



# AUDIO PROCESSING

**UNIVERSITA' DEGLI STUDI DI CATANIA**  
**DIPARTIMENTO DI MATEMATICA E INFORMATICA**  
**LAUREA TRIENNALE IN INFORMATICA**  
**A.A. 2021/22**  
**Prof. Filippo L.M. Milotta**

**ID PROGETTO:** 04

**TITOLO PROGETTO:** Chip C64 SID

**AUTORE 1:** Samuele Maria Gallina

**AUTORE 2:** Martin Gibilterra

**AUTORE 3:** Manuel Comis

## Indice

<b>1. Obiettivi del progetto .....</b>	<b>2</b>
LE PREMESSE .....	2
<b>2. Riferimenti Bibliografici .....</b>	<b>8</b>
<b>3. Argomenti Teorici Trattati .....</b>	<b>9</b>

## 1. Obiettivi del progetto

Il progetto si pone l'obiettivo di contestualizzare, storicamente e tecnologicamente, l'hardware del Commodore 64, descrivendo in modo particolarmente dettagliato le caratteristiche, il funzionamento e le potenzialità del suo chip audio dedicato: il MOS SID.

Il Commodore 64 è tutt'ora il computer più venduto di sempre, avendo piazzato sul mercato dal 1982 al 1994 circa 20 milioni di unità, distinguendosi per la sua facilità d'uso, il basso prezzo e le sue capacità grafiche e sonore, all'avanguardia per il periodo in cui venne creato.

Il Commodore 64 ha anche il merito di essere stato il primo home computer ad essere dotato di un chip dedicato alla sintesi sonora fornito di generatori di inviluppo ADSR, il MOS SID, che riesce a sintetizzare 3 voci indipendenti contemporaneamente.



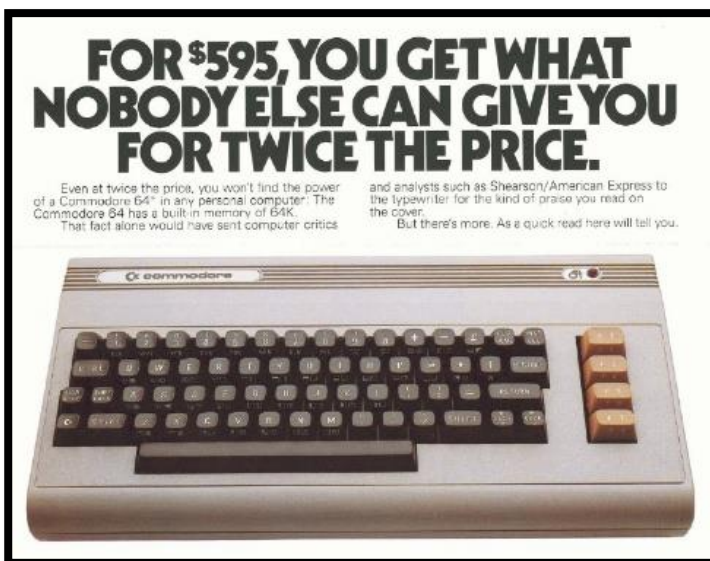
*L'originale Commodore 64, chiamato in Italia "il biscottone" per il suo form factor e il suo colore marroncino*

In secondo luogo, verrà fornita una dimostrazione pratica su come il MOS SID veniva programmato, in linguaggio BASIC, per la riproduzione di suoni più o meno complessi, usando l'emulatore VICE.

In conclusione, viene presentato un excursus temporale riguardante l'evoluzione dei chip e delle periferiche dedicati all'audio, dal post-rilascio del Commodore 64 fino alla fine degli anni '90 con l'esordio delle schede audio dedicate.

### Le premesse

Il Commodore 64 nacque con l'obiettivo di creare nuovi standard tecnologici riguardanti la grafica e l'audio per imporsi sul settore videoludico e degli home computer, con l'intenzione di rendere le tecnologie Commodore appetibili alle aziende produttrici di console e videogiochi da sala a tal punto da usarle nei loro prodotti.



*Réclame statunitense del Commodore 64 (1982)*

Esso fu pensato dapprima come un prodotto d'élite non rivolto al mercato consumer; tuttavia, verso la fine del 1982 Commodore decise di puntare sul mercato degli home computer, nel quale avrebbe incontrato diversi competitors come lo ZX Spectrum.

Nonostante ciò, Jack Tramiel, il CEO di Commodore, si rese conto che, grazie alla recente produzione di chip audio e video (rispettivamente SID e VIC-20) di ultima generazione, la sua azienda avrebbe potuto inserirsi in una fetta di mercato ancora inesplorata: le famiglie, che desideravano sempre più un computer a basso prezzo che potesse intrattenere anche a livello ludico.

### L'origine del Commodore 64

A quel punto, il boss di Commodore commissionò ai suoi dipendenti il design di un nuovo computer da esibire al CES di Chicago nel 1981. Nel giro di due giorni, gli ingegneri progettaronò quello che poi sarebbe diventato il Commodore 64, per poi presentarlo alla fiera. Il prodotto ebbe molto successo, anche a causa della mancanza di competitors all'evento stesso.

