UNIVERSIDADE FEDERAL DO PIAUÍ

CENTRO DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

MESTRADO ACADÊMICO EM ENGENHARIA ELÉTRICA

PABLO RODRIGUES LOPES

DESENVOLVIMENTO DE UM PROTÓTIPO DE RELÉ UNIVERSAL DE BAIXO CUSTO PARA EXECUÇÃO DE FUNÇÕES DE PROTEÇÃO EM TEMPO REAL

TERESINA

2022

PABLO RODRIGUES LOPES

DESENVOLVIMENTO DE UM PROTÓTIPO DE RELÉ UNIVERSAL DE BAIXO CUSTO PARA EXECUÇÃO DE FUNÇÕES DE PROTEÇÃO EM TEMPO REAL

Dissertação apresentada ao Curso de Mestrado Acadêmico em Engenharia Elétrica do Programa de Pós-Graduação em Engenharia Elétrica do Centro de Tecnologia da Universidade Federal do Piauí, como requisito parcial à obtenção do título de mestre em Engenharia Elétrica. Área de Concentração: Sistemas de Energia Elétrica

Orientador: Prof. Dr. Rui Bertho Junior

TERESINA

2022

FICHA CATALOGRÁFICA
Universidade Federal do Piauí
Biblioteca Comunitária Jornalista Carlos Castello Branco
Divisão de Representação da Informação

L864d Lopes, Pablo Rodrigues.
Desenvolvimento de um protótipo de relé universal de baixo custo para execução de funções de proteção em tempo real / Pablo Rodrigues Lopes. -- 2022.
66 f.

Dissertação (Mestrado) – Universidade Federal do Piauí, Programa de Pós-Graduação em Engenharia Elétrica, Teresina, 2022.
"Orientador: Prof. Dr. Rui Bertho Junior".

1. Raspberry Pi. 2. Redes Neurais. 3. Relés. 4. Sistemas de Proteção . 5. Sistemas em Tempo Real. I. Bertho Junior, Rui. II. Título.

CDD 621.3

Bibliotecária: Francisca das Chagas Dias Leite - CRB3/1004

PABLO RODRIGUES LOPES

DESENVOLVIMENTO DE UM PROTÓTIPO DE RELÉ UNIVERSAL DE BAIXO CUSTO PARA EXECUÇÃO DE FUNÇÕES DE PROTEÇÃO EM TEMPO REAL

Dissertação apresentada ao Curso de Mestrado Acadêmico em Engenharia Elétrica do Programa de Pós-Graduação em Engenharia Elétrica do Centro de Tecnologia da Universidade Federal do Piauí, como requisito parcial à obtenção do título de mestre em Engenharia Elétrica. Área de Concentração: Sistemas de Energia Elétrica

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. Rui Bertho Junior (Orientador)
Universidade Federal do Piauí (UFPI)

Prof. Dr. Aryfrance Rocha Almeida
Universidade Federal do Piauí (UFPI)

Prof. Dr. Paulo Murinelli Pesoti
Instituto Federal de São Paulo (IFSP)

Dedico essa dissertação de mestrado à minha família, que sempre me mostrou que correr atrás dos nossos sonhos, embora seja um caminho árduo, é bastante recompensador. Graças a vocês cheguei aonde estou.

AGRADECIMENTOS

À Deus, por me mostrar as oportunidades a minha frente e saber aproveitá-las no momento certo, bem como as pessoas maravilhosas que pos em meu caminho e que muito me ensinaram.

Aos meus pais, Sandra e Avelar, por sempre me apoiarem em minhas escolhas, por terem cuidado de mim por todos esses anos com muito amor, carinho e bons conselhos, e por sempre estarem pensando no meu bem e meu futuro. Ao meu irmão Bruno, pela amizade e companheirismo com os quais sempre pude contar. Você também me mostou que correr atrás dos nossos sonhos requer dedicação extrema, não importa quantas vezes precisemos fazer isso.

Ao meu professor orientador Rui Bertho Junior, por ter me apresentado a temática interessantíssima na qual este trabalho se baseou, bem como por toda a ajuda e suporte durante este trabalho de mestrado, através de seus conhecimento e esclarecimentos. Ao Filipe Bispo Lima, técnico responsável pelo Laboratório de Eficiência Energética do Centro de Tecnologia da UFPI, o qual prestou imensa ajuda nos testes necessários para a finalização desta pesquisa, presencialmente e remotamente quando necessário.

À todos os amigos do curso de engenharia elétrica e fora dele que de algum modo me incentivaram e me ajudaram a começar e terminar esse curso.

“Tudo o que temos de decidir é o que fazer com o tempo que nos é dado.”

