INSTITUTO FEDERAL
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
Bahia

Campus
Vitória da Conquista

COEEL
Coordenação de
Engenharia Elétrica

COORDENAÇÃO DE ENGENHARIA ELÉTRICA - COEEL

PROJETO FINAL DE CURSO - PFC

Instrumento de medição com interface em smartphone

LEONARDO ROMÃO QUEIROZ ARAÚJO

Vitória da Conquista-BA

9 de agosto de 2024

LEONARDO ROMÃO QUEIROZ ARAÚJO

Instrumento de medição com interface em smartphone

Projeto Final de Curso apresentado ao Curso de Graduação em Engenharia Elétrica do Instituto Federal de Educação, Ciência e Tecnologia da Bahia, *campus* Vitória da Conquista, como requisito parcial para obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Prof. Dr. Wilton Lacerda Silva

Vitória da Conquista-BA

9 de agosto de 2024

FICHA CATALOGRÁFICA ELABORADA PELO SISTEMA DE BIBLIOTECAS DO IFBA, COM OS DADOS FORNECIDOS PELO(A) AUTOR(A)

A663i Araújo, Leonardo Romão Queiroz.

Instrumento de medição com interface em smartphone / Leonardo Romão Queiroz Araújo; orientador Prof. Dr. Wilton Lacerda Silva -- Vitória da Conquista: IFBA, 2024.

89 p.

Trabalho de Conclusão de Curso (Bacharelado em Engenharia Elétrica) -- Instituto Federal da Bahia, 2024.

1.Instrumentos de medição. 2.Ohmímetro. 3.Capacímetro. 4.Osciloscópio. 5.RP2040. I.Silva, Wilton Lacerda, orient. II.TÍTULO.

CDU:681.2

Instrumento de medição com interface em smartphone

LEONARDO ROMÃO QUEIROZ ARAÚJO

A presente Monografia, apresentada em sessão realizada em **9 de agosto de 2024**, foi avaliada como adequada para a obtenção do Grau de Engenheiro Eletricista, julgada **aprovada** em sua forma final pela Coordenação do Curso de Engenharia Elétrica do Instituto Federal de Educação, Ciência e Tecnologia da Bahia, *campus* Vitória da Conquista.

BANCA EXAMINADORA:

Documento assinado digitalmente
gov.br
WILTON LACERDA SILVA
Data: 12/08/2024 11:35:53-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Wilton Lacerda Silva
IFBA campus Vitória da Conquista

Documento assinado digitalmente
gov.br
MARCELO MENDONCA DOS SANTOS
Data: 02/09/2024 14:10:04-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Marcelo Mendonça dos Santos
Instituição do membro da banca

Documento assinado digitalmente
gov.br
CLODOALDO GOMES MESSIAS
Data: 12/08/2024 11:42:10-0300
Verifique em https://validar.iti.gov.br

Prof. Esp. Clodoaldo Gomes Messias
Instituição do membro da banca

Vitória da Conquista-BA
9 de agosto de 2024

Dedico esta obra a Deus e à minha família.

"Um homem não pode receber coisa alguma, a não ser que lhe tenha sido dada do céu. [João 3:27]

AGRADECIMENTOS

Pela minha família, pelo apoio constante e sugestões para o meu desenvolvimento. Para minha tia Eliene, que concedeu moradia durante o curso. Agradeço ao meu orientador: Wilton Lacerda, pelo empenho, dedicação e parceria. Sou grato aos professores Marcelo e Clodoaldo pelas contribuições e melhorias do trabalho. Agradeço também aos meus colegas de curso pela cooperação durante o processo de aprendizado. Por fim, sou grato a todos os profissionais do IFBA de Vitória da Conquista pelo zelo que mantém a instituição.

RESUMO

O trabalho propõe o desenvolvimento de instrumento de medição de componentes (ohmímetro e capacímetro), em conjunto com a medição de sinais elétricos (osciloscópio) e a aquisição de dados por meio de microcontrolador RP2040. O projeto utiliza o aparelho celular Android como interface interativa, e a conectividade é alcançada por meio do cabo USB. Por isto, o trabalho foi dividido entre a programação para Android e para o microconntrolador, prototipação e apresentação dos testes realizados. Os resultados dos testes indicam que o aparelho multifuncional é estável e dentro das suas limitações cumpre com as funções para as quais foi projetado.

Palavras-chave: Ohmímetro, Capacímetro, Osciloscópio, RP2040, Aplicativo.

ABSTRACT

This work proposes the development of component measuring equipment (ohmmeter and capacimeter), the addition of signal measuring (oscilloscope), and the data acquisition made by the microcontroller RP2040. The project uses the Android cell phone as the user interface, and connectivity is achieved by the USB data cable. The work has been divided between programming for the Android and the microcontroller, prototyping, and the presentation of the tests carried out. The test results show the multifunction device is stable and, according to its limitations, fulfills the functions for which it was designed.

Keywords: Ohmmeter, Capacimeter, Oscilloscope, RP2040, App.

Lista de Figuras

Lista de Tabelas

Lista de Códigos

Glossário: Símbolos e Siglas

Notação	Descrição	Páginas
C_d	Capacitor desconhecido	8, 9, 31, 32
$D\%$	Desvio da amostra	54
N	Número de amostras	6
R_C	Resistor de carga do capacitor	8, 9, 31
R_d	Resistor desconhecido	7, 29, 31
R_{ref}	Resistor de referência	7, 29–31
T	Período	5
V	Tensão elétrica instantânea	6
V_m	Tensão medida	7–9
V_{CC}	Fonte de tensão conhecida	7, 9
V_{PP}	Valor de tenão pico à pico do sinal	66
V_{RMS}	Tensão RMS	6, 47, 58
f	Frequência (Hz)	5, 47
f_s	Frequência de amostragem	27, 28, 45, 47, 49, 51, 58–60, 63, 66
t	Variável de tempo	9

Notação	Descrição	Páginas
v	A função da tensão elétrica contínua	6
x_a	Valor amostrado do componente resistor ou capacitor	54
x_n	Valor nominal do componente resistor ou capacitor	54
ADC	Conversor analógico digital	4, 5, 13, 14, 19, 20, 27–34, 42, 45, 46, 57, 58, 60, 62, 63, 65
BMS	Bit mais significativo	15
C1	Chave 1	8
C2	Chave 2	8
DAC	Conversor digital analógico	14, 15
DC	Corrente contínua	8, 60
DFT	Transformada Discreta de Fourier	11, 12
DMA	Acesso direto à memória	5, 13, 14, 26–28, 32
FFT	Transformada Rápida de Fourier	11, 33, 34, 59
GND	Tensão de referência de aterramento	16

Notação	Descrição	Páginas
LED	Diodo emissor de luz	25, 32
LSB	Bit menos significativo	15, 17
MC	Microcontrolador	13, 25, 26, 29, 35, 36, 40, 42, 45, 46, 49, 50, 53, 64, 65
PC	Computador pessoal	15
pcb	Placa de circuito impresso	19, 51, 52
PWM	Modulação por largura de pulso	26, 27
RP2040	Microcontrolador da Raspberry pi	14, 15, 19, 24–26, 32, 33, 35, 36, 39, 40, 42, 45–47, 50, 56, 57, 65–67
SAC	Sistema de aquisição de dados	6, 9
SAR	Registrador de Memória Sucessiva	14

Notação	Descrição	Páginas
USB	Barramento Serial Universal	15–17, 43

Sumário

Capítulo 1

Introdução

Na última década, houve uma explosão no consumo de aparelhos celulares. Com toda esta popularidade, estes aparelhos a cada ano que passa ficam mais robustos em termos de hardware. Segundo a 34^a edição da Pesquisa Anual do FGVcia sobre o Mercado Brasileiro de TI e Uso nas Empresas, no Brasil existe a proporção de 1,2 smartphones para cada habitante , no total, 249 milhões de aparelhos no país. Por ser um aparelho acessível às pessoas, ele é ideal para projetos pela sua fácil disponibilidade e por ser um mini computador com um desempenho, a depender da aplicação, excelente.(MEIRELLES, 2023).

Paralelo a isto, os aparelhos eletrônicos oferecem facilidades, e por isto, o consumo deles é crescente a cada ano. Seja para o entretenimento ou para realizar algum trabalho. Porém, por diversas razões, estas ferramentas podem apresentar mau funcionamento, até mesmo pararem de funcionar. Por estes motivos, existem os equipamentos específicos que auxiliam no reparo de eletrônicos. Ao verificar um problema em algum equipamento, a depender do problema, se faz necessário o uso de multímetro e osciloscópio para diagnosticar o defeito, que pode ser diverso.

Segundo Braga, o osciloscópio tornou-se um equipamento indispensável na bancada eletrônica. Pois, as ondas elétricas não podem ser distinguidas pelo ser humano. Somente com o auxílio do aparelho é que podemos visualizar a forma de onda igual a uma "fotografia". (BRAGA, 2014) Já o multímetro é o canivete suíço para componentes elétricos, é a ferramenta indispensável para quem trabalha com eletricidade.

1.1 O problema

É possível unir diversos equipamentos de medição e integrar com as tecnologias oferecidas pelo celular?

Ter ciência e domínio das ferramentas eletrônicas é essencial para os profissionais de diversas áreas. Dito isto, estudantes e pesquisadores independentes, muita das vezes, não possuem recurso para ter acesso e utilizar as ferramentas de medição elétrica.

O projeto com características, como: eficiência, produtividade e segurança, é uma das funções da engenharia. Portanto, uma solução para este problema é desenvolver o equipamento de baixo custo e de simples manuseio. Além disto, ao desenvolver um produto de código aberto, permite-se a qualquer um a possibilidade de aprender, desenvolver e compartilhar o projeto.

1.2 Objetivo Geral

Projetar e construir um dispositivo de medição com múltiplas funções, incluindo: ohmimímetro, capacímetro, osciloscópio, com a interface conectada ao software via celular android, para assim atender os estudantes e pesquisadores em seus estudos em eletrônica.

1.2.1 Objetivos Específicos

- Criar uma interface que permita a interação com o usuário.
- Escolha do tipo de conexão entre o dispositivo móvel (celular) e o microcontrolador.
- Estudar e escolher o hardware do microcontrolador adequado ao projeto, que contenha um conversor analógico digital e protocolo de comunicação adequado.
- Desenvolver a programação para que o microcontrolador escolhido obtenha a amostras do sinal.
- Implementar, um método para salvar as informações adquiridas pelo aparelho.

- Comparar o equipamento construído com equipamentos reais.
- Utilizar um condicionador para aumentar o limite de leitura de tensão do microcontrolador.

1.3 Justificativa

Os equipamentos das bancadas de eletrônica são diversos e numerosos. O multímetro visa concentrar uma gama de funções em um único aparelho. Tendo em mente esta abordagem, neste trabalho buscamos concentrar em um único equipamento vários dispositivos de medição usuais na bancada de eletrônica. Outra proposta é minimizar o custo de adquirir vários dispositivos, ao construir um só com microcontrolador, e utilizar um aplicativo no smartphone para a interface do usuário. Portanto, além de economizar no bolso e no espaço da bancada, a proposta deste dispositivo multifunção tem como função proporcionar aos estudantes, pesquisadores, ”hobistas” e profissionais da área a recorrer a um equipamento compacto, de baixo custo e de fácil manuseio do usuário.

Capítulo 2

Metodologia

Este trabalho tem como base teórica livros, teses de graduação, além de artigos científicos, artigos técnicos e *datasheets*, com tematica sobre o funcionamento de osciloscópios digitais, ohmímetro, voltímetro, entre outros. Foi estudado o hardware de alguns microcontroladores para definir qual será o mais indicado para a função.

A primeira parte do projeto é definir qual microcontrolador será empregado. É desejável que possua as características de preço baixo (abaixo de 50 reais) e disponibilidade no mercado. Outro fator importante para a escolha do microcontrolador é a presença de um conversor analógico digital (ADC) com confiabilidade de amostragem, precisão e exatidão. E possuir acesso direto à memória para não haver interrupção de processamento enquanto ocorre a coleta e armazenamento de amostras. Alguns candidatos são o STM32 e RP2040 por possuírem esses recursos.

Solucionado o problema anterior, o próximo é definir quais linguagens de programação utilizarão tanto o microcontrolador quanto o aplicativo que interage com o usuário do celular. Outro problema é definir o tipo de conexão que será estabelecida entre os modulos. Após solucionar tais problemas, será então testado o funcionamento do dispositivo, bem como, a adição e teste de novas funções ao aparelho.

Por último, serão desenvolvidas as melhorias conforme o uso do equipamento. Tais melhorias, podem ser via software, como as funções para armazenamento dos dados adquiridos para a memória interna do aparelho celular, ou via hardware, adicionando novos componentes que auxiliarão na coleta de dados.

Capítulo 3

Referencial Teórico

3.1 Introdução

Nesta seção são descritas as técnicas aplicadas e princípios de funcionamento dos equipamentos desenvolvidos. Será descrita as especificações de hardware da placa de desenvolvimento, em especial do ADC, e do acesso direto à memória (DMA), do multiplexador e do padrão de comunicação empregado. Por fim, serão abordados trabalhos com o tema correlato ao nosso trabalho, para comparar as semelhanças e diferenças.

3.2 Tecnicas Aplicadas

3.2.1 Período e frequência

Segundo Haykin e Veen (2005), o período é a quantidade de tempo, em segundos, até que o sinal se repita. Já a frequência é a quantidade de vezes que um período se repete e é definida como:

$$f = \frac{1}{T} \tag{3.1}$$

Sendo que:

f $\Rightarrow$ Frequência (Hz);
T $\Rightarrow$ Período (s);

3.2.2 Tensão RMS e Pico

O valor da tensão eficaz é definido como a tensão em corrente contínua que produz uma potência média equivalente ao mesmo resistor em corrente alternada (ALEXANDER; SADIKU, 2013). Pode-se traduzir RMS como a raiz do valor quadrático médio, que significa exatamente sua fórmula matemática geral, dada por:

$$V_{RMS} = \sqrt{\frac{1}{T}\int_0^T v^2 dt} \tag{3.2}$$

A equação acima é válida para sinais no tempo contínuo. Já para sinais discretos, a equação da tensão RMS é dada pela raiz quadrada dos valores médios quadrados da tensão instantânea medida em espaços definidos no tempo (MEASUREMENT COMPUTING CORPORATION, 2012), dado por:

$$V_{RMS} = \sqrt{\frac{V_1^2 + V_2^2 + V_3^2 + \ldots + V_N^2}{N}} \tag{3.3}$$

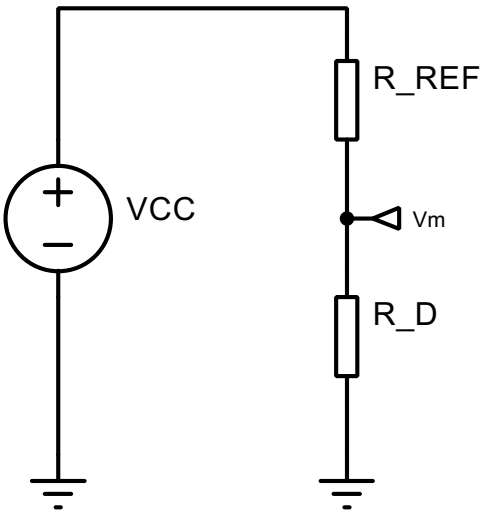
Sendo que:

V_{RMS} $\Rightarrow$ Tensão RMS (V);
v $\Rightarrow$ Valores instantâneaos de tensão (V);
V $\Rightarrow$ Um valor de tensão elétrica discreta (V);
N $\Rightarrow$ Número de amostras (adimensional);

A tensão de pico é o maior valor de tensão, em módulo, que o sinal atinge em um período. Dessa forma, se o pico do sinal for positivo, é nomeada de tensão máxima, senão tensão mínima. Também existe a tensão de pico a pico, que é a diferença entre a tensão máxima e mínima.

3.2.3 Medição de Resistência

A figura 3.1 representa o método divisor de tensão. O circuito é composto apenas por dois resistores e uma fonte de tensão conhecida. A corrente é injetada pela fonte de tensão sobre um resistor de referência em série a um resistor desconhecido. Entre os resistores é conectado o sistema de aquisição de dados (SAC) que mede a tensão.



FONTE: Próprio Autor

Figura 3.1 – *Medição indireta da resistência por divisão de tensão.*

Por meio da tensão medida V_m é possível computar o valor do resistor desconhecido, basta realizar a equação de divisor de tensão e isolar a variável R_d, como a equação a seguir:

$$\begin{aligned} V_m = \frac{R_d V_{CC}}{R_{ref} + R_d} \quad &\Rightarrow \quad R_d(V_{CC} - V_m) = R_{ref} V_m \\ R_d \quad &= \quad \frac{R_{ref} V_m}{V_{CC} - V_m} \end{aligned} \tag{3.4}$$

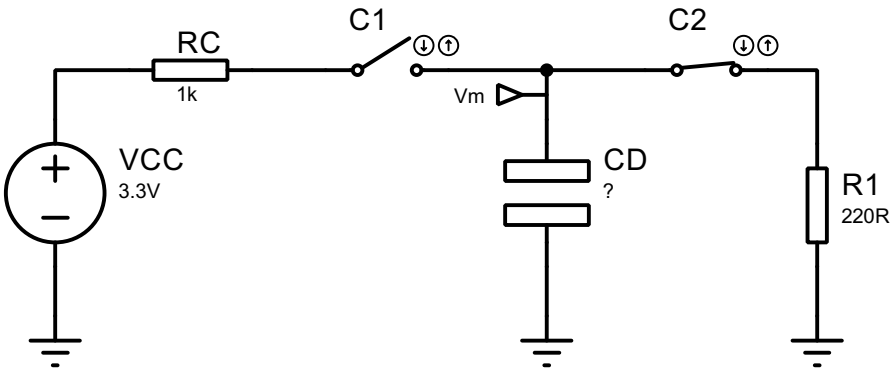
Sendo que:

R_d $\Rightarrow$ Resistor desconhecido (Ω);
R_{ref} $\Rightarrow$ Resistor de referência (Ω);
V_{CC} $\Rightarrow$ Valor da Fonte de tensão conhecida (V);
V_m $\Rightarrow$ Tensão medida (V);

Contudo, segundo Measurement Computing Corporation (2012), é necessário que o resistor de referência seja próximo ao valor do resistor desconhecido. Por este motivo, os ohmímetros possuem vários resistores de referência para determinar uma maior gama de valores.

3.2.4 Medição da Capacitância

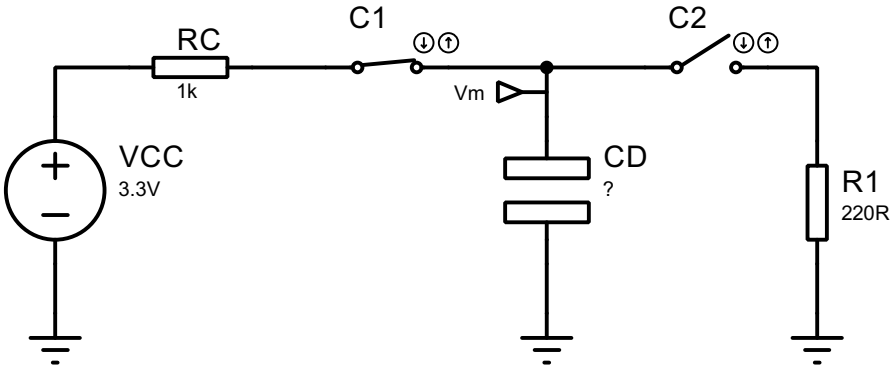
A técnica consiste em carregar e descarregar o capacitor por meio de uma fonte de tensão DC. A figura 3.2 representa a descarga de um capacitor desconhecido C_d com as chaves C1 aberta e C2 fechada, isolando os circuitos. Nesta primeira configuração, o capacitor descarregará até que a tensão em V_m seja igual à tensão de 0 V.