Nel 1982 iniziò la produzione in massa e l'interesse crebbe esponenzialmente tra i consumatori soprattutto grazie al fatto che Commodore distribuiva la propria libreria software nei più grandi negozi al dettaglio degli Stati Uniti, il che garantiva una visibilità senza precedenti per i loro prodotti.



*Jack Tramiel e la linea di prodotti Commodore, al CES di Chicago (1981)*

### Il contesto storico e tecnologico

Il Commodore 64 si posiziona nel pieno di un lungo periodo di alfabetizzazione informatica che nelle case statunitensi ed europee sdoganò lentamente l'uso del computer per le attività giornaliere di natura gestionale e ludica, in quanto fino ad allora i competitors avevano offerto solo soluzioni ad altissimo prezzo e destinate esclusivamente a scopi lavorativi o educativi, come ad esempio i primi PC IBM dei primi anni '80 oppure il britannico BBC Micro.

Per diminuire i costi di produzione e quindi il prezzo di vendita, il Commodore 64 venne proposto come una macchina "base" fornita di porte d'espansione che permettevano di collegare i propri dispositivi esterni. Commodore 64 abbatté gran parte dei limiti tecnici del suo tempo proponendo un hardware all'avanguardia delle dimensioni di una tastiera. Esso era caratterizzato da una grande capacità di elaborazione (principalmente RAM) per l'epoca rispetto ai competitors, il supporto a memorie di massa più portabili come le Datassette ma soprattutto dall'introduzione di un'uscita video antenna RF, che permetteva al Commodore 64 di essere una vera e propria macchina da gioco, determinandone il successo.

### Carrellata veloce dell'hardware del Commodore 64

Il Commodore 64 possiede delle componenti che per l'epoca erano all'avanguardia tra cui:

- Il microprocessore MOS Technology 6510 a 1 MHz;
- una DRAM di 64kB;
- 1 chip dedicato alla grafica (il VIC-II);
- 1 chip dedicato all'audio (il SID).

Il sistema operativo adottato dal Commodore 64 è molto semplice ed è costituito principalmente da 3 parti ovvero il KERNAL (il kernel creato per gli home computer a 8 bit dalla società Commodore), il monitor in linguaggio macchina e un interprete BASIC.

Le ultime 2 componenti permettono intuitivamente di programmare le azioni da far svolgere alla macchina.

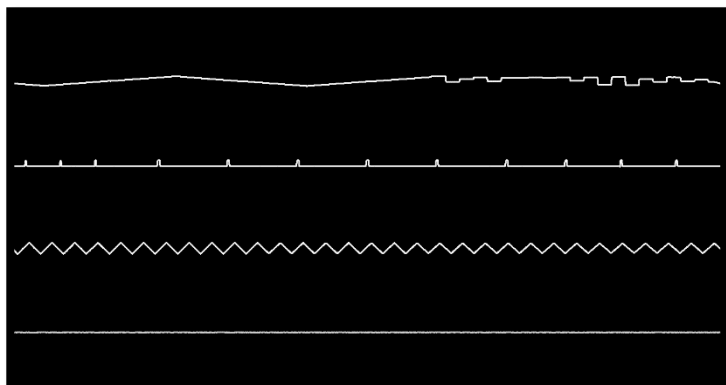
### Il MOS SID: una panoramica dell'hardware

Il chip SID è uno dei principali motivi di successo del Commodore 64. Progettato dal capo ingegnere Robert Yannes, che reputava i chip sonori dell'epoca come "primitivi e ovviamente progettati da persone che non sapevano niente sulla musica", è un circuito integrato in cui troviamo sia componenti digitali che analogiche; esso è in grado di sintetizzare 3 voci separate sfruttando 4 forme d'onda notevoli: quadra, dente di sega, triangolare e rumore pseudo-casuale.

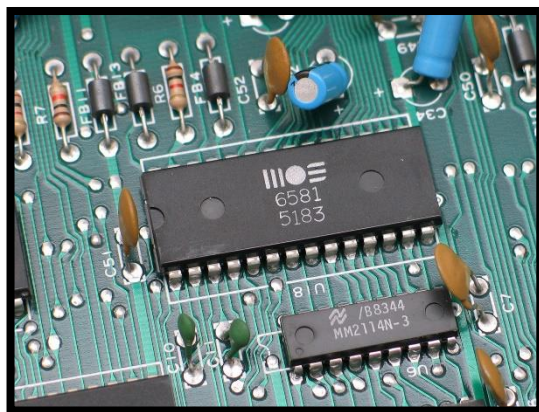
Ovviamente il SID è anche in grado di combinare queste 4 onde per crearne di più particolari e complesse. Oltre a questo, per sintetizzare i suoni il SID si occupa anche di applicare un inviluppo ADSR, modulare 2 voci tra loro e applicare dei filtri analogici con frequenza di taglio decidibile dal programmatore. I filtri analogici sono realizzati fisicamente grazie a dei condensatori esterni al chip.

Ci furono diverse versioni del chip SID, si ricordano il 6581 e l'8580. Con l'8580 si introdussero diverse migliorie, tra cui la possibilità di eseguire un AND logico tra onde arbitrarie, operazione che il 6581 permette solamente tra onda quadra e triangolare. L'8580 inoltre ha migliorie anche architetturali che permettono di dissipare meno calore e produrre dei suoni più chiari rispetto al 6581.

