



Universidade de Brasília – UnB  
Faculdade UnB Gama – FGA  
Engenharia Eletrônica

# **Implementação de Equipamento de Mixagem para DJs com Controle Automatizado de Filtros e Efeitos**

Autor: Joselito Prado Marques da Silva  
Orientador: Dr. Diogo Caetano Garcia

Brasília, DF  
2025





Joselito Prado Marques da Silva

## **Implementação de Equipamento de Mixagem para DJs com Controle Automatizado de Filtros e Efeitos**

Monografia submetida ao curso de graduação em Engenharia Eletrônica da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia Eletrônica.

Universidade de Brasília – UnB

Faculdade UnB Gama – FGA

Orientador: Dr. Diogo Caetano Garcia

Brasília, DF

2025

---

Joselito Prado Marques da Silva

Implementação de Equipamento de Mixagem para DJs com Controle Automatizado de Filtros e Efeitos/ Joselito Prado Marques da Silva. – Brasília, DF, 2025-  
91 p. : il. (algumas color.) ; 30 cm.

Orientador: Dr. Diogo Caetano Garcia

Trabalho de Conclusão de Curso – Universidade de Brasília – UnB  
Faculdade UnB Gama – FGA , 2025.

1. processamento de sinais. 2. mixer. I. Dr. Diogo Caetano Garcia. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Implementação de Equipamento de Mixagem para DJs com Controle Automatizado de Filtros e Efeitos

CDU 02:141:005.6

---

Joselito Prado Marques da Silva

## **Implementação de Equipamento de Mixagem para DJs com Controle Automatizado de Filtros e Efeitos**

Monografia submetida ao curso de graduação em Engenharia Eletrônica da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia Eletrônica.

Trabalho aprovado. Brasília, DF, 01 de junho de 2024 – Data da aprovação do trabalho:

---

**Dr. Diogo Caetano Garcia**  
Orientador

---

**Dr. Henrique Gomes de Moura**  
Convidado 1

---

**Dr. Wellington Avelino do Amaral**  
Convidado 2

Brasília, DF  
2025



**A dedicatória é opcional. Caso não deseje uma, deixar todo este arquivo em branco.**

*Este trabalho é dedicado às crianças adultas que,  
quando pequenas, sonharam em se tornar cientistas.*



# Agradecimentos

A inclusão desta seção de agradecimentos é opcional, portanto, sua inclusão fica a critério do(s) autor(es), que caso deseje(em) fazê-lo deverá(ão) utilizar este espaço, seguindo a formatação de *espaço simples e fonte padrão do texto (sem negritos, aspas ou itálico)*.

**Caso não deseje utilizar os agradecimentos, deixar toda este arquivo em branco.**



**A epígrafe é opcional. Caso não deseje uma, deixe todo este arquivo em branco.**

*“Não vos amoldeis às estruturas deste mundo,  
mas transformai-vos pela renovação da mente,  
a fim de distinguir qual é a vontade de Deus:  
o que é bom, o que Lhe é agradável, o que é perfeito.  
(Bíblia Sagrada, Romanos 12, 2)*



# Resumo

Um sinal de áudio possui características únicas de elementos que estão presentes em determinadas faixas de frequência. E a soma desses elementos compõe uma música. Assim, é necessário que haja um controle acerca das bandas de frequência que compõem um sinal. O *mixer* é o equipamento responsável pelo controle de ganhos e atenuações dessas bandas. Suas aplicações iniciaram na telefonia, passaram por mesas de produção musical, estações de rádio até que chegaram à mesa de som de um *DJ*. O *mixer* focado na atividade de um *DJ* se consolidou a partir de um único modelo baseado no controle de ganho de três bandas. Assim, o desenvolvimento de um *mixer* que controla dois canais utilizando apenas um controle é realizado. A transição entre duas músicas é otimizada pela utilização de dois filtros passa-altas, que são deslocados conforme um botão central é utilizado. O equipamento é capaz de realizar a leitura de sinais analógicos advindos de reprodutores de música padronizados pela indústria, como *CDJs*. Incrementos de opções de efeitos como *reverb* e *delay* são feitos, bem como o controle de seus parâmetros de implementação é controlável através de um botão *slider* horizontal. Esse sistema é implementado utilizando uma *Raspberry Pi* para leitura, escrita e processamento de sinais, bem como a lógica implementada em C visando à otimização da latência. Provas de conceito foram realizadas através do *PureData* tanto para a filtragem de sinais e funcionamento dos filtros passa-altas em função do posicionamento do botão, que dita a posição das frequências de corte dos filtros, além do controle dos parâmetros dos efeitos como ganho dos sinais amostrados após 1s para o efeito *reverb* e atraso das amostras em ms para o *delay*.

**Palavras-chave:** processamento de sinal, áudio, mixer.



# Abstract

*An audio signal has unique characteristics of elements that are present in specific frequency ranges. And the sum of these elements makes up a piece of music. Thus, it is necessary to have control over the frequency bands that make up a signal. The mixer is the equipment responsible for controlling the gains and attenuations of these bands. Its applications began in telephony, went through musical production desks, radio stations, and eventually reached the DJ's sound desk. The mixer focused on a DJ's activity was consolidated from a single model based on the control of three-band gain. Thus, the development of a mixer that controls two channels using only one control is carried out. The transition between two songs is optimized by using two high-pass filters, which are shifted as a central button is used. The equipment is capable of reading analog signals from music players standardized by the industry, such as CDJs. Incremental options for effects such as reverb and delay are included, as well as the control of their implementation parameters is adjustable through a horizontal slider button. This system is implemented using a Raspberry Pi for reading, writing, and processing signals, as well as the logic implemented in C aiming at optimizing latency. Proofs of concept were carried out using PureData both for signal filtering and the functioning of the high-pass filters based on the button's positioning, which dictates the cutoff frequencies of the filters, in addition to controlling the parameters of the effects such as signal gain sampled after 1s for the reverb effect and sample delay in ms for the delay.*

**Key-words:** *signal processing, audio, mixer.*



# Lista de ilustrações

Figura 1 – sinal de tempo contínuo (OPPENHEIM; WILLSKY, 2010) . . . . .	30
Figura 2 – sinal de tempo discreto (OPPENHEIM; WILLSKY, 2010) . . . . .	30
Figura 3 – espectro do sinal amostrado com $W_s > 2W_m$ (OPPENHEIM; WILLSKY, 2010) . . . . .	31
Figura 4 – quantização de amplitudes em tempo discreto (OPPENHEIM; WILLSKY, 2010) . . . . .	32
Figura 5 – ciclo de propagação do som (SAAD, 2019) . . . . .	35
Figura 6 – Rosnie - inventado por Alex Rosner (UBBS, 2015) . . . . .	38
Figura 7 – CMA-10-2DL - primeiro mixer estéreo comercial (MARQUES, 2019) .	38
Figura 8 – toca-disco SL-1200 (32BITMASCHINE, 2008) . . . . .	39
Figura 9 – mixer criado pela Rane (RANE, 2008) . . . . .	39
Figura 10 – CDJ-300 - primeira CDJ da Pioneer (PIONEER, 2024a) . . . . .	40
Figura 11 – mixagem virtual com o <i>Apple Vision Proo</i> (CARDOSO, 2024) . . . . .	40
Figura 12 – configuração mais encontrada (PIONEER, 2024b) . . . . .	41
Figura 13 – <i>rotary mixer</i> (MIXERS, 2024) . . . . .	41
Figura 14 – Espectro de áudio (GLEESON, 2024) . . . . .	43
Figura 15 – transmissão com fio desbalanceado (NEIMAR, 2023) . . . . .	45
Figura 16 – esquemático de fio balanceado (NEIMAR, 2023) . . . . .	45
Figura 17 – plug de 1/4" (ELECTRONICS, 2024) . . . . .	46
Figura 18 – plug de 1/8" (ELECTRONICS, 2024) . . . . .	46
Figura 19 – plug de 2,5 mm (ELECTRONICS, 2024) . . . . .	46
Figura 20 – cabo RCA (RS, 2024) . . . . .	47
Figura 21 – Conector XLR . . . . .	47
Figura 22 – fluxograma geral do <i>mixer</i> . . . . .	53
Figura 23 – bloco de leitura de sinais . . . . .	54
Figura 24 – bloco de filtragem . . . . .	55
Figura 25 – bloco de efeitos . . . . .	56
Figura 26 – bloco de conversão digital-analógico . . . . .	56
Figura 27 – Silent Drum de Jaime Oliver . . . . .	57
Figura 28 – expressão para a $f_{c2}$ . . . . .	59
Figura 29 – lógica de funcionamento do botão central no <i>PureData</i> . . . . .	60
Figura 30 – botão de seleção de efeito no <i>PureData</i> . . . . .	60
Figura 31 – botão de quantidade de efeito no <i>PureData</i> . . . . .	60
Figura 32 – lógica de seleção de efeito no <i>PureData</i> . . . . .	61
Figura 33 – variação do volume dos efeitos no <i>PureData</i> . . . . .	61
Figura 34 – implementação da variação do volume dos efeitos no <i>PureData</i> . . . . .	62

Figura 35 – soma dos sinais filtrados e dos efeitos no <i>PureData</i> . . . . .	62
Figura 36 – botão <i>knob</i> para frequência central e efeitos (ROBOCORE, 2024) . . . . .	63
Figura 37 – botão para seleção do efeito (EVEA, 2024) . . . . .	63
Figura 38 – PF8591 - conversor analógico-digital de 8 bits (SARAVATI, 2024) . . . . .	64
Figura 39 – <i>Raspberry Pi</i> para processamento dos sinais (MOGNON, 2016) . . . . .	64
Figura 40 – Diagrama de Conexões . . . . .	65
Figura 41 – música 1 no domínio do tempo sem filtragem . . . . .	76
Figura 42 – música 1 no domínio da frequência sem filtragem . . . . .	76
Figura 43 – música 1 no domínio do tempo com frequência de corte de 20 Hz . . . . .	77
Figura 44 – música 1 no domínio da frequência com frequência de corte de 20 Hz . . . . .	77
Figura 45 – música 1 no domínio do tempo com uma frequência de corte de 300 Hz . . . . .	77
Figura 46 – música 1 no domínio da frequência com uma frequência de corte de 300 Hz . . . . .	78
Figura 47 – música 1 no domínio do tempo com uma frequência de corte de 4 kHz . . . . .	78
Figura 48 – música 1 no domínio da frequência com uma frequência de corte de 4 kHz . . . . .	78
Figura 49 – música 1 no domínio do tempo com uma frequência de corte de 22 kHz . . . . .	79
Figura 50 – música 1 no domínio da frequência com uma frequência de corte de 22 kHz . . . . .	79
Figura 51 – canal de efeito <i>reverb</i> isolado com música 1 a um volume nulo . . . . .	80
Figura 52 – canal de efeito <i>reverb</i> isolado com música 1 a um volume de 0.3312 . . . . .	80
Figura 53 – canal de efeito <i>reverb</i> isolado com música 1 a um volume de 0.6625 . . . . .	81
Figura 54 – canal de efeito <i>reverb</i> isolado com música 1 a um volume de 1.0 . . . . .	81
Figura 55 – <i>reverb</i> com 0 dB após 1 segundo . . . . .	82
Figura 56 – <i>reverb</i> com 25 dB após 1 segundo . . . . .	82
Figura 57 – <i>reverb</i> com 50 dB após 1 segundo . . . . .	83
Figura 58 – <i>reverb</i> com 100 dB após 1 segundo . . . . .	83
Figura 59 – <i>delay</i> de 0 ms . . . . .	84
Figura 60 – <i>delay</i> de 250 ms . . . . .	84
Figura 61 – <i>delay</i> de 500 ms . . . . .	84
Figura 62 – <i>delay</i> de 1000 ms . . . . .	85

# Lista de tabelas



# **Lista de abreviaturas e siglas**

AD	Analógico para Digital
BPM	Batidas por Minuto
CD	Compact Disc
CDJ	Compact Disc Jockey
CD-R	Compact Disc-Recordable
DA	Digital para Analógico
dBu	Decibéis em relação a 1 Volt
DJ	Disc Jockey ou Discoterário
DSP	Digital Signal Processor (Processador de Sinal Digital)
DVD	Digital Versatile Disc (Disco Digital Versátil)
FLAC	Free Lossless Audio Codec
FC	Frequênciade Corte
FX	Efeitos
HPF	High-Pass Filter (Filtro Passa-Alta)
I2C	Inter-Integrated Circuit
MIDI	Musical Instrument Digital Interface
MP3	MPEG-1 Audio Layer 3 (Camada 3 de Áudio MPEG-1)
MPEG-2	Moving Picture Experts Group - Layer 2 (Grupo de Especialistas em Imagens em Movimento - Camada 2)
PCM	Pulse Code Modulation (Modulação por Código de Pulso)
RCA	Conektor Radio Corporation of America
RMS	Root Mean Square (Valor Quadrático Médio)
SPI	Serial Peripheral Interface

STFT	Short Time Fourier Transform (Transformada de Fourier de Tempo Curto)
TV	Televisão
UART	Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
XLR	Conecotor XLR, usado em equipamentos de áudio profissional

# Listas de símbolos

$dB$	Decibel, unidade de medida para a intensidade do som
$\Delta t$	Intervalo de tempo
$\int$	Integral
$\infty$	Infinito
$\pi$	Pi, constante matemática aproximadamente igual a 3.14159
$\Sigma$	Somatório
$d\omega$	Diferencial angular na Transformada de Fourier
$e$	Base do logaritmo natural
$f$	Frequência (em Hz)
$f_{c1}$	Frequência de corte do canal 1
$f_{c2}$	Frequência de corte do canal 2
$f_c$	Frequência de corte central
$j$	Unidade imaginária
$k$	Multiplicador de 1000
$n$	Índice de amostragem ou sequência discreta
$\omega$	Frequência angular
$\tau$	Tempo de atraso
$T$	Período (em segundos)
$W_m$	Largura de banda máxima do sinal
$W_s$	Frequência de amostragem



# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>25</b>
<b>1.1</b>	<b>Justificativa</b>	<b>25</b>
<b>1.2</b>	<b>Objetivos</b>	<b>25</b>
1.2.1	Objetivo Geral	26
1.2.2	Objetivos Específicos	26
<b>1.3</b>	<b>Estrutura do Documento</b>	<b>26</b>
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA E ESTADO DA ARTE</b>	<b>29</b>
<b>2.1</b>	<b>Sinal: Conceito</b>	<b>29</b>
<b>2.2</b>	<b>Sinal Contínuo</b>	<b>29</b>
2.2.1	Sinal Discreto	30
2.2.2	Transformada de Fourier	30
2.2.3	Teorema da Amostragem	31
2.2.4	<i>Aliasing</i>	32
2.2.5	Teorema da Quantização	32
<b>2.3</b>	<b>Filtros</b>	<b>33</b>
<b>2.4</b>	<b>Som e Música</b>	<b>35</b>
2.4.1	Conceito físico	35
2.4.2	Audição Humana	36
2.4.3	História da Gravação e Reprodução de Som	36
2.4.4	Disc Jockeys <i>DJs</i>	36
2.4.5	Parâmetros físicos	42
2.4.5.1	Equipamentos	44
<b>2.5</b>	<b>Mixer</b>	<b>44</b>
2.5.1	Potência de Sinal de Áudio	44
2.5.2	Cabeamento	45
2.5.3	Conectores de 1/4", 1/8"(3,5 mm) e 2,5 mm	45
2.5.4	RCA	47
2.5.5	XLR	47
2.5.6	Protocolos Digitais	48
2.5.7	Amplificação de Potência	48
2.5.8	Equalização	48
2.5.9	<i>Trim</i> e Volume	48
2.5.10	Conversão AD/DA	49
<b>3</b>	<b>METODOLOGIA</b>	<b>51</b>

<b>3.1</b>	<b>Proposta Geral</b>	<b>51</b>
<b>3.2</b>	<b>Levantamento de Requisitos</b>	<b>52</b>
3.2.1	Requisitos Funcionais	52
3.2.2	Requisitos Não Funcionais	52
<b>3.3</b>	<b>Fluxograma do Mixer</b>	<b>53</b>
3.3.1	Bloco de Leitura de Sinais	54
3.3.2	Bloco de Filtragem	54
3.3.3	Bloco de Efeitos	55
3.3.4	Bloco de Conversão DA	56
<b>3.4</b>	<b>Prova de Conceito</b>	<b>57</b>
3.4.1	<i>PureData</i>	57
3.4.2	Implementação de Filtragem	58
3.4.3	Implementação de Efeitos	59
<b>3.5</b>	<b>Proposta de Implementação</b>	<b>62</b>
3.5.1	Implementação em Hardware	62
3.5.2	Implementação em Software	66
3.5.3	Protótipo de Interface de Usuário	73
<b>4</b>	<b>RESULTADOS PRELIMINARES</b>	<b>75</b>
<b>4.1</b>	<b>Resultados de Filtragem</b>	<b>75</b>
<b>4.2</b>	<b>Resultados de Efeitos</b>	<b>80</b>
4.2.1	Automação de Efeitos	80
4.2.2	<i>Reverb</i>	81
4.2.3	<i>Delay</i>	83
<b>5</b>	<b>CONCLUSÃO</b>	<b>87</b>
	<b>REFERÊNCIAS</b>	<b>89</b>

# 1 Introdução

Sons são formados por elementos que estão presentes em faixas de frequências. Muitas vezes, deseja-se o controle desses elementos. Assim, a manipulação dessas bandas é realizada através de filtros com ganhos e atenuações. O equipamento responsável por esse manuseio é o *mixer*. Ao se analisar a história desse equipamento, percebe-se que a forma de mixagem parte do controle de bandas de frequência. Inicialmente, cortes bruscos entre canais eram realizados e reproduções eram feitas utilizando toca-discos com um microfone captando esse sinal.

Ao longo da segunda metade do século XX, o progresso da eletrônica permitiu a sofisticação de cada processo envolvido no ato de mixagem. Inicialmente, utilizou-se vinis e hoje se pode utilizar um banco de músicas hospedado na nuvem enquanto se utiliza óculos de realidade aumentada para realizar a mixagem.

O processo de mixagem evoluiu ao longo do tempo, porém a ideia de utilizar recortes de músicas para criar uma ambientação única permanece e permeia toda a história, e norteia o futuro do trabalho realizado por um *DJ*.

## 1.1 Justificativa

Ao se analisar o desenvolvimento dos equipamentos utilizados para mixar músicas, percebe-se uma constante quanto aos parâmetros utilizados para realizar transições entre músicas, que reside na alteração do ganho de bandas de frequências, usualmente três: baixas, médias e altas frequências.

Além disso, o uso combinado de filtragem e *crossfader* é amplamente empregado na prática de mixagem. Essa combinação pode simplificar o trabalho do *DJ*, proporcionando uma transição mais suave e eficiente entre as faixas.

A proposta de utilizar um botão central para controlar dois canais oferece uma vantagem adicional ao permitir um controle mais intuitivo e centralizado dos efeitos aplicados. Isso pode facilitar a aplicação e ajuste de efeitos de forma mais direta e rápida, aumentando a flexibilidade e a criatividade durante a mixagem.

## 1.2 Objetivos

Nessa seção, o objetivo geral e os objetivos específicos almejados por esse projeto são citados.

### 1.2.1 Objetivo Geral

Esse projeto visa a criação de um *mixer* implementado em um sistema embarcado que unifique o controle da mixagem feita por *DJs* em contrapartida ao modelo tradicional de mixagem baseado em equalização de três bandas.

### 1.2.2 Objetivos Específicos

Quanto aos objetivos específicos, elencou-se pontos ao longo do desenvolvimento desse sistemas que servirão como pontos de referência para garantir que o funcionamento do todo.

1. Realizar de forma automatizada o deslocamento de frequências de corte dos filtros passa-altas
2. Implementar dois tipos de efeitos
3. Converter o sinal obtido após o processamento para analógico a fim de ser reproduzido por um sistema de som

O Objetivo Específico 1 envolve a obtenção dos sinais de controle, conversões de analógico para digitais, a atuação correta nos filtros passa-altas de ambos canais.