FONTE: Próprio Autor

Figura 3.2 – *Circuito de descarga do capacitor desconhecido.*

Depois, as posições das chaves serão comutadas conforme a figura 3.3, onde C1 fecha e C2 abre. Agora, o capacitor irá carregar em série com o resistor R_C, de modo que a tensão V_m está em função do tempo em um circuito RC (ALEXANDER; SADIKU, 2013). Obedecendo à seguinte equação:



FONTE: Próprio Autor

Figura 3.3 – *Circuito de carga do capacitor desconhecido.*

$$V_m = V_{CC} - V_{CC}e^{-t/R_C C_d} \tag{3.5}$$

Sendo que:

t ⇒ Tempo (s);
R_C ⇒ Resistor conhecido de carga do capacitor (Ω);
C_d ⇒ Capacitor desconhecido (F);
V_{CC} ⇒ Valor da Fonte de tensão conhecida (V);
V_m ⇒ Tensão medida (V);

Para o circuito de carga do capacitor, de acordo com Alexander e Sadiku (2013), a equação $R_C C_d = \tau$ é chamada de constante de tempo. Portanto, no instante em que $\tau = t$, temos:

$$V_m = V_{CC}(1 - e^{-1}) \approx 0,632 V_{CC} \tag{3.6}$$

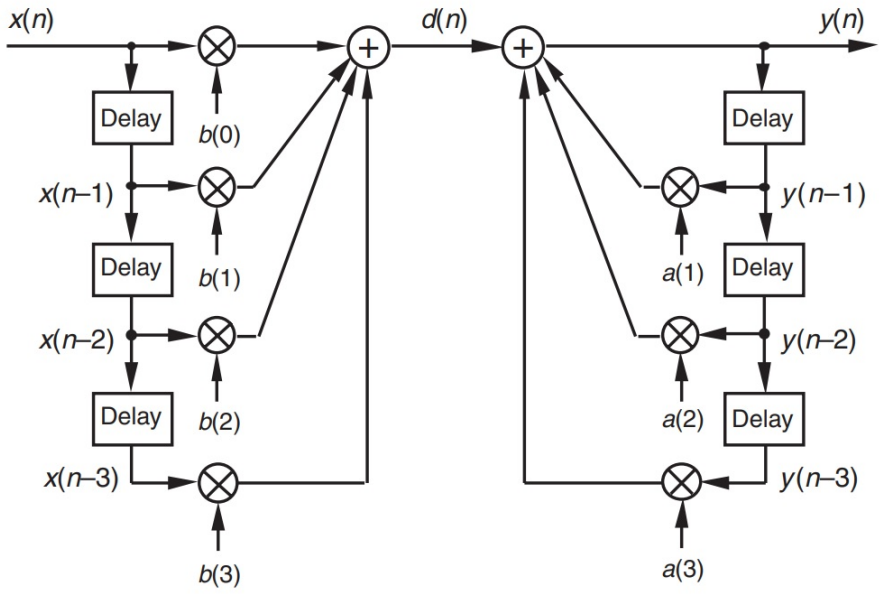
Partindo da equação 3.6, é possível determinar a capacitância, bastando conhecer os valores: do resistor de carga, da tensão inicial do capacitor e do tempo que o capacitor carrega de 63,2%. Este último é obtido por meio do SAC. Então, podemos deduzir que a capacitância será determinada pela seguinte equação:

$$C = \frac{t}{R} \tag{3.7}$$

3.2.5 Filtro IIR

Trata-se de um filtro digital de resposta ao impulso de duração infinita. Este nome foi dado, pois o filtro possui realimentação das amostras de entrada e saída. Portanto, após iniciar o filtro com amostras finitas não nulas, resultará em respostas infinitas não nulas. As vantagens de utilizar o filtro IIR, segundo Lyons (2011), são eficientes e rápidos, podendo ser empregado em tempo real em hardware. Contudo, necessitam ser bem projetados.

A figura 3.4 é o diagrama de blocos do filtro IIR, onde a entrada atual é representada por $x(n)$ e a saída atual por $y(n)$. Existe um tempo hábil entre as amostras atuais e as próximas amostras que, na prática, é o tempo de amostragem, denominado em inglês: *delay*. As amostras $x(n-1)$, $x(n-2)$ e $x(n-3)$ são as amostras



FONTE: Retirado de (LYONS, 2011) página 256

Figura 3.4 – *Diagrama de blocos do filtro IIR.*

de entrada anteriores tendo como referência a amostra atual $x(n)$. O mesmo vale para as amostras de saída. E os círculos em x representam multiplicação e em cruz soma. Por fim, "b"e "a"representam os coeficientes do filtro IIR. A seguir temos a representação do diagrama de blocos em equação matemática.

$$\begin{aligned} y(n) = \quad & b(0)x(n) + b(1)x(n-1) + b(2)x(n-2) + b(3)x(n-3) + b(4)x(n-4) \\ & a(1)y(n-1) + a(2)y(n-2) + a(3)y(n-3) + a(4)y(n-4) \end{aligned} \tag{3.8}$$

Esta equação é apenas multiplicação e soma, basta obter os valores dos coeficientes "a"e "b"e multiplicar com as respectivas amostras de saída (gerado pelo sistema) e entrada (fornecidas pelo sistema). O problema é o cálculo para determinar os coeficientes, visto que se não for calculado corretamente levará o sistema a instabilidade. Mas graças a programas como Matlab ou pacotes de programação como o Scypy, é possível projetar o filtro com certa facilidade. Basta estabelecer os critérios do filtro, como: o tipo, a banda de passagem e de corte, sendo calculados os valores dos coeficientes, bem como a ordem do filtro.

3.2.6 Transformada Rápida de Fourier (FFT)

A transformada discreta (DFT) determina o domínio frequência de uma série em função do tempo. O problema é que para calcular uma DFT de muitos pontos, é necessário um tempo considerado mesmo para computadores atuais. Segundo Lyons (2011), Cooley e Tukey desenvolveram o algoritmo da FFT otimizado para os sistemas de computadores. Para isto, foram alocados na memória valores que são recorrentes durante o cálculo da transformada e modificada a estrutura das operações matemáticas. Uma limitação do algoritmo é que a transformada permite apenas o número de amostras em potência de dois.

A estrutura mais simples é conhecida como radix-2, que reparte a DFT de tamanho N em duas metades. Partindo da DFT, é possível encontrar um valor constante recorrente nas equações chamado de *twiddle factor* W_N, definido por:

$$W_N = e^{-j2\pi/N} \tag{3.9}$$

O principal uso desta constante é simplificar as expressões, por meio das seguintes propriedades matemáticas:

$$W_{N/2}^{n(m+N/2)} = W_{N/2}^{nm} \tag{3.10}$$

$$W_N^{m+N/2} = -W_N^m \tag{3.11}$$

Dessa forma pode-se simplificar a DFT original em duas somatorias que contêm twiddle factor:

$$\begin{aligned} X(m) &= \sum_{n=0}^{N-1} x(n) e^{-j2\pi nm/2N} \\ &= \sum_{n=0}^{(N/2)-1} x(2n) e^{-=j2\pi(2n)m/2N} + e^{-j2\pi m/N} \sum_{n=0}^{(N/2)-1} x(2n+1) e^{-j2\pi(2n)m/2N} \\ &= \sum_{n=0}^{(N/2)-1} x(2n) W_N^{2nm} + W_N^m \sum_{n=0}^{(N/2)-1} x(2n+1) W_N^{2nm} \end{aligned} \tag{3.12}$$

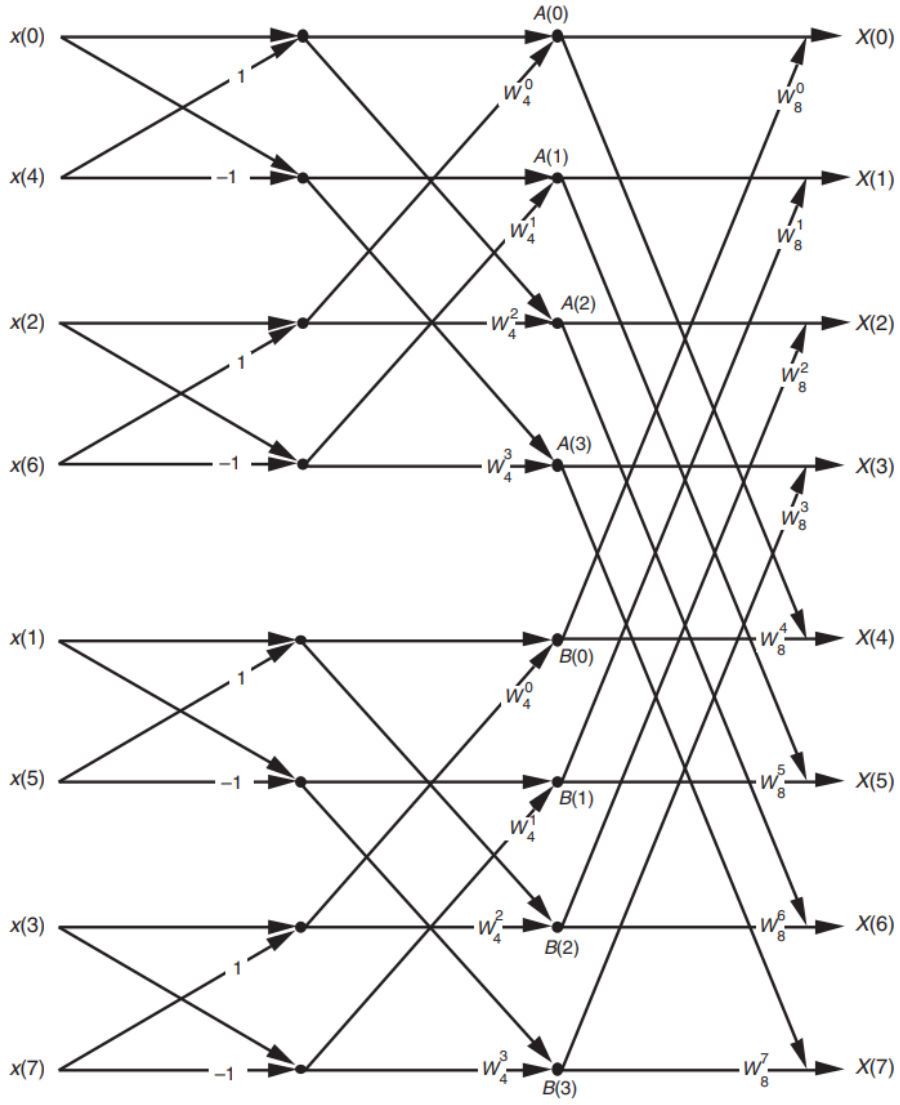
Agora basta aplicar as propriedades descritas nas equações 3.10 e 3.11 em 3.12, para então separar as amostras pares das impares da transformada, obtendo as

seguintes equações simplificadas:

$$X(m) = \sum_{n=0}^{(N/2)-1} x(2n)W_{N/2}^{nm} + W_N^m \sum_{n=0}^{(N/2)-1} x(2n+1)W_{N/2}^{nm}$$
$$= A(m) + W_N^m B(m) \quad (3.13)$$

$$X(m+N/2) = \sum_{n=0}^{(N/2)-1} x(2n)W_{N/2}^{nm} - W_N^m \sum_{n=0}^{(N/2)-1} x(2n+1)W_{N/2}^{nm}$$
$$= A(m) - W_N^m B(m) \quad (3.14)$$

Onde $A(m)$ e $B(m)$ são DFT's com quatro amostras. Esta equação final está organizada e é conhecida como estrutura borboleta.

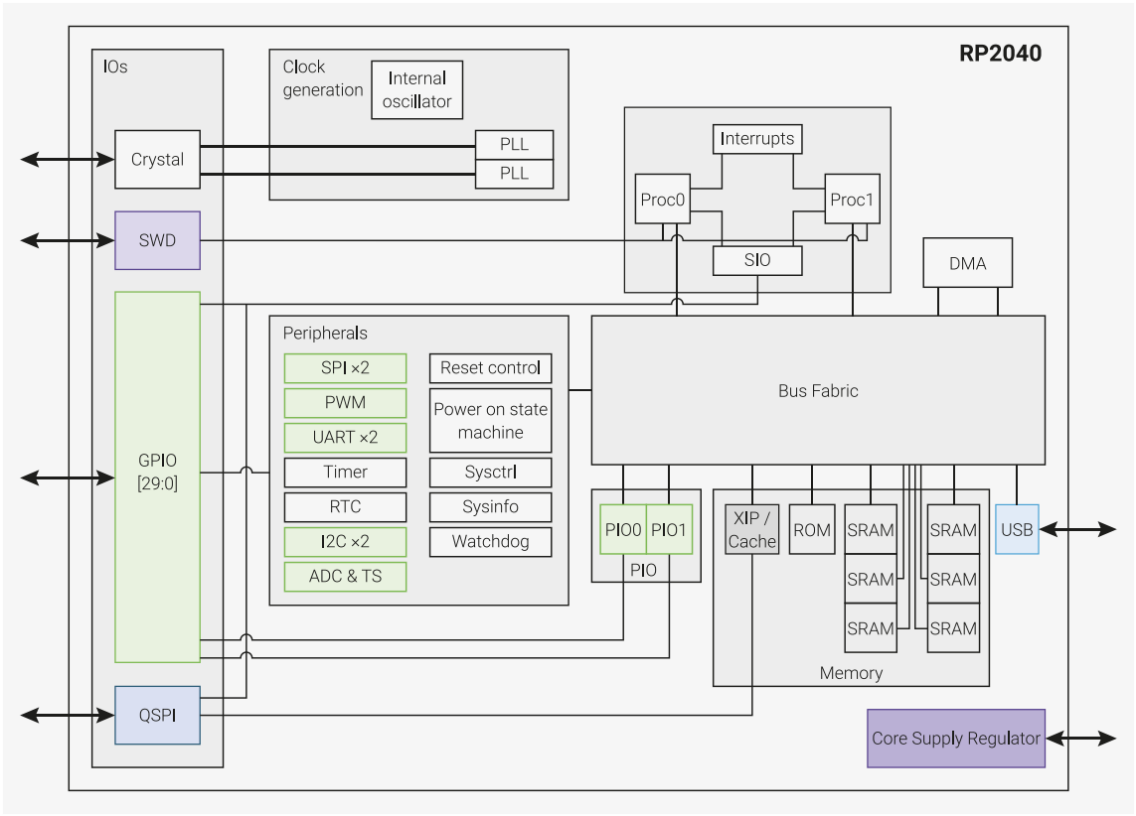


FONTE: Retirado de (LYONS, 2011) página 147

Figura 3.5 – *Exemplo de estrutura Radix-2 Borboleta de 8 amostras.*

A figura 3.5 representa esta estrutura, possui 3 etapas, sendo cada uma transformada de 2, 4 e 8 amostras. São salvos os valores constantes do twiddle factor e os resultados de cada etapa. Assim, economiz-se tempo de cálculo dos números complexos e etapas já calculadas.

3.3 Microcontrolador RP2040



FONTE: Retirado de (RASPBERRY PI, 2020b) página 11

Figura 3.6 – *Diagrama de blocos geral do RP2040.*

Foi escolhido este microcontrolador pelos componentes, familiaridade e documentação tanto da placa de desenvolvimento como do núcleo de processamento. A figura 3.6 é a visão geral do microcontrolador (ou MC) empregado no projeto. De acordo com a fabricante Raspberry Pi (2020b), o chip entrega alta performance, baixo custo e fácil utilização. Além disto, MC permite o desenvolvimento de softwares que interagem com o mundo físico.

Algumas de suas principais características presentes nos blocos da figura 3.6 são: dois núcleos de processamento, ADC de quatro canais, USB 1.1, 30 canais de multifunção, DMA. Além de possuir várias portas de uso geral com tipos de conexão multiplexado.

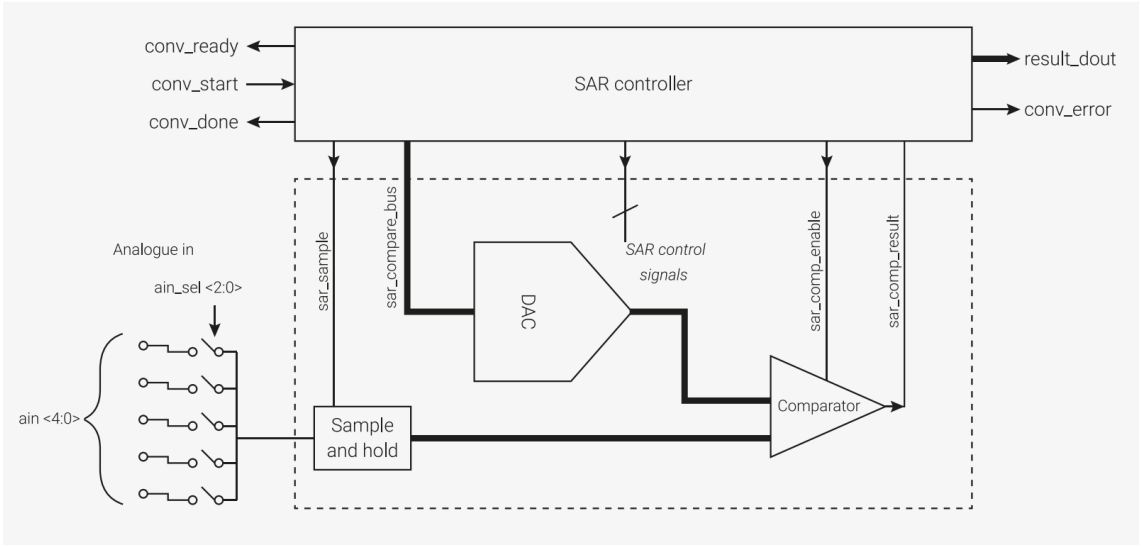
3.3.1 Acesso direto à memória (DMA)

Este hardware possui acesso mestre à escrita e leitura de dados conectados ao barramento de alta performance (Bus Fabric). Transmite uma quantidade de dados significativas, reduzindo o consumo de trabalho do processador. Para cada ciclo do processador do RP2040, o DMA pode fazer uma leitura e uma escrita de um tamanho de até 32 bits (RASPBERRY PI, 2020b). Pode-se configurar esta largura de banda em 32, 16 e 8 bits para os 12 canais independentes.

Existem muitas vantagens no uso desta arquitetura. Pode-se trabalhar sem utilizar os núcleos principais de processamento do RP2040. O DMA possui velocidade de transferência de dados elevada para escrita e leitura de memória.

3.3.2 Conversor Analógico Digital (ADC)

O RP2040 possui um ADC interno com registrador de aproximação sucessiva (SAR) de 12 bits. O conversor é um periférico do DMA, podendo ser utilizado em paralelo ao processador, armazenando as amostras na memória. Compartilha o oscilador de 48MHz com o USB e possui uma taxa de amostragem máxima de 500 ksps (RASPBERRY PI, 2020b). Sua entrada possui cerca de 1 pF de capacitância e uma impedância maior que 100 kΩ.



FONTE: Retirado de (RASPBERRY PI, 2020b) página 562

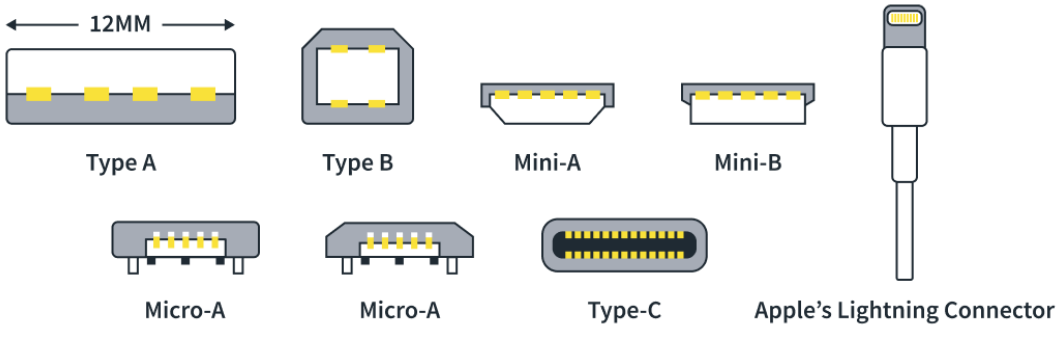
Figura 3.7 – *Diagrama de blocos da arquitetura SAR do ADC.*

A arquitetura do ADC é ilustrada na figura 3.7 e é composta pelos seguintes componentes: um controlador, um conversor digital analógico (DAC), um compa-

rador, controladores lógicos, registradores e um *sample and hold*. Este último lê a tensão e aguarda este valor na entrada do comparador até que o processo de conversão termine. Para isto, o controlador, inicialmente, zera os 12 bits do DAC. Depois, aciona o bit mais significativo (BMS) para 1, fazendo com que metade da escala do DAC seja utilizada (em escala de tensão do RP2040 2048). Desta forma, o comparador atua com as tensões do DAC e do sample and hold. Se a tensão for maior, o controlador aumenta o nível lógico do segundo BMS da escala para três quartos, senão muda a escala para um quarto. Por isso o nome aproximação sucessiva, pois esta operação ocorre até o bit menos sigificativo (LSB). A principal desvantagem é o tempo gasto até fazer as 12 comparações do DAC para cada amostra.

3.4 Protocolo USB

O padrão USB, popular nos computadores atuais, fez com que vários dispositivos eletrônicos diferentes o utilizem para transferência de dados. Surgiu para solucionar os problemas de conectividade nos anos 1990. Naquela época, cada dispositivo periférico do computador tinha seu próprio padrão, e para dispositivos especiais era necessária a instalação de componentes físicos no PC. O gerente de arquitetura I/O da Intel Bala Cadambi disse que um terço das scanners compradas, naquela época, foram devolvidas, pois os compradores não conseguiam instalar (INTEL CORPORATION, 1998). Situação em que, nos dias de hoje, basta apenas conectar os cabos USB e de alimentação que o dispositivo estará funcionando.



FONTE: Retirado de (SHAH, 2024)

Figura 3.8 – *Conectores USB que surgiram ao longo dos anos.*

A figura 3.8 mostra os diferentes conectores USB, (alguns descontinuados) os quais possuem diferentes velocidades e potências, dependendo do dispositivo e da compatibilidade do host. O protocolo USB foi construído baseado na hie-

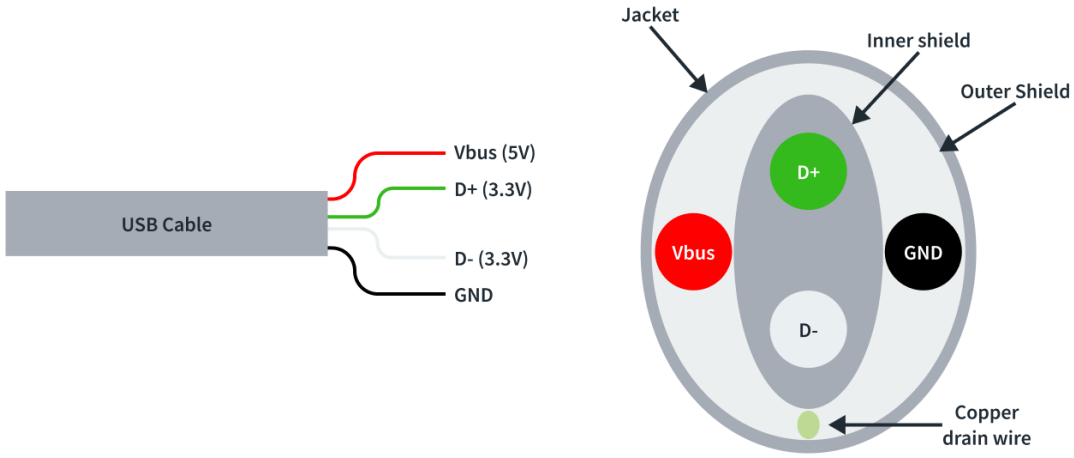
rarquia mestre e servo semelhante à I2C e SPI. Contudo, aceita apenas um host (equivalente ao mestre) que fica responsável por detectar novos dispositivos, gerenciamento de dados, energia e todas as demais atividades que ocorrem no barramento. Em uma mesma porta USB pode-se conectar até 127 dispositivos.