(O Senhor dos Anéis - A Sociedade do Anel)

RESUMO

Este trabalho apresenta uma plataforma protótipo de baixo custo capaz de executar as principais funções de um relé de proteção, utilizando como base o minicomputador *Raspberry Pi 3B+*, possibilitando que seus usuários possam implementar funções de proteção em *hardware* e realizar ensaios em tempo real. Foram realizados testes que comprovaram a capacidade do *Raspberry Pi 3B+* em fornecer respostas em tempo real, através de configurações específicas em seu Sistema Operacional. Um circuito experimental foi construído para enviar os sinais de faltas elétricas de uma mala de testes de relés de proteção para o *Raspberry Pi*, através do condicionamento dos sinais de tensão e corrente e sua digitalização. Além disso, uma biblioteca de funções foi desenvolvida para executar algoritmos de proteção mais simples, como a detecção de faltas trifásicas simétricas, bem como um algoritmo de proteção mais complexo baseado em redes neurais, capaz de diferenciar faltas trifásicas de transitórios gerados por variações de grandes cargas. Os resultados obtidos mostram que o relé protótipo é capaz de executar funções de proteção simples e complexas dentro de um tempo de atuação pré-definido, além de realizar amostragem dos sinais de tensão e corrente de forma satisfatória.

Palavras-chave: Raspberry Pi. Redes Neurais. Relés. Sistemas de Proteção. Sistemas em Tempo Real.

ABSTRACT

This work presents a low-cost universal relay prototype capable of performing the main functions of a power grid protection relay, based on a Raspberry Pi 3B+ single-board computer. Its users will be able to implement power protection functions in hardware and carry out tests in real-time.Tests carried out proved the Raspberry Pi 3B+'s ability to provide real-time response, by specific settings in its Operational System. An experimental circuit was built to send electrical fault signals from a Relay tester to the Raspberry Pi, by conditioning the voltage and current signals and digitizing them. In addition, a library of functions was developed to run simple power protection algorithms, such as symmetrical three-phase fault detection, as well as a more complex power protection algorithm based on neural networks, capable of differentiating three-phase faults from transients created by large load variations. The results show that the developed relay is capable of running simple and complex protection functions within a pre-defined runtime, in addition to accurate sampling of voltage and current signals.

Keywords: Neural Network. Power Protection Systems. Raspberry Pi. Real-time Systems. Relays.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ALGORITMOS

SUMÁRIO

1 INTRODUÇÃO

A evolução e expansão dos sistemas de potência trazem consigo constantes desafios para o fornecimento seguro e confiável de energia elétrica a seus usuários. Não só a demanda por energia elétrica aumentou significativamente, como também o uso expressivo da geração distribuída e cargas não-lineares por parte dos consumidores, levando a rede elétrica aos seus limites de estabilidade e segurança (KUFFEL; FORSYTH; PETERS, 2016). Em resposta a esse cenário, novas filosofias e dispositivos de proteção foram desenvolvidas ao longo dos anos para contemplar essas mudanças e muitas outras. Um dos equipamentos que constantemente recebe melhorias é o relé de proteção, desde seu formato como dispositivo eletromecânico até evoluir como um *Intelligent Electronic Device* (IED), também conhecido como relé numérico.

Os relés microprocessados tiveram um grande aumento em seu poder computacional e na complexidade de seus sistemas internos, capazes de integrar múltiplas funções de proteção e executar tarefas em tempo real, antes, durante e após a detecção de uma falta elétrica (WANG; DINAVAHI, 2016). Os relés numéricos comerciais são amplamente configuráveis, desde a parametrização de suas funções de proteção, autodiagnóstico de suas funcionalidades, armazenamento de dados de falta elétricas em forma de oscilográficas (de modo simultâneo ao algoritmo de proteção), e até a avaliação de sua resposta em ambientes simulados. No entanto, o *hardware* e *software* utilizados nesses equipamentos são Propriedade Intelectual (PI) de seus fabricantes. Logo, seus algoritmos não podem ser alterados por seus usuários. Com isso, pesquisadores que queiram testar um novo algoritmo ou filosofia de proteção em condições próximas da realidade dos sistemas de potência, geralmente desenvolvem suas próprias plataformas e dispositivos para essa finalidade. Na literatura, os dispositivos em *hardware* mais utilizados para testes em tempo real de algoritmos de proteção, por parte dos pesquisadores, são: *Field Programmable Gate Arrays* (FPGA's), Plataformas-protótipo e Microcontroladores (WANG; DINAVAHI, 2016; MONARO *et al.*, 2012; MARTINS, 2017).