L'originale 6581 aveva un difetto: ogni volta che il volume di uno dei canali cambiava valore, il chip produceva un brevissimo pop anomalo. Quest'anomalia fu sfruttata molto dai musicisti che componevano musica sul Commodore 64 per simulare strumenti a percussione (e quindi creare virtualmente un quarto canale audio digitale con una risoluzione di 4 bit) o per sintetizzare la voce umana. In particolare, veniva modificato il



*Vista all'oscilloscopio dei 3 canali audio (più il quarto canale "virtuale") durante la riproduzione di un brano (Tetris – Wally Beben, 1988)*



valore del registro del master volume in modo repentino, consentendo al programmatore/musicista di introdurre dei sample digitali primitivi in PCM a 4 bit. Tuttavia, questa pratica era CPU-intensive e le capacità dell'hardware limitate costringevano a fare dei compromessi in termini di potenza di calcolo.

Nonostante le migliorie apportate dall'8580, il 6581 fu preferito da diversi utenti, tra cui i musicisti, in quanto il suono fortemente distorto prodotto permetteva di

simulare meglio gli strumenti musicali più "grezzi" come la chitarra elettrica. In generale, il volume generato dal 6581 era più alto di quello dei successori, a fronte di numerosi problemi elettronici che abbassavano inevitabilmente la vita media del chip, creando un mercato consistente di SID danneggiati.

### Caratteristiche tecniche

Il chip è dotato di 3 canali audio indipendenti che possono riprodurre un suono che spazia in un range di 8 ottave e un range di frequenze da 16 a 4000 Hz e che possono essere modulati in ampiezza fino a 48 dB. Come detto prima, i 3 canali sono in grado di riprodurre 4 onde elementari, generate dagli oscillatori audio che possono essere sincronizzati tra loro nel tempo, più le loro combinazioni. In particolare, il rumore pseudo-casuale è generato grazie a uno shift register a 23 bit, il Fibonacci LFSR.

Il chip supporta anche la modulazione ad anello, utile per generare effetti audio "metallici". La modulazione ad anello si serve di un circuito chiamato "modulatore ad anello" che prende in ingresso due onde e restituisce in due uscite diverse la somma e la differenza tra di esse. Questa tecnologia si evolverà in seguito nella sintesi FM.

Per ogni oscillatore vi sono dei controlli di volume delle componenti dell'inviluppo ADSR, grandissima novità per l'epoca. Vi è la possibilità di applicare tre filtri sonori ovvero il passa-basso, il passa-alto e il passa-banda.

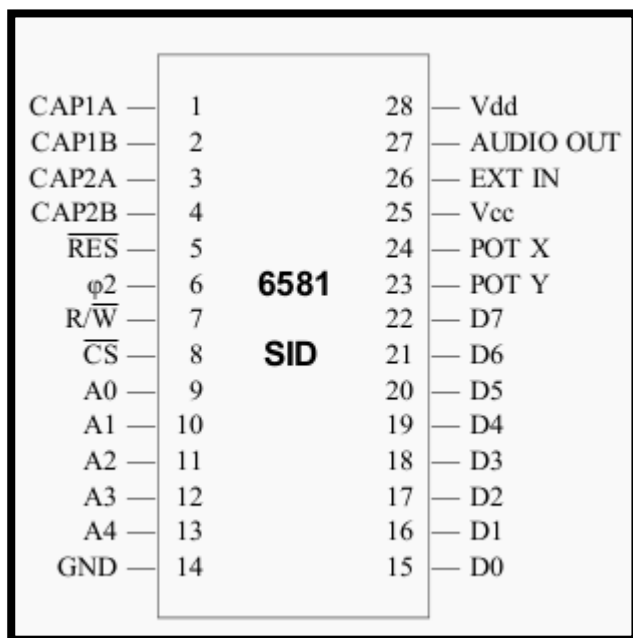
Il chip vanta la presenza di 2 convertitori ADC (analogico-digitale) a 8 bit, utili per collegare dei controller da gioco, nonché un input audio esterno.

Come funziona il SID?

Per riprodurre i suoni il Commodore 64 riserva una parte di RAM al SID, ovvero la regione che va da \$d400 (54272) fino a \$d41c (54300). Ogni cella della RAM riservata al SID ha un compito in particolare.

Ad esempio, le celle di memoria di indirizzo \$d400 e \$d401 sono usate per immagazzinare l'informazione sulla frequenza del suono del canale 1 (\$d400 è il byte alto e \$d401 il byte basso perché soli 8 bit sarebbero troppo pochi per rappresentare la frequenza). Un altro esempio lo rappresentano gli indirizzi \$d405 e \$d406. I bit 7...4 del \$d405 rappresentano la durata dell'attack mentre i bit da 3 a 0 rappresentano la durata del decay. Allo stesso modo da 7 a 4 in \$d406 vi è il livello di sustain e da 3 a 0 la durata della release. La tabella completa della configurazione dei registri di controllo è consultabile [qui](#).

Da questi esempi si nota che la gestione della memoria era ottimizzata; infatti, sfruttava la possibilità di immagazzinare 4 numeri in 2 celle di memoria facendo un'opportuna suddivisione dei bit.



Il chip audio ha bisogno di comunicare con le altre componenti della macchina e fisicamente questo succede tramite i pinout schematizzati nell'immagine (qui si fa riferimento alla versione 6581).