Em relação aos efeitos, o Objetivo Específico 2 visa embarcar desde a concepção matemática, obtenção dos dados da frequência central e a consequente intensidade do efeito.

Uma vez que a conclusão dos Objetivos acima garante um sinal processado no domínio digital, é necessário garantir que o sinal seja adequadamente convertido para analógico, a fim de ser reproduzido em equipamentos externos. Portanto, o Objetivo Específico 3 agrega a conversão do sinal de saída para que o sistema de som conectado ao sistema consiga reproduzir o canal de saída.

## 1.3 Estrutura do Documento

Esse documento parte do conceito de sinal e sistema para que o processo de amostragem e reconstrução de um sinal seja realizado. Posteriormente, uma conceituação acerca de som e música é realizada percorrendo o conceito físico e uma definição expandida utilizando o pensamento de John Cage.

Além disso, uma retrospectiva do desenvolvimento de *mixers* utilizados por *DJs* desde sua criação até o momento presente é realizada, bem como a exposição de funcionalidades usuais de um *mixer* é descrita para que se delimitem os processos essenciais de um *mixer*.

[ESSA PARTE DEVE SER REESCRITA APÓS A ADAPTAÇÃO DOS REQUISITOS DO PROJETO]

Em seguida, a metodologia do projeto é detalhada com um levantamento de requisitos do projeto. Um fluxograma geral e seus subfluxogramas são detalhados para que cada função fique clara quanto ao seu funcionamento.

Uma seção correspondente à prova de conceito se encontra presente, na qual uma simulação do sistema foi realizada contando com suas almejadas funções de filtragem e efeitos.

Para clarificar a proposta de implementação desse projeto, explanações acerca da interface de usuário e do processamento do sinal são realizadas.

Ao fim, os resultados obtidos da simulação do sistema foram explanados e a conclusão do objetivo final esperado para esse projeto foi realizada.



## 2 Fundamentação Teórica e Estado da Arte

Nessa seção, serão abordados conceitos básicos para a compreensão da monografia (teoria, problema e proposta), começando pela definição de um sinal segundo a literatura, representado por um sinal elétrico, tanto analógico quanto digital; conceituar música e a importância da mixagem realizada por *DJs*, culminando no equipamento utilizado, dando enfoque ao *mixer*, seja na sua história, evolução e no seu estado da arte.

### 2.1 Sinal: Conceito

"Os sinais, que são funções de uma ou mais variáveis independentes, contêm informações sobre o comportamento ou natureza de algum fenômeno, enquanto os sistemas respondem a algum sinal em particular, produzindo outros sinais ou algum comportamento desejado." ([OPPENHEIM; WILLSKY, 2010](#)).

Como exemplifica ([OPPENHEIM; WILLSKY, 2010](#)), tensões e correntes ao longo do tempo são funções, ou seja, sinais, enquanto o circuito em si pode ser compreendido como um sistema que reage à entrada aplicada, ao produzir sinais de saída.

Sensores são dispositivos capazes de mensurar grandezas físicas através da captação de sinais elétricos. Dessa forma, a criação desses instrumentos permitiu a compreensão de fenômenos físicos. Assim, o monitoramento de grandezas permite que se atue em sistemas físicos a fim de se obter um resultado desejado.

Com o desenvolvimento de sensores na história da instrumentação, foi possível a obtenção da quantização de parâmetros físicos através de sinais elétricos. Dessa forma, a compreensão de fenômenos físicos e, consequentemente, a construção de sistemas que podem alterar os sinais conforme a resposta desejada.

### 2.2 Sinal Contínuo

Um sinal pode ser classificado como contínuo caso sua variável independente seja contínua, ou seja, se houver um valor para cada instante de tempo (variável independente).

Por exemplo, na Figura 1, entre o intervalo de 0 a  $t$ , há infinitos valores tanto de tempo quanto para o parâmetro em função do tempo ([OPPENHEIM; WILLSKY, 2010](#)). Um exemplo pode ser o valor de corrente em um resistor alimentado por uma tensão ou um som que gera uma pressão acústica no ar captada pelo sistema auditivo. A Figura 1 é um exemplo de um gráfico para um sinal contínuo no tempo.

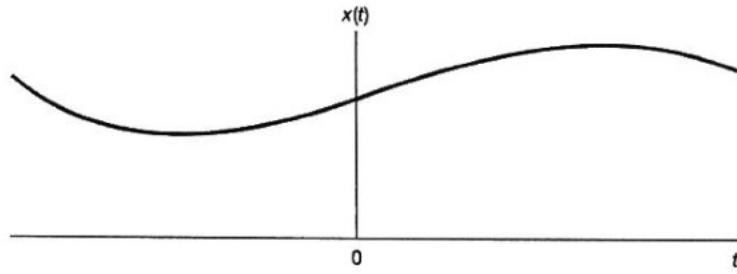


Figura 1 – sinal de tempo contínuo (OPPENHEIM; WILLSKY, 2010)

### 2.2.1 Sinal Discreto

No caso de um sinal discreto, a sua variável independente é o tempo discreto, que pode ser definido por um conjunto de números inteiros, de forma que caso  $n$  não seja inteiro, não há valor definido.

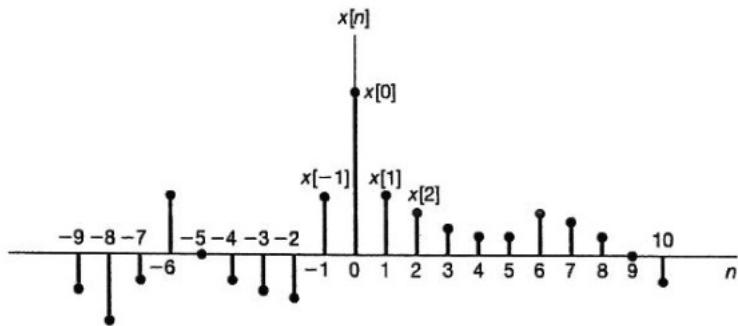


Figura 2 – sinal de tempo discreto (OPPENHEIM; WILLSKY, 2010)

Dessa forma, entre as amostras consecutivas da Figura 2, não há infinitos valores como há no tempo contínuo.

### 2.2.2 Transformada de Fourier

Uma ferramenta matemática crucial para a análise de sinais na frequência é a transformada de Fourier, que realiza uma modificação do termo independente, saindo do tempo ou espaço e indo para frequências, também denominada de equação de análise, que pode ser visualizada na equação 2.1.

$$X(e^{j\omega}) = \sum_{n=-\infty}^{+\infty} x[n]e^{-j\omega n} \quad (2.1)$$

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(e^{j\omega})e^{j\omega n} d\omega \quad (2.2)$$

De forma análoga, a transformada inversa, ou a também chamada de equação de síntese, realiza o processo inverso, modificando o sinal representado no domínio da frequência para o seu domínio original, segundo (OPPENHEIM; WILLSKY, 2010), seja tempo ou espaço. A equação da transformada inversa pode ser visualizada na equação 2.2

### 2.2.3 Teorema da Amostragem

Devido ao desenvolvimento da computação nas últimas décadas e pela consequente diminuição de custos de produção e de aquisição de dispositivos capazes de realizar o processamento digital de sinais, tornou-se muito vantajoso a utilização de sinais no tempo discreto, para que possam ser trabalhados digitalmente e, posteriormente, ou não, serem convertidos novamente para o tempo contínuo, sem a perda da informação inicial.

O processo de obter um sinal discreto a partir de um sinal contínuo é denominado amostragem, e para que, a partir do sinal amostrado, reconstrua-se o sinal contínuo original, o sinal e o processo de amostragem devem preencher alguns requisitos.

Para que um sinal seja amostrado, é necessário que o intervalo de amostragem, ou seja, o espaçamento entre duas amostras, seja regular. A cada período  $T$ , uma amostra é adquirida, resultando em uma frequência de amostragem  $f = 1/T$ . Ao passar essa função do domínio do tempo para o domínio da frequência, obter-se-á uma banda de frequências que compõe o sinal no domínio do tempo.

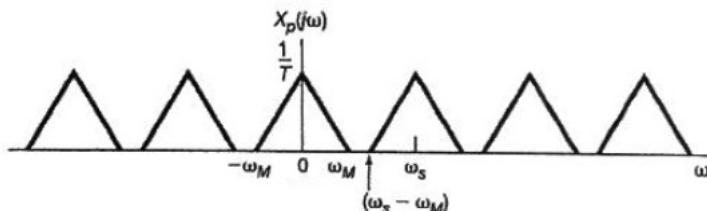


Figura 3 – espectro do sinal amostrado com  $W_s > 2W_m$  (OPPENHEIM; WILLSKY, 2010)

Ao observar a Figura 3, sendo a componente  $\omega_M$  a maior frequência angular presente no sinal do tempo,  $\omega_s$  a frequência de amostragem, constata-se que  $\omega_M < (\omega_s - \omega_M)$  segundo Oppenheim e Willsky (OPPENHEIM; WILLSKY, 2010). Dessa forma,  $\omega_s > 2\omega_M$ . Caso a frequência de amostragem seja menor que  $\omega_M$ , haverá sobreposição das bandas adjacentes e, por conseguinte, a reconstrução do sinal não será possível. Ao respeitar esse requisito, o sinal pode ser recuperado ao utilizar um filtro passa-baixas com a frequência de corte maior que  $\omega_M$  e menor que  $\omega_M - \omega_s$ . Essa análise é o Teorema da Amostragem que infere que a:

$$\omega_s > 2\omega_M \quad (2.3)$$

em que

$$\omega_s = \frac{2\pi}{T} \quad (2.4)$$

O parâmetro  $\omega_M$ , que deve ser menor que metade da frequência de amostragem  $\omega_s$ , recebe o nome de frequência de Nyquist (OPPENHEIM; WILLSKY, 2010).

Esse teorema foi explicitado na literatura por Shannon (SHANNON, 1949), mas foi apontado anteriormente por Nyquist (NYQUIST, 1928). A partir dele, comprehende-se a suficiência da representação de um sinal pela série de Fourier por  $2TW$  amostras, no qual  $T$  é a duração de uma função e  $W$  é a frequência mais alta que compõe o sinal.

## 2.2.4 Aliasing

Quando a frequência de amostragem não está de acordo com o critério de Nyquist, ou seja, menor que o dobro da frequência mais alta, não se tem a reconstrução do sinal, já que, ao realizar a filtragem passa-baixa, na banda  $\omega_M$  haverá componentes da banda adjacente.

## 2.2.5 Teorema da Quantização

Após a amostragem de sinais, obtém-se um conjunto de sinais. Para que esses sinais, pertencentes ao domínio contínuo, possam ser processados, é necessário que se faça a quantização digitalmente. A quantização é um processo no qual se determina intervalos de valores, atribuindo cada valor amostrado a um valor quantizado em função do intervalo no qual o valor se insere. Para cada sinal amostrado, é atribuído um valor quantizado.

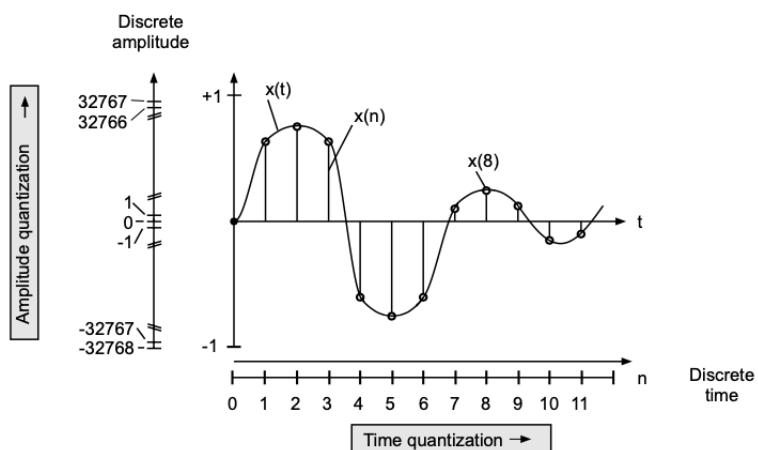


Figura 4 – quantização de amplitudes em tempo discreto (OPPENHEIM; WILLSKY, 2010)

Para determinar com precisão os valores dos intervalos da quantização, o maior e o menor valor obtido pós-amostragem são os limites dos valores de amplitudes discretos. Na Figura 4, observa-se o processo a partir do qual, para cada valor obtido na amostragem, há um intervalo correspondente que passará a representar esse sinal.

O intervalo de amplitude discreta da Figura 4 varia de -32768 a +32767, totalizando 65536 possíveis amplitudes discretas, ou seja,  $2^{16}$  possíveis amplitudes. Dessa forma, cada amplitude pode ser representada por uma palavra binária de 16 *bits*. A quantidade de *bits* a ser utilizada determinará a quantidade de intervalos possíveis, o que, por conseguinte, determinará a precisão da quantização, devido a um erro gerado.

Cada processo de quantização deve levar em conta a precisão necessária e os limites do intervalo de valores possíveis. Ao final, o procedimento de quantização é essencial para as conversões AD (analógico para digital).

A quantização, o processo de digitalizar a amplitude, é descrita pelo Teorema de Quantização de Widrow (WIDROW; KOLLAR; LIU, 1996), segundo Zölzer (ZÖLZER, 2008).

## 2.3 Filtros

Os filtros são sistemas essenciais que desempenham um papel de modificação seletiva de componentes durante um processamento de sinais (OPPENHEIM; SCHAFER, 2013), de forma que, um filtro ideal deixaria que certas frequências passassem sem modificações (banda de passagem), enquanto bloquearia completamente outras (banda de rejeição) (OPPENHEIM; SCHAFER, 2013). Porém, em um filtro real, o comportamento na região da modificação entre as bandas de passagem e rejeição, denominada de banda de transição, não funciona idealmente, havendo na verdade atenuações e amplificações em diferentes graus ao longo desse intervalo de frequência (OPPENHEIM; SCHAFER, 2013).

A implementação de filtros pode se dar de duas formas: através de circuitos analógicos ou digitais, sendo este utilizado no processamento de sinais digitais, de maneira que a sua lógica de circuitaria digital pode ser transformada em uma lógica de programação (OPPENHEIM; SCHAFER, 2013). Circuitos digitais oferecem algumas vantagens em relação aos analógicos, como: a flexibilidade de serem reprogramados de forma simples conforme haja novas necessidades; reproduzibilidade, de maneira que filtros digitais podem ser reproduzidos garantindo alta precisão para que os seus resultados sejam consistentes; imunidade a ruídos devido aos componentes utilizados em um circuito digital em relação ao analógico; e a possibilidade de implementar funções extremamente complexas cujo comportamento analógico pode ser extremamente difícil de ser alcançado (OPPENHEIM; SCHAFER, 2013).

Dentro da categoria de filtros digitais, há inúmeras formas de classificar os tipos existentes. Primeiramente, classifica-se em função da resposta no domínio da frequência observando onde há um comportamento de banda passante e banda de rejeição (OPPENHEIM; SCHAFER, 2013). Além disso, "baixas", "altas" e "banda" são termos que se referem a uma determinada banda sendo, respectivamente, um intervalo entre zero e determinada frequência, uma determinada banda entre duas frequências e, ao final, a banda entre uma frequência até o final da banda existente (OPPENHEIM; SCHAFER, 2013). A combinação das duas classificações gera os inúmeros filtros como: passa-baixa, *low pass filter (LPF)*; passa-altas, *high pass filter (HPF)*; passa-banda, *band pass filter (BPF)*; rejeita-faixa, *band-reject filter (BRF)* e outros (OPPENHEIM; SCHAFER, 2013).

Há outra classificação extremamente importante ditada em função da resposta do filtro ao ser excitado por um impulso, podendo ser finita, denominada de Resposta Finita ao Impulso (*Finite Impulse Response - FIR*), ou infinita, Resposta Infinita ao Impulso (*Infinite Impulse Response - IIR*) (OPPENHEIM; SCHAFER, 2013). Algumas características de cada filtro são essenciais para a escolha do filtro ideal para uma aplicação a um sistema de áudio, por exemplo, a resposta da linearidade da fase e do recurso computacional necessário para cada caso.

Ao se realizar um projeto de filtros, primeiramente, deve-se saber qual tipo e classificação se deseja em um filtro a partir do comportamento geral desejado (OPPENHEIM; SCHAFER, 2013). Em uma etapa posterior, em especificações mais precisas, deseja-se projetar um filtro com uma banda de transição específica, em uma determinada frequência de corte, com comportamentos delineados acerca de ondulações na transição, erros de aproximações e resposta da fase (OPPENHEIM; SCHAFER, 2013).

Algumas especificações são essenciais, como: frequência de corte, frequência que delimita o comportamento da banda de rejeição e da banda passante; largura da banda de transição, faixa de frequência adequada da transição entre a banda passante e a de rejeição; erros de aproximação, desvios máximos aceitáveis para ganho ou atenuação em relação à resposta ideal; resposta em fase, especificações de atraso de grupo e distorções de fase; complexidade computacional, desempenho do processamento em tempo real (OPPENHEIM; SCHAFER, 2013).

Para se analisar qual o filtro ideal para determinada aplicação, deve-se saber as vantagens e desvantagens de cada tipo e classificação que se adequem à situação (OPPENHEIM; SCHAFER, 2013). Além disso, pode-se utilizar parâmetros quantitativos e qualitativos para uma análise mais precisa, como a resposta em frequência (magnitude e fase), atraso de grupo, diagramas de Bode, resposta ao impulso, resposta ao degrau, frequência de corte, atenuação da faixa de rejeição, ondulação da faixa de passagem e erro quadrático médio aplicados a diferentes filtros, para que se obtenha o melhor desempenho possível, além da avaliação qualitativa (OPPENHEIM; SCHAFER, 2013).

## 2.4 Som e Música

Nesta seção, será apresentada uma abordagem histórica, conceitual e técnica sobre a natureza do som e a mixagem realizada por um *DJ*. A partir do som, a música será definida e grandezas físicas serão abordadas. Posteriormente, serão discutidos cenários históricos e técnicos sobre o início das gravações musicais e da discotecagem. Em seguida, com um enfoque mais atual, será apresentado um contexto contemporâneo acerca da mixagem realizada por um *DJ*, que realiza uma seleção musical com base nas características de cada música e faz transições para criar uma experiência musical única.

### 2.4.1 Conceito físico

Segundo Moyses ([NUSSENZVEIG, 2006](#)), "... corpos em vibração produzem sons ...". Dessa forma, é necessário que haja um meio para que o som se propague. Esse meio pode ser líquido, viscoso, sólido ou gasoso (como a atmosfera). De acordo com o mesmo autor, "... ondas sonoras na atmosfera são ondas longitudinais, associadas a variações de pressão, ou seja, compressões e rarefações ...".

Dessa forma, a oscilação de um objeto provoca constantemente compressão e rarefação, alterando a densidade na camada adjacente ao meio pelo qual o som será transmitido e gerando uma diferença de pressão que causa um deslocamento adjacente. Portanto, o ciclo de propagação de um som pode ser visualizado na Figura 5.

