviamente avrà un impatto notevolmente ridotto in termini di spazio rispetto ad un file audio. Ovviamente in un file audio si ha direttamente la forma d'onda, nel MIDI la forma d'onda viene sintetizzata a valle da un controller con sintetizzatore. Anche il costo è molto diverso: un registratore multitraccia a 48 tracce verrebbe a costare migliaia di dollari mentre un sequencer multitraccia MIDI a 48 tracce qualche decina/centinaia di euro. Il costo nel MIDI viene riversato tutto sulla qualità del sintetizzatore.

Nell'audio ovviamente si può registrare tutto ciò che viene catturato da un microfono, e non solo quelli mediati dalla musica.