Per fare qualche esempio si può parlare del pin EXT IN che intuitivamente è il pin per connettere al chip un dispositivo audio esterno.

Il pin AUDIO OUT invece serve per indirizzare l'audio verso l'uscita (dispositivi come le casse audio).

Approfondimenti disponibili [qui](#).

Una simulazione di generazione suoni tramite il SID

Per programmare il SID in modo tale da permettergli di generare le corrette frequenze, bisogna fare delle accortezze preliminari.

Il SID può rappresentare un insieme limitato e discreto di frequenze fisiche, la cui mappatura non segue un rapporto 1:1. Infatti, bisogna usare un fattore di normalizzazione pari a 16,94, in quanto il SID rappresenta la frequenza con un numero a 16 bit e il range 16-4000 Hz deve essere rimappato su un intervallo di valori da 0 a 65535.

Indicando la frequenza fisica in Hz con  $f$  e il corrispondente digitale con  $f_{SID}$ ,

$$f_{SID} = 16,94 * f$$

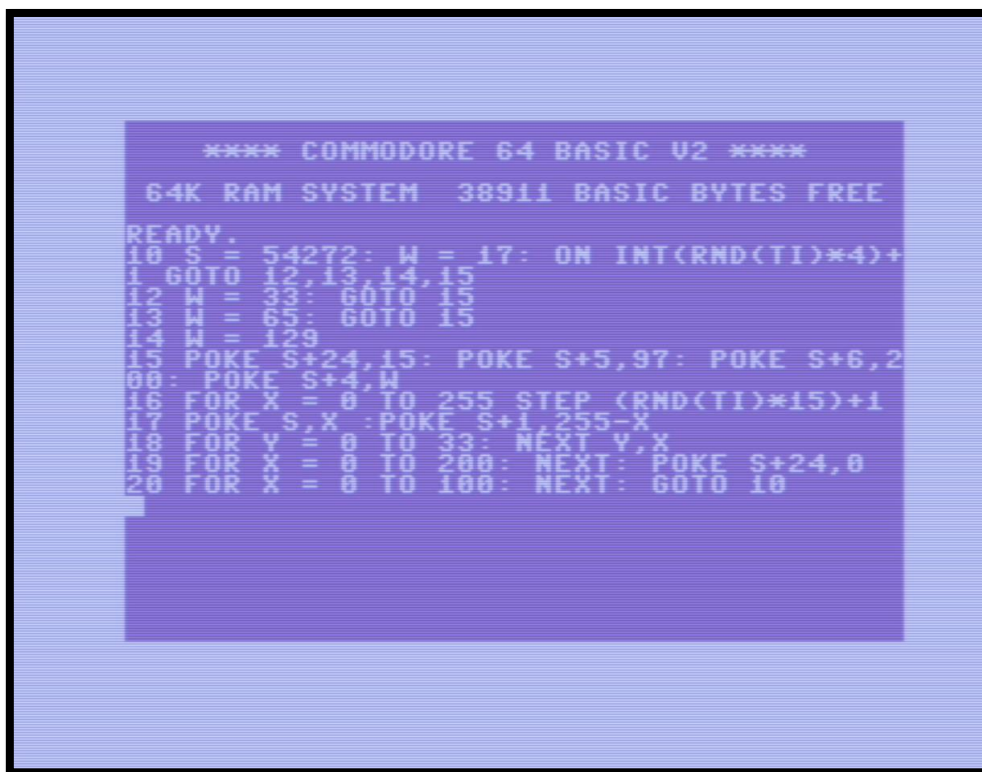
Ad esempio, il La fondamentale (440 Hz, La della quarta ottava), nel SID viene rappresentato con il valore 7.454.



Questa normalizzazione consente di regolare la frequenza di una nota, nonostante la rappresentazione in numeri interi, con una precisione di 1/17 di Hertz anziché 1/12, come prevede la scala tonale fisica, permettendo quindi un tuning dell'audio più preciso rispetto ad altri chip dell'epoca.

Per simulare la generazione di un suono con il chip SID, si fornisce di seguito uno script in linguaggio BASIC. Per semplicità, lo script usa solamente la prima delle tre voci del SID (registri 54272 – 54278).

Lo script viene eseguito su un emulatore del KERNAL di Commodore 64 chiamato VICE.

A screenshot of a Commodore 64 BASIC V2 interpreter window. The window has a light blue background. At the top, it displays '\*\*\*\* COMMODORE 64 BASIC V2 \*\*\*\*' and '64K RAM SYSTEM 38911 BASIC BYTES FREE'. Below this, it says 'READY.'. The script starts with line 10: 'S = 54272: W = 17: ON INT(RND(TI)\*4)+'. Line 11 is 'GOTO 12,13,14,15'. Line 12 is 'W = 33: GOTO 15'. Line 13 is 'W = 65: GOTO 15'. Line 14 is 'W = 129'. Line 15 is 'POKE S+24,15: POKE S+5,97: POKE S+6,2'. Line 16 is 'POKE S+4,W'. Line 17 is 'FOR X = 0 TO 255 STEP (RND(TI)\*15)+1'. Line 18 is 'POKE S,X: POKE S+1,255-X'. Line 19 is 'FOR Y = 0 TO 33: NEXT Y,X'. Line 20 is 'FOR X = 0 TO 200: NEXT: POKE S+24,0'. Line 21 is 'FOR X = 0 TO 100: NEXT: GOTO 10'. The script ends with line 21.