As FPGA's são circuitos integrados programáveis, formados por pequenos blocos lógicos, cuja funcionalidade desejada pode ser atribuída pelo usuário. Esses dispositivos possuem muitas características desejadas por sistemas de proteção digitais, como seus blocos lógicos configuráveis, arquitetura inerentemente paralela, baixa latência durante a interface com periféricos, além de alta velocidade de processamento (ORDONEZ; PENTEADO; DA SILVA, 2005). Trabalhos como o de Wang e Dinavahi (2016), que propuseram um relé de proteção multifuncional em tempo real em uma FPGA do modelo Virtex-7 Xilinx®, e Mitra e Chattopadhyay (2019), que

desenvolveram uma função de proteção de sobrecorrente adaptativa utilizando uma Rede Neural Artificial (RNA), embarcado em uma FPGA do modelo Cyclone IV Intel®, mostram a grande utilidade desta classe de dispositivos em sistemas de proteção. No entanto, o alto custo financeiro na compra de uma FPGA pode ser um empecilho para o uso dessa plataforma. Além disso, para programar esse dispositivo, é necessário que o usuário tenha um profundo conhecimento da arquitetura da FPGA utilizada, para explorar ao máximo seus recursos.

Entende-se por plataformas-protótipo como circuitos embarcadas robustos, geralmente utilizados em aplicações industriais, os quais foram adaptados como relés digitais. Uma de suas vantagens em comparação com as FPGA's são seus blocos lógicos fixos, permitindo ao usuário o uso de linguagens de programação de alto nível; ainda mantendo poder computacional. Os trabalhos de Monaro *et al.* (2012) e Pellini *et al.* (2013) são exemplos de aplicações bem-sucedidas desses dispositivos em pesquisas na área de proteção de sistemas de potência. Monaro *et al.* (2012) desenvolveram um sistema integrado utilizando um computador *PC / 104 Consortium* capaz de executar algoritmos de proteção em tempo real, o qual foi utilizado para executar um conjunto de novas funções de proteção para geradores síncronos, com base em lógica *fuzzy* (MONARO, 2013). Quanto a Pellini *et al.* (2013), estes adaptaram um computador industrial da marca IBM para emular a operação de um religador automatizado para alimentadores de distribuição, o qual foi adaptado por Dantas, Pellini e Manassero Junior (2018) para executar seu algoritmo de proteção diferencial no domínio do tempo. Contundo, assim como as FPGA's, tais plataformas possuem alto custo aquisitivo.

Por fim, os microcontroladores são comumente utlizados por pesquisadores em diversas aplicações na engenharia, devido à especificidade que esses dispositivos possuem para executar tarefas (ORDONEZ; PENTEADO; DA SILVA, 2005). Além disso, em comparação com as FPGA's e as plataformas-protótipo, eles são muito mais acessíveis e baratos. Um exemplo de sua aplicação na proteção de sistemas de potência é o trabalho de Joy *et al.* (2022), o qual utilizou o microcontrolador Arduino Nano como base do seu relé de proteção para alimentadores, o qual continha funções de proteção de sobrecorrente, sub e sobretensão, proteção de terra, dentre outras. Outro exemplo é o trabalho de Martins (2017), que usou uma placa de desenvolvimento MSP432P401R da *Texas Instruments* para executar um algoritmo de proteção que atua como um religador adaptativo para linhas de transmissão de alta tensão. Embora os microcontroladores utilizem programação em baixo nível, ferramentas online como TensorFlow (2021), permitem que algoritmos mais complexos sejam embarcados nesse tipo de dispositivo.

Em suma, uma plataforma ideal para testar funções de proteção para sistemas de potência deve possuir as seguintes características: baixo custo e acessibilidade, capacidade de lidar com programação de alto nível (capaz de executar um Sistema Operacional, por exemplo), e ser capaz de executar algoritmos de proteção em tempo real. Com as características apresentadas, este tipo de plataforma pode ser utilizada para a validação de algoritmos de proteção. Um tipo de plataforma que possui tais características são os minicomputadores. Uma vez que pouco se sabe sobre a aplicação desse tipo de plataforma em sistemas de proteção, este trabalho investiga a possibilidade de se utilizar um determinado minicomputador, o *Raspberry Pi 3B+*, como um sistema dedicado em tempo real para executar funções de proteção, desde funções mais simples, como a detecção de faltas trifásicas simétricas, até algoritmos de proteção mais complexos, como os baseados em *machine learning* (inteligência computacional) (MITRA; CHATTOPADHYAY, 2019; MONARO, 2013). Esse tipo de aplicação de baixo custo possibilitará que outros pesquisadores possam realizar testes em tempo real de seus algotimos de proteção.

1.1 Objetivos

O principal objetivo deste trabalho é o desenvolvimento de uma plataforma de baixo custo, *hardware* e *software*, capaz de executar algoritmos de proteção em tempo real. Ela deve ser capaz de identificar faltas elétricas e responder a elas em tempo real (tempo pré-definido pelo usuário). Para atingir esse objetivo geral, os seguintes objetivos específicos foram definidos:

- Avaliar a capacidade do *Raspberry Pi 3B+* em entregar uma resposta em tempo real aceitável para sistemas de proteção;
- Criar uma biblioteca de comandos em *software* que englobe desde amostragem de sinais de tensão e corrente, até medição e funções de proteção;
- Construir uma plataforma em *hardware* que englobe os principais elementos de um relé digital, comparando sua performance a um relé comercial;
- Modelagem de um alimentador em média tensão para gerar sinais de falta trifásicas, a serem fornecidos ao relé protótipo;
- Desenvolver um algoritmo de proteção baseado em redes neurais, verificando a capacidade do *Raspberry Pi 3B+* em executar algoritmos de proteção complexos em um tempo pré-determinado.