Tabela 3.1 – *Relação entre as velocidades do padrão USB e as distancia da conexão cabeada.*

Velocidade da USB	Máxima distância de cabo
Low Speed (1.5 Mb/s)	3m
Full Speed (12 Mb/s)	3m
High Speed (480 Mb/s)	5m
SuperSpeed (5 Gb/s)	3m
SuperSpeed+ (10 Gb/s)	2m

FONTE: Retirado de (SHAH, 2024)

Além da velocidade, no quesito potência elétrica, uma porta USB tem a tensão em torno de 4.4 a 5.25 V a depender dos dispositivos utilizados. E para o padrão 2.0, pode alimentar uma carga em até 500 mA. Já para o padrão de carregamento rápido, o USB-C pode entregar até 240 Watts.



FONTE: Retirado de (SHAH, 2024)

Figura 3.9 – *Composição do cabo USB.*

Os principais componentes que estão presentes no cabo USB são:

Vbus ⇒ Fornece uma tensão constante que pode está entre 4.4 à 5.25 V;
GND ⇒ Tensão de referêcia de aterramento;
D+ e D- ⇒ São cabos de dados com tensão de transmissão diferencial de 3.3 V;

3.4.1 Transmissão de dados USB

A transmissão feita pelos cabos de dados é diferencial para que o ruído elétrico seja reduzido. Assim sendo, quando D+ e D- estão com valores diferentes de tensão, ocorre a interação entre o dispositivo e o host. O sinal K é sempre oposto ao sinal J e para velocidade "Full Speed"são alternados os níveis lógicos destes dois sinais da tabela 3.2.

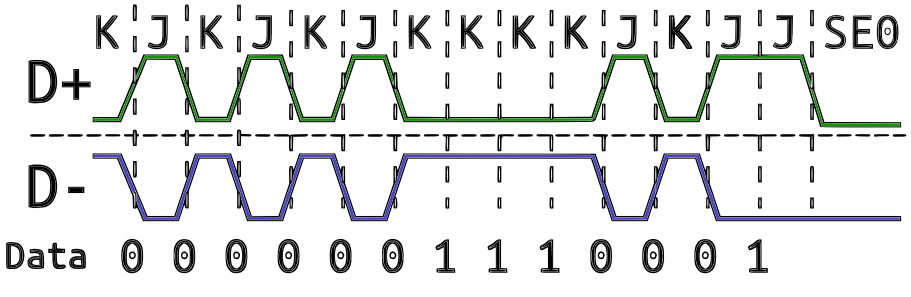
Tabela 3.2 – *Possíveis estados dos sinais de D+ e D- para velocidade "Low Speed"*

Sinal	Estado	D+	D-
J	Estado de espera ou nível lógico zero	low	high
K	nível lógico um	high	low
SE0	Fim de uma transação ou remoção do dispositivo	low	low
SE1	Situação impossível	high	high

FONTE: Retirado de (SHAH, 2024)

O USB funciona decodificando a alternacia de sinais K e J no padrão NZRI:

bit 0 ⇒ Ocorre quando o sinal alterna entre os sinais J depois K ou virse versa
bit 1 ⇒ Ocorre quando não há alternacia do sinal J ou K;



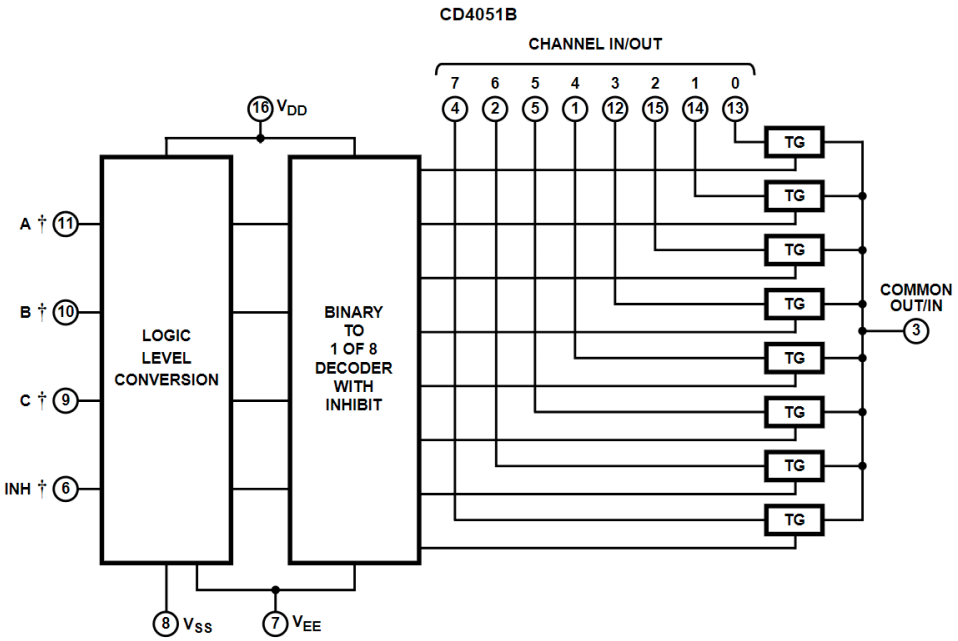
FONTE: Próprio Autor

Figura 3.10 – *Transmissão de modo diferencial "Full Speed" codificada em NZRI.*

A comunicação USB é half-duplex, ou seja, não se pode fazer transmissão e recepção ao mesmo tempo, e inicia-se sempre pelo host com os dispositivos respondendo a ele (SHAH, 2024). Os dados são transferidos por meio de um conjunto de informações chamado de *frame*, começando pelo LSB. Dentro do *frame* existem pacotes que estão sempre presentes nas transações e outros situacionais. Além disso, existem instruções de pacotes (a ordem mais baixa) que se inicia com o pacote de sincronização (*sync*) de 8 bits imutáveis de sequência: "KJKJKJKK". Depois,

o pacote de identificação (PID), o endereço do dispositivo (ADR), *endpoint* (EP), *payload data* (PD), o pacote de dados que é a informação da transmissão (DP) e a verificação cíclica de redundância (CRC) que verifica e corrige os erros na transmissão. Por fim, o pacote final que encerra a transação entre os dispositivos (EOP).

3.5 Multiplexador CD4051B



FONTE: Retirado de (TEXAS INSTRUMENTS, 1998) página 2

Figura 3.11 – *Diagrama de blocos do multiplexador CD4051B.*

A figura 3.11 apresenta a estrutura do multiplexador de 8 canais de entrada e 1 canal de saída. O controle é feito pelas entradas A, B e C e funciona com tensões entre 3 e 20 V. Além disso, por ser feito com transistores CMOS, possui baixa impedância com o canal ligado e alta com o canal desligado. Também possui uma entrada para restringir a conversão lógica (INH). A tabela da verdade 3.3 traz a combinação de 3 bits para as 8 portas disponíveis do chip. É um componente comum, barato e facilmente encontrado.

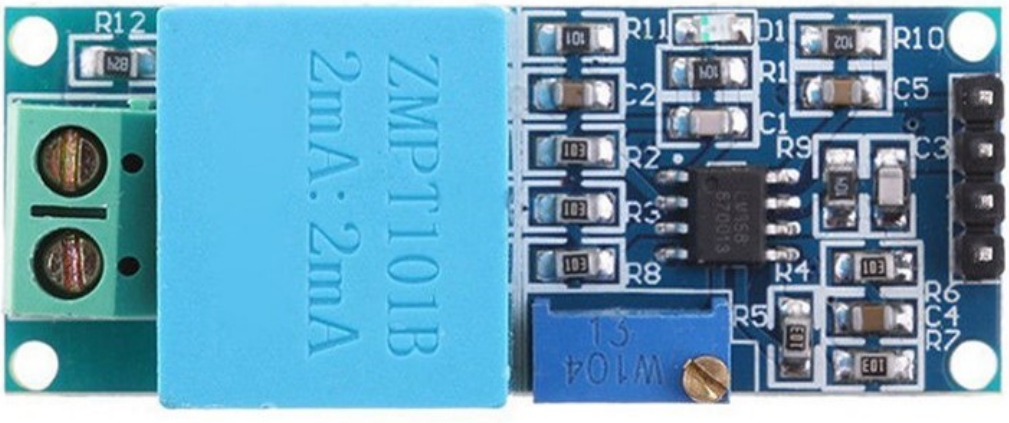
Tabela 3.3 – *Tabela da Verdade do multiplexador CD4051B.*

INH	C	B	A	Canal Ativo
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	X	X	X	Nenhum

FONTE: (TEXAS INSTRUMENTS, 1998) página 4

3.6 Modulo ZPTM101B

Como o ADC é limitado a 0 a 3.3 V, é necessário um circuito adicional para medir tensão alternada. Portanto, para mensurar a tensão da rede elétrica residencial com este projeto, é necessário transformar a tensão com uma relação próxima de 200:1. E ainda é preciso adicionar uma componente DC para o RP2040 ser capaz de ler a tensão. No mercado, existem módulos (circuitos compactos) prontos para esta finalidade, como o ZMPT101B, como mostra a figura 3.12, que é um transformador com dois amplificadores operacionais e um potenciômetro para ajuste na pcb.



FONTE: (CASA DA ROBÓTICA, 2024)

Figura 3.12 – *Modulo ZMPT101B capaz de converter até 700 V de pico à pico AC para tensões entre 1.5 à 4 V ajustável.*

O lado de alta do trafo é ligado na rede elétrica, já na parte de baixa o circuito

eletrônico deixa disponível a entrada para alimentar o amplificador (+5V e GND) e uma saída para o ADC.

3.7 Principio de funcionamento dos Aparelhos

3.7.1 Multímetro

O ohmímetro é o aparelho de medição da capacidade de um componente ou um circuito se opor à passagem de corrente elétrica. Temos a representação de um multímetro digital na figura 3.13 com as funções ohmímetro, capacímetro, etc; com suas pontas de prova do tipo agulha. Este último tem a conexão do tipo banana e serve para conectar o aparelho com os dispositivos elétricos. Existem vários tipos desse aparelho (miliohmímetro, megaohmímetro) podendo ser analógico ou digital.

O aparelho funciona ao injetar uma corrente conhecida no resistor ou circuito, então mede a tensão resultante e calcula a resistência por meio da Lei de Ohm ($V = IR$) (MADHU, 2020). Deve-se checar se o componente que se queira medir está em uso (energizado). Pois, neste estado não se pode aferir com o ohmímetro.

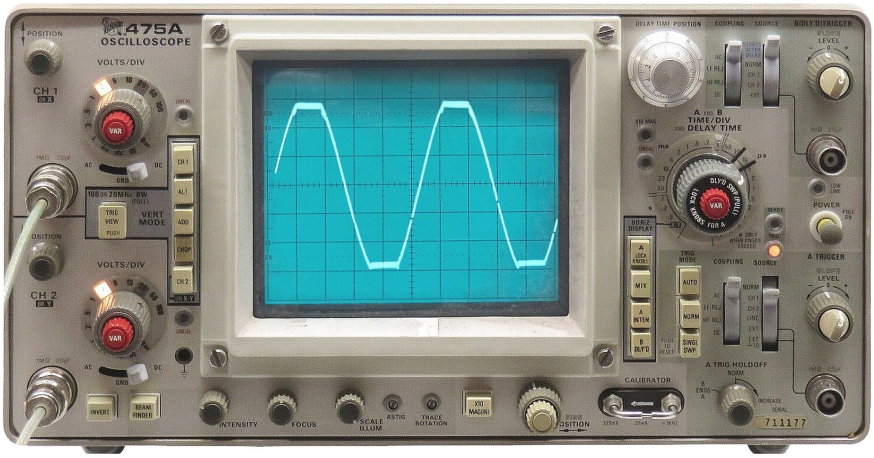
Já o capacímetro descarrega o capacitor e usa um resistor para carrega-lo, enquanto realiza medições da tensão no capacitor durante o período de carga e descarga utilizando um algoritmo para calcular a capacitância (ANG, 2022).

FONTE: Retirado de (MADHU, 2020)

Figura 3.13 – *Multímetro com funções de ohmímetro e capacímetro.*

3.7.2 Osciloscópio

Este é um aparelho de multiplos propósitos. Consiste em exibir gráficos cartesianos de sinais elétricos onde a tensão está em função do tempo. O osciloscópio, como representado na figura 3.14, são de uso geral. Mas também existem os específicos como de eletrocardiograma e de sistemas de ignição automotivo.



FONTE: (TEKTRONIX, 2016)

Figura 3.14 – *Osciloscópio analógico.*

Tanto os aparelhos digitais quanto os analógicos possuem alguns controles e características comuns. A tela, por exemplo, deve possuir as marcações de escala tanto como de tempo quanto de amplitude de tensão. Os principais controles são: horizontal, vertical e trigger (TEKTRONIX, 2016).

A seção horizontal ajusta a escala de tempo do aparelho que é dada por. segundos-por-divisão (Sec/Div) e a posição que é exibida no eixo X. Já para a parte vertical altera a escala do sinal e sua posição no eixo Y. O eixo está em volts-por divisão (volts/div). O controle é feito por botão (*knob*) de seleção no dispositivo. Por fim, o trigger é o controle que estabiliza os sinais na tela.

Similar ao multímetro, o osciloscópio conta com um par de sondas que permite a conexão do instrumento com outro aparelho. Desta vez, as pontas de prova possuem um botão seletor que altera o fator de atenuação ou não, podendo atenuar o sinal em até 10 vezes. Isto também ajuda a isolação da capacitancia do seu cabo no sinal que é analisado.

3.8 Trabalhos Relacionados

Aqui serão comentados alguns trabalhos que possua alguma relação com o tema e escolhido. Será apresentado um breve resumo dos trabalhos e, então, serão proposto melhorias dos trabalhos pesquisados.

Santos (SANTOS, 2019) defendeu em seu trabalho de final de curso o desenvolvimento e a prototipação osciloscópio digital, baseado em kit microcontrolador STM32, com a interface android e a comunicação por rede sem fio. Para isto, ele implementou um servidor utilizando Linux e programado em javascript com o intuito de fazer a ponte de comunicação entre o microcontrolador e um smartphone android. Segundo o autor, a escolha do dispositivo android é para diminuir os custos da prototipação. O autor defende também melhorias, tanto para o software quanto para hardware. Os principais pontos citados foram: aumento de funções do aplicativo, circuitos adicionais para aumentar a capacidade de leitura de tensão.

Duarte e Santos (SANTOS; DUARTE, 2013) apresentaram o projeto de um osciloscópio digital para sinais de até 4 MHz com suporte a filtros FIR e IIR em tempo real e análise espectral. O trabalho explica como utilizar o microcontrolador Arduino Uno para desenvolver um protótipo de osciloscópio com a interface desenvolvido no computador por meio do software Matlab. Graças a esta divisão de processamento, foi possível utilizar o Arduino somente para a obtenção e armazenamento dos dados, enquanto que o computador com maior poder computacional fazendo o uso do Matlab pode aplicar filtros digitais. Por este trabalho ter 10 anos desde sua realização, os materiais empregados estão defasados. Contudo, os fundamentos teóricos permanecem os mesmos, sendo assim, os metodos podem ser atualizados para estar coerentes à tecnologia atual.

Shou (SHOU, 2011) seguiu a mesma proposta do trabalho anterior, desenvolvendo um osciloscópio digital de baixo custo. Para isto utilizou o microcontrolador PIC18F4550 para obter as amostras e envia-las tanto para um cartão micro-sd para armazenar os dados, quanto para um display para a visualização, ou para o computador também para visualização por meio do ambiente de simulação. E por meio deste software foi feita a comparação do osciloscópio desenvolvido com o simulado.

Já Fernandes (FERNANDES, 2020) projeta um osciloscópio dividindo em dois dispositivos: o primeiro, o microcontrolador, faz a amostra de dados, e o segundo, um microprocessador, onde foi desenvolvido um software para interface homem-

máquina. Esta divisão ocorre pois o autor fez um estudo sobre o hardware do microcontrolador stm32 que possui um conversor analógico-digital com aproximação sucessiva. Outro diferencial deste trabalho é que foi empregado o estágio de condicionamento do sinal, uma vez que o ADC é limitado por amostrar uma tensão entre 0 à 3.3V. Então foi desenvolvida uma placa para o projeto ler tensões entre -12 à +12 V, sendo o estágio de entrada do sinal.

Os trabalhos comentados possuem semelhanças, entretanto, no presente trabalho propõe-se desenvolver vários equipamentos de medição em um único aparelho. Cabe ressaltar que o desenvolvimento do osciloscópio é o que demanda maior tempo de projeto tanto parte de software e hardware.

Capítulo 4

Desenvolvimento

4.1 Introdução

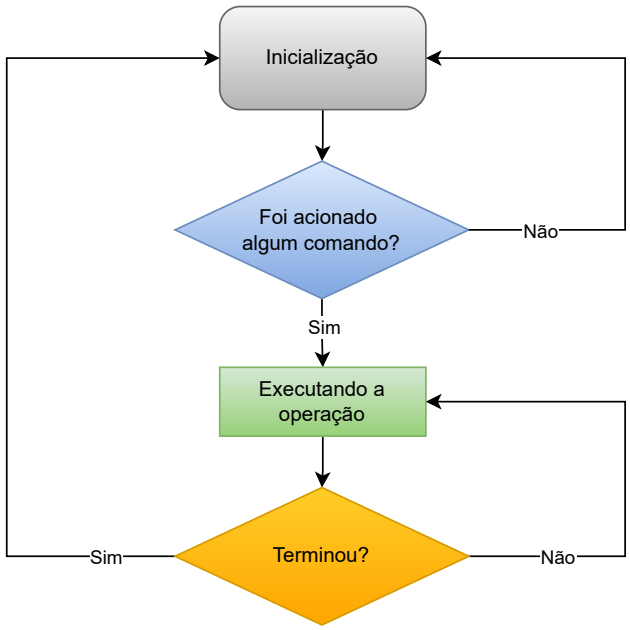
Esta seção aborda as funcionalidades dos códigos para o RP2040 e para a interface do celular bem como suas aplicações. Como grande parte do trabalho envolve programar para construir equipamentos de medição, então a teoria é empregada no código do algoritimo. Além disso, mais da metade dos códigos representa o desenvolvimento da interface em especial a parte dos gráficos. É explicado como é feita a interação entre os dois dispositivos. Também há a descrição sobre a prototipação do projeto, testes e resultados obtidos.

Como os códigos desenvolvidos ficaram extensos, é explicado apenas o funcionamento específico das ideias principais por trás de cada função. Entretanto, todo o código está disponível na página do Github do autor, presente no capítulo de apêndice.

4.2 Códigos para o RP2040

Foi utilizado o ambiente de desenvolvimento Arduino IDE junto à placa de desenvolvimento Raspberry Pi Pico para transcrever o código paro o RP2040. Esta IDE oferece suporte a linguagem de programação C e C++ e fornece várias bibliotecas de códigos abertos que agilizaram o desenvolvimento do projeto. Estas bibliotecas podem ser encontradas no site oficial do Arduino, páginas de hospedagem, ou em fóruns.

O diagrama da figura 4.1 apresenta a simplificação geral da programação cíclica e repetitiva. A inicialização é apenas o retorno para o início do loop, será posteriormente abordado a inicialização para cada função. Depois, o MC fica constantemente verificando se algum comando registrado foi acionado e determina distintas operações ou funções para cada comando. Existem operações que são interrompidas apenas por outro comando, que finaliza o ciclo e outros que são executados apenas uma vez. Por fim, o programa indica sempre quando recebe um comando e quando termina de executar. O programa foi pensado desta maneira para facilitar a interação entre o microcontrolador, a interface no celular e o usuário. Além de evitar possíveis erros e travamentos inesperados no RP2040, pois ele está, constantemente, verificando um novo comando ou está ocupado.



FONTE: Próprio Autor

Figura 4.1 – *Diagrama geral do funcionamento da programação*

O código 4.1 é a implementação do que foi explicado sobre o diagrama da figura 4.1, mas resumida para apenas um trabalho. Na função de loop (execução infinita) existe a primeira condição que verifica se chegou alguma informação na porta serial. Se não há informação a condição se repetirá, mas se existir, esta informação será salva na variável ”ch”. Os comandos no projeto são letras, neste caso, se for a letra ”O” maiúscula executará a função ”res”, responsável pelo cálculo da resistência. Será indicado um sinal no LED, da placa de desenvolvimento, para representar a duração de tempo desta função. O LED se apagará quando a função é encerrada e a programação volta novamente à primeira condição. A diferença

para a função loop no código original é que existem mais casos para as condições do *switch* para cada variável aceita por "ch". E não há chances de vir uma variável não listada, porque estão associadas diretamente há botões na interface não permitindo que o usuário faça modificação.

Código 4.1 – *Exemplo do Código representado no diagrama geral.*

```
void loop() {
    if (Serial.available()) {
        char ch = Serial.read();
        switch (ch) {
            case 'O':
                digitalWrite(led, HIGH);
                res();
                digitalWrite(led, LOW);
                break;
            ...}}}
```

4.2.1 Inicialização do ADC, DMA e PWM

Com auxílio das bibliotecas feitas por Philhower (2021) e Hoang (2021) é possível recorrer ao hardware DMA e modulação por largura de pulso (PWM) do RP2040 utilizando o Arduino IDE. Isto se deve ao MC só possuir suporte nativo a linguagem micropython, sendo necessário códigos adicionais (adaptar tipo "driver") para utilizar algumas de suas funções em C++.