La riga 10 assegna alla variabile S il valore 54272, cioè il primo registro di controllo del SID. Ciò viene fatto per semplificare la gestione dei 29 registri, che verranno indirizzati per spiazzamento.

Viene assegnato il valore 17 a W, che sarà la variabile che contiene le informazioni su quale tipo di onda generare. In questo caso, il valore 17 è uguale alla codeword binaria 00010001, dove il bit 4 se posto a 1 fa generare all'oscillatore un'onda triangolare e il bit 0 è il bit di gate, che se posto a 1 inizia l'involuppo del suono e quando posto a 0 lo termina (la gestione del bit di gate si può paragonare alla pressione e rilascio di un tasto nel pianoforte).

I valori W = 33, 65 e 129 corrispondono in binario alle codeword che attivano sul canale designato rispettivamente l'onda dente di sega, l'onda quadra e il rumore pseudocasuale.

In base al timestamp di esecuzione del Commodore 64 (cioè il valore in sessantesimi di secondo da quando il sistema è stato avviato), identificato da **ti**, viene generato un numero decimale tra 0 e 1 casuale grazie alla funzione **rnd**, che viene trasformato in intero con la funzione **int**. A quel punto, in base al numero ottenuto, verrà eseguita una tra le 4 linee poste dopo il goto: in questo caso, se il numero ottenuto è 1, verrà eseguita la linea 12.

Dalla linea 15 viene usata l'istruzione **POKE**: non è altro che una semplice assegnazione del valore dopo la virgola nella cella di memoria che ha indirizzo uguale al numero a sinistra della virgola. Nella linea 15 viene assegnato al valore massimo il controllo del master volume (registro 54296, valore 0000 1111), viene posto un attack lungo e un decay basso ponendo il valore 97 (registro 54277, valore 0110 0001), e vengono

configurati un alto livello di sustain e un'alta durata di release (registro 54278, valore 1100 1000). Infine, viene assegnato il valore **w** della waveform al registro 54276.

Il codice soprascritto cicla all'infinito scalando linearmente la frequenza dell'onda generata casualmente, dal valore più alto al valore più basso, creando una scala musicale al contrario. Il suono udito sarà diverso la maggior parte delle volte grazie alla pseudocasualità fornita dalla funzione rnd(ti) basata sul timestamp del Commodore 64.

#### I successori del chip SID: sintesi FM, samples, schede audio

I chip audio successivi al SID iniziavano ad allontanarsi sempre più dal concetto di generatore sonoro programmabile (quale era il SID). Dunque, si iniziò ad usare la sintesi FM (Frequency Modulation), che permette di generare timbri complessi modulando la frequenza di un suono con un altro. Le onde sonore vengono modulate consentendo quindi di ottenere timbri e sonorità finora impossibili da replicare come suoni distorti, strumenti a percussione, corde pizzicate e altro.

Ma come funziona praticamente la sintesi FM? Si ipotizzi di avere un'onda dall'andamento sinusoidale che quindi produce un suono ben preciso; come già accennato, si potrebbe modulare quest'onda con un'altra onda, ad esempio un'altra onda sinusoidale ma dalla frequenza decisamente più bassa. Potenzialmente non ci sono limiti a questa operazione in quanto ognuna di queste onde può essere modulata anche più volte da più onde diverse per creare sonorità sempre diverse. Ognuna di queste onde di modulazione viene chiamata operatore, mentre la sequenza con cui sono applicati gli operatori viene chiamata algoritmo.

La sintesi FM è stata alla base di alcune delle più vecchie generazioni dei sintetizzatori Yamaha, delle prime schede audio per PC, AdLib e Sound Blaster, e di molti dei chip audio dei cellulari per le suonerie polifoniche. Questa tecnologia ebbe un'ampia diffusione alla fine degli anni '80 quando la maggior parte delle schede arcade era dotata di un chip audio con 8 canali FM. L'FM era però inadatto alla riproduzione acustica dei suoni naturali, caratterizzati da sviluppi complessi che cambiano continuamente e in modo irregolare, incompatibile quindi coi modelli più semplici della sintesi FM.

Uno dei più importanti esponenti del tempo nei meriti della sintesi FM fu la SoundBlaster, che faceva uso del sintetizzatore FM Yamaha YM3812 già utilizzato dalle schede AdLib. Tra l'altro un'altra peculiarità di questa scheda audio era anche la sua capacità di decompressione ADPCM che le consentiva di riprodurre audio in formato digitale mono a 8 bit con una frequenza fino a 23 kHz e registrare con una frequenza di campionamento fino a 12 kHz. Il successo fu tale da soppiantare nel mercato le schede AdLib e prendersi lo scettro della scheda audio più diffusa dell'epoca per PC.

Uno dei chip successivi al SID fu il Paula, utilizzato dai Commodore Amiga, che implementò una grande novità dal punto di vista prestazionale in quanto permetteva di elaborare il suono in output senza bisogno della CPU, basandosi per la prima volta sui samples. Nell'Amiga, la CPU comunica al Paula dove nella RAM è conservato il sample, per quanto riprodurlo e tutte le informazioni necessarie alla riproduzione. La scarsa disponibilità di spazio di memorizzazione costringeva gli sviluppatori ad utilizzare un formato dal buon rapporto qualità/spazio occupato.