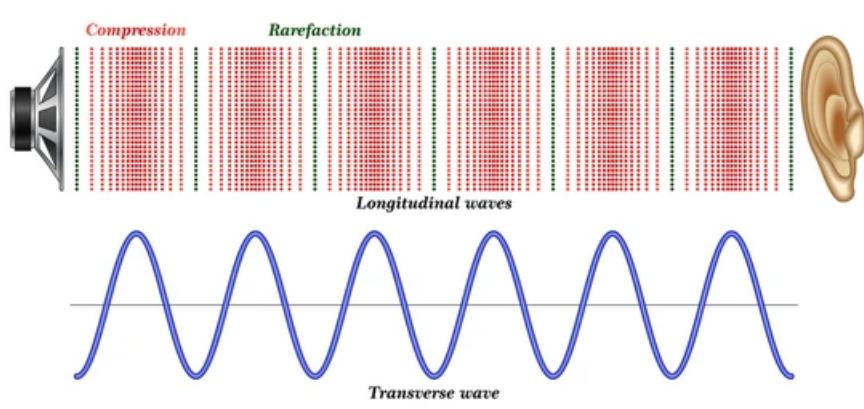


Figura 5 – ciclo de propagação do som ([SAAD, 2019](#))

Conforme indica Moyses ([NUSSENZVEIG, 2006](#)), há parâmetros quantitativos que influenciam a percepção auditiva: intensidade, altura e timbre. A intensidade se relaciona com a amplitude da onda sonora. A altura se refere à banda na qual o som está localizado, seja nas frequências baixas (características graves) ou nas altas frequências (características agudas), em relação à faixa de frequência audível pelo ouvido humano. Já o timbre são sons que possuem a mesma frequência principal, que o autor chama de "tom fundamental

da frequência", mas que possuem outras componentes em frequências superiores, tornando o som reconhecível, apesar de ter o mesmo tom fundamental.

### 2.4.2 Audição Humana

Como Farnell observa ([FARNELL, 2010](#)), embora o corpo humano possa sentir vibrações de 1 a 20 Hz, o ouvido humano é capaz de perceber sons a partir de 20 Hz até 10 kHz ou 20 kHz, dependendo da idade do ouvinte. A faixa normal na qual a voz humana se localiza está entre 300 e 3000 Hz; no entanto, harmônicos provenientes de sons reais podem superar esse limite, inclusive ultrapassando os 20 kHz. Dessa forma, ao utilizar o Teorema de Amostragem, presume-se que a taxa de amostragem mínima necessária para a reconstrução perfeita de um som amostrado seja de, no mínimo, 40 kHz.

### 2.4.3 História da Gravação e Reprodução de Som

A partir do som, a humanidade foi capaz de criar uma expressão artística denominada música, que permite a expressão de ideias, sentimentos, identidades culturais e formas de entretenimento. Além disso, a evolução da tecnologia no século XX proporcionou uma grande difusão da música devido às novas possibilidades de gravação e reprodução.

O marco inicial da gravação de som é atribuído a Thomas Edison, conforme Roads ([ROADS, 1996](#)), que inventou o fonógrafo em 1877. Ao longo da história, a forma de gravação, armazenamento e reprodução de música foi aperfeiçoada, passando por gramofones, gravadores de fios, fita magnética e vinil. Esses dispositivos utilizavam gravação analógica.

O primeiro formato digital amplamente utilizado foi o CD. Outros formatos digitais tentaram alcançar a mesma popularidade do CD, como o MiniDisc, DVD de áudio e Blu-Ray áudio; no entanto, somente os arquivos digitais, com destaque para o MPEG-2 Audio Layer III, mais conhecido como MP3, conseguiram alcançar e superar a popularidade do CD. Esse sucesso deve-se ao uso eficiente de métodos de compressão e compartilhamento, que possibilitaram ao MP3 se tornar o principal meio de compartilhamento de música, acompanhando a ampliação do acesso a computadores e dispositivos digitais capazes de reproduzir música de forma acessível.

### 2.4.4 Disc Jockeys *DJs*

Cada novo formato de mídia possibilitou novas formas de circulação da música. Cada mudança de formato provocou uma transformação na forma como a música é apreendida, começando com apresentações ao vivo, passando por reproduções via rádio, e culminando na reprodução em qualquer lugar com um aparelho adequado. Isso levou ao

surgimento de especialistas em curadoria musical, que influenciaram e modificaram o consumo de música ao transmitir músicas a partir de vinis.

Segundo Brewster e Broughton (BREWSTER; BROUGHTON, 2014), houve uma grande luta para que *DJs* que transmitiam suas seleções nas rádios recebessem apoio das gravadoras. Após a Segunda Guerra Mundial, a Capital Records formalizou esse apoio ao reconhecer o potencial de divulgação dos *DJss* em rádios. O sucesso desses *DJs*, que influenciavam a população das regiões cobertas por suas rádios, teve a capacidade de desenvolver novos gêneros e tendências, pois eram criadores de tendências.

Em 1943, Jimmy Savile observou que um amigo havia conseguido conectar a saída de um gramofone a um rádio valvulado, transformando-o em uma saída de áudio. Isso inspirou Savile a criar um pequeno salão de dança onde gravações de jazz seriam tocadas, permitindo que as pessoas dançassem sem a presença de uma banda. Assim, nasceu um clube, conforme relatado por Brewster e Broughton (BREWSTER; BROUGHTON, 2014). Em seu segundo evento, Savile substituiu o rádio valvulado por um alto-falante. Com o aumento do público, ele implementou essa ideia em vários clubes na Inglaterra e, em um projeto específico, teve a ideia de usar dois toca-discos para diminuir o tempo de transição entre as músicas.

Em 1957, conforme Brewster e Broughton (BREWSTER; BROUGHTON, 2014), Bob Casey promovia festas em ginásios escolares usando um toca-discos conectado a um pequeno alto-falante. Utilizando o sistema de áudio do ginásio, ele posicionava um microfone na saída do alto-falante e transmitia o som amplificado pelo sistema de áudio. Seu pai, um engenheiro de som, desenvolveu um sistema em 1955 com dois toca-discos e controle de volume, permitindo a comutação entre discos e comentários enquanto o volume da música era reduzido.

De acordo com Brewster e Broughton (BREWSTER; BROUGHTON, 2014), em 1964, durante a Feira Mundial em Nova Iorque, Alex Rosner apresentou o primeiro sistema estéreo com dois canais de áudio, um avanço significativo na experiência de audição. Inspirados em sistemas de som da Broadway, David Mancuso e Alex Rosner criaram, para o The Loft (uma famosa festa alternativa), um sistema de som com *subwoofers* (alto-falantes dedicados aos tons graves) e *tweeters* (alto-falantes dedicados aos médios).

Em 1971, Alex Rosner desenvolveu o primeiro *mixer* estéreo para *DJs*, o Rosnie, para o Haven Club (um clube famoso da época), como ilustrado na Figura 6. Esse equipamento contava com duas saídas: uma para fones de ouvido e outra para as caixas de som, permitindo a seleção independente de canais para o retorno da música ao *DJ* e a saída principal. Além disso, o controle *on/off* também foi implementado para o microfone.

O Rosnie foi o primeiro *mixer* para aplicações fora de estúdios a controlar o ganho de bandas de frequência. Ao contrário das soluções anteriores que apenas controlavam

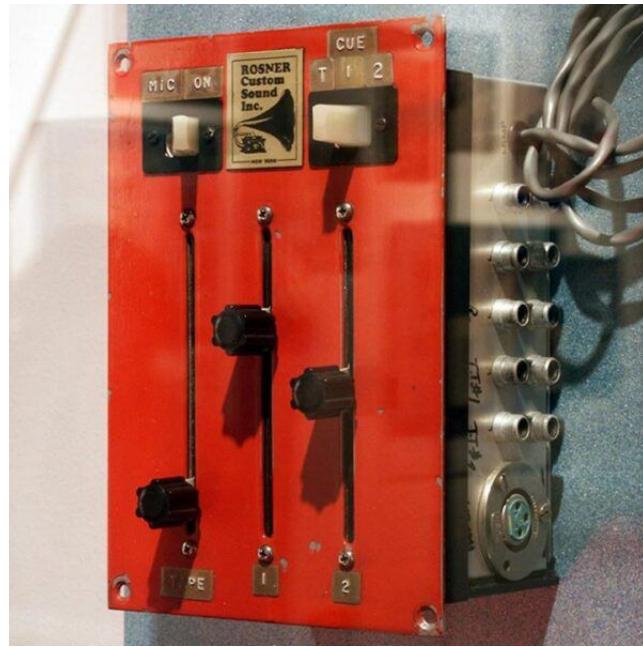


Figura 6 – Rosnie - inventado por Alex Rosner (UBBS, 2015)

o volume dos canais, esse equipamento permitiu mixar dois discos simultaneamente de forma que as transições entre músicas fossem extremamente suaves, muitas vezes não se percebendo a transição entre uma música e outra. No entanto, esse equipamento não foi criado com fins comerciais.

Louis Bozak, auxiliado por Alex Rosner, criou o CMA-10-2DL em 1971, visto na Figura 7, o primeiro *mixer* estéreo comercializado que instantaneamente se tornou padrão nos clubes da época.



Figura 7 – CMA-10-2DL - primeiro mixer estéreo comercial (MARQUES, 2019)

Em 1972, a Technics criou o toca-disco SL-1200, como mostrado na Figura 8, que se tornou um padrão entre os clubes devido ao *driver* do motor, que proporcionava

durabilidade, estabilidade e precisão, auxiliando na mudança do BPM e simplificando o *beat matching* (sincronia de batidas) entre duas músicas.



Figura 8 – toca-disco SL-1200 ([32BITMASCHINE, 2008](#))

A partir de uma colaboração entre duas gigantes da indústria da música, a *Sony* e a *Philips* criaram o CD em 1979, uma nova mídia capaz de armazenar e reproduzir músicas de forma extremamente compacta, leve e mais barata do que o vinil. No ano seguinte, a mesma colaboração desenvolveu o *Red Book Audio*, o formato de arquivo utilizado no CD, que emprega PCM (*Pulse Code Modulation*) na sua codificação. Esses avanços culminaram em 1982 com o início do consumo de CDs, que popularizou ainda mais o consumo de música ao redor do mundo.

Em 1986, a empresa Rane criou um mixer, visto na Figura 9, focado para *DJs* cuja qualidade se aproximava bastante daquela praticada em estúdios de música: o *MP 24 DJ Club Mixer*. Este desenvolvimento possibilitou um aumento significativo na qualidade do som reproduzido em clubes.



Figura 9 – mixer criado pela Rane ([RANE, 2008](#))

Em 1991, conforme Brewster e Broughton ([BREWSTER; BROUGHTON, 2014](#)), foi criado o MP3, um formato que permitiu a compressão de arquivos de áudio, eliminando

conteúdo redundante. Rapidamente, esse método possibilitou o compartilhamento massivo de músicas pela internet.

Nos anos 90, a *Pioneer*, que viria a se tornar uma gigante no mercado de equipamentos para *DJs*, começou a lançar uma série de produtos que facilitavam e ampliavam a atividade do *DJ*. Em 1992, a empresa lançou a primeira CDJ, um equipamento que integrava várias funções necessárias à mixagem, com suporte para CD. Em 2001, a *Pioneer* lançou uma CDJ com leitor de cartões de memória, e em 2007, uma CDJ capaz de ler dispositivos USB. Isso permitiu que *DJs* não precisassem mais carregar toda a sua discografia, bastando um dispositivo de memória digital para ter sua coleção à disposição.



Figura 10 – CDJ-300 - primeira CDJ da *Pioneer* ([PIONEER, 2024a](#))

Na década de 2010, grandes empresas possibilitaram que *DJs* tocassem músicas armazenadas na nuvem e disponíveis em plataformas de streaming. Em 2024, a *Apple* lançou o *Vision Pro*, um óculos de realidade aumentada capaz de emular equipamentos de *DJ*. Com o *Vision Pro*, um *DJ* pode demonstrar suas habilidades e sua coleção com apenas um óculos de VR. Esse equipamento pode ser visto na Figura 11.



Figura 11 – mixagem virtual com o *Apple Vision Pro* ([CARDOSO, 2024](#))

Apesar das muitas soluções inovadoras para mixagem, a configuração mais comum atualmente é composta por duas CDJs e um *mixer*, como mostrado na Figura 12.



Figura 12 – configuração mais encontrada (PIONEER, 2024b)

No entanto, ainda existem configurações que priorizam a qualidade sonora. Para esse público, continuam sendo produzidos equipamentos analógicos que remetem aos layouts de *mixers vintage*, como os *rotary mixers*, apresentados na Figura 13.



Figura 13 – *rotary mixer* (MIXERS, 2024)

A mixagem não se limita ao entretenimento em clubes. O papel de um curador musical remonta à época das rádios. Com a evolução da tecnologia de reprodução, armazenamento e gravação de música, a curadoria musical se tornou mais acessível.

A mixagem transcende a mera função de entretenimento. Desde a expansão do conceito de música proposta por John Cage ([CAGE, 2019](#)), no qual música se torna qualquer som, houve várias expansões conceituais tanto teóricas quanto práticas. Uma *mix* pode ser tanto o ato de um produtor musical ponderar a presença de cada instrumento ou canal em uma música quanto o ato de criar uma nova música a partir de várias, resultando em uma colagem sonora.

Uma *mix* possui estrutura e enredo, com início, meio e fim, e serve a diversos propósitos como entretenimento, relaxamento, meditação, cura, dança, apreciação e pesquisa, entre outros. Além disso, os ambientes de circulação são variados, incluindo páginas de *streaming*, festivais, festas e até mesmo sessões individuais de *bedroom djing* (*DJs* que tocam em sua própria casa, geralmente aprendendo a mixar). Nesse cenário, a liberdade do criador se estende para incluir não apenas misturas de músicas de diferentes gêneros, mas também a incorporação de trechos de áudio de entrevistas, livros, filmes, paisagens sonoras e arte sonora, resultando em uma experiência única.

Além disso, a adição de performances ao vivo, com a utilização de sintetizadores e qualquer dispositivo capaz de gerar som, amplia ainda mais as possibilidades criativas. Desde o uso de sinais elétricos provenientes de eletrodos até a integração de elementos inusitados, como performances ao vivo, as opções são verdadeiramente infinitas, limitadas apenas pela imaginação do criador.

#### 2.4.5 Parâmetros físicos

Quando um *DJ* mixa músicas, diversos parâmetros devem ser considerados durante a execução de uma música ou no momento da transição entre duas, como batidas por minuto (BPM), tom, volume, ganho e a presença de componentes em bandas de frequência, como graves, médios e agudos.

A evolução dos equipamentos para *DJs* pode ser descrita pelo aprimoramento do controle de cada um desses parâmetros, visando um design que permita decisões rápidas com a menor latência possível na saída de áudio e, ao mesmo tempo, melhore o processamento do áudio para obter a melhor qualidade possível.

Para criar uma atmosfera musical ideal, o controle das bandas de frequência é crucial. Ele permite a adição e a subtração de elementos de uma música para criar uma nova versão, que pode ser usada durante a transição entre músicas ou para dar uma nova roupagem a uma música existente. Por exemplo, pode-se adicionar batidas a uma música que não as possui ou adicionar vocal a uma música que é somente instrumental.

O equalizador foi criado, segundo Izhaki ([IZHAKI, 2012](#)), pelos *Bell Labs* com o objetivo de ajustar o que era transmitido ao ouvido devido à atenuação de altas frequências durante a transmissão do sinal pelos fios. No entanto, para a produção musical, o

equalizador é utilizado para manipular o conteúdo das bandas de frequências dos diversos elementos musicais. A Figura 14 ilustra os dois tipos de divisões de bandas apresentados.

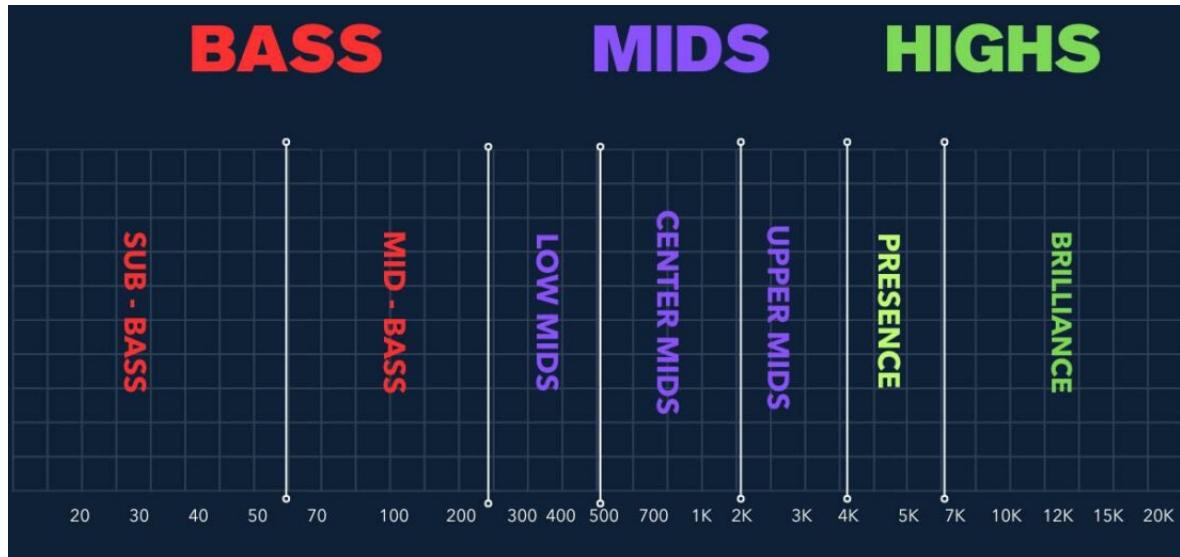


Figura 14 – Espectro de áudio (GLEESON, 2024)

Para a produção musical, considera-se a equalização em sete bandas:

- Subgrave: entre 20 e 60 Hz, onde estão elementos como bumbo e baixo.
- Graves baixos: entre 60 e 120 Hz, tonalidades associadas ao bumbo e ao baixo.
- Médios graves: entre 120 e 250 Hz, onde estão as frequências fundamentais que definem os tons naturais dos instrumentos.
- Médios: entre 250 Hz e 2 kHz, onde estão os harmônicos de baixa ordem de vários instrumentos.
- Médios altos: entre 2 e 6 kHz, onde há harmônicos complexos.
- Agudos ou Brilho: entre 9 e 20 kHz, onde há pouca energia para muitos instrumentos, mas essa banda é importante por estar associada ao brilho na música.

A quantidade de bandas e a definição dos intervalos podem variar entre teóricos e equipamentos, portanto, esses intervalos não são uma regra fixa. No entanto, para *DJs*, a maioria dos equipamentos divide o espectro de áudio em três bandas: graves (de 20 a 250 Hz), médios (de 250 Hz a 4 kHz) e agudos (de 4 kHz a 20 kHz).

Essa escolha de bandas reduzidas deve-se à necessidade de simplicidade e rapidez no uso, além de que, em cada uma dessas três bandas, há elementos semelhantes bem definidos, que auxiliam na construção de uma boa mixagem.

#### 2.4.5.1 Equipamentos

Os equipamentos utilizados por *DJs* evoluíram significativamente ao longo do tempo. Inicialmente, utilizavam-se equipamentos improvisados para outras finalidades até que dispositivos específicos para mixagem foram desenvolvidos.

Existem inúmeras configurações possíveis para realizar uma mixagem, e a escolha dos equipamentos depende da fonte de música utilizada. Se a fonte for vinis, devem ser usados toca-discos. Se a fonte for um dispositivo de memória digital, como um pendrive ou cartão de memória, que armazena músicas organizadas por um *software* de gerenciamento, deve-se utilizar uma CDJ. Para performances ao vivo, onde o *DJ* produz música em tempo real, podem ser utilizados *notebooks*, sintetizadores, *samplers*, sequenciadores e muitos outros dispositivos especializados.

## 2.5 Mixer

Um dispositivo fundamental em qualquer *layout* de mixagem é o *mixer*, cuja função é misturar e controlar os canais de entrada de áudio. Embora existam várias possibilidades de controle, a mais comum é o ajuste das bandas de frequência para cada canal de entrada. Cada dispositivo de reprodução de música é considerado um canal de entrada. O *mixer* é capaz de controlar essas bandas de frequência, bem como somar os sinais de áudio para gerar um sinal de saída que é enviado para os alto-falantes.

Os critérios para classificar um *mixer* como bom são bastante subjetivos. Eles podem incluir o design do equipamento, a qualidade do áudio, a distorção do som, a precisão dos controles, a ergonomia, a portabilidade, o preço, os recursos adicionais como efeitos, referências históricas, e a integração com *software* ou outros equipamentos. Além disso, as preferências pessoais do *DJ* e seu estilo de mixagem influenciam quais aspectos são mais valorizados.