Código 4.2 – *Código do uso da inicialização ADC e PWM utilizando o DMA.*

```
#include "RP2040_PWM.h"
#include <ADCInput.h>
RP2040_PWM* PWM_Instance;
ADCInput adc27(A2);
ADCInput mike(A3);
void setup() {
    adc27.setBuffers(12, 32);
    adc26.setFrequency(sfreq);
    adc27.setFrequency(sfreq);
```

```
10
11      PWM_Instance = new RP2040_PWM(pinToUse, freq, duty);
12      PWM_Instance->setPWM(pinToUse, freq, duty);
13      ...}
```

Primeiro foram inicializadas às duas bibliotecas e criado uma instância para o PWM. Já para o ADC são utilizados as últimas duas das quatro entradas e então iniciado utilizando todo o buffer disponível do DMA. Depois é configurada a porta, a frequência e taxa de trabalho para o PWM e a taxa de amostragem do ADC. São utilizadas variáveis para estas configurações, pois é necessário que se altere a taxa de amostragem (f_s) no osciloscópio. O PWM é utilizado para testar as sondas deste aparelho.

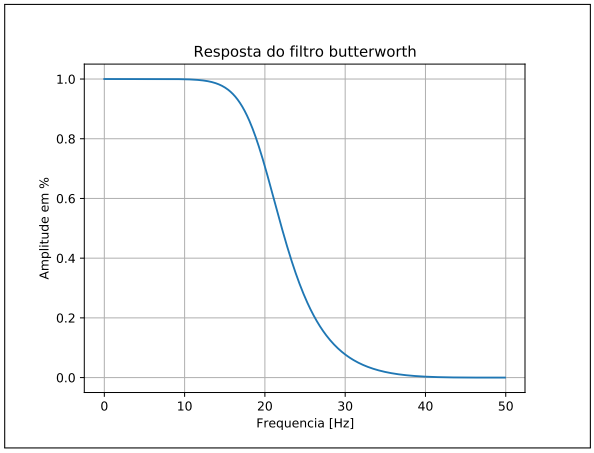
4.2.2 Projeto e algorítimo do Filtro IIR

O filtro foi projetado com a finalidade de amortecer as variações das amostras obtidas pelo ADC, obrigando com que os resultados oscilem menos. Portanto, para projetar esse tipo de filtro são utilizado ferramentas como o Matlab, ou bibliotecas matemáticas como o Scypy. Pois, o mínimo desvio ou a falta de aproximação leva a instabilidade deste tipo de filtro. Foi utilizado, no código 4.3, a importação da biblioteca Scipy e depois especificado as frequências de corte, amostragem e normalizada (”fc”, ”fs” e ”Wn”). Depois é definida a função que determina o design do filtro Butterworth (”butter”) com a ordem, a frequência normalizada e o tipo. Desta forma, são calculados os coeficientes da função de transferência do filtro e armazenado nas variáveis ”b” e ”a”.

Código 4.3 – *Código para calcular o filtro IIR com o desing Butterworth e tipo Passa-baixas.*

```
1 import scipy.signal as sig
2
3 fs = 100
4 fc = 20
5 Wn = 2 * fc / fs #Normalizar a frequência
6 b, a = sig.butter(4, Wn, btype='low')
7 z, p, k = sig.tf2zpk(b, a)
8 w, h = sig.freqz(b, a)
```

E para testar o filtro digital são calculado os polos, zeros, ganho e a resposta em frequência dos coeficientes "b" e "a", utilizando as funções "tf2zpk" e "freqz" para gerar gráficos da resposta e o plano z do filtro. Os resultados são mostrados nas duas figuras 4.2 e 4.3 a seguir. Os polos são representados com "x" e os zeros como pontos. O filtro está cortando em 20 Hz com a atenuação de -3dB.



FONTE: Próprio Autor

Figura 4.2 – *Resposta do filtro digital IIR com frequência de corte em 20Hz*

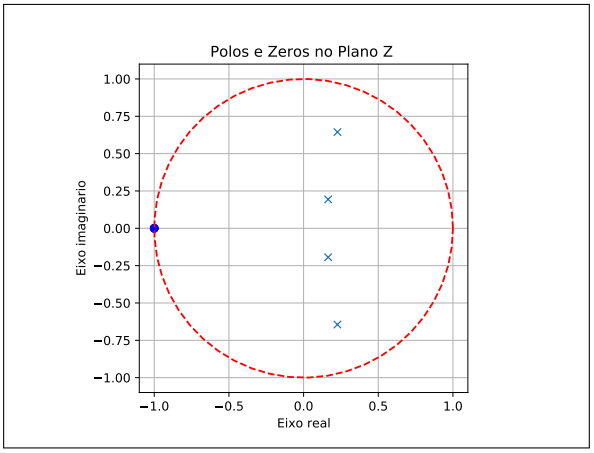


Figura 4.3 – *Polos e Zeros do filtro IIR Butterworth projetados*

A seguir o código 4.4 é a função do filtro em uso no projeto. Ele é chamado nas funções do ohmímetro e capacímetro. Inicialmente é configurado a frequência de amostragem (f_s) em 80 kHz do ADC e então ele é iniciado com o DMA. São declarados os valores dos vetores a e b, sendo os coeficientes previamente obtidos. E os vetores x e y que são a entrada e saída do sistema, respectivamente, zerados. Depois é feito um loop para o cálculo de 200 amostras de tensão do ADC no filtro utilizando a equação 3.8. Finalmente o valor filtrado da tensão é obtido e é desativado o DMA.

Código 4.4 – *Código do uso do filtro digital com os coeficientes a e b calculados anteriormente.*

```
1  float filtro() {
2    adc26.setFrequency(80000);
3    adc26.begin();
4    float x[5] = { 0, 0, 0, 0, 0 };
5    float y[5] = { 0, 0, 0, 0, 0 };
6    float b[5] = { 0.04658291, 0.18633163,
7    0.27949744, 0.18633163, 0.04658291 };
8    float a[5] = { 1., 0.7820952, -0.67997853,
9    0.1826757, -0.03011888 };
10   for (int i = 0; i < 200; i++) {
```

```
11      x[0] = mike.read();
12      y[0] = b[0] * x[0] + b[1] * x[1] + b[2] * x[2] +
13      b[3] * x[3] + b[4] * x[4] + a[1] * y[1] + a[2] * y[2]
14      + a[3] * y[3] + a[4] * y[4];
15      x[4] = x[3]; y[4] = y[3]; x[3] = x[2]; y[3] = y[2];
16      x[2] = x[1]; y[2] = y[1]; x[1] = x[0]; y[1] = y[0]; }
17    adc26.end(); return y[0];
18  }
```

4.2.3 Código do Ohmímetro

Primeiro foi declarado as saídas de controle para o multiplexador CD4051B, mais os valores medidos dos resistores de referências apelidados de R_{ref} no código 4.5. Se faz necessário o uso destes resistores, pois existe variedade de valores de resistência muito superior à escala limitada de apenas 4095 graduações para o ADC. Com 3 bits é possível registrar as saídas do MC que seleciona cada uma das sete R_{ref} conectadas ao multiplexador.

Código 4.5 – *Código para as saídas para o controle e valores medidos das resistências de escala*

```
1 const byte rSelPins[3] = { 13, 14, 15 };
2 const int rRef[7] = { 219, 988, 5519, 10050, 99900,
3 563000, 757000 };
```

A função do código 4.6 do Ohmímetro tem início com um laço que irá obter os valores de tensão para cada R_{ref} conectado individualmente ao resistor desconhecido R_d (que se queira medir). Existe também uma saída para inibir o chaveamento do CD4051B conhecido como INH, além das 3 para selecionar um dos seus oito canais. Esta restrição serve para manter a estabilidade do componente. Depois, é chamado a função do filtro descrita no algoritmo 4.4 e é atribuída a variável "cOut". Por fim é desativado o CD4051B.

Código 4.6 – *Parte do algoritmo para o cálculo da resistência elétrica*

```
1 void res() {
2   for (byte count = 0; count < NUM_REF_RESISTORS; count++) {
```

```
3       digitalWrite(rSelPins[0], count & 1); // A
4       digitalWrite(rSelPins[1], count & 2); // B:
5       digitalWrite(rSelPins[2], count & 4); // C: MSB
6       digitalWrite(enableMux, LOW);
7       cOut = filtro();
8       digitalWrite(enableMux, HIGH);
9       ... }}
```

Após obter a amostra filtrada é utilizado o código 4.7 desenvolvido a partir da equação 3.4, faz a conversão para um valor da resistência elétrica. Mas antes o resistor de referência escolhido no laço é adicionado a impedância resistiva interna do CMOS do componente. Esta impedância do multiplexador é encontrada no datasheet. Então após adquirir o valor resistivo utilizando todas as R_{ref}, chegou a hora de comparar qual destes valores mais se aproxima da metade da escala do ADC.

Código 4.7 – *algoritmo para converter a amostra de tensão e relacionar ao valor da resistência do Resistor desconhecido*

```
1 ...
2       rR = rRef[count] + SWITCH_RESISTANCE;
3       rX = (rR * cOut) / (MAX_ANALOG_VALUE - cOut);
4 ...
```

É calculado a diferença entre a metade da escala 2048 do ADC subtraído do valor da tensão "cOurt", e é chamado de delta genérico. Com os deltas genéricos de todas as amostras de cada R_{ref} é possível encontrar os mais próximos. Ao utilizar o algoritmo 4.8 são obitidos os melhores dois deltas: o menor e o maior valor mais próximos ao valor médio do ADC como referência.

Código 4.8 – *Codigo para determinar qual amostra fica mais proxima da metade da escala do ADC*

```
1 ...
2       delta = (MAX_ANALOG_VALUE / 2.0 - cOut);
3       if (fabs(delta) < fabs(delta1)) {
4           delta2 = delta1;
5           r2 = r1;
```

```
6           delta1 = delta;
7           r1 = rX;
8         } else if (fabs(deltaBest2) > fabs(delta)) {
9           delta2 = delta;
10          r2 = rX;
11      }
12  ...
```

Existe a possibilidade de se medir um R_d que seja maior que todas as R_{ref} ou menor. Neste caso só podemos ter um delta válido mais próximo da metade da escala do ADC. Neste caso será computado apenas um delta e é obtido o valor da R_d a partir dele. Mas quando o valor de R_d está entre os valores das R_{ref} teremos dois deltas. Neste último caso é feito uma interpolação, no código 4.9, dos dois valores de delta e então é enviado para a interface o valor de R_d.

Código 4.9 – *Código para a aproximação e envio do valor de* R_d

```
1  ...
2      if (r1 >= 0 && r2 >= 0) {
3          if (delta1 * delta2 < 0) {
4              rX = r1 - delta1 * (r2 - r1)
5              / (delta2 - delta1);
6          } else {
7          rX = r1;
8      }}
9   Serial.println(rX);
```

4.2.4 Código do Capacímetro

Inicialmente é necessário descarregar o capacitor antes de conectar ao aparelho. Utiliza-se o resistor de descarga de 220 Ω e para a carga recorre ao resistor emprestado do ohmímetro de 1 kΩ. O código 4.10 funciona da seguinte forma, o multiplexador é iniciado e o capacitor desconhecido C_d é descarregado. Quando a tensão do C_d é muito baixa é iniciado um timer, depois é carregado o capacitor até que atinja 63,2% da tensão que é equivalente o produto ($R_C C_d$) resistor de carga

vezes o capacitor desconhecido que é equivalente a τ. Neste momento o timer é finalizado e obtido este produto e armazenado na variável "T". Como a unidade do temporizador é microssegundo o valor da capacitância obtida será a mesma ordem só que em microfarad. Ao utilizar a equação 3.7 é obtido o valor de C_d e então enviado para a interface.

Código 4.10 – *Código geral para o uso do capacímetro com o multiplexador.*

```
void vou() {
  digitalWrite(enableMux, LOW);
  digitalWrite(rSelPins[0], HIGH);
  digitalWrite(rSelPins[1], HIGH);
  digitalWrite(rSelPins[2], HIGH);
  while (filtro() > 20) {
  }
  float time_d = micros();
  digitalWrite(rSelPins[0], LOW);
  digitalWrite(rSelPins[1], HIGH);
  digitalWrite(rSelPins[2], LOW);
  while (filtro() < 2588) {
  }
  float time_c = micros();
  digitalWrite(enableMux, HIGH);
  float T = (time_c - time_d);
  float mf = T / rRef[1];
  Serial.println(mf);
}
```

4.2.5 Código do Osciloscópio

Como o tratamento de dados no osciloscópio é realizado na interface, o RP2040 só utiliza o DMA em conjunto ao ADC para obter as amostras. Por isto o código da função 4.11 é simples. Inicia o ADC e então enquanto o mesmo faz cerca de 300 amostras e envia para a interface por meio da USB, o LED da placa de desenvolvimento fica aceso. Depois, desliga o LED e o DMA. Esta função é chamada sempre que necessário compor o sinal no osciloscópio.

Código 4.11 – *Código da função do osciloscópio.*

```
1 void osc() {
2   adc27.begin();
3   digitalWrite(led, HIGH);
4   for (int i = 0; i < limit; i++) {
5     Serial.println(adc27.read());
6   }
7   adc27.end();
8   digitalWrite(led, LOW);
9 }
```

A função do trigger utiliza-se o mesmo código do osciloscópio acrescido de um laço e é representado no código 4.12. Este laço fica entre as linhas dois e três do código 4.11 do osciloscópio. Como os sinais que se queira analisar são periódicos, é possível utilizar algoritmo que detecta a alternância do sinal. Para isto, faz necessário especificar qual valor de tensão que ocorre esta mudança no sinal. O osciloscópio é possível selecionar qual nível de tensão que ocorrerá o ”trigger”. Já o do projeto como o ADC do RP2040 só consegue medir até 3.3 V, foi utilizado novamente a faixa da metade da escala entre 1.65 V para colocar o trigger. Isto permite que o código seja simples apenas com uma linha. Porém, impossibilita a alteração vista nos osciloscópios tradicionais.

Código 4.12 – *Código adicional para o uso do trigger do osciloscópio.*

```
1 while (adc27.read() < 2040 || adc27.read() > 2050) {}
```

4.2.6 Código da FFT

Graças ao auxílio da biblioteca desenvolvida por Condes e Overbohm (2020) é possível realizar FFT do tipo radix 2 com a possibilidade de alterar o número de amostras com facilidade, acelerando o desenvolvimento do projeto. Portanto, no código 4.13 é, inicialmente, declarado a classe ”FFT” e para isto é necessário declarar os valores rais e imaginários, o número de amostras e a taxa de amostragem.

Código 4.13 – *Código da função de FFT do osciloscópio.*

```
ArduinoFFT<float> FFT
= ArduinoFFT<float>(vReal, vImag, samples, sfreq);
```

Já para a função da FFT tem o começo semelhante ao do código do osciloscópio, amostra o sinal por meio do ADC atribuindo a variável do valor Real "vReal" e atribui zero em todo vetor de valores imaginários. Posteriormente, é desativado o ADC e é utilizado os comandos da biblioteca para o cálculo da transformada. Após isto, "vReal" as amostras de vReal estão no domínio da frequência e são enviadas a interface. Como a interface que controla a taxa de amostragem não é necessário o envio desta informação para criação da parte gráfica.

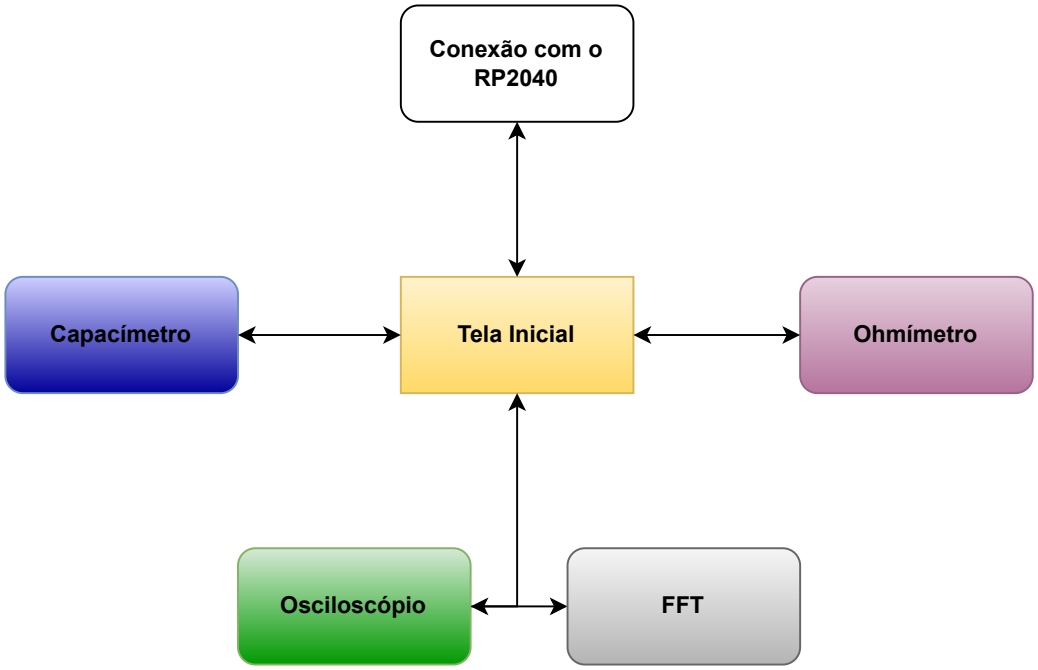
Código 4.14 – *Código da função de FFT do osciloscópio.*

```
void analiza() {
  adc27.begin();
  for (int i = 0; i < samples; i++) {
    vReal[i] = adc27.read();
    vImag[i] = 0;
  }
  adc27.end();
  FFT.windowing(FFTWindow::Hamming, FFTDirection::Forward);
  FFT.compute(FFTDirection::Forward);
  FFT.complexToMagnitude();
  for (int i = 0; i < samples; i++) {
    Serial.println(vReal[i]);
}}
```

4.3 Códigos da Interface

Para a interface foi desenvolvido um aplicativo para celular utilizando o ambiente de desenvolvimento Android Studio. Este é um aplicativo que permite diversos kits de desenvolvimento em código aberto. Um deles é o Flutter que recorre à linguagem de programação Dart com o foco em desenvolver interfaces interativas. Todos estes recursos foram desenvolvidos pela Google e distribuído como código aberto e de livre acesso.

A principal vantagem do Flutter é que permite desenvolver para as plataformas: Android, IOS, macOS, Linux e Windows. Porem, isto depende, também, da acessibilidade no desenvolvimento. Por exemplo, neste trabalho só foi possível desenvolver para Android, porque parte do código não possui suporte para demais plataformas.



FONTE: Próprio Autor

Figura 4.4 – *Digrama das telas do programa do celular*

O programa é divido no total em 6 páginas, quadros ou telas: inicial, conexão, ohmímetro, capacímetro, osciloscópio, FFT. A tela inicial é a principal tela de acesso para as demais, sempre é necessário voltar a ela para acessar as demais. É simplesmente composta por botões que alteram para a tela selecionada. Serão tiradas fotos para explicar os componentes de cada tela da interface.

Além dos códigos de cada tela existe o modelo, é por meio dele que será possível fazer toda a comunicação e conversão de dados com RP2040. O modelo é um conjunto de códigos que permitem classes, funções e variáveis globais. O motivo disto é que cada tela que é iniciada no flutter não salva os dados registrados das demais, é como se existe vários programas independentes. Ou seja, sem o código do modelo para interliga as páginas, cada tela teria que solicitar a conexão com o MC toda vez.

Apenas as figuras do osciloscópio e do FFT que não poderiam ser simuladas,

pois requerem conexão ao RP2040. Para estas duas telas serão tomadas screenshots de um celular na versão do Android 9. As demais foram retiradas de um aparelho simulado com a versão do Android 11 no ambiente Android Studio.

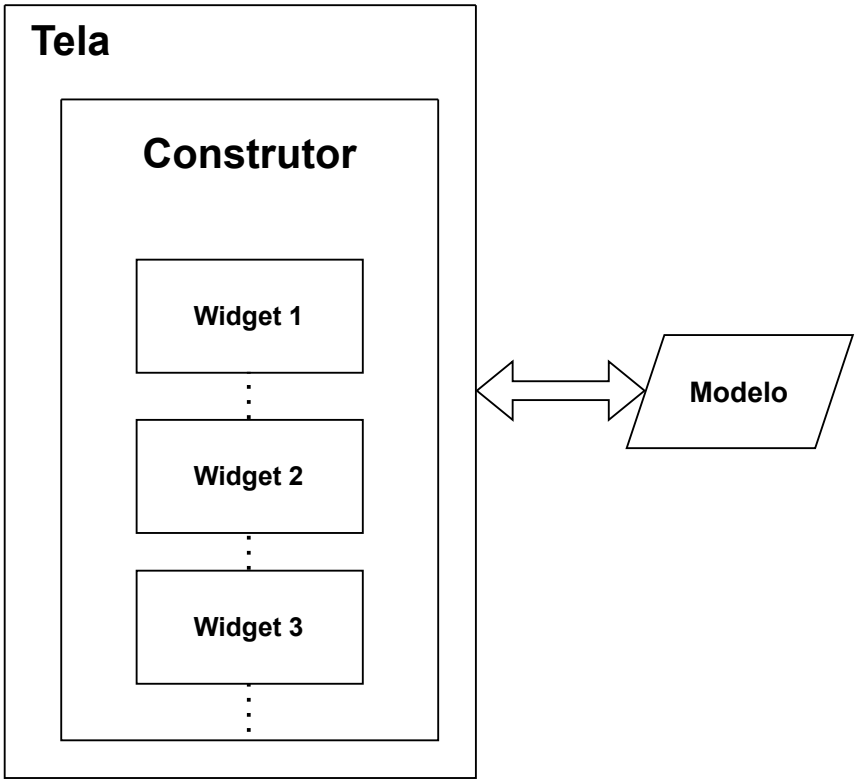
A organização dos códigos para cada tela no flutter segue a mesma organização para todas as telas representado na figura 4.5. O widget é uma função especial do Flutter, é uma ferramenta que possui diversas aplicações e pode ser composta por: animações, imagens, listas, barras de tarefas, ícones interativos, botões, etc. É possível criar seu próprio pacote de widgets e existem milhares destes códigos disponíveis gratuitamente em repositórios no pub.dev ou no Github. No projeto os widgets são geralmente empregados como botões e textos. Eles ficam contidos no construtor que é uma maneira de organizar os widgets na tela, em formato de matriz, customizado livremente. O modelo oferece dados para a tela, já que não existem variáveis globais, e os widgets contido na tela também pode modificar as informações do modelo. Desta forma, as informações contida em cada tela pode ser compartilhada para as demais pelo modelo. Mas no projeto o modelo é utilizado para a comunicação com o MC por meio do código para USB.