Il formato in questione è noto come modulo musicale (MOD Amiga) realizzato tramite trackers, software specifici per editare musica a partire da samples monofonici. L'editor del tracker contiene diverse colonne, ognuna corrispondente a un canale. Per ogni colonna vengono create delle "pagine" da un certo numero di righe ognuna delle quali contiene informazioni di riproduzione in un certo istante di tempo come la nota riprodotta, il sample usato e effetti vari. Dunque, il file MOD contiene una lista di istruzioni di riproduzione nonché i vari samples audio utilizzati per il brano.

## 2. Riferimenti Bibliografici

[1] [Articolo italiano sulla storia del Commodore 64 – Andrea Longhi](#)

Questo articolo in italiano descrive in maniera semplice e concisa la storia di Commodore e del Commodore 64.

[2] [MOS SID – Wikipedia](#)

Questa pagina di Wikipedia contiene tutte le informazioni storiche e tecnologiche relative al SID: le sue origini, una descrizione dell'hardware e la sua scheda tecnica.

[3] [Specifiche tecniche del SID – C64-wiki](#)

Questa pagina contiene la descrizione dettagliata di tutti i registri di controllo utilizzati dal SID, nonché dettagli tecnici aggiuntivi sull'hardware.

[4] [How Oldschool sound/Music worked – The 8-Bit Guy \(2015\)](#)

Questo video riassume perfettamente come veniva generato il suono nei computer e console tra gli anni 80 e gli anni 90, citando anche il chip Yamaha YM3812 fondamentale per l'evoluzione delle schede audio per PC.

[5] [Commodore 64 Story & Review – Nostalgia Nerd \(2015\)](#)

Da questo video sono state tratte tutte le informazioni storiche su Jack Tramiel e l'azienda Commodore, sull'origine del Commodore 64 e sul contesto storico-tecnologico in cui è stato creato.

[6] [Schede audio – Wikipedia](#)

Questa pagina spiega cos'è una scheda audio, fornendo cenni storici e citando le differenze principali con le tecnologie precedenti a essa.

[7] [Cenni di storia della chip music – Alessio “AlextheLioNet” Bianchi \(2008\)](#)

Questa serie di pagine fornisce un excursus storico dettagliato sul funzionamento dei chip audio dall'epoca dei generatori sonori programmabili, passando per la sintesi FM fino al Paula e la musica tramite samples.

[8] [FM SYNTHS in under 4 minutes – ANDREW HUANG \(2018\)](#)

Questo video spiega in meno di 4 minuti come funziona la sintesi FM nella pratica: si parla di modulazione di onde, operatori portanti e modulanti, algoritmi.

[9] [Sound Blaster – Wikipedia](#)

La pagina di Wikipedia della Sound Blaster descrive nel dettaglio i vari modelli di questa scheda audio, di cui si parla nell'ultimo paragrafo degli obiettivi del progetto.

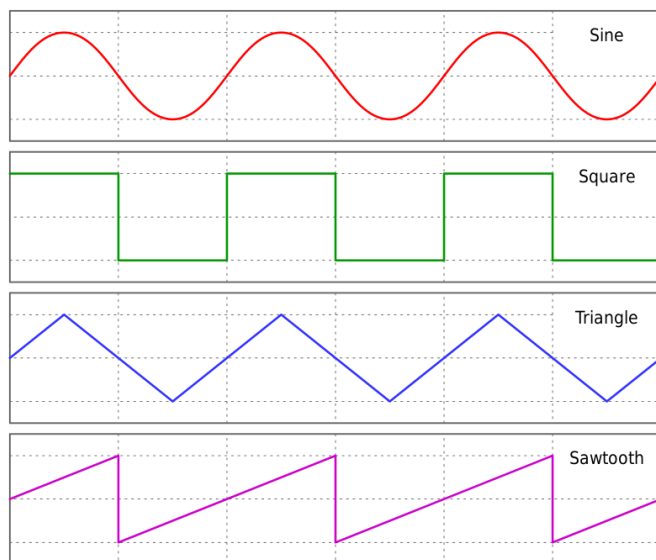


### 3. Argomenti Teorici Trattati

Si analizzano di seguito alcuni argomenti teorici, su cui non si è approfondito nella sezione degli obiettivi, che sono stati studiati e sfruttati per la progettazione del chip SID.

#### Forme d'onda non sinusoidali

Le forme d'onda riproducibili dal chip SID sono l'onda quadra, a dente di sega, triangolare e rumore pseudo-casuale. In più ovviamente sono riproducibili le loro possibili combinazioni.



L'onda quadra è un segnale molto importante in teoria dei segnali e in elettronica. Essa è composta dall'alternanza di 2 soli valori ovvero 0 e 1. L'onda quadra perfetta è solo un modello matematico in quanto è impossibile per un componente elettronico generarla. Il motivo è legato alla trasformata o serie di Fourier che riesce a trovare i termini elementari di un'onda. Il valore di ogni termine varia sulla base di 3 coefficienti principali presenti nella formula della serie di Fourier.