De acordo com Winer (WINER, 2012), o desenvolvimento da conexão entre dispositivos de áudio deve considerar três aspectos principais: o sinal de áudio, as impedâncias (de entrada e saída), e o tipo de conector.

### 2.5.1 Potência de Sinal de Áudio

É comum encontrar entradas *phono* em *mixers*, destinadas a dispositivos como toca-discos, que emitem sinais na ordem de miliVolts. Esses sinais precisam ser amplificados por um amplificador de potência antes de serem processados pelo sistema do *mixer*. Alguns dispositivos têm uma tensão de saída tão baixa que requerem um pré-amplificador para elevar o sinal ao nível de linha adequado para amplificação adicional.

Outra entrada comum é a *line*, que já apresenta sinais em nível de linha. Segundo Winer (WINER, 2012), esses sinais possuem dois níveis padrão: -10 dBV para equipamentos não平衡ados e +4 dBu para equipamentos balanceados. Equipamentos de linha profissional geralmente são balanceados, com tensões RMS em torno de 1,23 V, enquanto equipamentos não平衡ados, como aparelhos de som e TVs, têm tensões em torno de 0,316 V.

### 2.5.2 Cabeamento

Condutores desbalanceados possuem um fio para transmitir o sinal e um terra comum para referência (BARTLETT; BARTLETT, 2009). A Figura 15 ilustra a transmissão de um sinal com ruído através de cabos desbalanceados.

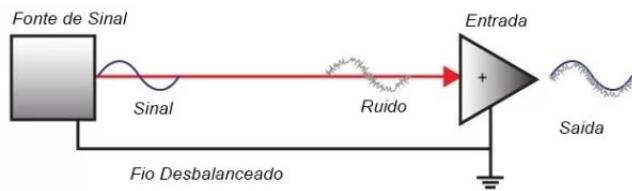


Figura 15 – transmissão com fio desbalanceado (NEIMAR, 2023)

Por outro lado, condutores balanceados utilizam dois condutores para transmitir o sinal, cobrindo-os com uma blindagem. Ao final da transmissão, um diferenciador é usado para reduzir o ruído adquirido durante a transmissão (BARTLETT; BARTLETT, 2009).

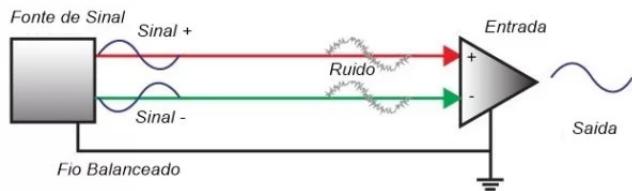


Figura 16 – esquemático de fio balanceado (NEIMAR, 2023)

A Figura 16 mostra um esquemático de fios balanceados no qual há a presença de um condutor adicional, diferente do desbalanceado.

### 2.5.3 Conectores de 1/4", 1/8"(3,5 mm) e 2,5 mm

Para conectar dispositivos, é necessário considerar tanto a transmissão de sinais elétricos quanto os conectores. Cada tipo de conector tem suas vantagens relacionadas a aspectos como quantidade de canais, distância, potência, blindagem e interferência.

Um dos conectores mais conhecidos é o 1/4" (Figura 17), amplamente utilizado em instrumentos como guitarras, violões e teclados, além de amplificadores e *mixers*. Este



Figura 17 – plug de 1/4"(ELECTRONICS, 2024)

conector pode ser usado para sinais estéreos não balanceados ou sinais mono balanceados (BARTLETT; BARTLETT, 2009).



Figura 18 – plug de 1/8"(ELECTRONICS, 2024)

O conector 1/8" (Figura 18), também conhecido como 3,5 mm, é muito encontrado em celulares, dispositivos de reprodução de música e alto-falantes. Internamente, ele possui a mesma construção do conector de 1/4" e pode transmitir os mesmos tipos de sinais.



Figura 19 – plug de 2,5 mm (ELECTRONICS, 2024)

O conector de 2,5 mm, frequentemente utilizado em *headsets* para adicionar um microfone, possui três fios em vez de dois. A Figura 19 ilustra esse tipo de conector.

### 2.5.4 RCA

O cabo RCA (Figura 20), amplamente utilizado nos primórdios dos sistemas de telefonia e também conhecido como *phono* devido ao seu uso em toca-discos, pode transmitir sinais de áudio mono de forma não balanceada. Para transmitir um sinal estéreo, são usados dois cabos RCA.



Figura 20 – cabo RCA ([RS, 2024](#))

Caso um cabo RCA (Figura 20) mono seja conectado, ele pode ser automaticamente duplicado para os dois canais, permitindo que o sinal mono seja transmitido por ambos os canais de áudio ([BARTLETT; BARTLETT, 2009](#)).

### 2.5.5 XLR

O conector XLR, inicialmente chamado de Cannon, foi desenvolvido pela empresa Cannon para a transmissão de sinais平衡ados em aplicações profissionais, oferecendo boa rejeição a ruídos e mantendo a integridade do sinal.



Figura 21 – Conector XLR

A Figura 21 mostra esse conector, que é amplamente utilizado em áudio profissional, especialmente em microfones, mesas de som e outros equipamentos que exigem alta qualidade de sinal.

### 2.5.6 Protocolos Digitais

Conectores para transmissão de dados digitais são amplamente utilizados em *mixers* e em outros equipamentos de áudio. Conectores como *Ethernet*, *USB*, *Bluetooth* e *MIDI* são comuns para diversas configurações e interações entre dispositivos.

### 2.5.7 Amplificação de Potência

Os principais níveis de tensão e potência conectados a *mixers* são o nível *Phono* e o nível de linha. A tensão do *Phono* varia entre 0,1 e 5 mV, enquanto a tensão de um nível de linha é em torno de 2,0 V<sub>RMS</sub>. Segundo (SELF, 2013), é essencial que exista um intervalo adequado de níveis de tensão para garantir que o sinal se comporte corretamente dentro do amplificador. O sinal não deve ser tão baixo a ponto de os ruídos se tornarem mais proeminentes do que o próprio sinal, nem tão alto a ponto de causar saturação após a amplificação.

Portanto, quando um sinal *Phono* entra em um *mixer*, ele precisa ser amplificado internamente para alcançar um nível de linha adequado.

### 2.5.8 Equalização

Como discutido anteriormente, as frequências sonoras audíveis podem ser divididas em três grandes bandas: agudos, médios e graves.

Tradicionalmente, após a amplificação do sinal, o *mixer* permite o controle do ganho ou a atenuação dessas bandas de frequência. Durante o processo de mixagem, diferentes combinações de ganho podem ser ajustadas conforme necessário, e, em um estágio posterior, todos os canais são somados para produzir o sinal final.

### 2.5.9 *Trim* e Volume

Quanto à amplitude do sinal de áudio, há dois controles que frequentemente estão presentes em *mixers*: *trim* e volume. O *trim* é um controle sobre o ganho que um sinal obtém antes de ser processado pelo *mixer* e passar pelos controles de equalização. Serve para que o usuário consiga ajustar dois tipos de sinais para que apresentem o mesmo nível ou a combinação desejada; uma mais presente que a outra.

Já o volume é geralmente visto em um controle de *fader*. Esse botão serve para controlar a amplitude do sinal enquanto o sinal é equalizado, ou seja, posteriormente à amplificação do sinal.

### 2.5.10 Conversão AD/DA

No processamento de sinais de áudio, podemos dividir o tipo de processamento em dois grandes blocos: analógicos e digitais. O processamento analógico é realizado por circuitos analógicos, enquanto o processamento digital é feito por computadores, DSPs e outros dispositivos. A principal diferença reside no estado do sinal durante o processamento. Se o sinal é processado em formato analógico, o *mixer* é analógico, e se o processamento é digital, o *mixer* é digital.

Dispositivos que reproduzem música, como CDJs e toca-discos, podem emitir sinais analógicos ou digitais, dependendo do equipamento e da saída utilizada. Portanto, se o *mixer* requer sinais digitais e a saída do reproduutor de música for analógica, é necessário realizar a conversão de analógico para digital e vice-versa.

Da mesma forma, o sinal de saída do *mixer* precisa ser analógico, pois será amplificado e enviado para os alto-falantes. Assim, se o processamento for realizado de forma digital, o sinal deve passar por uma conversão digital para analógico antes de ser amplificado.



# 3 Metodologia

Essa seção é dividida em dois grandes blocos, que elucidam duas grandes etapas do projeto. Primeiramente, tem-se a prova de conceito, no qual fluxogramas visam explanar a lógica de funcionamento, apontando qual é a entrada e a saída daquele bloco, bem como, qual é o processamento realizado. A prova de conceito foi implementada em um *software* de síntese musical chamado *PureData*.

Em seguida, encontra-se a seção da implementação final realizada. Essa seção é dividida em duas subseções: uma dedicada ao *hardware*, que explica como os sinais de controle foram obtidos e como o sinal de saída se dá para que possa ser reproduzido em um sistema de som, e outra dedicada ao *software*, que engloba a aquisição do sinal através de arquivos de música no formato *.wav*, explicações de aquisição de sinais analógicos convertidos para digitais, que são responsáveis pelos controles de frequência, escolha do efeito e a quantidade de efeito aplicado, além do processo de construção do filtro utilizado na aplicação.

## 3.1 Proposta Geral

A evolução dos equipamentos de mixagem começou com a mixagem em vinil, e ao longo do tempo, novas funcionalidades foram adicionadas, como o *jogger* para atraso e avanço da música, *fader* para controle de volume e *knobs* para ajuste de frequências. Com o avanço da microeletrônica, esses equipamentos evoluíram para a eletrônica digital, incorporando *DSPs* para processar formatos de áudio de alta qualidade, como *WAV* e *FLAC*.

Atualmente, existem dois tipos principais de equipamentos de mixagem: os mais caros, que não requerem um computador, e os mais acessíveis, que dependem de um computador, mas oferecem melhor qualidade de processamento. DJs iniciantes frequentemente enfrentam dificuldades para realizar ajustes finos nas bandas de frequência em *mixers* clássicos de três bandas. Esta habilidade é crucial para destacar elementos musicais e criar novas atmosferas. À medida que um DJ desenvolve seu estilo, a mixagem torna-se mais automática, facilitando a transição entre músicas.

Este projeto propõe a criação de um *mixer* com um controle central que manipula dois arquivos de música. Em vez dos tradicionais controles de três bandas de frequência, o *mixer* utilizará filtros passa-altas. O controle central ajustará simultaneamente as frequências de corte de ambos os canais, permitindo uma transição suave entre as músicas.

Além disso, o sistema incluirá dois efeitos, *delay* e *reverb*, que serão controlados

através de um botão giratório, que determinará a intensidade do efeito aplicado.

Em resumo, o sistema processará arquivos de áudio que já estejam dentro da *Raspberry Pi*, utilizando filtros passa-altas e permitindo ao usuário selecionar e controlar a intensidade dos efeitos, com botões *knobs* para ajuste e um botão *on/off* para seleção dos efeitos.

## 3.2 Levantamento de Requisitos

O levantamento de requisitos é uma etapa crucial no desenvolvimento de qualquer sistema, pois define as funcionalidades e características necessárias para atender às necessidades dos usuários finais. Nesta seção, são apresentados os requisitos funcionais e não funcionais, que especificam o comportamento esperado do sistema, assim como as restrições e qualidades que devem ser atendidas. Os requisitos foram organizados em categorias que abrangem desde aspectos de desempenho e interface do usuário até segurança e manutenção, assegurando uma visão completa e detalhada do que o sistema deve entregar.

### 3.2.1 Requisitos Funcionais

Nessa subseção, são levantados os requisitos que descrevem o que o sistema deve realizar, bem como funcionalidades e comportamentos.

- O sistema deve permitir ao *DJ* utilizar arquivos *.wav*.
- O sistema deve permitir ao *DJ* controlar a frequência de corte.
- O sistema deve permitir ao *DJ* controlar a presença dos efeitos.

### 3.2.2 Requisitos Não Funcionais

Já nesta subseção, há a descrição de como o sistema deve se comportar em relação ao desempenho, segurança, usabilidade e outros atributos qualitativos.

- O sistema deve responder aos comandos do *DJ* com latência mínima, garantindo uma experiência de mixagem fluida.
- O sistema deve ser confiável e estável, capaz de lidar com longos períodos de uso contínuo sem falhas.
- O sistema deve oferecer uma qualidade de som de alta fidelidade, garantindo que o áudio reproduzido seja claro e com mínimas distorções.
- A interface do mixer deve incluir botões físicos ou controles táteis para ajuste de frequência e efeitos de áudio.

- A interface do mixer deve ser organizada de forma lógica e intuitiva, com controles agrupados por função para facilitar a navegação.
- A interface deve possuir indicações claras das funções dos botões de interação com o usuário.
- O sistema deve incluir interface de saída de áudio padrão *RCA*.
- O sistema deve suportar uma ampla gama de frequências de áudio, garantindo que os graves sejam reproduzidos com profundidade e os agudos sejam nítidos e claros.
- O sistema deve ser projetado para minimizar o risco de danos aos equipamentos de áudio conectados.

### 3.3 Fluxograma do Mixer

O *mixer* proposto é composto por sub-blocos de funcionamento interligados, conforme ilustrado na Figura 22.

De forma geral, o sistema opera em um ciclo contínuo de leitura de sinais, tanto das músicas quanto dos controles, e realiza modificações de parâmetros para processar os sinais, que, por fim, são reproduzidos. Cada ciclo pode ser representado na Figura 22.

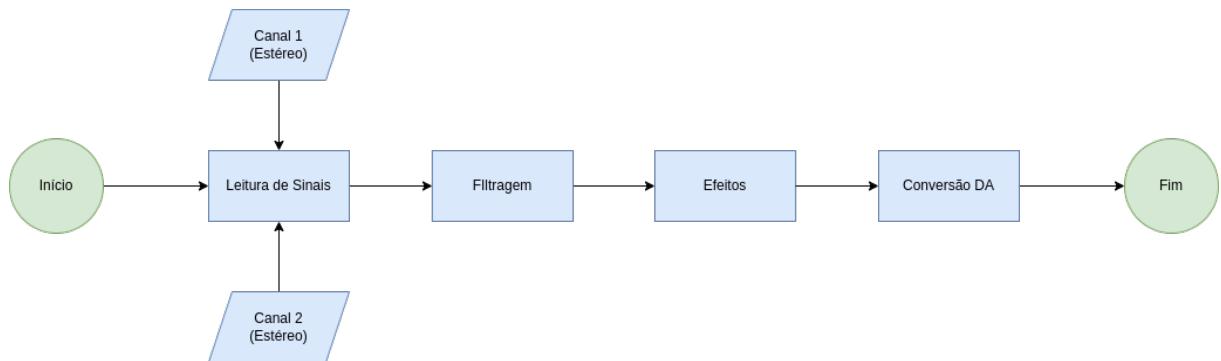


Figura 22 – fluxograma geral do *mixer*

Os sinais provenientes de arquivos de música *.wav* locais entram no sistema e são divididos em *buffers*. Simultaneamente, é feita uma leitura dos valores atuais dos controles disponíveis na interface.

Com os valores dos parâmetros obtidos, a filtragem dos sinais e o processamento dos efeitos são realizados novamente para ajustar o comportamento dos sinais conforme os novos valores dos controles.

Finalmente, os sinais filtrados e os efeitos são combinados e convertidos para o formato analógico, para que possam ser reproduzidos.

Nas subseções abaixo, cada bloco presente no fluxograma geral do sistema (Figura 22) será detalhado em seus aspectos conceituais e processuais por meio de um subfluxograma, que descreve o processamento interno.

### 3.3.1 Bloco de Leitura de Sinais

Os sinais lidos incluem os sinais provenientes de arquivos digitais e os de controle provenientes de dois potenciômetros e um botão de duas posições. Os potenciômetros são utilizados para ajustar a frequência central e a quantidade de efeito desejado, enquanto o botão de duas posições é utilizado para selecionar o efeito desejado.



Figura 23 – bloco de leitura de sinais

Assim, conforme a Figura 23, todos os sinais analógicos de controle passarão por uma conversão analógico-digital, para que, em seguida, algumas variáveis tenham seus valores atualizados. Enquanto isso, os sinais de música obtidos de arquivos digitais já se encontram em formato digital, mas são transformados em sequência de *buffers*.

### 3.3.2 Bloco de Filtragem

No bloco de filtragem, os sinais já estão no domínio digital. Primeiramente, deve-se ler a posição do botão central, que é obtida a partir da conversão de um sinal analógico

proveniente de um potenciômetro, convertido de um sinal elétrico para um sinal digital.

Com a posição do botão, que estará em um intervalo de valores quantizados, realiza-se a normalização e conversão para um valor de frequência de corte, variando entre 20 e 22.050 Hz.

Este valor de frequência de corte é utilizado para atualizar o filtro passa-altas do canal 1, que então realiza a filtragem do sinal correspondente.

Para o canal 2, um novo valor de frequência de corte é calculado usando a expressão da Equação 3.1. O filtro passa-altas deste canal é então ajustado conforme a nova frequência de corte, conforme ilustrado na Figura 24.

No bloco de filtragem, todos os sinais analógicos já estão codificados de forma que podem ser processados digitalmente. A frequência de corte central ( $fc$ ) varia entre 20 e 22.050 Hz, e os valores são atribuídos aos parâmetros  $fc_1$  e  $fc_2$ .

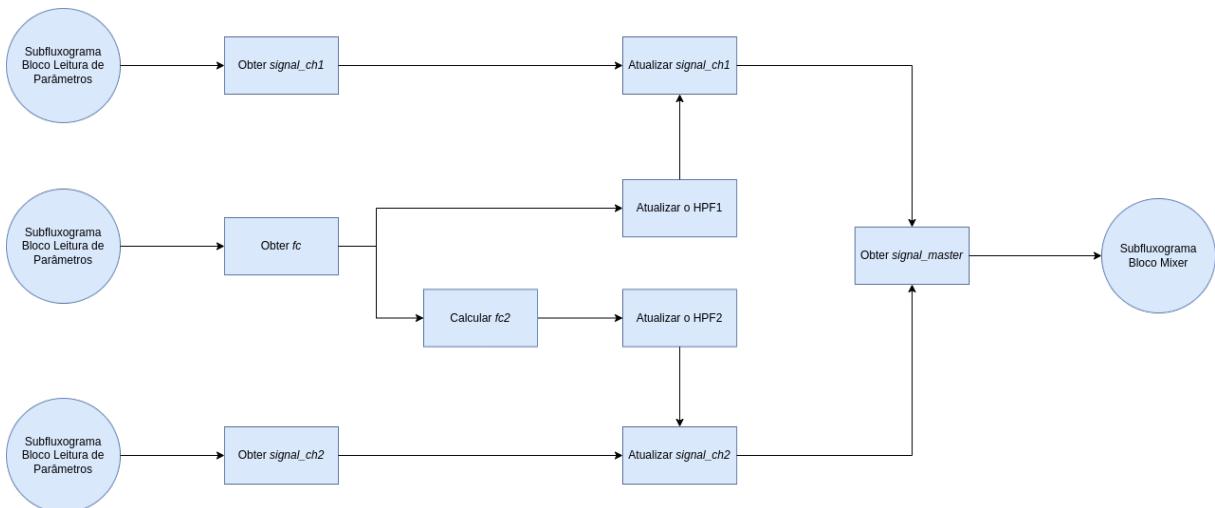


Figura 24 – bloco de filtragem

No subfluxograma da Figura 24,  $fc$  representa a frequência de corte central, lida e convertida a partir do botão central;  $fc_1$  é a frequência de corte para o filtro passa-altas 1 (HPF1) e  $fc_2$  é a frequência de corte para o filtro passa-altas 2 (HPF2).

Ao final deste processo, os sinais dos dois canais são combinados, resultando no sinal *signal\_master*, que é enviado ao bloco de processamento de efeitos.