Por fim, nesta seção é apresentado os códigos mais importantes de cada tela e explicar como ela é utilizada.

4.3.1 Código da Tela de Início

A tela de início da figura 4.6 possui apenas 5 botões e um título do equipamento foi nomeado de ”MultEquip”. A lupa é um botão e é necessário que clique nele para conectar ao equipamento antes de utilizar qualquer uma das 4 funções disponíveis. A organização da tela se dar pelo construtor que organiza em uma coluna os 4 botões. É recomendado orientar o celular em modo paisagem, mas pode ser utilizado em modo retrato.

Todas as telas possuem um widget chamado ”AppBar” ou barra de tarefas do aplicativo a utilização com mais funcionalidades é a da tela inicial descrito no algoritmo 4.15. Primeiro foi centralizado o título e adicionado o texto do título ”MultEquip”. Depois é definido a ação como um botão ícone a lupa da figura 4.5 da tela inicial. Este widget possui a função de navegação que direciona para a tela de conexão que é um ”builder” construtor de outra tela e a executa. Portanto, acaba por fechar a tela inicial e abre a tela de conexão. Por fim, foi adicionado uma ”tooltip” é uma dica de texto ao segurar este botão.



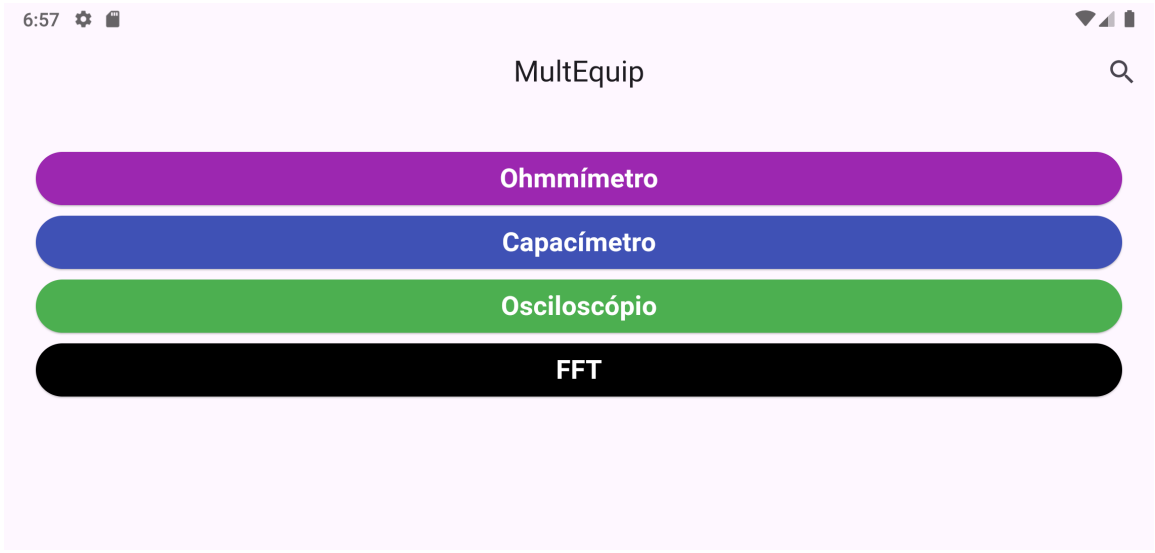
FONTE: Próprio Autor

Figura 4.5 – *Organização da estrutura de código para cada tela.*

Código 4.15 – *Código da barra de tarefas do aplicativo.*

```
appBar: AppBar(
    centerTitle: true,
    title : const Text('MultEquip'),
    actions: <Widget>[
        IconButton(icon: const Icon(Icons.search),
            tooltip: 'Próxima página',
            onPressed: () {
                Navigator.of(context).push(MaterialPageRoute
                (builder: (context) => const Connect()));
                },),],), ...
```

Para os botões todos utilizam a mesma função só altera a navegação para a página diferente e o texto interno. No código 4.16 é utilizado o botão do tipo ressaltado "ElevetedButton" que possui relevo e animação. Quando pressionado



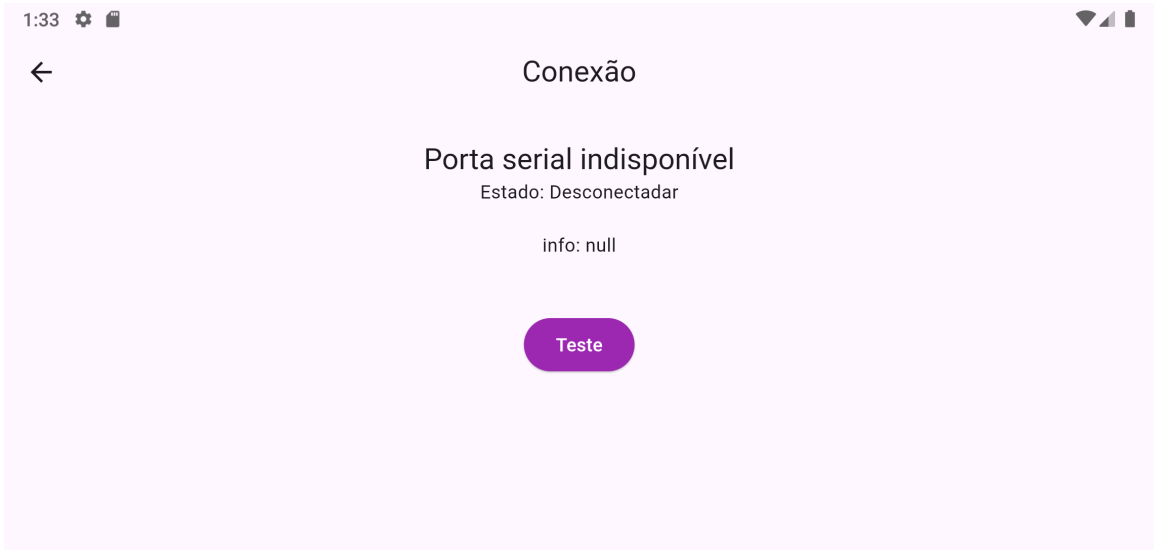
FONTE: Próprio Autor

Figura 4.6 – *Tela inicia do aplicativo deste trabalho.*

executa a função de navegação e troca a tela para outra neste caso do Ohmímetro, pode-se utilizar a estilização para modificar tudo no botão: cores, fonte, animação, etc. A linguagem é baseada em parentesco, por exemplo: o filho do botão é um texto e seu pai é um construtor.

Código 4.16 – *Código dos Botões.*

```
1 ...
2 ElevatedButton(
3     onPressed: () {
4         Navigator.push(context,MaterialPageRoute
5             (builder: (context) => const Ohm()));},
6         style: ElevatedButton.styleFrom
7             (backgroundColor: Colors.purple),
8         child: const Text('Ohmmímetro',
9             style: TextStyle(
10                fontSize: 20, fontWeight: FontWeight.bold,
11                color: Colors.white))), ...
```



FONTE: Próprio Autor

Figura 4.7 – *Tela de conexão USB.*

4.3.2 Código da Tela de Conexão

Para a conexão USB foi necessário fazer o uso de um pacote desenvolvido por Bessems e Herranz (2023) dada a complexidade e tempo disponível. O Android Studio e Flutter permite a integração com outras linguagens de programação para o desenvolvimento, este pacote em especial foi desenvolvido em Java, inicialmente, e então portado.

A figura 4.7 apresenta a tela de conexão e a organização também é centralizada e estruturada em uma coluna. Possui o título do nome da tela e textos que indica qual dispositivo está disponível, o estado da conexão e informações adicionais ofertada pelos endereçamentos do padrão USB. No final tem um botão para testar a conexão com o RP2040 ,ao aperta-lo, envia ao servo uma informação e recebe outra indicando que ocorreu a comunicação das duas vias.

Já para o código inicia com o ”AppBar” contendo apenas o título e o botão seta na parte superior no canto esquerdo. Este botão indica que esta é a tela filha da tela inicial, e ao aperta-lo volta ao início novamente, todas as telas filhas possui esta funcionalidade sem a necessidade de programa-la.

Código 4.17 – *Código da página conexão USB.*

```
1 ...
2 Text(model.ports.isNotEmpty ? "Porta serial disponível"
3     : "Porta serial indisponível",
```

```
4       style: Theme.of(context).textTheme.titleLarge),
5       ...model.ports,
6       Text('Estado: ${model.status}\n'),
7       Text('info: ${model.port.toString()}\n'),
8       Text(_value ?? ""),
9       ElevatedButton( style: const ButtonStyle(
10          backgroundColor: WidgetStatePropertyAll
11              (Colors.purple)),
12          onPressed: () async {
13              _value = await model.transacao(98, 1);},
14          child: const Text('Teste',
15              style: TextStyle(color: Colors.white),)),...
```

Já o código 4.17 descreve apenas a parte dos textos e botão da página. Se faz uso do operador ternário "?" para executar escolhas com o texto. Este operador funciona da seguinte forma, primeiro estabelece a condição se for verdadeiro o primeiro texto fica disponível senão o segundo texto, neste caso indica se o serial está ou não disponível. Os proximos textos possuem constantes e variáveis, que são iniciados com $, e são retiradas do modelo. Outro operador para o texto utilizado é o verificador de nulidade "??" as variáveis precisam ser verificadas caso seja do tipo nulo. E sua utilidade é para que a variável fique visível como texto no quadro, caso não seja nula.

Ao pressionar o botão é iniciado uma função assincrona (que não é executada imediatamente) que realiza a trasação com o servo RP2040 e o resultado é atribuido a variável local "_value". Esta função transação aceita dois argumentos o primeiro é a conversão decimal em ASC2 e o segundo é a deadline em milisegundo. Para a comunicação com o servo foi empregado, apenas, números decimais em ASC2, sendo: letras maísculas, minusculas e simbolos numéricos. Assim é possível enviar informações que serão executadas rapidamente pelo servo e retorna o resultado ao host. A informação chega ao MC como um caractere enquanto a resposta para o aplicativo de interface (ou host) recebe já esta em string.

4.3.3 Código do Modelo

Várias variaveis do código 4.17 são da classe ”model”, por exemplo ”model.ports” e esta é a classe que contém todas as funções globais do sistema. Este código não possui interface mas costroi uma lista na página da conexão USB utilizando o construtor ”ports”.

A função do código 4.18 inicia esvaziando o vetor construtor ”ports” e é feito uma busca de todos os servos disponíveis pela USB. Então é checado caso algum dispositivo for desconectado. A função ”notifyListernerts” atualiza os valores modificados até então para todas as páginas do código. Depois, é feito um laço para adicionar todos os dispositivos em ”ports” no formato de lista. Nesta lista contêm: o nome do dispostivo, o seu fabricante, e um botão. Estas informações são obrigatórias para as transações no padrão USB ocorrerem. Já o botão possui uma condição se o dispositivo estiver conectado é escrito desconectar, senão aparece conectar, e realiza estas duas funções.

Código 4.18 – *Código da página conexão USB.*

```
void getPorts() async {
    ports = [];
    devices = await USBSerial.listDevices();
    if (!devices.contains(devicer)) { connectTo(null);}
    notifyListeners();
    for (var device in devices) {
      ports.add(ListTile(
          leading: const Icon(Icons.USB),
          title : Text(device.productName!),
          subtitle : Text(device.manufacturerName!),
          trailing: ElevatedButton(
            child:
                Text(devicer == device ?
                "Desconectar" : "Conectar"),
            onPressed: () {
                connectTo(devicer == device ? null
                : device).then((res) {getPorts();});
            },
          )));}notifyListeners();}...
```

4.3.4 Código do Ohmímetro e Capacímetro

Foi reutilizado o código da página do ohmímetro para o capacímetro. Porque nas duas interfaces só precisa aparecer o valor do componente que se queira medir. As diferenças são alguns detalhes dos títulos e unidades de medida, mas o algoritmo empregado para compor a interface é o mesmo.

FONTE: Próprio Autor

Figura 4.8 – *Página do Ohmímetro que é semelhante ao Capacímetro*

A tela da figura 4.8 é composta por um ”Appbar”, um texto e um botão. A barra de tarefas possui um título e um botão de calibração. Para calibrar o dispositivo, deve-se fazer um curto-circuito no ADC do RP2040 e apertar o botão ”Calibrar”. Assim, por meio do software é possível corrigir o deslocamento do offset do MC. Já para o texto ”Insira o Resistor” é substituído com o valor calculado da resistência elétrica no RP2040. Por último, o botão ”Amostrar” ao pressionar ordena ao MC realizar a função do ohmímetro e converte para um número com 3 casas a cima da vírgula, convertendo o valor da unidade em ohm ou kiloohm.

É apresentado o código 4.19 para o botão ”Amostrar”. É utilizada a função escrever (”write”) que faz o envio do comando 79 para o RP2040.

Código 4.19 – *Código do botão ”Amostrar”.*

```
1  ElevatedButton(
2 onPressed: () async {
3     model.port?.write(Uint8List.fromList([79]));
4     },,...
```

Além da função escrever existe a funçao de ouvir o canal USB que é sempre iniciada para cada página separadamente, connforme é explicado no código 4.20. Anterior a isto ao entrar na tela todo valor que o host recebe é alocado na variável "a" que é o valor calculado do resistor, e então é verificado se este valor está entre zero a mil ou de mil a milhão. Se o valor é menor que mil, apenas é arredondado o valor em um número inteiro e a unidade é dada em Ω. Já para valores maiores de mil, é arredondado para aparecer apenas os 3 primeiros números e a unidade é kΩ. E para valores maiores que um milhão, nada é feito e o valor da unidade é vazio (o equipamento é incapaz de medir este valor). Por fim "setState" atualizado o texto "Insira o resistor" caso seja meço um valor válido entre a escala descrita de zero ao milhão. O mesmo é feito para o capacimetro, apenas altera a unidade para μF.

Código 4.20 – *Código da função que recebe os valores da conexão USB ao iniciar a tela do Ohmímetro.*

```
ElevatedButton(
                    @override
 void initState() {
   USB model = ScopedModel.of(context);
   model.transaction!.stream.listen((event) {
     value = double.parse(event);
     double? a = value;
     if (a! < 1000) {
       explode = false;
       value = a;
       unit = "\Omega";
       value = value!.round() as double?;
     } else if ((a > 1000) && (a < 1000000)) {
       explode = false;
       unit = "k\Omega";
       value = a /= 1000;
     } else {
       setState(() {
         explode = true;
         unit = "";
         value = a;
       });
```

```
23          }
24          setState(() {});
25        });
26  ,...
```

No capacímetro, a estrutura da tela é igual ao ohmímetro, possui a função de escrita com a ordem diferente (olhe a tabela das ordens da interface ao microcontrolador em 4.1) e de ouvir ajustando para valor comercial de capacitância.

4.3.5 Código do Osciloscópio

Para gráfico, foi utilizado um pacote feito pela empresa Syncfusion que possui suporte ao Flutter. Este pacote possui diversos tipos de gráficos e todos podem ser customizados, um deles é o gráfico de linha com animação que foi empregado neste aparelho. Entretanto, o pacote oferece a facilidade de, com poucas alterações no código, poder alterar características do gráfico para teste, facilitando o desenvolvimento. Porém, existe o desafio de conciliar a obtenção e a organização dos dados para, então, construir o gráfico com eles.

É explicado apenas partes específicas do código de autoria própria. Pois além de ser extenso, existe a documentação detalhada das demais partes envolvendo a configuração e estilização do gráfico nos domínios públicos da Syncfusion.

FONTE: Próprio Autor

Figura 4.9 – *Tela do Osciloscópio ao entrar na página.*

A figura 4.9 apresenta a página do osciloscópio, ao iniciar pela página principal inicial. A barra do aplicativo possui o título e o botão de volta semelhante às demais telas filhas, mas com botões de ”play”, ”pause” e de imagem. Além disso, apresenta no canto esquerdo da tela o botão trigger e as seguintes legendas: de frequência do sinal, de tensão máxima, miníma e de pico a pico.

Existe a função de inicialização da página que pode iniciar funções, classes e variáveis. De fato, existe a necessidade de, para este código, iniciar a função o quanto antes, que é a função que verifica o fluxo de dados recebidos do USB. Então, para isto, é utilizada, novamente, a função para ouvir da porta USB, ”port.listen”, logo ao iniciar a página e fica ativa durante todo o processo. Toda vez que o aplicativo recebe uma resposta do RP2040, que são as amostras de tensão lidas pelo ADC para compor o gráfico do osciloscópio, esta operação na linha 5 do código 4.21 é executada. Convertendo este valor recebido em 12 bits em tensão, para economizar processamento do MC. E armazenados os dados na variável”_lista” para que, depois que terminar a transação de dados, possa continuar a construir o gráfico.

Por última, executa a função de atualizar o fluxo de dados do gráfico ”updateDataSource”. que é explicada adiante.

Código 4.21 – *Código para verificar e salvar o valor da tensão em um vetor.*

```
1 @override
2   void initState() {
3 ...
4 model.transaction!.stream.listen((event) {
5       _lista.add(3.3 * double.parse(event) / 4095);});
6        if (_lista.length == 260) {
7         updateDataSource();
8       }...
9     super.initState();}
```

Ao lado do título existe o seletor de frequência que, ao apertá-lo, é possível escolher entre: 100 Hz, 500 Hz, 1 kHz, 5 kHz, 10 kHz, 50 kHz, 100 kHz e 500 kHz das f_s do ADC. A página está configurada para sempre começar 50 kHz de frequência ao entrar. Ao pressionar alguma frequência, o código 4.22 é ativado. O widget é uma combinação de botão e uma lista para abrigar todas as opções. Todos os valores acionam a função de escrever na porta USB ”port.write”. Neste algoritmo

é enviado 48, que em ASC2 converte para o número 0, que é a ordem para o MC alterar a taxa para 100 Hz. E assim é o mesmo para as demais até o número 55, que corresponde ao número 7 e 500 kHz de taxa de amostragem do ADC. No final da seção dos códigos da interface, existe a tabela 4.1 que faz a relação entre os números do código que são convertidos em ASC2 e sua respectiva função para o RP2040. É apresentado apenas o código para 100 Hz, as demais são semelhantes, alterando apenas os valores mencionados.

Código 4.22 – *Parte do código para seleção da frequência de amostragem.*

```
...
actions: <Widget>[
    DropdownButton(
        value: dropdownValue,
        items: [DropdownMenuItem(
            value: '100',
            child: const Text('100 Hz'),
            onTap: () {
                model.port!.write(Uint8List.fromList([48]));
            },),...
```

Agora, para o botão "play" da interface do osciloscópio, ao apertá-lo, a seguinte ação do código 4.23 é executada. Existem três opções para atualizar a tela do gráfico do osciloscópio, são elas: adicionar, remover e atualizar os valores. No código primeiro, é estabelecido o valor 83 na variável "count", para ser utilizado novamente mais a seguir. Também são limpos os valores no registrador do gráfico "chartData" e na variável "_lista". Para então ser enviado o comando 83 ao MC que significa a ordem para o RP2040 amostrar e enviar os dados do osciloscópio que é verificado pelo código anterior.

Código 4.23 – *Código ao apertar o botão play*

```
onPressed: () {
    count = 83;
    chartData!.clear();
    _lista.clear();
    model.port!.write(Uint8List.fromList([83]));
    }, ...
```

Este código 4.24 é da função iniciada após o recebimento de todos os dados recebidos ao apertar o botão "play" na inicialização da tela. O RP2040 envia 300 amostras para o celular e são armazenadas na variável "_lista" e utilizadas 256 amostras no laço, o restante não é utilizado. Neste laço também utilizam-se as funções que obtêm o maior valor e o menor valor salvo na "_lista" que são "Vmax" e "Vmin" da interface. E também pega cada amostra, eleva ao quadrado e soma na variável "vpow", para o cálculo parcial da V_{RMS}. Depois, o código determina a frequência do sinal amostrado de modo semelhante à condição do trigger explicada na seção dos códigos do RP2040. Para isto, é preciso localizar a amostra que começa e a que termina num período do sinal. Sabendo estas duas amostras e a taxa de amostragem, é possível determinar a frequência do sinal, nas linhas 15 e 16. E é feito o cálculo da tensão de pico-a-pico que é a diferença entre "Vmax" e "Vmin". E por fim, termina o cálculo de V_{RMS}.