1.2 Estrutura do trabalho

A Seção 2 deste trabalho aborda a estrutura básica do *Raspberry Pi 3B+*, os principais conceitos relacionados aos sistemas em tempo real e os testes realizados neste minicomputador para verificar sua latência e atrasos inerentes.

Os componentes em *hardware* utilizados para compor o relé protótipo são apresentados na Seção 3 deste trabalho, bem como a biblioteca de proteção desenvolvida que englobam as funções de amostragem, medição e proteção dessa plataforma.

A Seção 4 apresenta a modelagem de um alimentador em média tensão no software ATP/ATPDraw, de modo a gerar os sinais de faltas trifásicas, aos quais o relé protótipo será submetido. Além disso, é apresentada a rede neural desenvolvida para diferencias faltas trifásicas desse alimentador em comparação com sinais transitórios gerados por grandes variações de carga.

Em seguida, a Seção 5 apresenta os resultados dos testes no relé protótipo, com foco seu processo de amostragem, tempo de execução para uma falta de sobrecorrente. Seu desempenho é comparado com um relé comercial de proteção para avaliar a eficácia e limitações da plataforma desenvolvida. Além disso, a sua precisão ao executar a rede neural desenvolvida também é mensurada.

Por fim, a Seção 6 apresenta as conclusões do trabalho e sugestões para a continuidade da pesquisa.

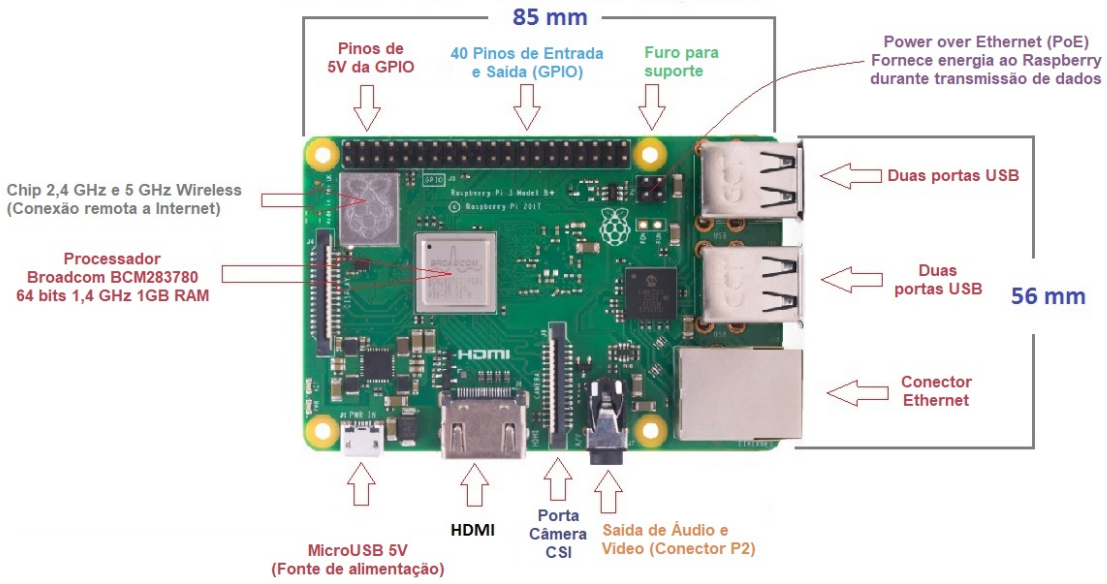
2 RASPBERRY PI E SISTEMAS EM TEMPO REAL

Esta Seção apresenta os principais conceitos relacionados a plataforma *Raspberry Pi* e sistemas em tempo real, além de uma revisão bibliográfica sobre o uso desse minicomputador em pesquisas relacionadas à sistemas de potências. Na sequência, são apresentadas as configurações necessárias para que o *Raspberry Pi* entregue uma resposta em tempo real. Por fim, foram realizados testes de latência no *Raspberry Pi 3B+* para avaliar se sua resposta em tempo real é satisfatória para sistemas de proteção.

2.1 Raspberry Pi 3B+

O *Raspberry Pi* é uma série de plataformas educacionais focadas no ensino da computação e marketing digital (RASPBERRY PI FOUNDATION, 2021). Ele tem dimensões físicas e recursos semelhantes aos de um microcontrolador Arduino, mas possui grande poder computacional e memória, o que o torna capaz de executar Sistemas Operacionais (SO). Juntamente com seu baixo custo, esse tipo de plataforma tornou-se popular entre vários campos acadêmicos, especialmente em pesquisas relacionadas a Internet das Coisas (IoT) (AQUEL, 2018). Além disso, existem inúmeras comunidades *online* dedicadas a divulgar soluções das mais variadas aplicações, utilizando estes minicomputadores. A Figura 1 apresenta o minicomputador *Raspberry Pi 3B+* e seus principais componentes.