### 3.3.3 Bloco de Efeitos

Os efeitos do *mixer* podem ter seus parâmetros de reverberação e atraso configurados conforme o botão de quantidade de efeito. O parâmetro de reverberação é ajustado para 1 segundo, e o intervalo de atraso (em milissegundos) é configurado de acordo com a quantidade de efeito desejado.

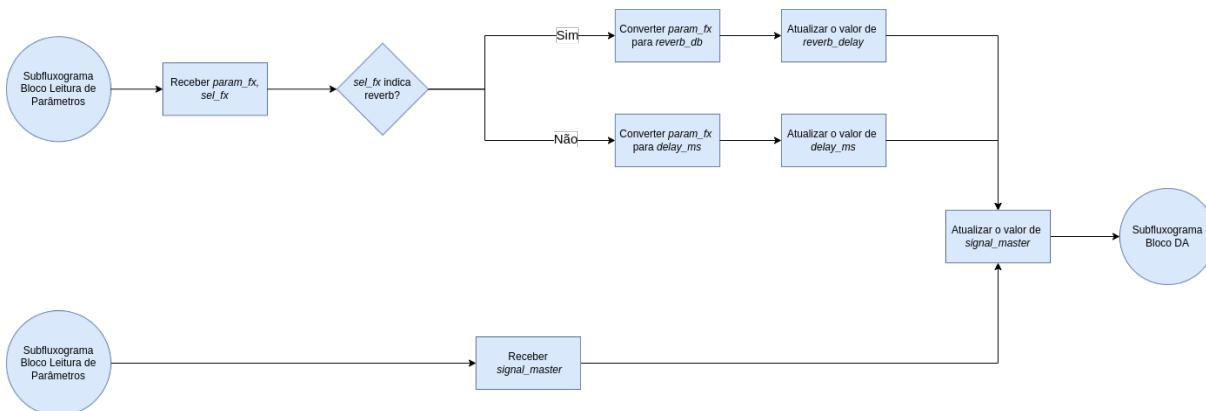


Figura 25 – bloco de efeitos

Além disso, o usuário pode selecionar qual efeito deseja utilizar através de um botão de duas posições, conforme ilustrado na Figura 25. Os parâmetros de seleção e quantidade de efeitos são obtidos do bloco de leitura de sinais. Ao final deste bloco, o sinal filtrado, já com o efeito aplicado, é atribuído ao `signal_master`.

### 3.3.4 Bloco de Conversão DA

O sinal de saída obtido pelo bloco *Efeito* precisa ser convertido de digital para analógico para que possa ser reproduzido em um sistema de som.



Figura 26 – bloco de conversão digital-analógico

Conforme mostrado na Figura 26, o sinal digital passará por um processo de conversão digital-analógico, para que possa ser finalmente reproduzido por caixas de som.

## 3.4 Prova de Conceito

Nesta seção, é apresentada uma implementação em um ambiente virtual que simula a lógica de funcionamento do sistema.

### 3.4.1 PureData

O *PureData* (PUCKETTE, 2002) é um ambiente de música computacional programável, projetado para análise, síntese e processamento de áudio em tempo real através de sinais digitais.

Esse ambiente permite a criação de sistemas de processamento de áudio utilizando blocos programáveis, com funções implementadas tanto pelos seus criadores quanto pela extensa comunidade de usuários.



Figura 27 – Silent Drum de Jaime Oliver

O *PureData* contém uma extensa comunidade que desenvolve diversos tipos de projetos como instalações de arte interativas, geradores de música ambientes totalmente automatizados, uma bateria que utiliza visão computacional para controle da música, como se encontra na Figura 27, emuladores de equipamentos musicais. Outras aplicações incluem gravação, processamento e edição de áudio, *samplers* e instalações de arte interativas que podem integrar sensores ou sistemas de áudio multicanal, além de projetos de projeção mapeada e muitos outros.

No *PureData*, foi possível desenvolver uma prova de conceito que abrange a lógica do botão central para controle das frequências de corte, bem como o funcionamento dos efeitos. Para simular os sinais de entrada, foram utilizados arquivos *.wav* locais.

A demonstração do sistema se divide em duas principais funcionalidades: filtragem e efeitos.

### 3.4.2 Implementação de Filtragem

A filtragem é realizada lendo dois arquivos de música no formato WAV, utilizando as funções `open`, `start` e `stop` para localizar, iniciar e parar a reprodução, respectivamente. Em seguida, o comando `readsf~ 2 1e+06` é utilizado para configurar a leitura dos sinais em estéreo com um milhão de amostras no *buffer*. Este processo é repetido para ambos os arquivos.

Para aplicar a filtragem, utiliza-se a função `hip~`, que implementa um filtro passa-altas. No entanto, o argumento da função varia entre o canal 1 e o canal 2. Como mostrado na Figura 43, o canal 1 ("FC do HPF1") recebe diretamente o parâmetro  $fc$ , proveniente do *slider* azul.

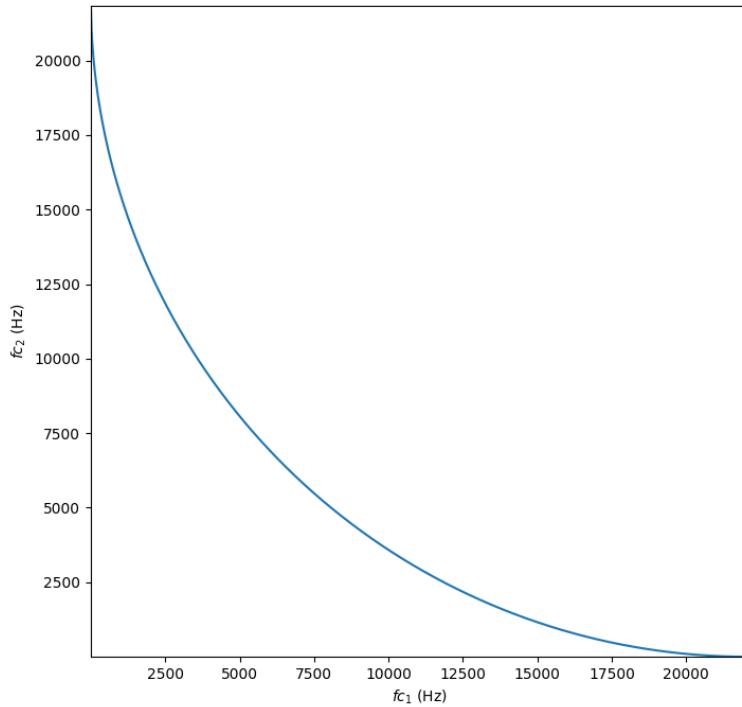
Por outro lado, o filtro passa-altas do canal 2 ("FC do HPF2") utiliza um valor ajustado por uma expressão anterior, conforme a Equação 3.1. Esse ajuste assegura que uma pequena variação em  $fc_1$  resulte em uma grande variação em  $fc_2$ , e vice-versa. Além disso, em frequências centrais, a variação entre os canais torna-se mais semelhante. A Equação 3.1 descreve um círculo com raio igual à frequência de amostragem, com o botão centralizado no ponto (22050, 22050).

$$fc_2 = 22050 - \sqrt{22050^2 - (fc - 22050)^2} \quad (3.1)$$

O ajuste na Equação 3.1 é projetado para ponderar mudanças nas frequências, pois mudanças lineares não são eficazes com um controle centralizado, como demonstrado pelas frequências dos canais na Figura 28.

Mudanças na ordem de centenas no canal 1 têm pouco impacto no canal 2, uma vez que as baixas frequências têm um ganho maior em relação às altas frequências. Essa lógica também se aplica ao outro extremo do controle de frequência.

Na Figura 29, a frequência obtida do botão central, denotada como  $f_c$ , varia de 0.2 a 22050 Hz. A frequência de corte do canal 1, referida como FC do HPF1, é equivalente a  $f_c$ . A frequência de corte do canal 2, identificada como FC do HPF2, é calculada pela Equação 3.1. O sinal resultante, `send~ music`, é a soma dos sinais filtrados dos canais 1 e 2.

Figura 28 – expressão para a  $fc_2$ 

### 3.4.3 Implementação de Efeitos

A implementação dos efeitos utiliza três parâmetros principais: um botão `toggle` para alternar entre os efeitos *delay* e *reverb*; um *slider* para ajustar parâmetros internos dos efeitos; e a frequência de corte do botão central, que automatiza o volume do efeito.

O botão `toggle` é usado para alternar entre os efeitos. Com duas posições disponíveis, sempre um efeito está ativo. Para mudar o efeito, basta alterar a posição do botão, conforme mostrado na Figura 30.

O *slider* é utilizado para ajustar os parâmetros internos de cada efeito. Seus valores variam de 0 a 1, e o botão está ilustrado na Figura 31.

Cada efeito utiliza o parâmetro *fx* do *slider* e o ajusta conforme necessário. No caso do *reverb*, o valor de *fx* é multiplicado por 100 para determinar a quantidade de *dB* que permanece na música após 1s. Para o *delay*, o valor é multiplicado por 1000, transformando-se no intervalo de tempo em *ms* que o efeito permanecerá na música.

A lógica de seleção do efeito é apresentada na Figura 32. Os comandos `receive~` inserem os efeitos como entrada. Cada volume do efeito é multiplicado pelo valor do `toggle`; um deles é multiplicado pelo valor atual enquanto o outro é multiplicado pelo inverso, permitindo que o botão `toggle` funcione como um alternador entre os efeitos.

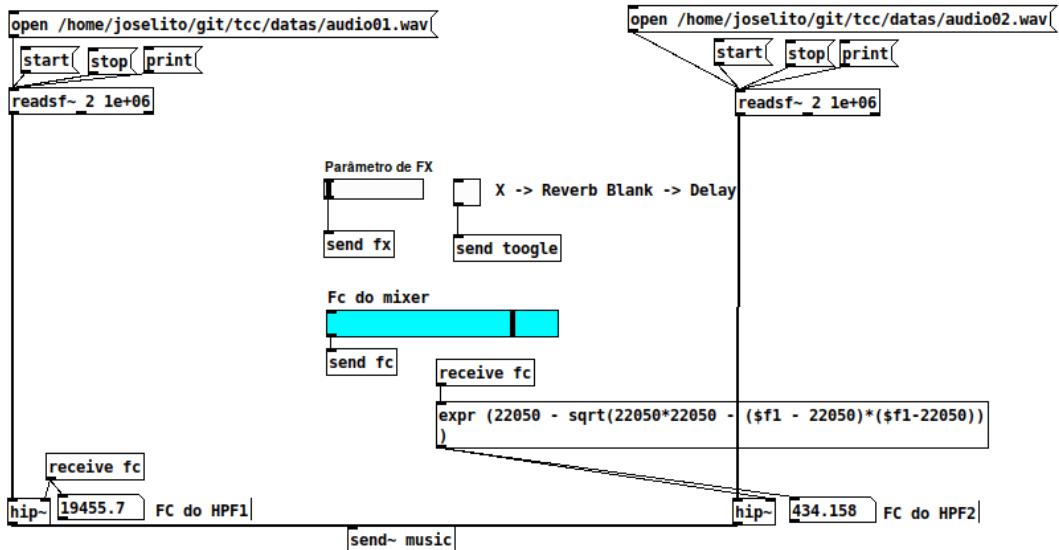


Figura 29 – lógica de funcionamento do botão central no *PureData*

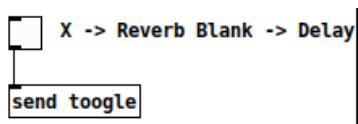


Figura 30 – botão de seleção de efeito no *PureData*

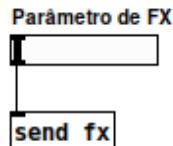
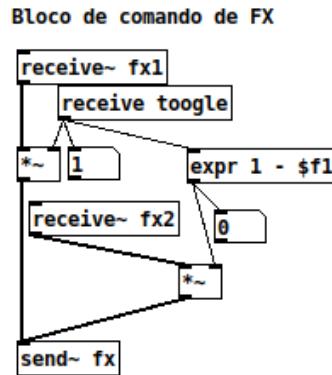
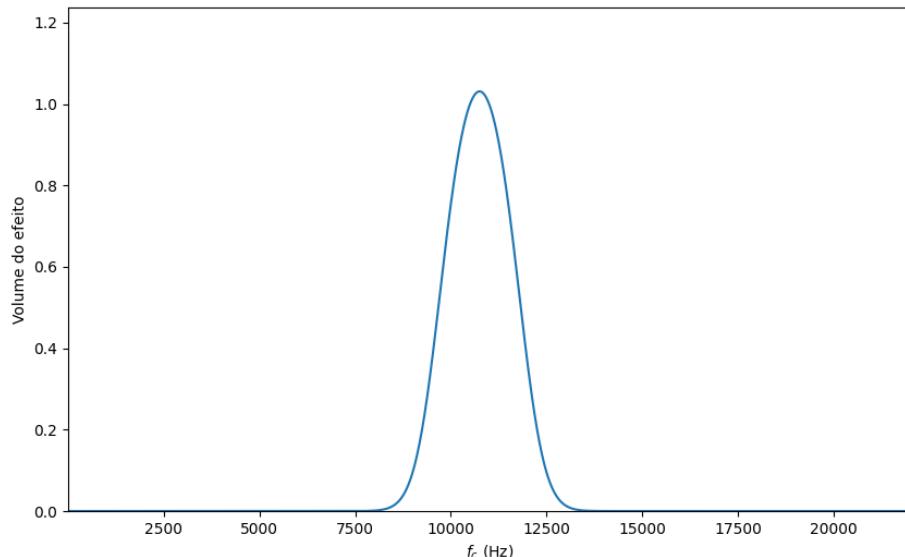


Figura 31 – botão de quantidade de efeito no *PureData*

No sistema, o volume do efeito é ajustado automaticamente com base na posição do botão central, ou seja, nas frequências de corte. A Figura 33 ilustra a variação do volume dos efeitos em função da frequência central.

Figura 32 – lógica de seleção de efeito no *PureData*Figura 33 – variação do volume dos efeitos no *PureData*

No *PureData*, o bloco de automação do volume dos efeitos é implementado usando as operações mostradas na Figura 34.

Finalmente, o sinal dos efeitos é multiplicado pelo volume dos efeitos e, em seguida, somado ao sinal filtrado, resultando no sinal de saída. Esse sinal é processado por um bloco de conversão digital-analógico e, por fim, é reproduzido. O bloco que realiza a soma dos sinais está representado na Figura 35.

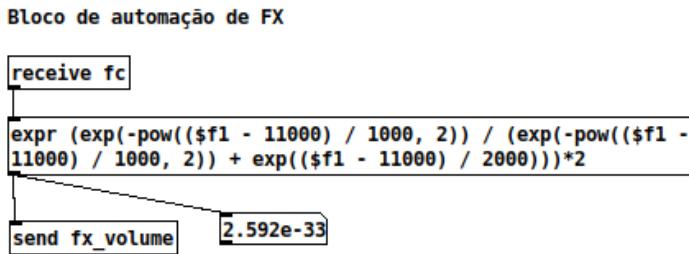


Figura 34 – implementação da variação do volume dos efeitos no *PureData*

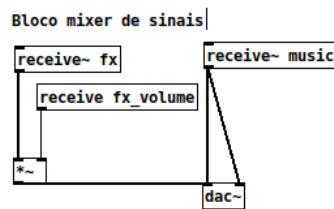


Figura 35 – soma dos sinais filtrados e dos efeitos no *PureData*

## 3.5 Proposta de Implementação

Em linhas gerais, a proposta de implementação desse projeto envolve a utilização de uma *Raspberry Pi* para a escolha das músicas a serem enviadas ao *mixer*, para a aquisição de sinais analógicos responsáveis pelos controles de frequência e efeitos, além de, através do conector de 3,5 mm da placa, realizar a ligação ao equipamento de sistema de som para a reprodução.

### 3.5.1 Implementação em Hardware

Esta seção descreve a proposta de implementação em *hardware* para o sistema de mixagem de áudio, abrangendo conectores, botões e outros componentes essenciais.

#### Botões

A interação do usuário é crucial para ajustar parâmetros como frequência de corte, quantidade de efeitos e seleção do efeito desejado. Para isso, serão utilizados dois *knobs* e uma chave.

Os *knobs* serão equipados com potenciômetros. Eles são alimentados por uma tensão, e sua posição é medida por uma tensão de saída proporcional à tensão de entrada. Essa configuração é preferida devido à precisão oferecida pelo ajuste com dois dedos, que proporciona uma mudança de frequência precisa. A Figura 36 mostra um exemplo esperado de um *knob* para ajustar a frequência central e de intensidade de efeitos.



Figura 36 – botão *knob* para frequência central e efeitos ([ROBOCORE, 2024](#))

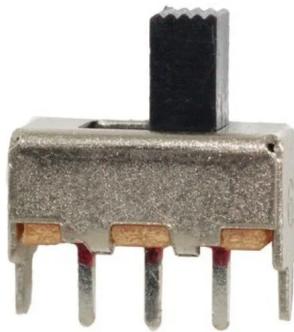


Figura 37 – botão para seleção do efeito ([EVEA, 2024](#))

A chave será utilizada para selecionar entre dois efeitos distintos, gerando dois níveis de tensão: um nulo e outro de alimentação. Cada nível corresponderá a um efeito diferente. Um exemplo de botão para essa finalidade é mostrado na Figura 37.

Para ajustar a quantidade de efeitos desejada, será utilizado um botão semelhante ao *slider*, com a mesma abordagem de controle e ajuste.

### Conversão AD

A conversão analógica-digital será empregada na aquisição dos sinais de controle, ou seja, para dois potenciômetros, utilizando um conversor PCF8591, Figura 38, que conta com uma resolução de 8 bits.

Para que esses sinais de controle sejam lidos, a comunicação I2C da *Raspberry Pi* deve estar habilidada para a leitura.

### Raspberry Pi

A unidade de processamento é o dispositivo responsável pela realização do processamento dos sinais.

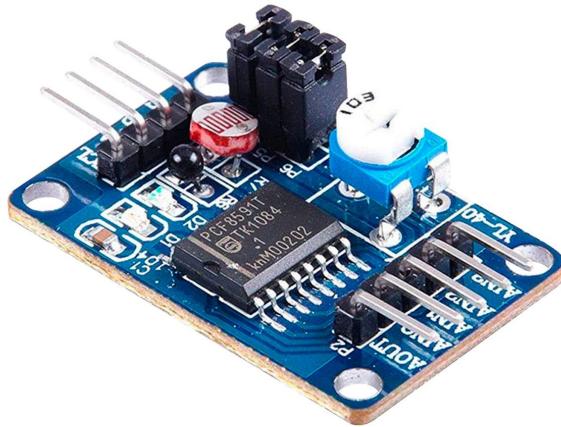


Figura 38 – PF8591 - conversor analógico-digital de 8 bits ([SARAVATI, 2024](#))



Figura 39 – *Raspberry Pi* para processamento dos sinais ([MOGNON, 2016](#))

Para essa tarefa, será utilizada uma *Raspberry Pi*, que oferece um bom poder de processamento e flexibilidade em termos de linguagens de programação suportadas. A *Raspberry Pi* é mostrada na Figura 39.

### Conversão DA

A conversão digital-analógico é necessária para que o sinal final, após o processamento, possa ser reproduzido por sistemas de áudio que utilizam sinais analógicos. A *Raspberry Pi* possui uma saída/entrada de áudio com conector de 3.5mm, conforme ilustrado na Figura 18.

### Diagrama de Conexões

O diagrama de pinagem, Figura 40, contém o conversor AD e a *Raspberry Pi* 4 B, e compreende as conexões necessárias para que os sinais de controle sejam lidos pelo

conversor PCF8591 e transferidos via I2C para a *Raspberry*.