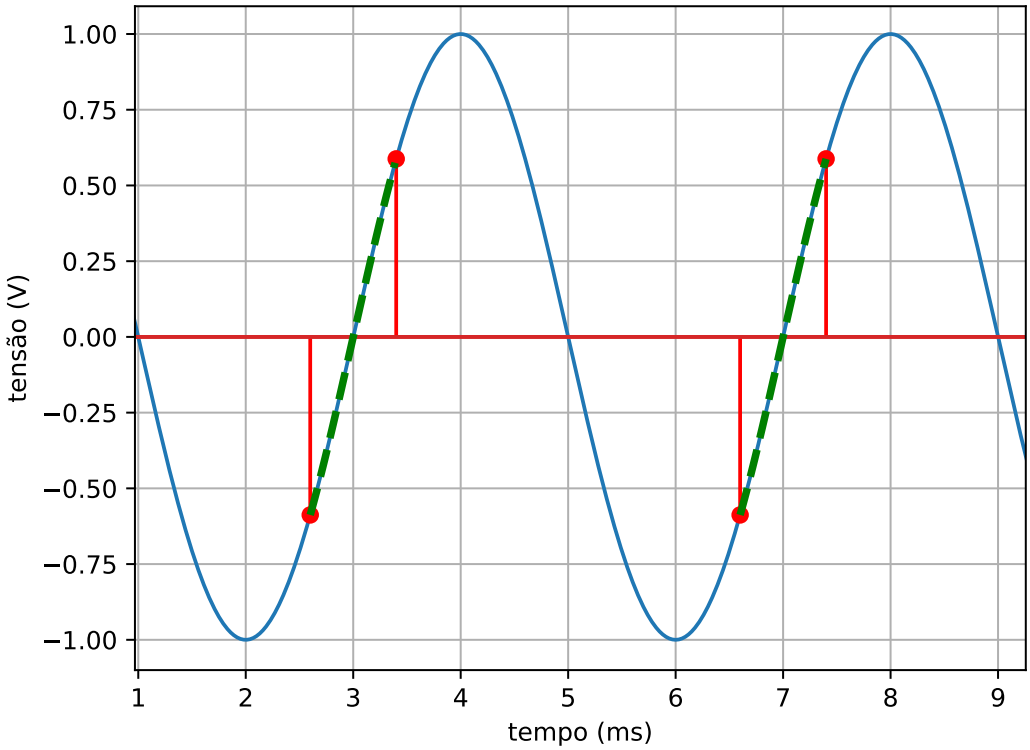
O gráfico da figura 4.10 mostra o sinal em azul e os pontos em vermelho são a limitação feita pela condição das linhas 5 e 6 do código 4.24. Já para o lado verde representa as possíveis amostras aceitáveis pela condição. E funciona da seguinte forma: se a amostra anterior é negativa e a atual é positiva, salva-se a posição da amostra atual. Fazendo isso, duas vezes já temos as duas posições das amostras e é possível calcular a frequência deste sinal, basta realizar a simples equação 4.1.

$$f = \frac{f_s}{P[n] - P[n-1]} \tag{4.1}$$

f ⇒ Frequência do sinal;
f_s ⇒ Frequência de amostragem;
$P[n]$ ⇒ Posição atual do vertor amostrado do sinal;
$P[n-1]$ ⇒ Posição anterior do vertor amostrado do sinal;

Código 4.24 – *Parte um do algoritmo da função updateDataSource.*

```
1     for (int i = 0; i < 256; i++) {
2       chartData!.add(LiveData(i, _lista[i]));
3       vpow += pow(_lista[i], 2);
4       vMax.value = max(vMax.value, _lista[i]);
5       vMin.value = min(vMin.value, _lista[i]);
6       if ((_lista[i] - 1.65) < 0 &&
7           (_lista[i + 1] - 1.65) > 0 &&){
```



FONTE: Próprio Autor

Figura 4.10 – *Gráfico para demonstrar como é obtido a frequência e o trigger.*

```
8            if (startDetect == false) {
9            startValue = i;
10           startDetect = true;}
11           else if (startDetect == true) {
12           endValue = i;
13           oneTime = true;}}}
14       fHz.value = (int.parse(dropdownValue)
15       / (endValue - startValue)).round();
16       vPp.value = vMax.value - vMin.value;
17       vRMS.value = sqrt(vpow/256);;...
```

Para o código 4.25 é a continuação anterior da parte anterior, focando apenas em atualizar a parte dos dados no gráfico. Existe a condição de que na primeira vez desta função os valores são adicionados e, para a segunda em diante, são apenas atualizados. Ao adicionar os dados no gráfico, é preciso atualizar o controlador dele, é uma forma de gastar menos recurso para redesenhar a tela do gráfico,

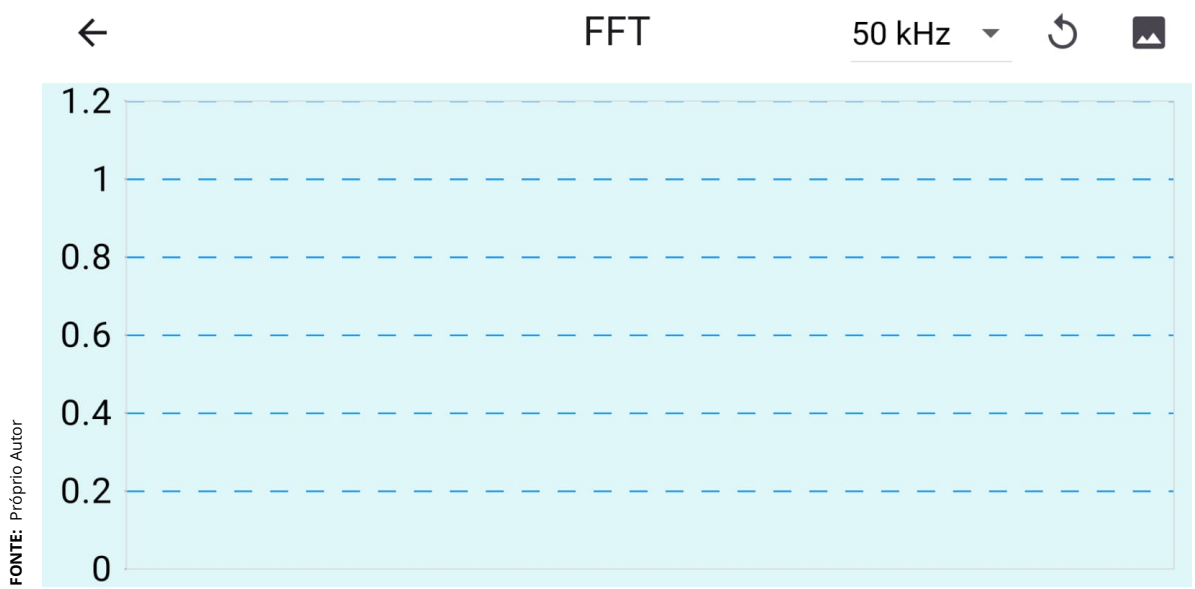
caso contrário é necessário atualizar todos os pontos da tela da interface. Depois é novamente esvaziado os valores do gráfico, porém sem atualizar a tela. E por último, é ordenado mais 300 amostras ao MC, que serão adicionados na próxima vez que o temporizador acionar a função, completando o ciclo.

Código 4.25 – *Parte dois do algoritmo da função updateDataSource.*

```
...
 if (time == 0) {
       _chartSeriesController?.updateDataSource(
       addedDataIndexes: _zeros);
       time++;
    } else {
       _chartSeriesController?.updateDataSource(
       updatedDataIndexes: _zeros);}
    _lista.clear();
    chartData!.clear();
    USB model = ScopedModel.of(context);
    model.port!.write(Uint8List.fromList([count]));
```

4.3.6 Código do Trigger

Assim como a página do capacímetro é semelhante à do ohmímetro, acontece o mesmo com a página do FFT e osciloscópio. A figura 4.11 demonstra que o FFT possui até menos funcionalidades que a página anterior. Possui o mesmo seletor de f_s, o botão de "screenshot" e o botão de atualizar a tela. Este último botão é muito semelhante aos botões "trigger" e "play" do osciloscópio, mas como não possui animação nesta tela, é ainda mais simples seu funcionamento. Ao apertá-lo, envia o código em ASC2 para o MC e recebe as amostras já no domínio da frequência, basta apenas por estes dados no gráfico. E ao apertá-lo uma segunda vez, apaga os valores e substitui pelos novos, por este motivo o ícone é atualizar.



FONTE: Próprio Autor

Figura 4.11 – *Página da interface do FFT.*

4.3.7 Tabela dos comando utilizados para comunicação com o RP2040

A tabela 4.1 a seguir apresenta todos os comandos utilizados no trabalho. Na interface, são declarados os valores decimais nas funções de transação e escrita no serial USB. O próprio código converte os valores decimais em caracteres, seguido o padrão ASC2. O caractere é enviado por meio da conexão USB e chega ao dispositivo MC e executa a função descrita. Após o término da execução, é enviado o resultado para a interface. Desta forma, conclui-se o processo de comunicação entre os dois aparelhos.

4.4 Prototipação

Para a criação de um protótipo, primeiro foi desenhado o esquemático para que facilite a montagem do circuito. O esquemático apresentado na figura 4.12 contém todos os componentes do projeto, são eles: a placa de desenvolvimento com RP2040, o multiplexador CD4051B e um conector tipo parafuso de quatro polos para eletrônica. Depois, foram montados em um protoboard os componentes para testes iniciais do projeto. Constatando que, uma vez na protoboard, o projeto se mantém estável, o próximo passo foi desenvolver um protótipo na placa de fenolite. Para este método, é necessário projetar a placa do circuito impresso.

Tabela 4.1 – *Comandos da interface utilizados para comunicação com o RP2040*

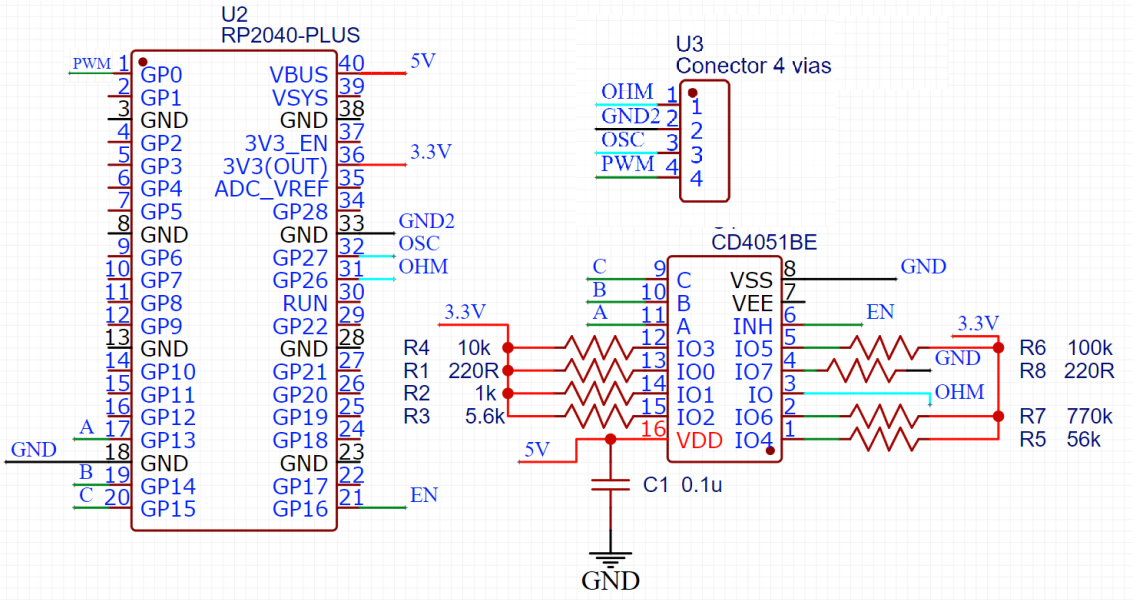
Decimal	Caractere ASC2	Descrição
48	0	Muda f_s para 100 Hz
49	1	Muda f_s para 500 Hz
50	2	Muda f_s para 1 kHz
51	3	Muda f_s para 5 kHz
52	4	Muda f_s para 10 kHz
53	5	Muda f_s para 50 kHz
54	6	Muda f_s para 100 kHz
55	7	Muda f_s para 500 kHz
98	b	Testar a comunicação na página teste
79	O	Comando da função ohmímetro
83	S	Comando da função osciloscópio
67	C	Comando da função capacímetro
75	K	Comando para calibração
97	a	Comando para função FFT
84	T	Comando para função do trigger

FONTE: Próprio autor

Então, foi utilizado o programa EasyEDA para este propósito e para o esquemático também. Este programa possui versão online gratuita e oferece uma diversidade de bibliotecas (esquemas de componentes) que são disponibilizadas pelos fabricantes e colaboradores.

A figura 4.13 representa o projeto das trilhas e posicionamento dos componentes na placa de circuito impresso. Com este projeto é possível transferir as trilhas e pads para a placa de fenolite. Para isto, bastou pintar a pcb com uma tinta não solúvel em água e foi utilizada uma gravadora a lazer para reproduzir o padrão na pintura. Depois do esquema da figura 4.13 gravado na placa de fenolite, colocou em um recipiente com percloreto de ferro e água para a dissolução das partes indesejadas de cobre. Com a placa corroída, foi necessário fazer os furos, montar os componentes e soldar.

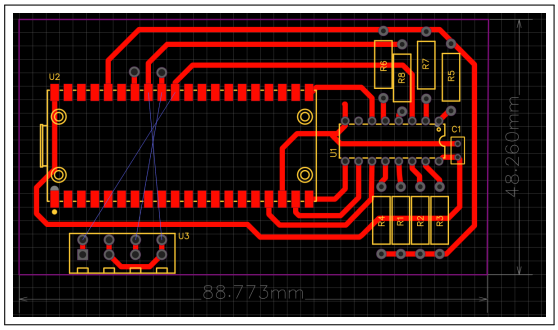
Por ter cometido alguns erros na placa anterior, foi produzida outra placa com os mesmos componentes, modificando apenas a placa de desenvolvimento. Dessa vez, foi produzida na placa de fenolite já perfurada. As trilhas foram feitas apenas com fios ”jumper” . Com os componentes posicionados, basta soldar. O resultado das duas placas é apresentado na figura 4.15.



FONTE: Próprio autor

Figura 4.12 – *Esquemático completo do projeto feito no EasyEDA.*

Foi feito um pequeno gabinete (ou case) para comportar a pcb e evitar um curto-circuito na superfície da placa. Para isto, foi feito o desenho em 3D desta peça no programa Autodesk Fusion, ilustrado na figura 4.14. A seguir, uma aproximação dos valores de todos os componentes que compõe o trabalho na tabela 4.2. Então, o total dá algo em torno de 130 reais, que consta também o valor gasto com energia elétrica e estanho durante o processo de produção.



FONTE: Próprio autor

Figura 4.13 – *Desenho das conexões (trilhas e pads) para confecionar a pcb de uma camada.*

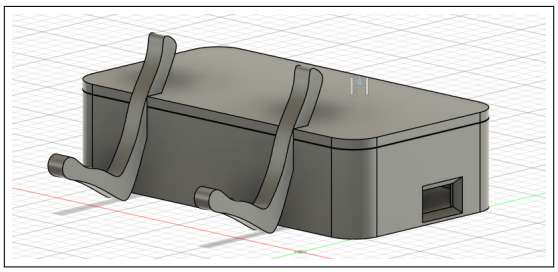


Figura 4.14 – *Desenho do gabinete do projeto no Autocad Fusion.*

FONTE: Próprio autor

4.5 Testes, resultados e discussões

Com o protótipo pronto e com o programa compilado no celular, a versão final do projeto é apresentada na figura 4.16 mostrando a parte da frente do gabi-

Tabela 4.2 – *Tabela com os preços dos componentes e o total gasto.*

Componente	Preço R$
1x Raspberry pi pico	20,00
1x CD4051	3,00
1x Placa de fenolite perfurada 7x9 cm	5,00
1x Cabo para jumper 1m 10 vias	15,00
1x Borne com 4 vias	4,00
8x resistores 1/4 W 1%	2,50
1x Conector BNC femêa	4,00
1x Conector de pressão 2 vias para aúdio	4,00
1x Capacitor cerâmico ou poliester de 100 nF	0,20
1x Custo ao produzir o gabinete	5,00
1x ZMPT101B	20,00
1x Cabo com dois conenctores USB-c	20,00
1x Sonda para osciloscópio	50,00
TOTAL	**147,70**

FONTE: Próprio autor

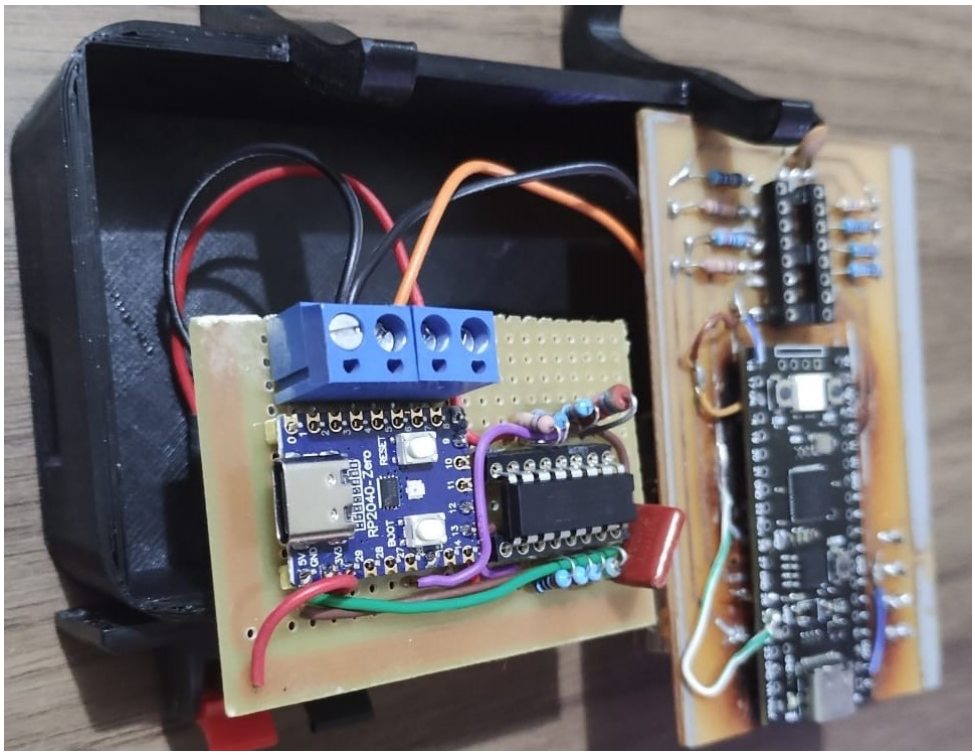
nete como suporte ao celular.

Aqui são apresentados os dados obtidos por meio de testes do equipamento ao medir valores de resistência e capacitância. Assim, ao testá-lo, será possível estabelecer os limites de medição do aparelho. Para isto, são utilizados dois equipamentos de medição para comparar e validar os valores amostrados.

O primeiro é o modelo LCR-T4 que é projetado com o mesmo MC do arduino uno e possui a resolução de 0.1 Ω para o limite de medição entre 0.1 Ω a 50 MΩ para a resistência elétrica. Para a capacitância, este aparelho pode medir entre 25 pF e 100 mF.

O segundo é o multímetro digital de modelo 118A, com, também, a resolução de 0,1 Ω e com a faixa um pouco maior de 0.1 Ω a 60 MΩ comparado ao equipamento anterior. Já para capacitancia, o limite de medição é entre 20 nF e 60 mF. Os equipamentos são ilustrados nas figuras 4.17 e 4.18.

Também é feito o cálculo de desvio das amostras e comparado com as tolerâncias dos resistores. Já para o capacitor, uma tolerância de até 15% será utilizada ao analisar as amostras. O cálculo do desvio em porcentagem é mostrado a seguir.



FONTE: Próprio autor

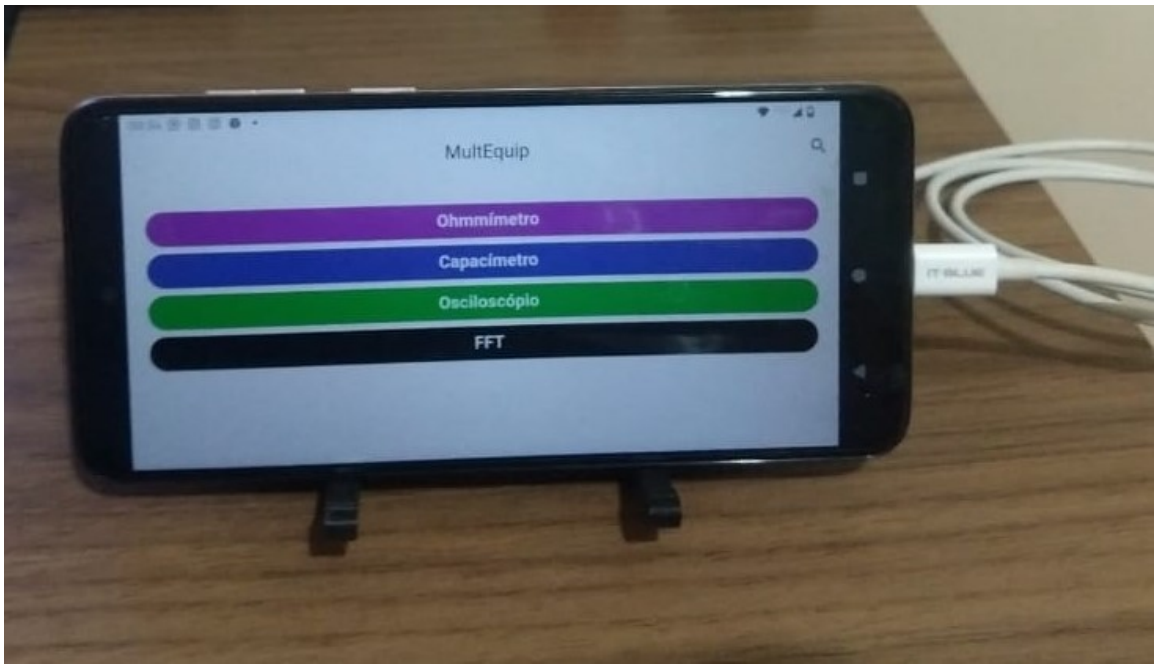
Figura 4.15 – *As duas placas construidas no trabalho em cima do gabinete, em destaque a placa perfurada.*

$$D\% = \frac{|x_n - x_a|}{x_a} \times 100 \tag{4.2}$$

$D\%$ ⇒ Desvio da amostra em porcentagem;
x_n ⇒ Valor nominal do componente resistor ou capacitor;
x_a ⇒ Valor amostrado do componente resistor ou capacitor;

A figura 4.19 mostra todos os componentes que foram mensurados de diferentes tipos e tolerâncias.