Il processo inverso alla serie è la cosiddetta sintesi di Fourier che somma i termini elementari per creare l'onda. Però per ottenere esattamente l'onda quadra servono infiniti

termini (in particolare servono solo i termini dispari in quanto quelli pari per la formula di Fourier risulteranno essere uguali a 0).

Per un componente elettronico è ovviamente impossibile sommare infiniti termini: questo è esattamente il motivo per cui l'onda quadra che genera il SID è solo un'approssimazione.

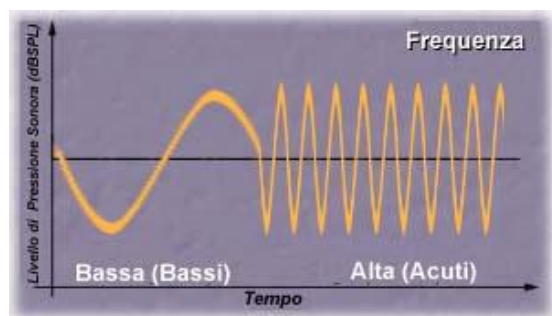
Lo stesso identico discorso si riflette sull'onda triangolare, anch'essa ha bisogno di infiniti termini (solo i dispari) per essere sintetizzata.

Per l'onda dente di sega invece i termini presenti sono sia i pari che i dispari.

Si conclude dunque affermando che le forme d'onda non sinusoidali generate dal chip SID sono solo delle approssimazioni in quanto servirebbero infiniti valori da sommare per generarle.

#### Ottave e frequenza dei suoni

Il chip in ognuno dei suoi canali può riprodurre un suono avente una frequenza da 16 a 4000 Hz in un range di 8 ottave.



La frequenza è semplicemente ciò che determina l'altezza di un suono, ovvero quanto è grave o acuto.

Essa è facilmente calcolabile, nel caso di periodicità del segnale, come inverso del periodo.  $f = 1/T$

Nel caso di toni puri si vede dallo spettro delle frequenze, ottenuto grazie alla trasformata o serie di Fourier, che la frequenza è solo 1. Nel caso di toni complessi periodici il numero di frequenze sarà maggiore ad 1 ma sarà comunque un numero discreto, cosa che non succede nel

caso di toni non periodici in quanto lo spettro calcolato è continuo.

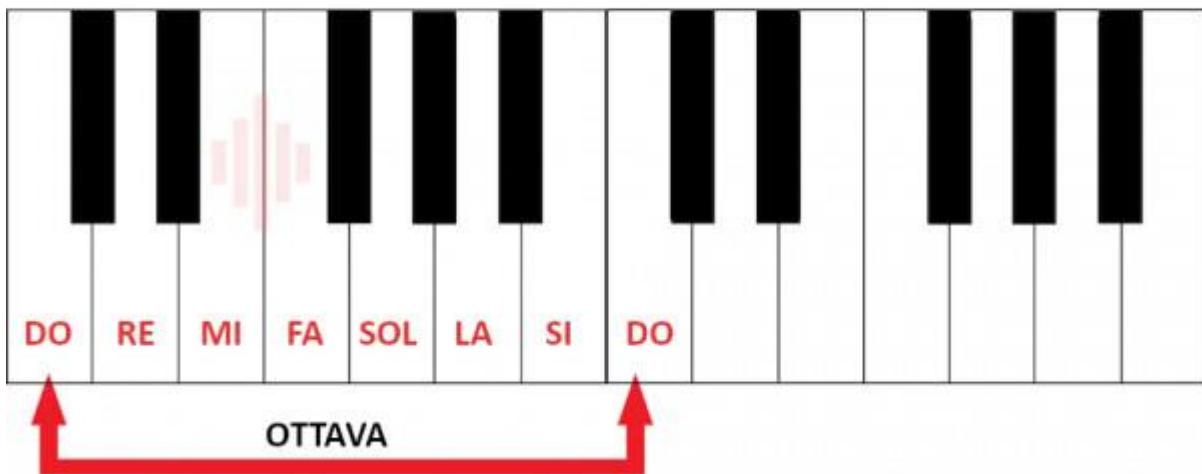
Qualunque sia lo spettro (continuo o discreto) la frequenza predominante, ovvero con ampiezza maggiore, è detta fondamentale.

La frequenza fondamentale nello spettro di un suono emesso determina la nota musicale percepita. Detto in maniera più semplice: un suono è caratterizzato da diverse frequenze, la fondamentale ne determina la nota mentre le altre frequenze ne determinano il timbro.

Si può ora definire l'ottava; essa altro non è che l'intervallo tra note uguali in cui la successiva ha frequenza doppia.

Esempio: se la prima ottava inizia con Do a frequenza pari a 16,35 Hz segue che la seconda ottava inizia pure con la nota Do ma con frequenza pari a 32,70 Hz.

Ogni ottava compete di 12 semitoni, un semitono consiste nell'aumento di  $2^{(1/12)}$  Hz tra note adiacenti. Ad esempio se Do ha frequenza pari a 16,35 Hz la nota successiva Do# avrà frequenza uguale a 17,32 Hz.