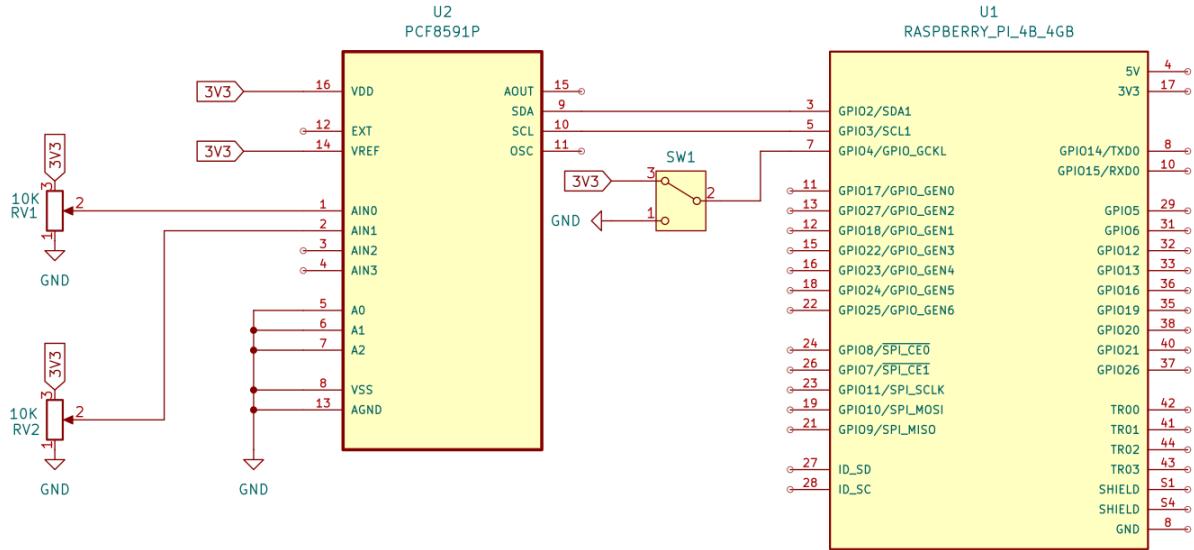


Figura 40 – Diagrama de Conexões

No circuito integrado PCF8591, as entradas A0, A1 e A2 correspondem à configuração do endereço I2C, necessário para que a *Raspberry* consiga identificar o dispositivo e receber as leituras realizadas. Naturalmente, esse dispositivo possui endereço *0x48*. Porém, quando os pinos A0, A1 e A2 estavam abertos, havia uma flutuação do endereço. Para estabilizar o endereço, foi necessário o aterramento dessas entradas de endereço.

Além disso, as entradas utilizadas para as leituras dos *knobs* seletor de frequência de corte (AIN0) e da intensidade do efeito (AIN1), utilizou-se as duas primeiras entradas. Para que se obtenha as leituras específicas dessa entrada, no código de aquisição de sinal, selecionou-se as entradas *0x40* e *0x41*, correspondentes respectivamente às entradas AIN0 e AIN1.

Outro componente utilizado foi a chave de duas posições, no diagrama 40, denominado de *SW1*. Essa chave é capaz de mandar ao GPIO da *Raspberry* o nível lógico alto (3V3) ou baixo (GND). Com esses níveis, é possível descobrir se o usuário deseja que o efeito 1 ou o 2 seja executado naquele momento.

Para a reprodução do sinal de saída, utilizou-se a entrada de 3,5 m nativa da *Raspberry*.

### Lista de Materiais

Para a implementação desse projeto, fez-se necessária a utilização dos componentes abaixo:

- 1 Raspberry Pi

- 1 CI conversor AD/DA PCF8591
- 2 potenciômetros de  $10\text{ k}\Omega$
- 1 chave HH de 2 posições
- 1 cabo 3,5 mm
- 1 fonte 5 VDC com cabo USB-C
- 1 computador para acesso remoto

Para realizar a alimentação da *Raspberry*, faz-se necessária a fonte com um cabo USB-C.

### 3.5.2 Implementação em Software

Para a implementação do processamento de sinais de áudio, o código foi estruturado em várias bibliotecas, cada uma contendo funções específicas para cada etapa do processamento. O fluxo começa com a declaração das variáveis necessárias para iniciar a aquisição dos sinais. Em seguida, os arquivos de áudio no formato *.wav* são lidos e processados utilizando uma lógica de *buffers*, que segmenta os dados para permitir um processamento contínuo e eficiente.

Com os segmentos de dados em mãos, diferentes processamentos são realizados, como filtragem e aplicação de efeitos. Após o processamento, os sinais são então reproduzidos.

[INSERIR UM DIAGRAMA DE ESTADOS PARA ESSA APLICAÇÃO]

Além disso, de forma cíclica, são realizadas aquisições de sinais de controle que permitem ajustar, em tempo real, parâmetros como a frequência de corte, a seleção dos efeitos e a intensidade aplicada.

#### Definição e Alocação Inicial de Buffers

No início da função principal, `int main() {}`, são declarados ponteiros que terão papel fundamental ao longo do processamento de áudio. Esses ponteiros, `*sinal1` e `*sinal2`, irão armazenar os sinais de áudio completos, um para cada canal de áudio. Além disso, são definidos ponteiros para buffers, `**buffers_sinal1` e `**buffers_sinal2`, que serão usados para armazenar segmentos do sinal durante o processamento.

O tamanho do *buffer* é definido pela variável `buffer_size`, que pode ser ajustada de acordo com a necessidade da aplicação. No exemplo, o tamanho padrão do *buffer* foi definido como 1024 amostras, o que permitirá um processamento em blocos de dados,

facilitando a aplicação de filtros e efeitos. A quantidade total de *buffers* a ser utilizada durante a execução será determinada posteriormente com base no tamanho total dos sinais de áudio.

Este trecho inicial de código configura as variáveis básicas necessárias para garantir que o processamento ocorra em partes menores (segmentos), o que é essencial para operações eficientes em tempo real.

### Aquisição de Sinais de Áudio

Para iniciar o processamento, a aquisição dos sinais é realizada por meio da leitura de arquivos *.wav* locais, sendo que duas funções principais são responsáveis por essa tarefa: `ler_wav_estereo` e `ler_dois_wav_estereo`.

A função `int ler_wav_estereo()` armazena os dados do arquivo *WAV* em um *buffer*. O arquivo de áudio é aberto e, caso ocorra algum erro, uma mensagem é exibida. Após a abertura do arquivo, seu cabeçalho é analisado para obter informações essenciais como número de canais, taxa de amostragem, *bits* por amostra e tamanho total. Com esses dados, a memória é alocada para armazenar os sinais, que são então lidos e salvos.

Já a função `ler_dois_wav_estereo()` faz uso de `ler_wav_estereo()` para ler os dois arquivos a partir dos caminhos definidos. Ao invocar `ler_wav_estereo()`, os sinais são alocados em *arrays*.

### Geração dos Buffers

O processamento de áudio realizado por esta aplicação é dinâmico, pois se adapta às novas variáveis de controle, como frequência de corte, escolha e intensidade de efeitos. Para garantir um processamento contínuo e eficiente, foi utilizada a função `gerar_buffers_circulares`, que cria *buffers* de tamanho regular, os quais são utilizados como entradas para as funções de processamento. A função `gerar_buffers_circulares` é responsável por gerar os *buffers* circulares a partir dos sinais de áudio, alocando as memórias necessárias para armazená-los e preenchendo-os de forma a permitir o processamento contínuo dos sinais.

Antes de iniciar processamento, a função `gerar_buffers_circulares` gera os *buffers* circulares a partir dos dois sinais de áudio (`sinal1` e `sinal2`), alocando as memórias necessárias para armazená-los. Em vez de utilizar *buffers* clássicos, a função adota o conceito de *buffer* circular, permitindo que, ao atingir o final de um *buffer*, o processamento continue a partir do início, assegurando um fluxo contínuo dos sinais. Primeiramente, a função calcula o número de *buffers* necessários com base no tamanho total dos sinais e no tamanho de cada *buffer*. Em seguida, ela aloca memória para os ponteiros que armazenarão os *buffers* e, dentro de um laço, aloca memória para cada *buffer* individualmente. Após

garantir a alocação bem-sucedida, a função preenche os *buffers* com os sinais, utilizando acesso circular para garantir que, ao atingir o final do sinal, os dados continuem sendo lidos a partir do início. Se algum erro ocorrer durante a alocação, a função retorna -1 após imprimir uma mensagem de erro; caso contrário, ela retorna 0, indicando que os *buffers* foram criados e preenchidos com sucesso.

### Aquisição dos Parâmetros de Controle

1) Explicar onde esses parâmetros são utilizados 2) Explicar quais componentes realizam essa aquisição 3) Explicar Qual a resolução desses sinais 4) Explicar com qual frequência ele é atualizados 5) Explicar como se obtém a frequência 5) Explicar o código que realiza essa aquisição e atribuição

### Conversão de Frequências de Corte

1) Explicar como a frequência 2 é obtida

### Parâmetros do Filtro FIR

A obtenção de um filtro tradicionalmente parte de especificações como banda de transição desejada em determinada frequência de corte, atenuações na banda rejeitada, erros e outros parâmetros. Porém, para a obtenção do filtro para essa aplicação, utilizou-se um método empírico. Esse método consistiu na obtenção da concentração de potência em uma música de referência antes e depois de uma filtragem, realizada no *PureData*.

Através de um código em *Python*, aplicou-se uma DFT para se obter a contribuição, em relação à potência, da faixa entre 20 e 300 Hz. Em seguida, uma filtragem padrão do *software* foi realizada e novamente se obteve a concentração de potência dessa faixa. Com as duas concentrações, antes e depois da filtragem, obteve-se um parâmetro para realizar uma iteração para elencar a ordem necessária para que a rejeição da banda obtivesse o mesmo comportamento.

Assim, utilizando outro código em *Python*, implementou-se um filtro que retornasse essa concentração após a filtragem. O código iterou a ordem do filtro até que se obtivesse uma atenuação maior que a obtida pelo *PureData* e se chegou a ordem 121.

A seguir, foi necessário descobrir qual janelamento produziria esse desempenho de atenuação através de um teste que retornou a concentração de potência na banda rejeitada com ordem 121 em 11 tipos de janelas. O janelamento que obteve melhor resultado foi o *Hamming*. Dessa forma, o filtro para a aplicação foi obtido.

## Matriz de Coeficientes do Filtro FIR

Quando o valor analógico do potenciômetro é quantizado, obteve-se 256 níveis de quantizações já que o conversor analógico digital escolhido possui resolução de 8 *bits*. Dessa forma, o código possui 256 frequências possíveis para realizar o processamento, ou seja, 256 combinações possíveis de coeficientes para o filtro. Dessa forma, antes de se realizar o processamento, esses coeficientes são calculados e alocados em um vetor, para que, conforme novas frequências de corte sejam obtidas, utilize-se os coeficientes já criados e alocados nesse vetor, ao invés de realizar o cálculo novamente. Essa solução foi aplicada para evitar o cálculo dos coeficientes a cada ciclo.

A função `generate_hamming_highpass_filter()` realiza duas etapas principais para gerar os coeficientes do filtro passa-altas, pois utiliza o método de janelamento para projeto de filtro FIR, ou seja, obtém-se o filtro ideal para determinada frequência de corte, e em seguida, obtém-se os coeficientes para a janela escolhida, conforme a ordem determinada. De forma mais detalhada, primeiramente, calcula-se a resposta ideal do filtro utilizando a função *sinc*. A resposta ideal é obtida na frequência de corte normalizada em função da frequência de *Nyquist*; e os coeficientes são obtidos para permitir a passagem das altas frequências enquanto atenuam as baixas. Aplica-se a janela de *Hamming* para suavizar a resposta do filtro ideal e minimizar o efeito *Gibbs* nos extremos do filtro, melhorando sua resposta em frequência.

A matriz resultante contém os coeficientes dos filtros para todas possíveis frequências de corte, cuja quantidade é limitada pela quantização dos valores analógicos no vetor `frequencias_log[]`.

## Aplicação do Filtro FIR

A função `aplicar_filtro_FIR_buffer` aplica o filtro FIR a um sinal armazenado em um *buffer* circular. O objetivo da função é percorrer cada amostra do sinal e aplicar os coeficientes do filtro FIR. A função recebe como parâmetros o `buffer_sinal`, que é o sinal de entrada, o `buffer_sinal_filtrado`, que armazenará o sinal após a filtragem, o `buffer_size`, que indica o tamanho do *buffer*, os `coeficientes`, que contém os coeficientes do filtro FIR, e a `ordem`, que define o número de coeficientes a serem usados no cálculo da soma ponderada.

Após processar todas as amostras anteriores, a função limita o valor do `acumulador` para garantir que o valor filtrado não ultrapasse os limites de 16 bits, ajustando-o para o valor máximo (`MAX_16BIT`) ou mínimo (`-MAX_16BIT`) permitido. Essa etapa é importante para evitar distorções causadas por estouros de valores. Finalmente, o valor filtrado é armazenado no `buffer_sinal_filtrado[j]`, que contém o sinal de saída filtrado.

O processo de aplicar os coeficientes do filtro FIR a cada amostra do sinal, através da soma ponderada das amostras anteriores, é o mecanismo gerado a partir de um comportamento do sinal no domínio da frequência, a partir da frequência de corte e da ordem do filtro, que modificam os sinais no domínio do tempo, permitindo que a operação de rejeição de determinada banda seja efetiva nos sinais representados no tempo. E é dessa forma que se pode sentir, ao escutar o som filtrado, que determinados elementos foram excluídos dos sinais.

### Efeito - Delay

Para o processamento do *delay* no sinal de áudio, implementou-se uma biblioteca denominada *delay.h*, que contém duas funções: *aplicar\_delay()*, responsável por aplicar o efeito nos *buffers* e *liberar\_delay()*, responsável por limpar os *buffers* após a utilização dos *buffers* processados.

O arquivo *delay.c* declara globalmente um ponteiro para o *buffer* circular onde as amostras são armazenadas, bem como uma variável para a localização dentro do *buffer*. Os parâmetros para essa função são *float \*buffer*, *int buffer\_size*, *float wetness* e *float feedback*, sendo respectivamente, o ponteiro para o *buffer* que contém as amostras a serem processadas, o tamanho do *buffer* de áudio utilizado, o parâmetro que determina a composição do sinal final entre o sinal filtrado e o sinal com efeito e a quantidade de realimentação do sinal atrasado.

A função começa definindo o valor do delay: o delay é inicialmente definido como 500 milissegundos e depois convertido para amostras (*delay\_samples*), com base na taxa de amostragem. Se esse valor exceder o máximo permitido (*MAX\_DELAY\_SAMPLES*), ele é limitado.

Para cada amostra no buffer de áudio, a função obtém a amostra atrasada do *delay\_buffer* com base na posição atual. Em seguida, a nova amostra é uma mistura do sinal original (*buffer[i]*) com o sinal atrasado, controlada pelo parâmetro *wetness*. O sinal atrasado também recebe uma realimentação proporcional ao parâmetro *feedback*, o que faz com que ele seja repetido várias vezes, criando um efeito de eco contínuo.

O *delay\_buffer* é atualizado com a nova amostra somada ao sinal de feedback. A posição do *delay\_buffer* é incrementada cicличamente, reiniciando quando atinge o tamanho máximo de delay (*delay\_samples*). Finalmente, a amostra processada é armazenada no *buffer[i]*, substituindo o valor original.

### Efeito - Reverb

O efeito *reverb* utiliza o conceito presente no efeito *delay* para ser implementado, com a diferença de que neste efeito, uma série de ecos são aplicados tão rapidamente e

de forma difusa que se cria uma sensação de espacialidade ao som. Portanto, os ecos do *reverb* são muito mais curtos e mais numerosos em relação ao *delay*, para que se dê uma sensação de profundidade, como se o som estivesse sendo aplicado a um ambiente real.

Para a implementação desse efeito tão usual na mixagem de *Djs*, defini-se os seguintes parâmetros: **buffer\_size**, que dita a quantidade de amostras em um *buffer*, **sample\_rate**, que é a frequência de amostragem do sinal corrente, e **maxDelay**, que é o máximo atraso a ser utilizado no *delay*, que nesse caso foi 0.5 segundos, devido a um cálculo realizado de tempo por batida na música, levando em consideração 125 BPM.

O processo de aplicação do efeito *reverb* se inicia com a alocação de um *buffer* para armazenar o atraso de cada amostra, levando em conta o parâmetro de atraso máximo.

Em seguida, utilizando o tamanho padrão do *buffer*, aplica-se o efeito de *reverb*, ou seja, o sinal atrasado do *delay* é recuperado em sinais posteriores, de forma que o parâmetro *wetness* indica a presença do efeito e do sinal original; quanto maior o *wetness*, mais a amostra com efeitos é aplicada ao sinal de saída. A reintrodução de amostras anteriores é controlada pelo parâmetro *feedback*, enquanto o peso para o sinal sem efeito e o com efeito é regulado pelo parâmetro *wetness*. Dessa forma, é realizado o efeito *reverb* nesse caso, por um banco de atrasos.

Portanto, o processamento desse efeito ocorre amostra por amostra presente no *buffer* utilizando as seguintes etapas:

O *loop for* (`int i = 0; i < numSamples; i++`) percorre cada amostra do sinal de áudio (*buffer*), processando uma amostra de cada vez.

A variável **drySignal** armazena o valor da amostra original (não processada), extraída do **buffer[i]**.

A variável **wetSignal** é preenchida com o valor da amostra do *buffer* de *delay* na posição atual **delayIndex** (*buffer* esse que é iniciado com zeros, mas que ao longo dos ciclos, funciona como a memória do efeito). Esse valor representa o sinal com o efeito de *reverb* aplicado (atraso e *feedback*).

A fórmula `buffer[i] = (1.0f - wetness) * drySignal + wetness * wetSignal` calcula a mistura entre o sinal original (seco) e o sinal processado (molhado), com base no valor de **wetness**. Quanto maior o valor de **wetness**, maior será a contribuição do sinal com *reverb* no resultado final.

A variável **feedbackSignal** é calculada como **wetSignal \* feedback**, que determina a quantidade de *feedback* a ser aplicada ao *buffer* de *delay*. O valor de **feedback** controla a intensidade do efeito de *reverb* (quanto maior o **feedback**, mais ecos serão gerados).

O *buffer* de *delay* **delayBuffer[delayIndex]** é atualizado com o valor do sinal

seco (`drySignal`) somado ao `feedbackSignal`. Isso garante que a amostra processada seja armazenada no *buffer* de *delay* para ser reutilizada em iterações futuras, criando o efeito de eco do *reverb*.

O índice `delayIndex` é incrementado e ajustado para garantir que ele seja cíclico. O cálculo `(delayIndex + 1) % maxDelay` faz com que o índice retorne ao início do *buffer* quando atingir o limite máximo (`maxDelay`), mantendo o processamento contínuo das amostras.

Em resumo, o código processa cada amostra do áudio, aplica a mistura do sinal original com o sinal atrasado (com o efeito de *reverb*), aplica *feedback* para intensificar os ecos, e atualiza o *buffer* de *delay* de forma cíclica para criar o efeito de *reverb* contínuo.

## Reprodução dos Buffers

A reprodução das amostras é acionada a cada final de ciclo de processamento dos *buffers*. Para tal tarefa, utilizou-se uma biblioteca de áudio chamada *PortAudio* para implementar duas funções que cobrem as seguintes tarefas: inicialização do *PortAudio* e a reprodução do *buffer*.

A função `inicializar_audio()` tem como objetivo inicializar o sistema de áudio, configurar o fluxo de saída de áudio e prepará-lo para a reprodução. Essas etapas foram implementadas, primeiramente, declarando uma variável global `PaStream *stream`, e em seguida, com a inicialização do sistema de áudio através da função `Pa_Initialize()`. Após a inicialização, configura-se o fluxo de dados com parâmetros como quantidade de canais de saída, tipo de dados, taxa de amostragem e tamanho do bloco, através da função `Pa_OpenDefaultStream()`. Por fim, com o sistema de áudio e o fluxo inicializados e configurados, inicia-se a transmissão com a função `Pa_StartStream()`. Portanto, a execução da função `inicializar_audio()` é realizada apenas uma vez.