Alguns resistores utilizados são de potência e não possuem faixa de tolerância, mas eles são bem próximos aos valores nominais. A tabela 4.3 contém os valores obtidos nos três equipamentos, sendo ”MultEquip” o nome dado ao equipamento projetado. Os desvios apresentados do maior e menor valor de resistência indicam as maiores fontes de erro. O desvio do resistor se deve a muitos fatores, são eles: ruídos e precisão do regulador de tensão da placa de desenvolvimento (RASPBERRY PI, 2020a), resistor de referência de 220 Ω maior que dez vezes o valor nominal, a impedância do multiplexador. Estas são as principais hipóteses para este problema, já para o desvio da resistência de 100 kΩ é o valor elevado



FONTE: Próprio autor

Figura 4.16 – *Visão frontal do projeto com o programa iniciado no celular.*

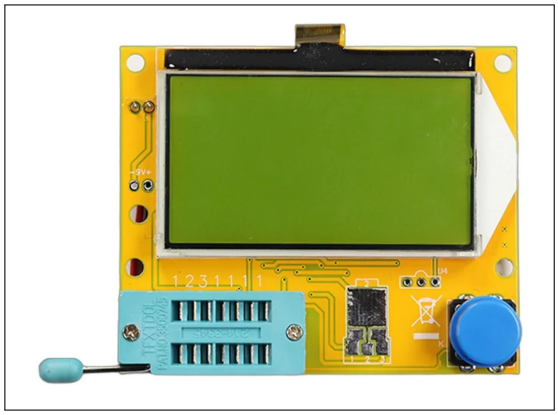
da resistencia, o que leva a circuitos equivalentes diferentes do projetado. Sendo este um erro dado pelo método adotado para obter valores de resistência.

Tabela 4.3 – *Tabela com a medição dos resistores e o valor do desvio do MultEquip*

Resistencia nominal (Ω) / Tolerância %	LCR-T4 (Ω)	Multimetro (Ω)	MultEquip (Ω)	Desvio (%)
100 k +/-1%	100,2 k	100,3 k	95,41 k	4,59
4k7 +/- 5%	4,69 k	4,66 k	4,68 k	0,47
240 +/-10%	238,7	239,6	239	0,42
22 k +/-10%	24,59 k	24,22 k	23,64 k	7,45
10 k	10,11 k	10,07 K	10,05 k	0,50
5k49 +/-1%	5,5 k	5,48K	5,49 k	0,07
56k +/-5%	56,1 k	55,91 k	53,47 k	4,52
18	17.8	19,2	22,88	27,11

FONTE: Próprio autora

O resistor de 22 kΩ que apresenta o segundo maior desvio, é provável que seja um erro na sua produção. Pois todos os aparelhos apresentaram o valor mais elevado do que o nominal descrito no código de cores do componente. E os resistores de 10 kΩ e o de 18 Ω são resistores de potência, não possuindo indicativo de tolerância do fabricante.



FONTE: (PINTUDY STORE, 2024)

Figura 4.17 – *Aparelho utilizado nos testes LCR-T4*

Figura 4.18 – *Aparelho utilizado nos testes Multímetro 118A*

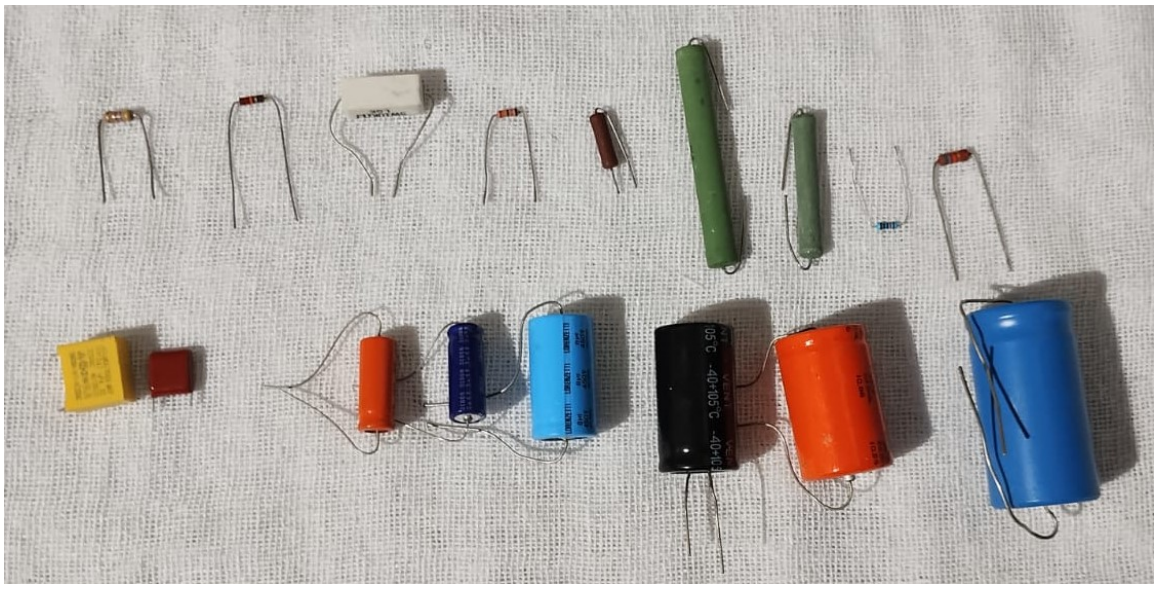
FONTE: (ANEG, 2024)

Tabela 4.4 – *Tabela com a medição dos capacitores e o valor do desvio do MultEquip*

Capacitância nominal (μF)	LCR-T4 (μF)	Multimetro (μF)	MultEquip (μF))	Desvio (%)
150	153,4	148	149,9	0,07
220	236,1	231	242	10,00
8	9,18	8,75	8,41	5,13
4700	4343	4131	4293	8,66
10	10,65	10,16	10,21	2,10
100	96,77	93,9	95,11	4,89
0,334	0,322	0,335	0,34	1,80
0,47	–	0,458	0,45	4,26

FONTE: Próprio autor

Agora, para os valores de capacitância apresentados na tabela 4.4 para os capacitores eletrolíticos, os valores obtidos são próximos dos outros dois equipamentos. Mas é o esperado, já que foi calibrado por eles. O maior problema é a impossibilidade de obter bons valores para capacitância na casa dos picofarad. Isto, pois, a função do tempo utilizado no RP2040 ”micros” já não é capaz de contabilizar a carga do componente (dado o valor tão pequeno). Então, novamente, para estes valores, o método utilizado não é adequado. Outro problema é para valores de capacitância elevado, por exemplo, de 4700 μF, que é demasiadamente demorado o período de carga e descarga do componente. Podendo levar até um minuto para obter o valor da amostra. O capacitor de 0,47 μF não foi possível medir com o LCR-T4 pela dificuldade de conectá-los.



FONTE: Próprio autor

Figura 4.19 – *Os componentes utilizados para a realização dos testes.*

4.5.1 Teste de amostras sucessivas

Este teste serve para demonstrar que, ao realizar amostra sucessiva de um capacitor ou resistor no equipamento projetado, resulta em valores diferentes. Isto ocorre devido ao fato de o ADC do RP2040 não possuir uma tensão de referência de alta precisão ou devido aos erros do próprio hardware. É sugerido pelo fabricante da placa de desenvolvimento (RASPBERRY PI, 2020a) que utilizasse o ”LM4040”como referência de tensão. Pois o empregado gera bastante ruído mesmo com um filtro RC embutido na placa de desenvolvimento. Portanto, esta seção tem o objetivo de esclarecer este problema encontrado ao decorrer do projeto. Foi utilizado apenas três componentes para este teste, sendo eles: resistores de 18 Ω e 10 kΩ e um capacitor eletrolítico de 100 μF.

A tabela 4.5 apresenta as variações dos valores amostrados do mesmo componente. Como já mencionado, ao medir a tensão é possível relacionar com a resistência por meio do método de divisão de tensão. E também, ao medir a tensão da carga do capacitor e utilizar a relação de ”τ” é possível relacionar com a capacitância. Portanto, o aparelho que mede tais valores de tensão, o ADC, não deve apresentar erros graves para que possa garantir exatidão e coerência em seus valores amostrados.

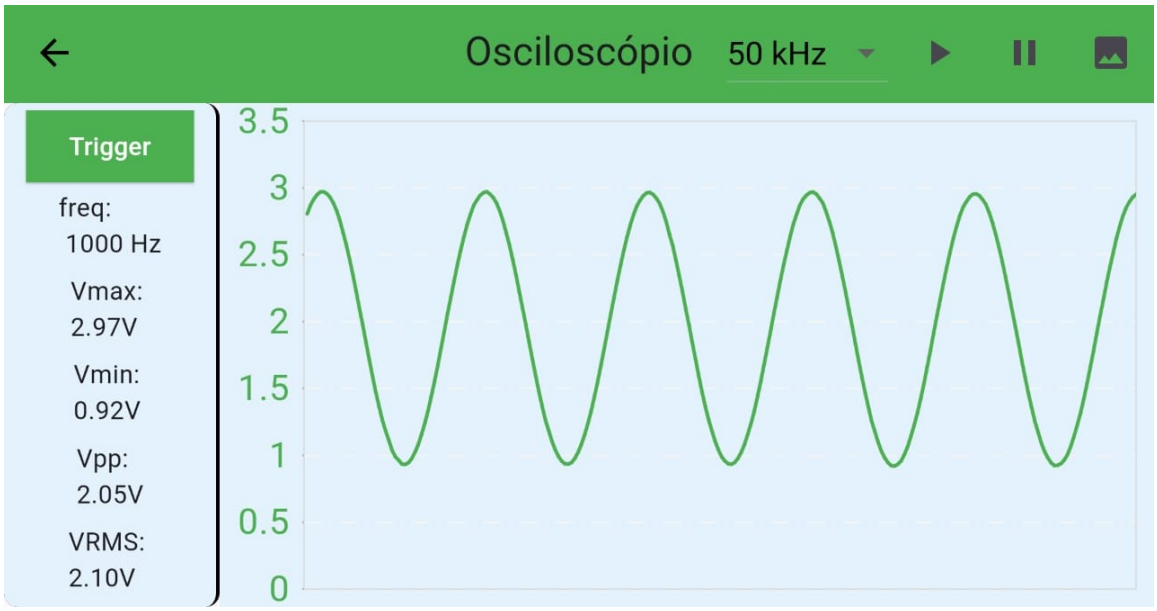
Tabela 4.5 – *Tabela com a medição dos capacitores e resistores sucessivos no MultEquip*

Resistor de 18 Ω	Resistor de 10 kΩ	Capacitor de 100 μF
23.23	10065	94.21
22.96	10075	95.31
23.05	10085	94.97
23.05	10055	95.40
22.96	10085	95.18
23.14	10055	95.13
23.05	10075	93.78
22.96	10026	95.38

FONTE: Próprio autor

4.5.2 Teste do Osciloscópio e da FFT

Para testar os códigos do osciloscópio e da FFT, é necessário injetar um sinal na entrada do protótipo. Portanto, foi utilizado um gerador de sinais que acompanha o osciloscópio LOTO OSC482 para gerar ondas senoidais, triangulares e trem de pulso. E com isto podemos comparar, por meio das imagens e tabelas, os valores obtidos de frequência do sinal, tensão máxima, mínima, pico a pico e a V_{RMS} entre os dois aparelhos para diferentes sinais.



FONTE: Próprio autor

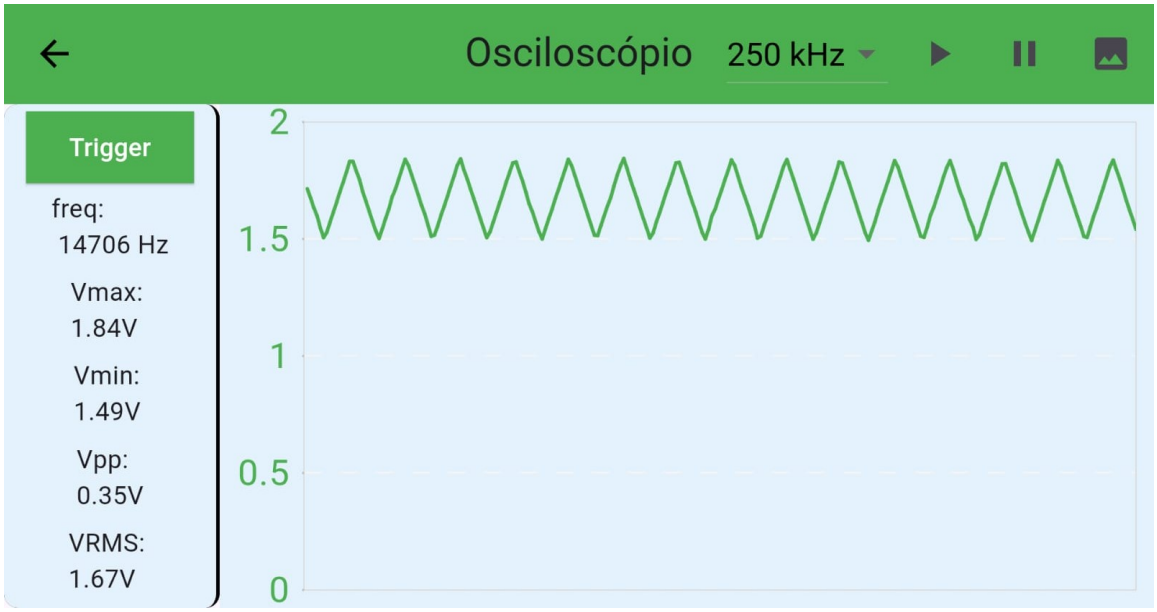
Figura 4.20 – *Sinal senoidal exibido por meio do aplicativo de celular do trabalho.*

Os valores obtidos pelos equipamentos são próximos e se diferenciam por dois motivos: a referência de tensão do ADC e pela diferença de f_s entre os apa-

Tabela 4.6 – *Dados obitidos do sinal da onda senoidal entre os dois equipamentos.*

Dados do sinal	OSC48)	MultEquip
f (Hz)	1000	1000
V_{MAX} (V)	2,945	2.97
V_{min} (V)	0.878	0.92
V_{PP} (V)	2,067	2.05
V_{RMS} (V)	2,031	2.10

FONTE: Próprio autor



FONTE: Próprio Autor

Figura 4.21 – *Sinal triangular exibido por meio do aplicativo de celular do trabalho.*

relhos. Como o regulador de tensão gera ruídos na ordem de dezenas de mV, é um indicador para as diferenças de V_{MAX} e V_{min} entre os instrumentos. Ademais, o OSC482 sempre irá amostrar o sinal com a máxima taxa de amostragem do aparelho, por isto indica valores com maior precisão. Enquanto que o aparelho projetado modifica sua f_s e utiliza menor número de amostras, para ajustar o sinal na tela do celular, portanto, menor precisão.

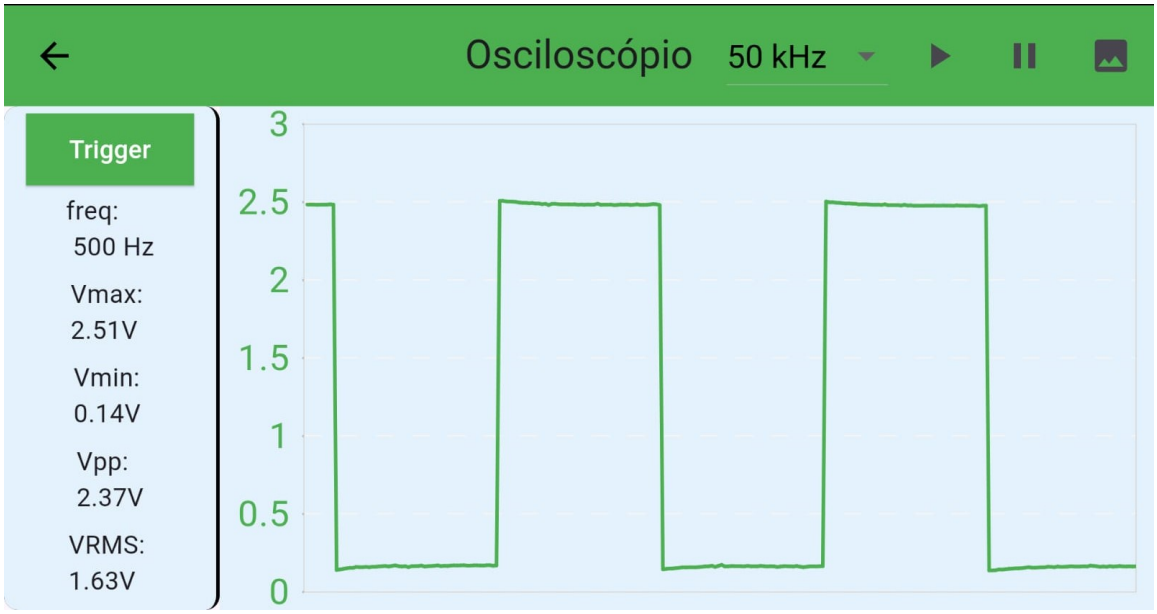
Agora, para a transformada, como ela é uma versão do osciloscópio mais simples, é mostrado apenas para o sinal de trem de pulso de 1 kHz na figura 4.23. Foi utilizado este sinal porque a FFT deste sinal é composta pela soma das frequências harmônicas ímpares. Ou seja, há mais picos ao cálcular este sinal do que os outros dois anteriores (senoidal e triangular).

Ao clicar na função do gráfico, é mostrada a legenda deste ponto com o dado

Tabela 4.7 – *Dados obitidos do sinal da onda triangular entre os dois equipamentos.*

Dados do sinal	OSC482	MultEquip
f (Hz)	14.990	14706
V_{MAX} (V)	1.820	1.84
V_{min} (V)	1.464	1.49
V_{PP} (V)	0.356	0.35
V_{RMS} (V)	1,640	1.67

FONTE: Próprio autor



FONTE: Próprio Autor

Figura 4.22 – *Sinal trem de pulso exibido por meio do aplicativo de celular do trabalho.*

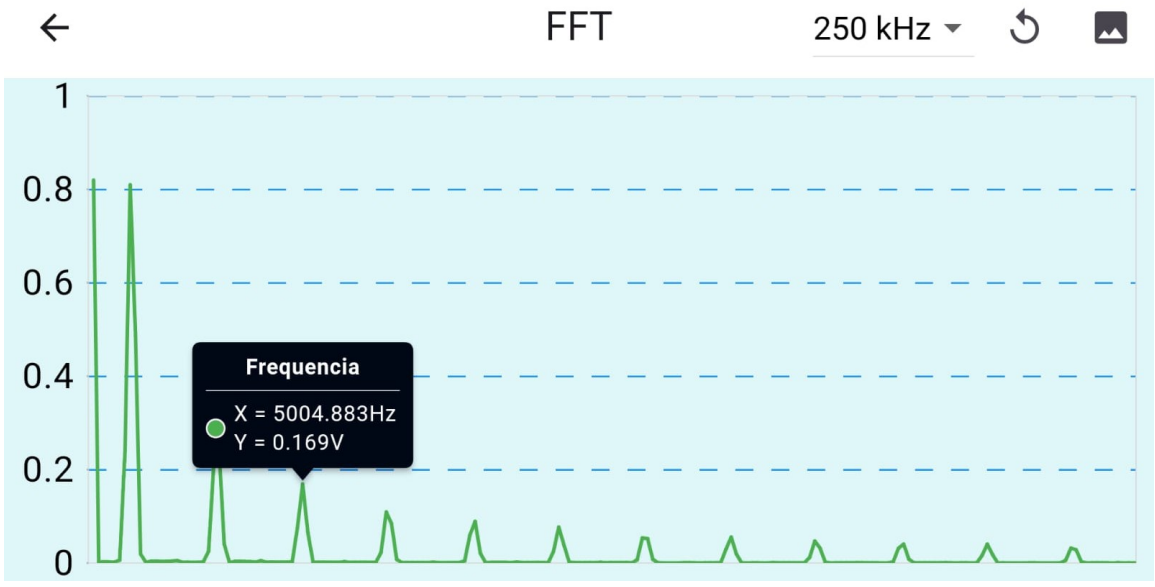
da frequência e da porcentagem da amplitude do sinal. O equipamento está levando em consideração o cálculo da tensão contínua e alternada. Então, sempre haverá amplitude para o componente DC em 0 Hz.

Ao término dos testes, o valor máximo f_s obtido foi de 250 ksps. Para conseguir chegar aos 500, como mencionado no desenvolvimento, é necessário um cristal oscilador de 48 MHz independente para o ADC. Entretanto, não houve muita distorção do sinal ao utilizar 256 pontos por vez na tela do gráfico por vez. E tão pouco é notada a interferência dos ruídos provocados pela regulação de tensão da placa de desenvolvimento. E apresentou valores próximos a equipamento comercial de custo de dezenas de vezes seu valor de produção.