Figura 1 – Minicomputador Raspberry 3B+ e seus principais componentes



Fonte: Adaptado de Aquel (2018).

Dentre os componentes que formam o *Raspberry Pi 3B+*, aqueles mais relevantes para o relé protótipo deste trabalho são (RASPBERRY PI FOUNDATION, 2021):

- CPU - processador quad-core ARMv8; 64 bits; 1,4 GHz; o que indica o significativo poder computacional desse minicomputador;
- GPIO – Sigla para *General Purpose Input/Output*; são portas programáveis que permitem a comunicação desse minicomputador com outros periféricos;
- Sistema operacional - Raspberry Pi OS (anteriormente chamado de Raspbian), o qual é uma distribuição Linux.

No âmbito acadêmico, há precedência no uso do *Raspberry Pi* como uma plataforma que integra funções relevantes para sistemas de potência, além de demonstrar sua capacidade em entregar resposta em tempo real.

John *et al.* (2017) realizaram a automação de uma subestação de 11 kV com o *Raspberry Pi*, atuando como um Controlador Lógico Programável (CLP). Utilizando a comunicação serial RS 485/USB com diversos transdutores (tensão, corrente, potência, etc.) e estabelecendo uma relação mestre/escravo com eles, os autores demonstraram que a subestação opera de forma eficiente ao se utilizar esse minicomputador.

Walter, Fakih e Gruttner (2013) utilizaram o *Raspberry Pi* como um simulador de um sistema de controle eletromecânico em tempo real, o qual se comunica com o ambiente *Simulink* do Matlab®, no qual um motor e sensores foram modelados. Todos os cálculos relativos à modelagem do motor, leitura dos sensores e atualização dos autuadores foram realizados em uma escala de tempo de 100 μs, validando o sistema de controle.

Leccese *et al.* (2016) desenvolveram um analisador de qualidade de energia de baixo custo com o *Raspberry Pi*. O sistema foi testado em algumas cidades italianas, a fim de averiguar a qualidade de energia do país. Os resultados mostraram que o desempenho do sistema proposto é equiparável aos analisadores de energia que já se encontram instalados pelo país.

Hernandez *et al.* (2017) utilizaram o *Raspberry Pi* como um simulador de fluxo de potência. O minicomputador foi alimentado pelo simulador em tempo real *Opal RT*, que forneceu os dados de tensão e corrente de um sistema de potência teste do IEEE 37 barras. Esse sistema é representado por uma matriz de admitância com 13689 elementos, dos quais somente 1031 eram conhecidos (7,5% dos valores). O trabalho citado mostrou que o *Raspberry Pi* foi capaz de fornecer a solução para esse fluxo de potência em uma escala de tempo na ordem de 500 μs, sendo uma resposta em tempo real satisfatória para sistemas de potência complexos.

2.2 Sistemas em tempo real

O significado do termo “tempo real” pode variar de acordo com o contexto no qual é utilizado. Neste trabalho, Sistemas Operacionais em Tempo Real (RTOS) são sistemas computacionais que executam tarefas em um tempo preciso. Com isso, eles são capazes de receber informações ou dados, processá-los e retornar uma resposta ou ação em um tempo pré-determinado (BUTTAZZO, 2013). Para sistemas de proteção, esse tipo de resposta é extremamente importante, uma vez que eles devem detectar faltas elétricas e operar periféricos com o mínimo de atraso. O resultado de uma dessas tarefas sendo executada precoce ou tardiamente pode resultar no fracasso dessa ação; resultando em danos físicos à infraestrutura elétrica de uma rede ou até mesmo causar ferimentos em pessoas (HERNANDEZ *et al.*, 2017).

De acordo com Buttazzo (2013), os principais conceitos associados aos sistemas em tempo real são:

- Determinismo: o sistema deve sempre responder a um evento específico (dados de entrada do sistema) sempre da mesma maneira;
- Previsibilidade: cada tarefa de um sistema deve acontecer dentro de um intervalo de tempo específico;
- Performance: os sistemas em tempo real devem ter baixa latência (tempo gasto entre o envio e o recebimento de uma informação), baixo atraso (desvio do tempo da resposta usual) e alta taxa de transferência de dados.