A função que realizará propriamente a reprodução do áudio é a `reproduzir_buffer()`, de modo que envia um *buffer* para o dispositivo de saída de áudio, previamente configurado. Para realizar essa funcionalidade, a função `Pa_WriteStream()` é invocada junto com os seguintes parâmetros: `stream`, variável global, `media_buffer`, amostras processadas após a aplicação dos efeitos, e `buffer_size`, o tamanho do *buffer*. Dessa forma, `reproduzir_buffer()` é invocada de forma recursiva, a cada final de ciclo de processamento de um *buffer*.

## Processamento dos Buffers

Com as funções que realizam cada etapa do processamento definidas e com suas entradas e saídas alinhadas, há uma função que reúne a chamada dessas funções anteriores para cada *buffer* denominada de `int processar_buffers_circulares`.

Em seguida, inicia-se uma sequência de declarações e alocações de vetores e *buffers* que serão consultados ou manipulados ao longo do processamento contínuo. Primeiramente, declara-se dois vetores essenciais para o processo de filtragem sendo um responsável pelas frequências de corte possíveis e outro pelos coeficientes do filtro FIR. Com os vetores declarados, chama-se as funções `gerar_pontos_logaritmicos()` e `gerar_matriz_coeficientes()`, que utilizarão esses vetores para alocar as frequências de corte e todos os conjuntos possíveis de coeficientes para essa aplicação.

Alocações de espaço para os ponteiros dos *buffers*, que serão utilizados para os sinais filtrados, são realizadas utilizando a função `malloc()`, de forma que ambos canais estão contemplados por esses ponteiros. Na sequência, os *buffers* em si são alocados de forma iterativa até que se crie espaços para o número total de *buffers* definidos pela duração das músicas. Para as duas etapas de alocações, são realizados testes que verificam o êxito da criação e apontamento das memórias solicitadas.

Subsequentemente, inicia-se propriamente o processo contínuo de processamento dos sinais. Adquiri-se os parâmetros de controle (frequência de corte, seleção e intensidade do efeito) que serão usados nas funções específicas de cada etapa, e os coeficientes do filtro FIR são escolhidos em função da frequência de corte e alocados em um vetor denominado `coeficientes_filtro[ORDER]`.

Assim, a função `aplicar_filtro_FIR_buffer()` é chamada utilizando como parâmetros os *buffers* do sinal original e do sinal filtrado, seus tamanhos, os coeficientes e a ordem do filtro, e aplicada aos dois canais; ou seja, aos *buffers* dos dois canais. Os sinais filtrados se encontram em `buffers_sinal_i_filtrado` e uma média entre os dois conjuntos de amostras é realizada e alocada em um outro *buffer* denominado `media_buffer`, de forma que a união entre os dois canais de entrada é realizada.

A próxima etapa do processamento diz respeito aos efeitos. Um teste do valor booleano da seleção do efeito é realizado para verificar qual efeito deve ser aplicado. Assim, chama-se a função adequada, seja `aplicar_delay()` ou `applyReverbEffectBuffer()`, utilizando como parâmetros de entrada `media_buffer`, `buffer_size` e `wetness`, que determina a intensidade do efeito aplicado.

Em seguida, a reprodução do *buffer* final obtido pós aplicação do efeito utilizando a função para essa etapa.

### 3.5.3 Protótipo de Interface de Usuário

A interface do usuário será projetada de forma semelhante a um *mixer* atual, com dois botões *sliders* horizontais e um botão de duas posições. Além disso, haverá um cabo 3,5 mm para que se possa conectar a um sistema de som externo.



# 4 Resultados Preliminares

Para validar o sistema proposto, foram realizados testes no ambiente virtual *PureData*, focando na filtragem controlada por um botão central e no funcionamento dos efeitos, incluindo o controle automático do volume e a configuração dos parâmetros de cada efeito.

O procedimento se inicia com a aquisição dos sinais advindos de arquivos de áudio. Em seguida, em função das frequências de corte, arquivos wav foram obtidos pelo PureData. Em seguida, esses arquivos foram lidos por *scripts* em *Python* para que Figuras que representem tanto o sinal no domínio do tempo quanto no domínio da frequência fossem obtidas.

## 4.1 Resultados de Filtragem

Nesta seção, são apresentados os resultados de ensaios realizados no ambiente virtual *PureData*, simulando a filtragem em diferentes frequências de corte.

Utilizaram-se duas músicas: ([JOCAFI, 2019](#)) e ([TYV, 2019](#)), das quais foram selecionadas janelas de dois segundos. A escolha desse intervalo considerou a presença de elementos de todas as bandas de frequência. A filtragem foi realizada usando filtros passa-altas com as seguintes frequências de corte:

- 0 Hz
- 20 Hz
- 300 Hz
- 4 kHz
- 22 kHz

O filtro implementado no *PureData* pode ser descrito como recursivo, um IIR (Infinite Impulse Response), que, através de uma equação de diferenças entre a amostra atual e a anterior, é capaz de calcular o valor da próxima amostra. Além disso, há inúmeras ordens para esse tipo de filtro, que são oriundas da quantidade de termos anteriores que são utilizados na equação de diferença.

Para cada frequência de corte, foram obtidas as representações do sinal tanto no domínio do tempo quanto no domínio da frequência, utilizando a Transformada de Fourier

de Curto Prazo (STFT - *Short Time Fourier Transform*). As figuras a seguir mostram as representações em função da frequência de corte para ambas as músicas de referência.

Na Figura 41, é mostrada a janela do arquivo de áudio no domínio do tempo, evidenciando a presença de *ticks* e elementos de maiores freqüências.

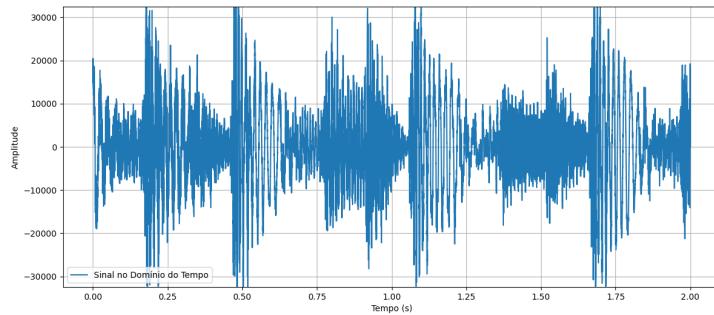


Figura 41 – música 1 no domínio do tempo sem filtragem

Na Figura 42, a STFT da mesma janela revela uma maior presença de elementos em baixa freqüência. Assim como na Figura 41, é possível observar a variação de elementos agudos ao longo da música.

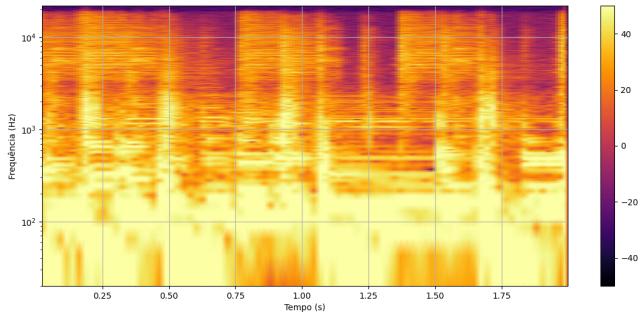


Figura 42 – música 1 no domínio da frequência sem filtragem

Após a aplicacão de um filtro passa-altas com freqüência de corte de 20 Hz, as alterações no sinal são sutis. A Figura 43 mostra o sinal no domínio do tempo com a filtragem aplicada.

A análise da Figura 44 confirma que as componentes em freqüência não sofreram grandes alterações após a aplicacão do filtro passa-altas com freqüência de corte de 20 Hz.

Em *mixers* convencionais, o botão correspondente às baixas freqüências geralmente controla a banda de 300 Hz. Portanto, no presente estudo, o controle de freqüência foi ajustado para aproximar a freqüência de corte de 300 Hz.

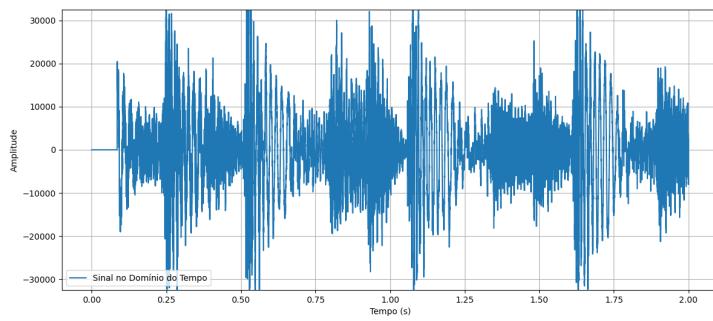


Figura 43 – música 1 no domínio do tempo com frequênciade corte de 20 Hz

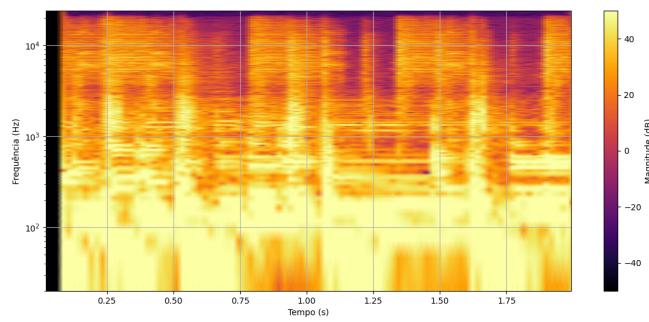


Figura 44 – música 1 no domínio da frequênciade corte de 20 Hz

Os resultados dessa filtragem podem ser visualizados nas Figuras 45 e 46. A Figura 45 mostra que houve uma atenuação significativa dos sinais, evidenciada pelos valores máximos da amplitude. Observa-se também a ausência de sinais de baixa freqüência, que estavam presentes como envelopes nos sinais não filtrados.

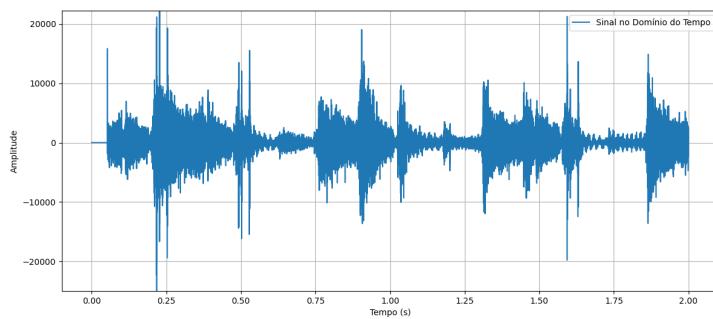


Figura 45 – música 1 no domínio do tempo com uma frequênciade corte de 300 Hz

A atenuação da banda de 300 Hz é confirmada na Figura 46, onde se observa uma redução de aproximadamente 30 dB nas componentes de baixa freqüência.

A próxima freqüência de corte utilizada foi de 4 kHz, que, conforme descrito no Capítulo 2, é onde se encontram os elementos médios. A Figura 47 mostra a atenuação

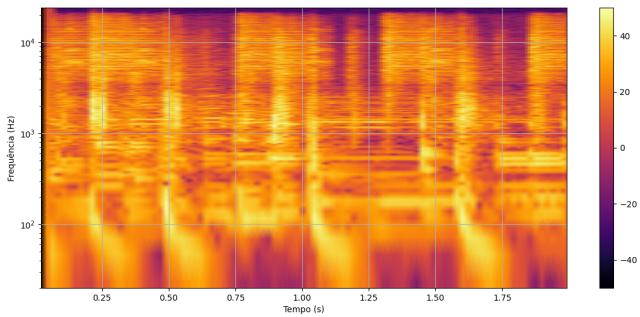


Figura 46 – música 1 no domínio da frequência com uma frequência de corte de 300 Hz

dos sinais em comparação com as filtragens anteriores.

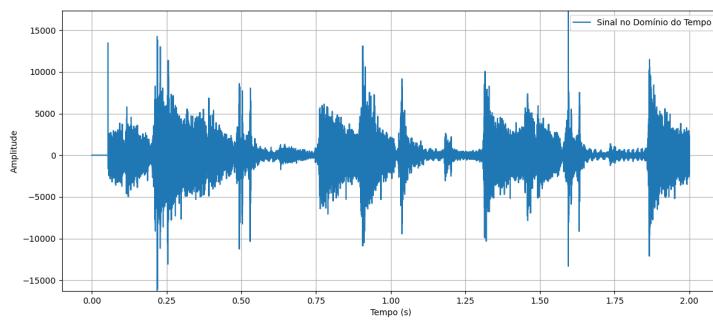


Figura 47 – música 1 no domínio do tempo com uma frequência de corte de 4 kHz

Na Figura 48, que apresenta a STFT, observa-se uma atenuação significativa das componentes na banda de 4 kHz, com algumas componentes chegando a 0 dB em determinados pontos.

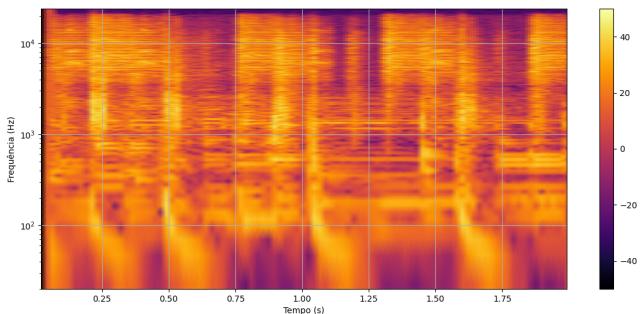


Figura 48 – música 1 no domínio da frequência com uma frequência de corte de 4 kHz

Para a música ([JOCAFI, 2019](#)), uma análise final foi realizada utilizando uma frequência de corte de 24 kHz. Este ajuste visou atenuar os elementos restantes, considerados agudos ou brilhantes. A Figura 49 mostra a atenuação das amplitudes dos sinais em comparação com as filtragens anteriores.

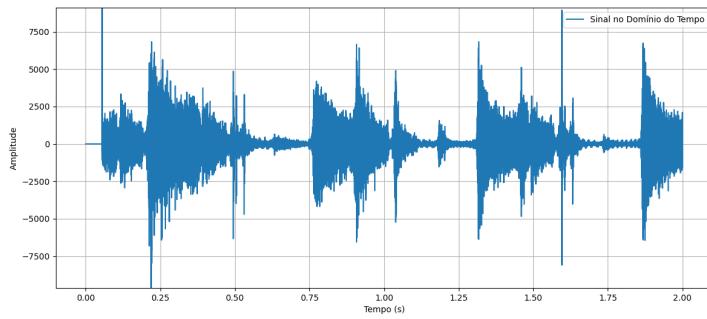


Figura 49 – música 1 no domínio do tempo com uma frequência de corte de 22 kHz

Além disso, na Figura 50, observa-se a atenuação da banda correspondente, com elementos variando de 10 dB a -40 dB. Comparado com a STFT da filtragem anterior, nota-se uma atenuação generalizada do sinal.

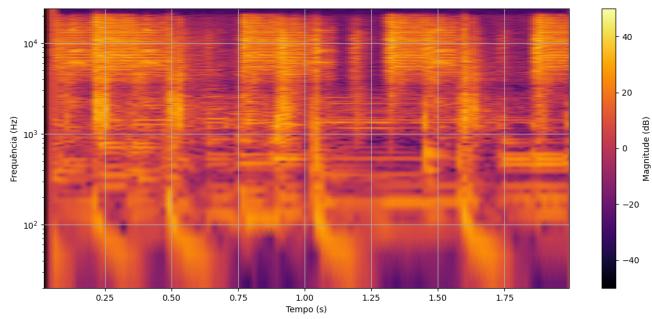


Figura 50 – música 1 no domínio da frequência com uma frequência de corte de 22 kHz

A análise realizada na música ([JOCAFI, 2019](#)) foi também aplicada à música ([TYV, 2019](#)). Os resultados obtidos no domínio do tempo e da frequência, utilizando a STFT, foram semelhantes aos observados para a música ([JOCAFI, 2019](#)).

É importante notar que, conforme o botão de frequência de corte avança, a frequência de corte do filtro do canal 1 aumenta, atenuando primeiro as menores frequências e, em seguida, as maiores frequências. Em contraste, para o canal 2, a frequência de corte começa alta e diminui, ampliando as componentes de frequência do sinal da música ([TYV, 2019](#)).

## 4.2 Resultados de Efeitos

Os efeitos são controlados pela frequência central, que determina o volume do efeito, e um botão de duas posições, que seleciona o efeito a ser utilizado. Testes foram realizados para verificar essas operações.

Primeiramente, simulou-se o volume do efeito em função da posição do botão central. Em seguida, foram realizadas simulações variando os parâmetros dos efeitos.

### 4.2.1 Automação de Efeitos

Para validar o controle automático do volume, isolou-se a saída dos efeitos e variaram-se as frequências, que geram diferentes volumes de efeitos. Os sinais no domínio do tempo foram analisados, conforme mostrado nas figuras abaixo.

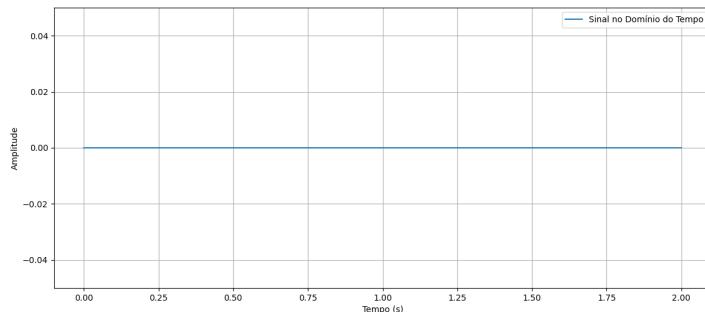


Figura 51 – canal de efeito *reverb* isolado com música 1 a um volume nulo

Na Figura 51, a frequência central utilizada foi a mínima. De acordo com a função mostrada na Figura 33, o volume esperado seria nulo. Assim, o sinal obtido também foi constante e nulo, como esperado.

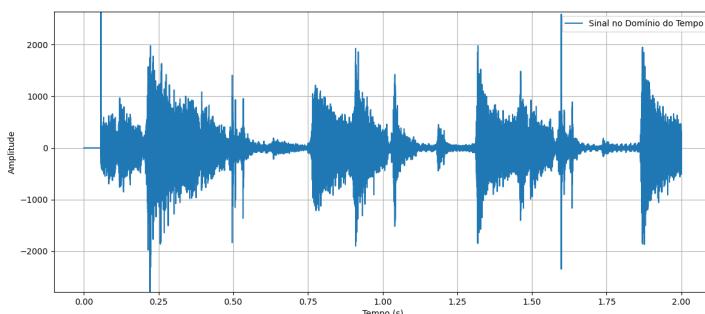


Figura 52 – canal de efeito *reverb* isolado com música 1 a um volume de 0.3312

Em seguida, variou-se a posição do botão central para atingir a frequência de 9454 Hz, resultando em um volume de 0.3312. A Figura 52 mostra um sinal similar ao da música (JOCAFI, 2019), mas com uma amplitude específica.

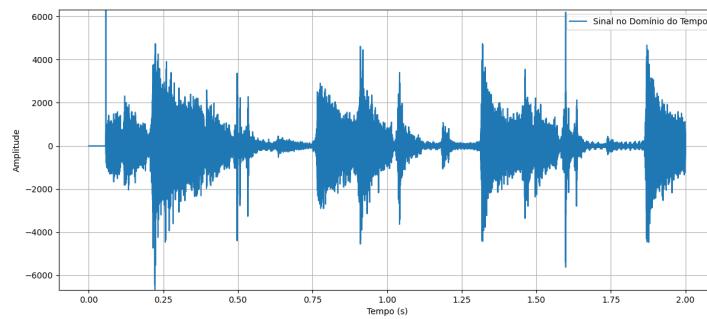


Figura 53 – canal de efeito *reverb* isolado com música 1 a um volume de 0.6625

A posição do botão central foi ajustada para observar a variação do volume do efeito. Na Figura 53, com a frequência de 9875 Hz, observou-se um volume maior.