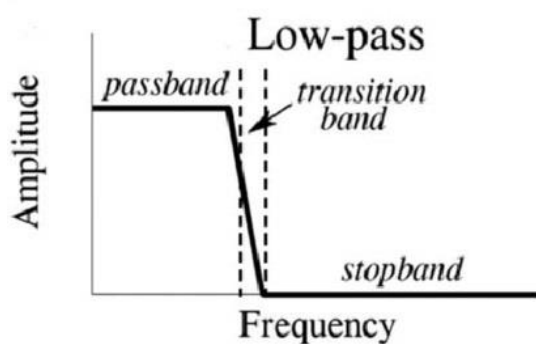


## Filtri di banda

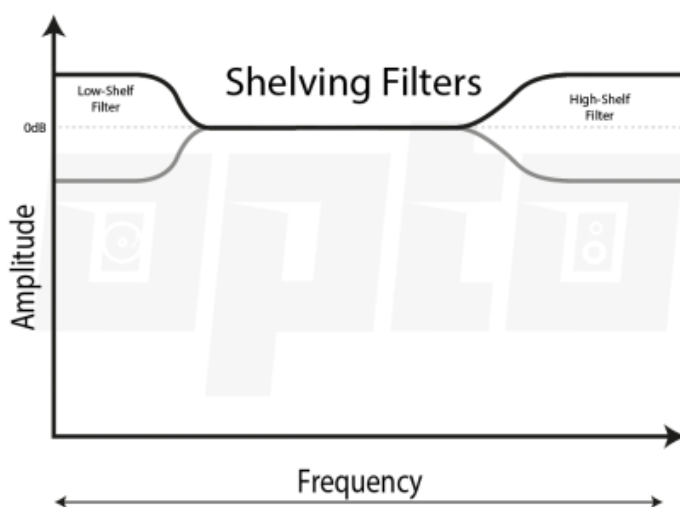
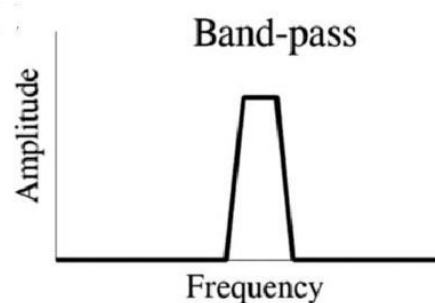
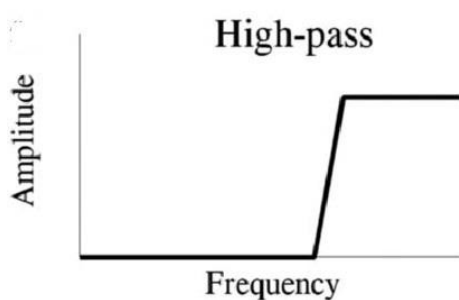
Il chip SID offre la possibilità di applicare tre filtri sonori ovvero il passa-basso, il passa-alto e il passa-banda.

Come suggeriscono i loro nomi, questi filtri servono per filtrare frequenze di vario tipo. Ad esempio, il passa-basso filtrerà le frequenze di basso valore (con basso si intende al di sotto di una soglia stabilita).

I classici filtri HPF, LPF e BPF (rispettivamente: High Pass Filter, Low Pass Filter e Band Pass Filter) annullano le frequenze che non sono all'interno dell'intervallo desiderato. In genere però il filtro oltre alla soglia ha un ulteriore parametro detto pendenza o rolloff che permette di azzerare l'ampiezza delle frequenze vicino alla frequenza di cutoff in modo graduale.



Come si vede dall'immagine, l'annullamento delle frequenze non è immediato ma graduale e segue una certa pendenza. Questa tecnica evita un annullamento brusco (troppo percepibile a livello uditivo).



Si possono poi generalizzare questi filtri aggiungendo un terzo parametro, il livello di dB a cui impostare l'ampiezza delle frequenze nell'intervallo desiderato, questi filtri sono detti Shelving filters.

Questi filtri possono o attenuare o incrementare l'ampiezza delle frequenze desiderate.

In pratica i filtri precedenti sono come gli Shelving filters con il livello di dB uguale a 0.

## Inviluppo ADSR

Per sintetizzare i suoni, il SID si occupa anche di applicare un inviluppo ADSR per gestire il volume del suono nel tempo.

Si ricorda la definizione di inviluppo: esso è l'andamento dell'ampiezza di un suono nell'intervallo che va dall'istante di generazione fino all'istante in cui si estingue. Un inviluppo particolare, che è usato dal SID, è l'ADSR.

L'ADSR è l'unione di quattro intervalli successivi tra loro che seguono quest'ordine:

- **Attack:** fase in cui l'ampiezza passa in modo rapido da valore 0 a un valore massimo
- **Decay:** fase in cui l'ampiezza dal suo valore massimo raggiunge un valore costante
- **Sustain:** fase in cui l'ampiezza rimane circa costante
- **Release:** fase in cui l'ampiezza dal valore costante passa a valore 0 (si estingue)

Nota: attack, decay e release sono intesi come intervalli temporali mentre per il sustain si parla invece di sustain level ovvero un valore di ampiezza. Il sustain potrebbe avere un intervallo decidibile in un generatore di suoni o in uno strumento musicale. Ad esempio, con un pianoforte, tenendo premuto a lungo un tasto, attack, decay e release mantengono la stessa durata mentre l'intervallo del sustain varierà, ma non il sustain level che non dipende da quanto è grande l'intervallo.