Dependendo do tipo de tarefa que um sistema em tempo real deve lidar, eles podem ser classificados nos seguintes subgrupos (BUTTAZZO, 2013):

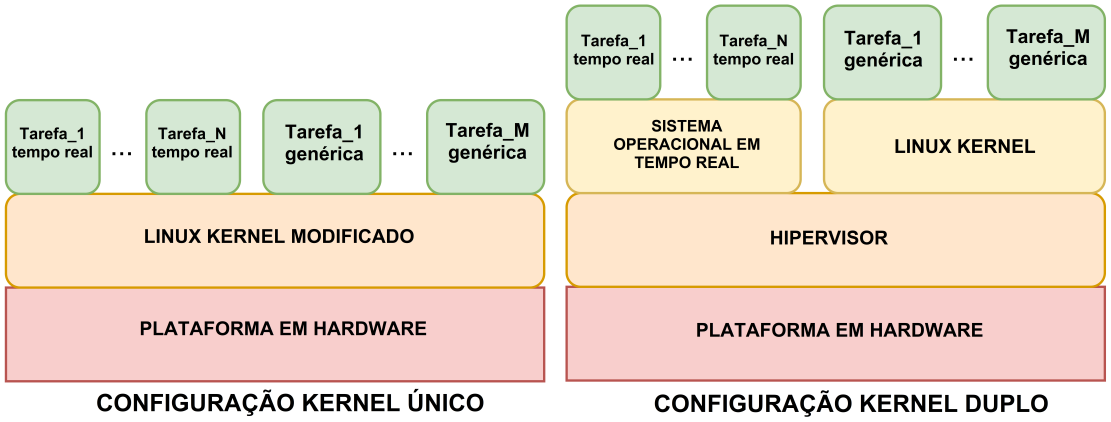
- Sistema em tempo real crítico (*hard real-time*): as tarefas geridas por ele devem ser executadas em um tempo pré-determinado. Qualquer desvio no tempo definido pode gerar consequências catastróficas ao sistema que afetam;
- Sistema em tempo real firme (*firm real-time*): as tarefas geridas por ele possuem um tempo pré-determinado para serem executadas. Caso um atraso ocorra na resposta do sistema, a informação fornecida não é mais útil. Porém, esses atrasos não danificam o ambiente que afetam;
- Sistema em tempo real leve (*soft real-time*): as tarefas geridas por ele possuem um tempo pré-determinado para serem executadas, porém mesmo que haja atrasos em sua resposta, os resultados ainda podem ser aproveitados pelo sistema que afetam.

As características mencionadas anteriormente não são atendidas por Sistemas Operacionais de Propósito Geral, como os sistemas operacionais Windows e Linux, uma vez que estes são focados no desempenho geral do sistema, ao invés de executar tarefas específicas com restrição de tempo (JOHANSSON, 2018). Portanto, IED's comerciais normalmente possuem sistemas operacionais desenvolvidos especificamente para aplicações em sistemas de proteção, a fim extinguir faltas elétricas em tempos de resposta críticos.

Devido à popularidade do SO Linux entre seus usuários para as mais diversas aplicações (por ser uma plataforma de código aberto), algumas abordagens foram criadas para torná-lo mais adequado para computação em tempo real. A mais comum delas é a configuração do *kernel* do Linux (núcleo do sistema operacional, responsável por gerenciar os dados dos aplicativos do sistema, bem como lidar com a comunicação de *hardware* interno e periféricos). De acordo com Johansson (2018), as principais abordagens para configuração em tempo real do *kernel* do Linux são ilustradas na Figura 2.

- Kernel Único - modifica o *kernel* do Linux para tornar seu desempenho e tempo de resposta mais previsível;
- Kernel Duplo - cria um *kernel* secundário virtual (também conhecido como hipervisor), que lida com o tempo de processamento, espaço de memória e etc. Esta configuração prioriza tarefas em tempo real ao invés das rotinas comuns do Linux.

Figura 2 – Modificação kernel único e duplo para computação em tempo real do Linux



Fonte: Adaptado de Johansson (2018).

2.3 Configurando o Raspberry Pi para atuação em tempo real

Conforme mencionado em Molloy (2016), o *Raspberry Pi* não foi originalmente criado para computação em tempo real, por se tratar de uma plataforma de cunho educacional. Pelo fato do Raspberry Pi SO ser um sistema de propósito geral como o Linux, é necessário configurar seu *kernel* para que ele entregue respostas em tempo real. Muitas das comunidades *online* dedicadas ao *Raspberry Pi* já fornecem alguns *patches* (arquivos de configuração) gratuitamente, os quais modificam o *kernel* do *Raspberry Pi*, possibilitando a execução de tarefas em tempo real. Na literatura, os *patches* mais comumente utilizados pelos usuários são o Preempt RT e o Xenomai.

O Preempt RT foi um projeto colaborativo *online*, iniciado em 2005, e desde então vem recebendo melhorias e adaptações para outras plataformas, como o *Raspberry Pi*. De acordo com Mckenney (2005), o principal objetivo do projeto é tornar o *kernel* do Linux mais preemptivo, o que significa melhorar a capacidade de interrupção de rotinas por tarefas de maior prioridade, além de diminuir a latência e atraso nessas tarefas. Esse *patch* segue a abordagem do *kernel* único. Contudo, uma de suas desvantagens é executar as tarefas em tempo real, embora com prioridade, no mesmo nível que as tarefas comuns do *kernel*. Um erro nas tarefas em tempo real pode causar falha de segmentação (erro no acesso aos dados de memória do sistema), resultando na falha de todo o sistema como consequência. Para retornar o sistema à normalidade, é necessário reinicializar o minicomputador.