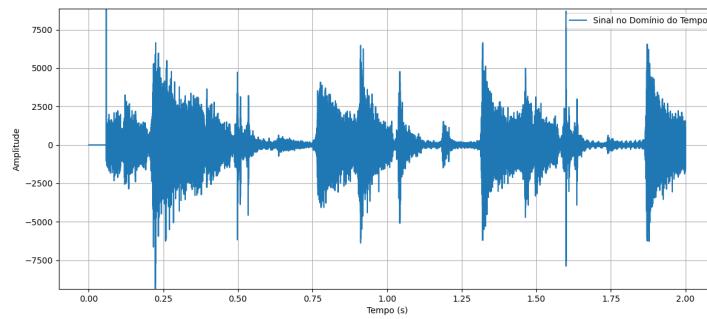


Figura 54 – canal de efeito *reverb* isolado com música 1 a um volume de 1.0

Para validar completamente a automação do volume, o botão central foi posicionado na frequência de 11025 Hz, que gera o volume máximo (1.0). A Figura 54 mostra a ampliação do volume em função da variação da frequência, confirmando a automação do volume.

#### 4.2.2 Reverb

Para validar o controle do efeito de *reverb*, o botão central foi posicionado para garantir que a frequência selecionada resultasse no volume máximo do efeito. Em seguida, ajustou-se a posição do botão de presença do efeito para observar a variação dos parâmetros. No caso do *reverb*, a quantidade de dB presente após 1 segundo é configurada conforme o botão de parâmetro.

Nas figuras a seguir, são apresentadas representações do sinal no domínio do tempo, ilustrando a variação desse parâmetro. O intervalo de dB proposto no sistema varia de 0 a 1000 dB.

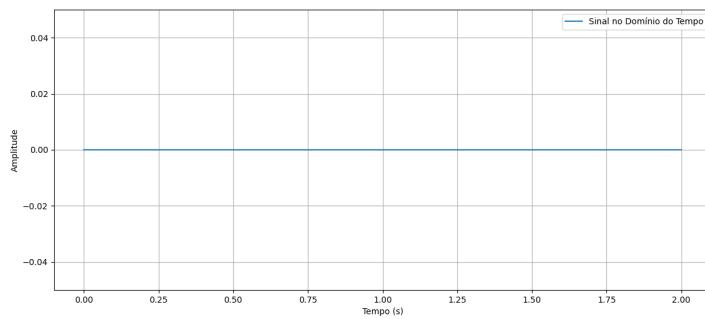


Figura 55 – *reverb* com 0 dB após 1 segundo

Na Figura 55, a posição inicial do botão de quantidade de efeito resulta em 0 dB após 1 segundo. Portanto, a forma de onda esperada também é nula, o que foi confirmado pelos resultados obtidos.

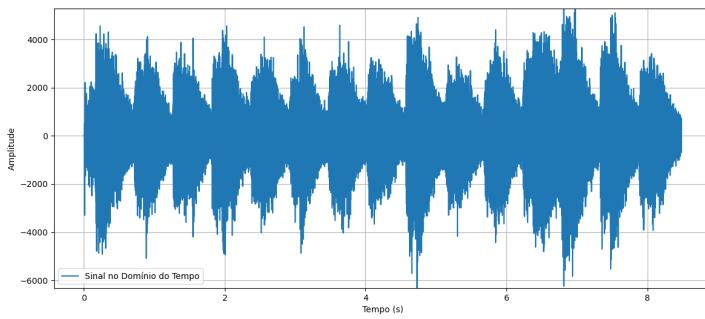


Figura 56 – *reverb* com 25 dB após 1 segundo

Em seguida, o botão de parâmetro de efeito foi ajustado para que 25 dB permanecessem após 1 segundo da amostra atual da música. Na Figura 56, observa-se a repetição de trechos da música devido à aplicação do efeito.

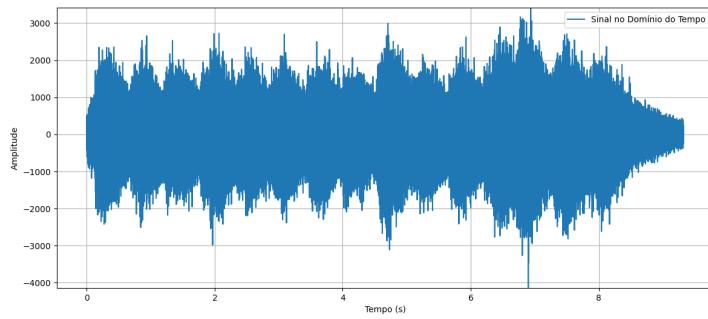


Figura 57 – *reverb* com 50 dB após 1 segundo

Com o aumento do ganho, o efeito de *reverb* torna-se mais perceptível. Na Figura 57, é possível observar uma maior presença do efeito, com um aumento na amplitude do sinal ao longo do tempo.

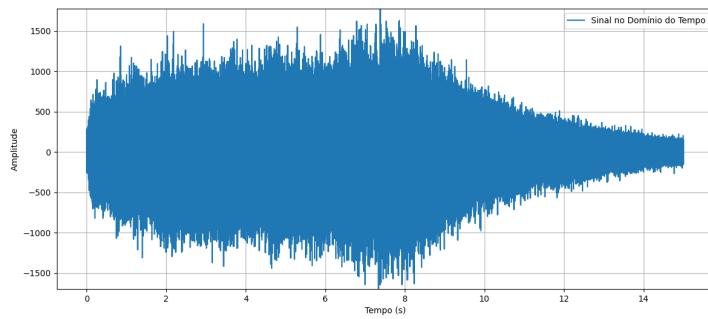


Figura 58 – *reverb* com 100 dB após 1 segundo

Finalmente, ao configurar 100 dB de ganho após 1 segundo, como mostrado na Figura 58, o efeito de *reverb* é ainda mais pronunciado, resultando em um som com pouca definição. Isso indica que uma grande quantidade de som permaneceu, comprometendo a clareza da música.

#### 4.2.3 Delay

O efeito de *delay* consiste em repetir o sinal após um determinado intervalo de tempo. As figuras a seguir ilustram o comportamento do efeito com diferentes configurações de atraso.

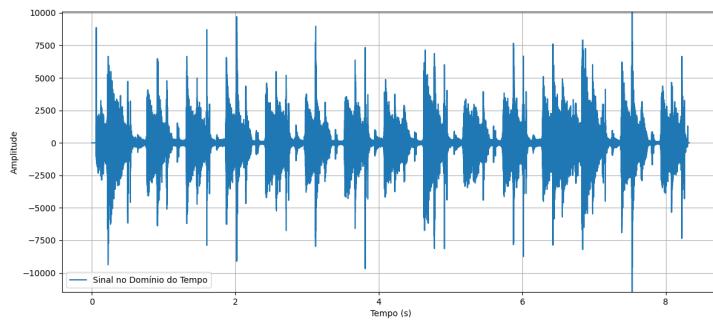


Figura 59 – *delay* de 0 ms

Na Figura 59, foi configurado um *delay* de 0 ms. Nesse caso, a cópia do sinal é obtida imediatamente, sem atraso.

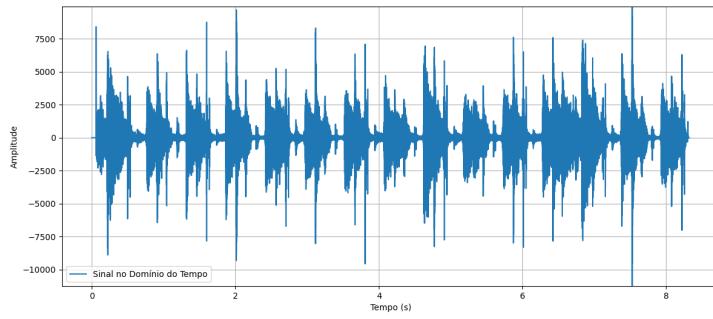


Figura 60 – *delay* de 250 ms

Na Figura 60, um *delay* de 250 ms foi configurado. Aqui, a cópia do sinal é reproduzida com um atraso de 250 ms em relação ao sinal original.

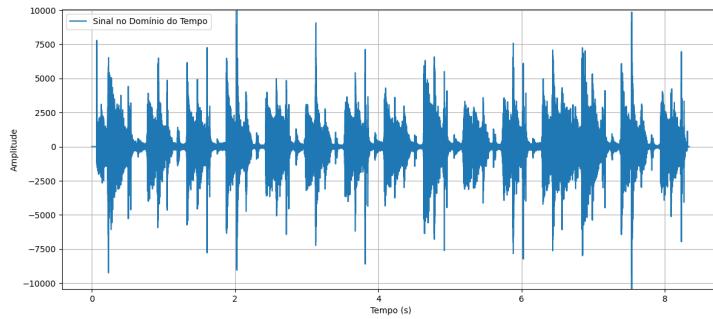


Figura 61 – *delay* de 500 ms

Na Figura 61, o *delay* foi configurado para 500 ms. Neste caso, a cópia do sinal é reproduzida com um atraso de 500 ms em relação ao sinal original.

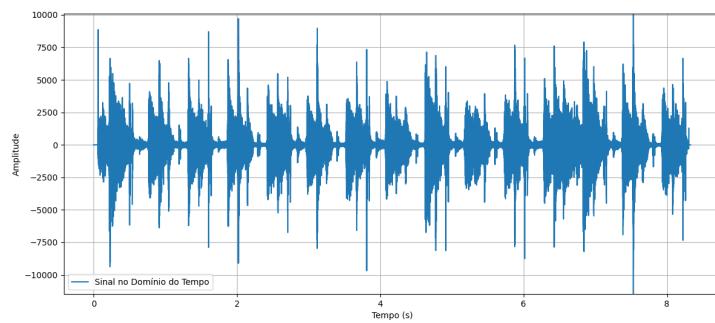


Figura 62 – *delay* de 1000 ms

Na Figura 62, foi configurado um *delay* de 1000 ms. A cópia do sinal é reproduzida com um atraso de 1000 ms em relação ao sinal original.



## 5 Conclusão

Este projeto visa o desenvolvimento de um dispositivo para *DJs* que permita a leitura de sinais provenientes de reprodutores de música e a realização de mixagens automáticas. O dispositivo modificará a frequência de corte de dois filtros passa-altas para realizar a transição entre músicas de forma suave e direta. Além disso, serão incorporados dois efeitos: *reverb* e *delay*, com parâmetros ajustáveis, como dB após 1s e ms de atraso.

O equipamento utilizará uma *Raspberry Pi* e um *Arduino Uno* para leitura, processamento de sinais e conversões necessárias. Toda a lógica de processamento será implementada em linguagem C para garantir alta velocidade e minimizar a latência.

O dispositivo será montado em uma caixa com uma interface de usuário composta por botões *sliders* horizontais e um botão de duas posições.

Para assegurar a usabilidade efetiva do equipamento, serão realizados testes detalhados com usuários e coletados *feedbacks* para aprimorar o desempenho e a funcionalidade dos filtros e efeitos.



# Referências

32BITMASCHINE. *File:Technics SL-1200MK2-2.jpg*. 2008. <[https://commons.wikimedia.org/wiki/File:Technics\\_SL-1200MK2-2.jpg](https://commons.wikimedia.org/wiki/File:Technics_SL-1200MK2-2.jpg)>. [Acessado em 02-09-2024]. Citado 2 vezes nas páginas 15 e 39.

BARTLETT, B.; BARTLETT, J. *Practical Recording Techniques: The Step-by-step Approach to Professional Audio Recording*. Focal Press, 2009. ISBN 9780240811444. Disponível em: <<https://books.google.com.br/books?id=E0uy8adetQoC>>. Citado 3 vezes nas páginas 45, 46 e 47.

BREWSTER, B.; BROUGHTON, F. *Last Night a DJ Saved My Life: The History of the Disc Jockey*. Grove Atlantic, 2014. ISBN 9780802194367. Disponível em: <<https://books.google.com.br/books?id=MxTnBAAAQBAJ>>. Citado 2 vezes nas páginas 37 e 39.

CAGE, J. *Silêncio – Conferências e escritos de John Cage*. [s.n.], 2019. (Literatura / Música). ISBN 9788555911026. Disponível em: <<https://www.cobogo.com.br/produto/silencio-conferencias-e-escritos-de-john-cage-633>>. Citado na página 42.

CARDOSO, B. *Desenvolvedor fala sobre processo de levar o app djay para o Vision Pro*. 2024. <<https://macmagazine.com.br/post/2024/01/12/desenvolvedor-fala-sobre-processo-de-levar-o-app-djay-para-o-vision-pro/>>. [Acessado em 02-09-2024]. Citado 2 vezes nas páginas 15 e 40.

ELECTRONICS, M. *1/4 "(6.35mm) Plugs - Amphenol Audio*. 2024. <<https://br.mouser.com/new/amphenol/amphenol-audio-q-plugs/>>. [Acessado em 03-09-2024]. Citado 2 vezes nas páginas 15 e 46.

EVEA. *Rotating Knob Button, 2 Fixed Positions*. 2024. <<https://www.evea-solutions.com/en/switches-and-leds/1478-rotating-knob-button-2-fixed-positions-on-off-2-contacts.html>>. [Acessado em 03-09-2024]. Citado 2 vezes nas páginas 16 e 63.

FARNELL, A. *Designing Sound*. [S.l.]: The MIT Press, 2010. ISBN 0262014416. Citado na página 36.

GLEESON, A. *The Audio Frequency Spectrum Explained*. 2024. <<https://www.headphonesty.com/2020/02/audio-frequency-spectrum-explained/>>. [Acessado em 02-09-2024]. Citado 2 vezes nas páginas 15 e 43.

IZHAKI, R. *Mixing Audio: Concepts, Practices and Tools*. Focal Press, 2012. ISBN 9780240522227. Disponível em: <<https://books.google.com.br/books?id=f-Rz8c73xh4C>>. Citado na página 42.

JOCAFI, A. C. . *Pé de Bode (Ney Faustini Edit)*. 2019. WAV, Gop Tun Records. Disponível em: <<https://goptun.bandcamp.com/track/p-de-bode-ney-faustini-edit>>. Citado 4 vezes nas páginas 75, 78, 79 e 80.

MARQUES, E. *Bozak CMA DL*. 2019. <<https://www.electronica-pt.com/esquema/old-tv/bozak-cma-10.2-dl-71774/>>. [Acessado em 02-09-2024]. Citado 2 vezes nas páginas 15 e 38.

MIXERS, E. *E & S - AUDIO - MIDI Design and Manufacture — electronique-spectacle.com*. 2024. <<http://www.electronique-spectacle.com/>>. [Acessado em 02-09-2024]. Citado 2 vezes nas páginas 15 e 41.

MOGNON, M. *NEC Displays lançará monitores com Raspberry Pi 3 integrado em 2017*. 2016. <<https://www.adrenaline.com.br/hardware/nec-displays-lancara-monitores-com-raspberry-pi-3-integrado-em-2017/>>. [Acessado em 03-09-2024]. Citado 2 vezes nas páginas 16 e 64.

NEIMAR. *Qual é a diferença entre cabo Balanceado e Desbalanceado*. 2023. <<https://proaudiosp.com.br/qual-e-a-diferenca-entre-cabo-balanceado-e-desbalanceado/noticias/neimar-pro-audio/>>. [Acessado em 02-09-2024]. Citado 2 vezes nas páginas 15 e 45.

NUSSENZVEIG, H. M. *Curso de física básica, 2 : fluídos, oscilações e ondas. calor*. 4.ed.. ed. São Paulo: Edgard Blücher, 2006. ISBN 9788521207481. Citado na página 35.

NYQUIST, H. Certain topics in telegraph transmission theory. *Transactions of the American Institute of Electrical Engineers*, v. 47, n. 2, p. 617–644, 1928. Citado na página 32.

OPPENHEIM, A.; WILLSKY, A. *Sinais e Sistemas*. 2<sup>a</sup>. ed. São Paulo: Pearson, 2010. 592p. p. ISBN 857605504X. Citado 5 vezes nas páginas 15, 29, 30, 31 e 32.

OPPENHEIM, A. V.; SCHAFER, R. W. *Processamento em Tempo Discreto de Sinais*. 3<sup>a</sup>. ed. São Paulo, Brasil: Pearson Universidades, 2013. Capa comum. ISBN 978-8581431024. Citado 2 vezes nas páginas 33 e 34.

PIONEER. *CDJ-300*. 2024. <<http://www.cddj.com/ppdj/products/cdj300>>. [Acessado em 02-09-2024]. Citado 2 vezes nas páginas 15 e 40.

PIONEER. *DJM-A9 - 4-channel professional DJ mixer (black) - pioneerdj.com*. 2024. <<https://www.pioneerdj.com/pt-pt/product/mixer/djm-a9/black/overview>>. [Acessado em 02-09-2024]. Citado 2 vezes nas páginas 15 e 41.

PUCKETTE, M. Using pd as a score language. In: *Proceedings of the International Computer Music Conference (ICMC)*. [s.n.], 2002. p. 184–187. Disponível em: <<https://puredata.info/>>. Citado na página 57.

RANE. *MP 24 Mixer Evolution*. 2008. <<https://www.ranecommercial.com/legacy/mp24evo.html>>. [Acessado em 02-09-2024]. Citado 2 vezes nas páginas 15 e 39.

ROADS, C. *The Computer Music Tutorial*. MIT Press, 1996. (Mit Press). ISBN 9780262680820. Disponível em: <<https://books.google.com.br/books?id=nZ-TetwzVcIC>>. Citado na página 36.

ROBOCORE. *Potenciômetro 20k Deslizante com Knob*. 2024. <<https://www.robocore.net/resistor-potenciometro/potenciometro-deslizante-com-knob>>. [Acessado em 03-09-2024]. Citado 2 vezes nas páginas 16 e 63.

- RS. *Product Detail*. 2024. <<https://seecommercestg.ingrammicro.com/site/productdetail?id=V933158>>. [Acessado em 03-09-2024]. Citado 2 vezes nas páginas 15 e 47.
- SAAD, F. *Sound Propagation - Image Vector*. 2019. <<https://www.shutterstock.com/image-vector/sound-reflection-reverberation-260nw-1282003960.jpg>>. [Acessado em 22-10-2024]. Citado 2 vezes nas páginas 15 e 35.
- SARAVATI. *Módulo Conversor A/D e D/A - 8 Bits - PCF8591*. 2024. <<https://www.saravati.com.br/modulo-conversor-a-d-e-d-a-8-bits-pcf8591.html>>. [Acessado em 30-01-2025]. Citado 2 vezes nas páginas 16 e 64.
- SELF, D. *Audio Power Amplifier Design*. Focal Press, 2013. ISBN 9780240526133. Disponível em: <<https://books.google.com.br/books?id=Poh4MAEACAAJ>>. Citado na página 48.
- SHANNON, C. E. Communication in the presence of noise. *Proceedings of the IEEE*, v. 37, p. 10–21, 1949. Citado na página 32.
- TYV. *The White Calf*. 2019. WAV, Gop Tun Records. Disponível em: <<https://goptun.bandcamp.com/track/the-white-calf-tyv-edit>>. Citado 2 vezes nas páginas 75 e 79.
- UBBS. *The Story Behind the First Ever Mixer*. 2015. <<https://stoneyroads.com/2015/05/the-story-behind-the-first-ever-mixer/>>. [Acessado em 02-09-2024]. Citado 2 vezes nas páginas 15 e 38.
- WIDROW, B.; KOLLAR, I.; LIU, M.-C. Statistical theory of quantization. *IEEE Transactions on Instrumentation and Measurement*, v. 45, n. 2, p. 353–361, 1996. Citado na página 33.
- WINER, E. *The Audio Expert: Everything You Need to Know about Audio*. Focal Press, 2012. ISBN 9780240821009. Disponível em: <<https://books.google.com.br/books?id=Hf0YQAUWGfgC>>. Citado 2 vezes nas páginas 44 e 45.
- ZÖLZER, U. *Digital Audio Signal Processing*. [S.l.]: Wiley, 2008. ISBN 9780470997857. Citado na página 33.