Tabela 4.8 – *Dados obitidos do sinal da onda trem de pulso entre os dois equipamentos.*

Dados do sinal	OSC482	MultEquip
f (Hz)	500	500
V_{MAX} (V)	2.480	2.51
V_{min} (V)	0.103	0.14
V_{PP} (V)	2.377	2.37
V_{RMS} (V)	1,719	1.63

FONTE: Próprio autor



FONTE: Próprio Autor

Figura 4.23 – *A FFT do sinal trem de pulso de 1 kHz, detalhe para a terceira harmônica.*

4.5.3 Medindo tensão AC

Foi necessário ajustar o ganho do amplificador pelo potenciômetro do modulo ZMPT101B, para então, ser analisado o sinal resultante de sua saída para o ADC deste componente quando é ligado à rede. A figura 4.24 a seguir mostra esta saída do sinal já calibrada utilizando o OSC482.

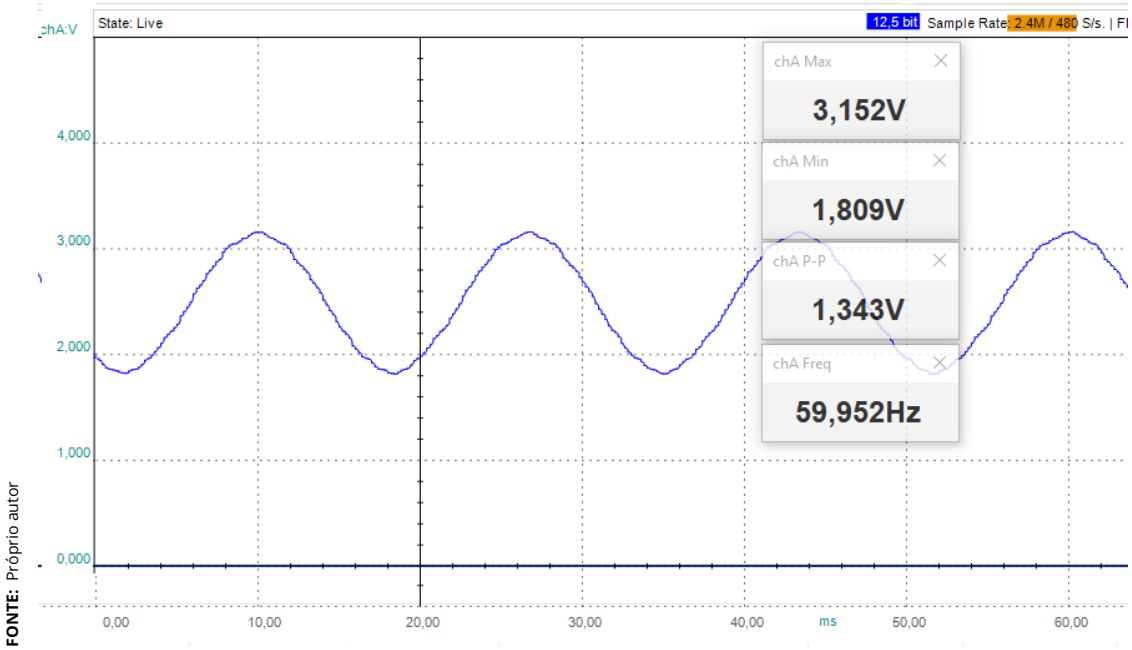
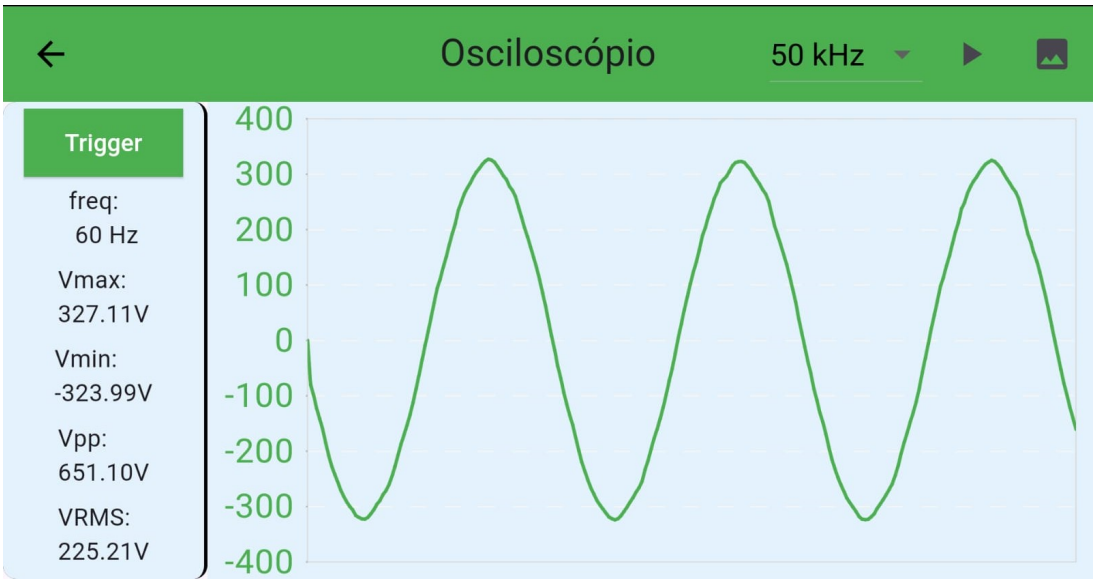


Figura 4.24 – *Sinal meço da saída do ZMPT101B pelo OSC482.*

Em seguida, depois de analisar o sinal, bastou corrigir os valores obtidos pelo ADC na página do código do osciloscópio. Para isto, foi modificada uma linha do código do osciloscópio, alterando no software os valores aceitos pelo ADC medidos para valores condizentes à rede elétrica. Esta linha é logo na inicialização da tela, onde converte os valores de 12 bits para corresponder de 0 a 3.3 V. Agora, o código 4.26 altera para os valores correspondentes à rede elétrica. Assim, é possível utilizar a mesma página do osciloscópio para ler também a tensão da rede, como mostra a figura 4.25.

Código 4.26 – *Código para adequar o osciloscópio para ler valores AC com o ZMPT101B*

```
1 _lista.add(507.83*(3.3 * double.parse(event) / 4095-2.51));
```



FONTE: Próprio autor

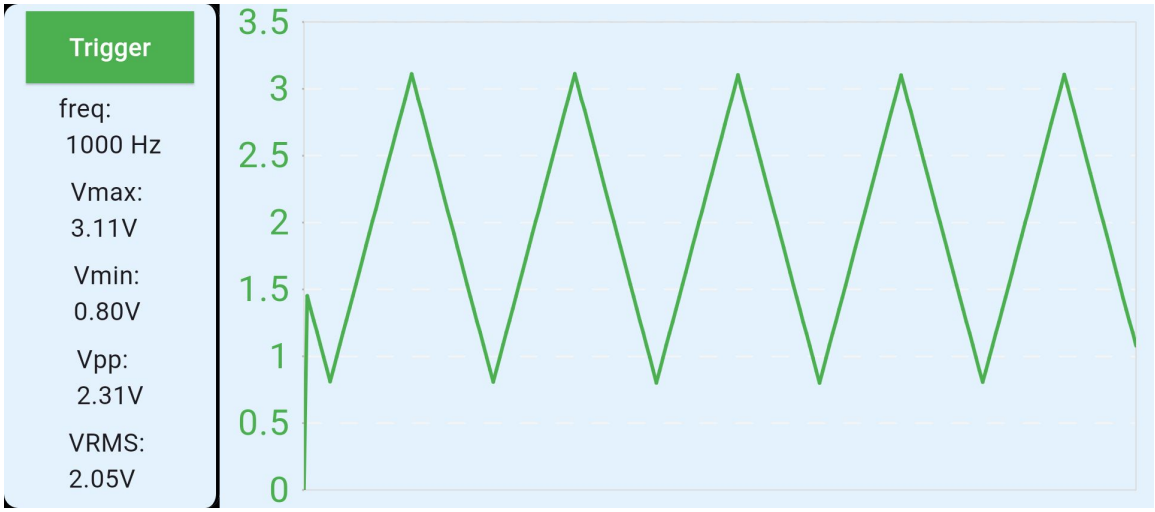
Figura 4.25 – *Sinal resultante meço no projeto utilizando o ZMPT101B conectado a rede elétrica.*

4.5.4 Problemas, limitações e bugs

A maior fonte de problemas e erros no projeto são encontrados nos códigos relacionados aos gráficos. Para o osciloscópio, existe o problema do valor inicial do gráfico demonstrado na figura 4.26. Este erro só prejudica a experiência visual do projeto, não afetando as demais funções.

Ao implementar o algoritmo para o ”trigger” , estabelece 1.65 V como o ponto principal, em que irá repetir o sinal diversas vezes. Se o sinal não alternar neste valor, é impossível dar o ”trigger” para fixá-lo na tela e tão pouco será feito o cálculo da frequência (possuem algoritmos similares). Os osciloscópios possuem um botão seletor que percorre o nível de tensão para o valor que o sinal se alterna em um período. Mas não foi possível desenvolver um seletor para movimentar o ”trigger” ficando preso ao valor estabelecido de 1.65 V. Não foram pesquisadas as soluções por causa das tensões pequenas aceitas pelo ADC de 0 a 3.3 V apenas.

Uma limitação imposta pelo código do gráfico é a impossibilidade de alterar os valores dos seus eixos. Como o eixo y só desloca entre 0 e 3.3 V representando a tensão, não há a necessidade de mudanças. Porêm, ao adicionar amostras no tempo no eixo x, ao modificar a f_s, é necessário modificar os valores de tempo e suas grandesas. Mas o código do gráfico não aceita variáveis, apenas valores fixos, sendo número ou nomes. Assim sendo, existe esta dificuldade de alterar valores de tempo e frequência (este erro também acontece na FFT) na parte gráfica do trabalho.



FONTE: Próprio autor

Figura 4.26 – *Erro da posição inicial em 0 no gráfico e logo em seguida retorna para o valor da amostra.*

Outro problema inconveniente na interface do aplicativo é que, ao desligar a tela do aparelho do celular ou minimizar o aplicativo, perde-se a conexão com o RP2040. É necessário reconectar o aparelho celular ao MC novamente para estabelecer a conexão. Este erro apresentou ao utilizar um celular na versão do Android 9.0, já outro na versão 11.00 não apresentou tal comportamento. Um bug que ocorre com frequência é ao apertar play na tela do osciloscópio e não alterar a tela. Isto acontecia quando acontecia durante o desenvolvimento e é um sinal de erro no código. Porêm, como não é sempre que ocorre, não é possível localizar a fonte de erro.

Por fim, em algumas ocasiões, não era possível estabelecer conexão com a placa de desenvolvimento. Este evento ocorreu, somente, durante as reuniões de orientação do projeto. Por este motivo, não foi descoberta a razão da causa do problema. Porém, também não ocorreu este erro com outro celular e com o protótipo do MC na placa perfurada.

Capítulo 5

Considerações Finais

O primeiro passo para a realização do trabalho foi definir os metodos utilizados para alcaçar os objetivos. O ohmímetro usa o metodo de divisão de tensão junto ao multiplexador para realizar possuir a escala automática e aumentar a impedância entre as portas do microcontrolador. Já o capacímetro utiliza o método de carga e descarga do capacítor sendo possível adaptar também no multiplexador. O osciloscópio faz uso do ADC como principal ferramenta de obtenção dos dados junto a programação para recriar a interface. Por fim, a FFT também é semelhante ao metodo utilizado pelo osciloscópio com a adição do algorítmo de Cooley–Tukey.

Já em relação aos objetivos, foi criada a interface, por meio do Flutter, e seus elementos são simples e intuitivos pois consistem apenas de botões e textos. No entanto, estabelecer a conexão entre o aparelho celular e o MC causou diversos problemas e instabilidade, pois o uso do protocolo usb é complexo. Contudo, superada esta barreira a maioria dos microcontroladores pode ser conectada ao aplicativo. Foi escolhido o RP2040 dada a disponibilidade, o preço, os componentes e suporte à conexão escolhida. Foi utilizado a linguagem C++ para programar o MC, pois permitiu: acessar diversas bibliotecas de código, acesso e configuração do hardware. Foram escolhidos os equipamentos de bancada eletrônica para o desenvolvimento em uma única interface. Já os dados são salvos como imagens do gráfico tanto para o osciloscópio, quanto para a FFT. Todos os 3 equipamentos foram comparados a equipamentos comerciais e de proposta similar de baixo custo e com muitas funções. Por fim, foi construido um circuito para acomodar a placa de desenvolvimento e aumentar a capacidade e a qualidade dos dados por meio do multiplexador e o modulo ZMPT101B.

O maior desafio do trabalho foi conciliar a conexão com a organização dos dados para compor a tela com o gráfico do osciloscópio sem que seja instável. Isto porque é necessário apurar os erros cometidos ao programar a interface do aplicativo de celuar, mas não foi possível utilizar ferramentas para depurar o código ao mesmo tempo que está conectado ao RP2040.

Considerando as tabelas 4.3 e 4.4 de testes e resulatados, o instrumento proposto é completamente operacional na faixa de 100 a 25000 Ω. Já para a função de capacímetro os resultados apontam para a faixa de valores entre 100 nF a 60 mF.

De acordo com os dados das tabelas 4.6, 4.7 e 4.8, para as tensões máximas e mínimas, existe uma diferença aproximada de 50 mV. O erro é propagado para o V_{PP}, contudo, estas tabelas apresentam valores de maior proximidade. Na contramão, para a tensão eficaz, o erro aumenta e apresenta máximo 89 mV de diferença. Em relação a frequência, os valores cálculados são consistentes, mas depende da quantidade de pontos disponíveis ao compor o gráfico. Conforme as figuras 4.20,4.21 e 4.22, até 15 kHz é possível identificar as formas de onda. Por fim, a figura 4.23 é até a décima segunda harmônica ímpar do sinal para f_s de 250 ksps.

5.1 Sugestões para Trabalhos Futuros

Como o trabalho desenvolveu várias funções e aparelhos, todos eles podem ser melhorados de diversas formas. Assim sendo, o método do ohmímetro proporcionou a calibração autônoma até certo ponto da sua faixa de operação. Desta forma, outros métodos como ponte de wheatstone podem ser explorados para valores mais altos de resistência elétrica. Já para o capacímetro, é interessante elaborar o código e o circuito para a leitura de valores de nanofarad e picofarad. Para o osciloscópio, o uso de um circuito de atenuação de sinal pode ser empregado para a medição dos sinais DC negativos e para sinais positivos de maior tensão.

Ao utilizar o RP2040, deve-se levar em consideração o problema da tensão de referência e o ruído gerado pelo regulador de tensão. Foi utilizado um filtro digital com o intuito de amenizar estes problemas. Contudo, seria ideal utilizar um regulador de tensão de precisão ou projetar um filtro analógico para reduzir o ruído da placa de desenvolvimento.

Explorar outros tipos de conexão pode resultar em uma melhor experiência de uso, principalmente as sem fio, como bluetooth e wifi. Desta forma, é necessário a escolha de outro microcontrolador que possua suporte a estas tecnologias, além do uso de bateria com circuito para a carga.

Por fim, pode-se adicionar mais equipamentos e funções no trabalho, como as encontradas no ohmímetro. As opções são diversas, como medir a corrente elétrica, teste de continuidade, teste de transistor, entre outras.

REFERÊNCIAS

ALEXANDER, C. K.; SADIKU, M. N. *Fundamentos de circuitos elétricos*. Porto Alegre: AMGH Editora, 2013. (Citado 3 vezes nas páginas 6, 8, and 9.)

ANEG. M118a multimetro digital. <https://pt.aliexpress.com/store/1102516986?spm=a2g0o.detail.0.0.65baOwEQOwEQ4d&sortType=bestmatch_sort> [Acessado em 28 de abril de 2024]. 2024. (Citado na página 56.)

ANG, B. How to use a digital multimeter. <https://www.keysight.com/blogs/en/tech/bench/2022/06/14/how-to-use-a-digital-multimeter> [Acessado em 2 de maio de 2024]. 2022. (Citado na página 20.)

BESSEMS, R.; HERRANZ, F. usb serial. <https://github.com/altera2015/usbserial> [Acessado em 16 de fevereiro de 2024]. 2023. (Citado na página 39.)

BRAGA, N. C. *Osciloscópio: Primeiros Passos*. [S.l.]: Newton C. Braga, 2014. (Citado na página 1.)

CASA DA ROBóTICA. Sensor de tensão ac 0 a 250v voltímetro zmpt101b. <https://www.casadarobotica.com/sensores-modulos/sensores/tensao/sensor-de-tensao-ac-0-a-250v-voltimetro-zmpt101b> [Acessado em 28 de abril de 2024]. 2024. (Citado na página 19.)

CONDES, E.; OVERBOHM, B. arduinofft. <https://github.com/kosme/arduinoFFT> [Acessado em 20 de março de 2024]. 2020. (Citado na página 33.)

FERNANDES, A. M. *DESENVOLVIMENTO DE UM OSCILOSCÓPIO DIGITAL COM O RASPBERRY PI*. Dissertação (Trabalho de conclusão de curso) — Universidade Tecnológica Federal do Paraná, 2020. (Citado na página 22.)

HAYKIN, S.; VEEN, B. V. *Sinais e Sistemas*. Porto Alegre: Bookman, 2005. (Citado na página 5.)

HOANG, K. RP2040 PWM Library. <https://github.com/khoih-prog/RP2040_PWM> [Acessado em 16 de fevereiro de 2024]. 2021. (Citado na página 26.)

INTEL CORPORATION. *Two decades of "plug and play" How USB became the most successful interface in the history of computing*. [S.l.], 1998. <https://pdf1.alldatasheet.com/datasheet-pdf/view/26890/TI/CD4051.html>, Acessado em 2 de maio de 2024. (Citado na página 15.)

LYONS, R. G. *Understanding digital signal processing, 3/E*. [S.l.]: Pearson Education, 2011. (Citado 4 vezes nas páginas 9, 10, 11, and 12.)

MADHU. Ohmmeter working and types. <https://www.codrey.com/electrical/ohmmeter-working-and-types/> [Acessado em 2 de maio de 2024]. 2020. (Citado na página 20.)

MEASUREMENT COMPUTING CORPORATION. *Data Acquisition Handbook*. [S.l.], 2012. <https://files.digilent.com/reference%2Fdata-acquisition-handbook.pdf>, Acessado em 30 de março de 2024. (Citado 2 vezes nas páginas 6 and 7.)

MEIRELLES, F. S. 34ª pesquisa anual do fgvcia da fgv/eaesp, 2023. <https://eaesp.fgv.br/sites/eaesp.fgv.br/files/u68/pesti-fgvcia-2023-resumoppt.pdf> [Acessado em 16 de fevereiro de 2024]. 2023. (Citado na página 1.)

PHILHOWER, I. E. F. Arduino-Pico. <https://github.com/earlephilhower/arduino-pico> [Acessado em 16 de fevereiro de 2024]. 2021. (Citado na página 26.)

PINTUDY STORE. Lcr-t4 multi-function transistor tester. <https://pt.aliexpress.com/item/1005006196080821.html#nav-store> [Acessado em 28 de abril de 2024]. 2024. (Citado na página 56.)

RASPBERRY PI. *Raspberry pi pico datasheet*. [S.l.], 2020. <https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf>, Acessado em 2 de maio de 2024. (Citado 2 vezes nas páginas 54 and 57.)

RASPBERRY PI. *RP2040 datasheet*. [S.l.], 2020. <https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf>, Acessado em 2 de maio de 2024. (Citado 2 vezes nas páginas 13 and 14.)

SANTOS, E. N. onorio dos. *Osciloscópio digital de baixo custo, baseado em KIT microcontrolador STM32, com tela em aplicativo android e envio de dados via rede sem fio.* Dissertação (Trabalho de conclusão de curso) — Universidade Federal do Paraná, 2019. (Citado na página 22.)

SANTOS, M. M.; DUARTE, G. H. *Projeto de um osciloscópio digital para sinais de até 4MHz com suporte a filtros FIR e IIR em tempo real e análise espectral.* Tese (Trabalho de conclusão de curso) — Universidade Tecnológica Federal do Paraná, 2013. (Citado na página 22.)

SHAH, J. *How USB Works: Introduction*. [S.l.], 2024. <https://www.circuitbread.com/tutorials/how-usb-works-introduction-part-1>, Acessado em 2 de maio de 2024. (Citado 3 vezes nas páginas 15, 16, and 17.)

SHOU, B. M. *Osciloscópio digital portátil de baixo custo*. Dissertação (Trabalho de conclusão de curso) — Universidade Federal do Paraná, 2011. (Citado na página 22.)

TEKTRONIX. *XYZs of Oscilloscopes*. [S.l.], 2016. <https://download.tek.com/document/03W_8605_7_HR_Letter.pdf>, Acessado em 2 de maio de 2024. (Citado na página 21.)

TEXAS INSTRUMENTS. *Datasheet CD4051B, CD4052B, CD4053B*. [S.l.], 1998. <https://pdf1.alldatasheet.com/datasheet-pdf/view/26890/TI/CD4051.html>, Acessado em 2 de maio de 2024. (Citado 2 vezes nas páginas 18 and 19.)

APÊNDICE A

Todos os códigos desenvolvidos no projeto

Como os códigos, desenvolvidos no Arduino Ide e no Android Studio, ao decorrer do projeto, são extensos, foram hospedados na página do Github do autor nas seguintes URL:

Da interface: <https://github.com/Ukobir/MultEquip-Interface/tree/main>

Do RP2040: <https://github.com/Ukobir/MultEquip-backend>