Quanto ao Xenomai, o qual utiliza a abordagem de *kernel* duplo, cria um co-kernel que lida com as tarefas de maior prioridade, delegando as tarefas de menor prioridade para o *kernel* não-preemptivo (GERUM, 2022). Por se tratar de dois *kernels* trabalhando de forma simultânea, foi criada uma API (Interface de Programação de Aplicações) com funções e rotinas específicas para o tempo real, sendo necessário aprendê-la antes de usar esse *patch* de maneira correta.

Na literatura, a exemplo dos trabalhos de Chalas (2015) e Johansson (2018), nota-se que o *patch* Xenomai tem melhor desempenho em tempo real do que o *patch* Preempt RT para a plataforma *Raspberry Pi*. Contudo, este trabalho de Mestrado optou pelo uso do *patch* Preempt RT, uma vez que requer do usuário apenas prioridade de manipulação de tarefas durante a codificação de um algoritmo.

Neste trabalho, foi utilizado o *patch* fornecido como código aberto por Tam (2018) para configurar o *Raspberry Pi 3B+* com o Preempt RT. Para minimizar a ocorrência de erros

como a falha de segmentação, utilizou-se da solução proposta por Roberts (2013): dedicar um dos *cores* (núcleos) do *Raspberry Pi* para as tarefas em tempo real (também em código aberto). Com essa configuração, Roberts (2013) mostrou que o *Raspberry Pi 3B +* pode atingir latências inferiores a 3 μs (cerca de cinco mil vezes menor que um ciclo de 60 Hz de um sinal CA), com pouca variação nas medições. Uma vez que a configuração citada faz uso da linguagem C/C++, foi utilizada essa linguagem de programação como base para a biblioteca de tempo real do relé protótipo.

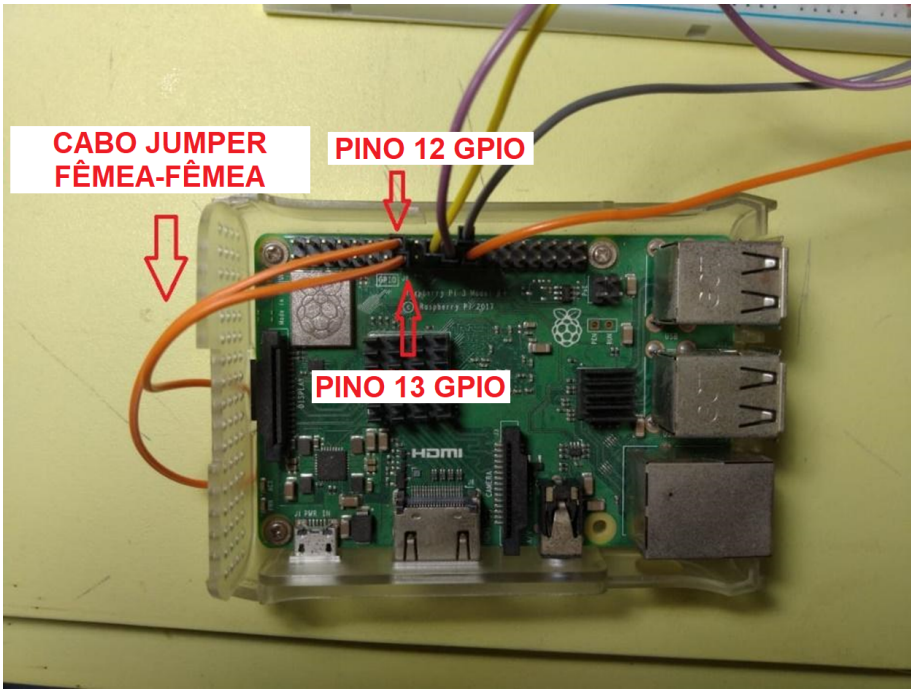
2.4 Performance em tempo real do Raspberry Pi 3B+

Chalas (2015) e Johansson (2018) mostram que o *Raspberry Pi* é capaz de entregar resposta em tempo real, testando os *patchs* já citados nas versões *2B* e *3B* dessa plataforma, respectivamente. Em suas conclusões, eles ressaltam que, embora as modificações do *kernel* tenham dado a esse minicomputador baixa latência e alto desempenho durante a execução de tarefas em tempo real, o *Raspberry Pi* OS não satisfez certos critérios para ser classificado como um sistema em tempo real crítico (*hard real-time*), capaz de executar uma tarefa ou função complexa com restrição de tempo de 10 μs. Uma vez que este trabalho combina a abordagem de *kernel* único (Preempt RT) com a abordagem de núcleo dedicado de Roberts (2013), os seguintes testes foram realizados a fim de avaliar a resposta em tempo real do *Raspberry Pi 3B +* nesta nova configuração:

- Teste de latência em *software* – a latência do *Raspberry Pi 3B+* foi medida após atrasos aleatórios de 1-1100 μs serem inseridos na biblioteca em tempo real. Para este teste, a tarefa escolhida está vazia, a fim de avaliar os atrasos inerentes da própria biblioteca. A latência é medida pela diferença entre o atraso real da tarefa e o atraso solicitado (gerado aleatoriamente). Este teste contempla atrasos curtos (de até 50 µs) e longos (de até 1100 µs);
- Teste de latência em *hardware* - dois pinos GPIO do *Raspberry Pi 3B +* foram conectados eletricamente. Esta tarefa foi executada em um *thread* separado (em outro núcleo do *Raspberry Pi*), criando bordas de subida em momentos específicos no pino de saída enquanto o pino de entrada lê continuamente os valores digitais. A latência (diferença entre esses dois momentos) foi medida na camada principal da biblioteca de tempo real. Para esse teste, os pinos 12 (configurado como saída) e 13 (configurado como entrada) foram conectados por

um por um cabo *jumper* fêmea-fêmea, como mostra a Figura 3

Figura 3 – Teste de latência em hardware do Raspberry Pi 3B+



Fonte: Autoria própria.

Os testes citados estão relacionados a eventos que podem ocorrer no mais simples sistema de proteção, como atrasos aleatórios originados da comunicação entre os dispositivos, ou má conexão elétrica entre eles. Grandes atrasos originados desses problemas podem acarretar na falha de um sistema de proteção, como um atraso em um sinal de *trip* (sinal de desligamento enviado por relés de proteção) para extinguir uma falta elétrica, ou instabilidade no sistema de potência devido a atrasos na desconexão dos trechos afetados por uma falta (WEI; CHEN, 2010).

A Tabela 1 apresenta os resultados dos testes de latência feitos no *Raspberry Pi 3B+*, o qual executou os testes citados por quatro horas, realizando várias medições de latência a cada minuto (tempo de atualização do teste). O contador em *hardware* do *Raspberry Pi 3B+*, que realiza incrementos a uma taxa de 1 MHz, foi utilizado para medir as latências com resolução de microssegundos.

Os testes de latência mostram que a operação do *patch* Preempt RT enquanto um dos núcleos (*cores*) do *Raspberry Pi 3B+* é dedicado para as tarefas em tempo real, melhora significativamente a precisão deste minicomputador. Para todos os testes, a latência medida é em torno de 1 μs, com precisão acima de 99%, uma vez que a latência média e a variância de cada teste são muito próximas. O valor mínimo de latência para o teste GPIO é zero devido à precisão

Tabela 1 – Resultado dos testes de latência para o Raspberry Pi 3B+

	Latência (μs)			
Testes	**Média**	**Mínima**	**Máxima**	**Variância**
Atrasos curtos (até 50 μs)	1,03	1,0	41,0	1,03
Atrasos Longos (até 1100 μs)	1,02	1,0	42,0	1,02
Hardware (GPIO)	0,8	0	32,0	0,8
Número de amostras em quatro horas: 16.364.428,00				

Fonte: Autoria própria.

do contador usado, o que significa que valores de latência menores que 1 μs não puderam ser medidos neste teste.

Embora esses testes mostrem que o *Raspberry Pi 3B+* é capaz de computação em tempo real, o algoritmo executado durante o teste foi a própria biblioteca em tempo real. Assim, é necessário avaliar se o *Raspberry Pi 3B+* é capaz de manter a resposta em tempo real durante a execução de tarefas mais complexas, como um algoritmo de proteção. Dessa forma, o tipo de sistema real que ele é capaz de lidar pode ser definido (sistema em tempo real leve, firme ou crítico).

2.5 Considerações Finais

Este capítulo apresentou o *hardware* a ser utilizado como base do relé protótipo deste trabalho: (*Raspberry Pi 3B+*), bem como as configurações necessárias no *kernel* do seu SO para que ele possa entregar uma resposta em tempo real. Os testes de latência apresentados mostraram que o *Raspberry Pi 3B+* possui atrasos mínimos em sua biblioteca em tempo real, tornando-o uma plataforma promissora para executar algortimos de proteção em tempo real. Logo, o próximo na construção do relé protótipo deste trabalho será escolher os componentes eletrônicos adequados, bem como codificar uma biblioteca de funções de relés digitais dentro do *Raspberry Pi 3B+*.

3 PLATAFORMA PROPOSTA PARA O RELÉ PROTÓTIPO

Esta Seção apresenta a configuração em *hardware* e *software* para a implementação do relé protótipo, detalhando os componentes utilizados, suas conexões, e as principais funcionalidades da plataforma proposta.

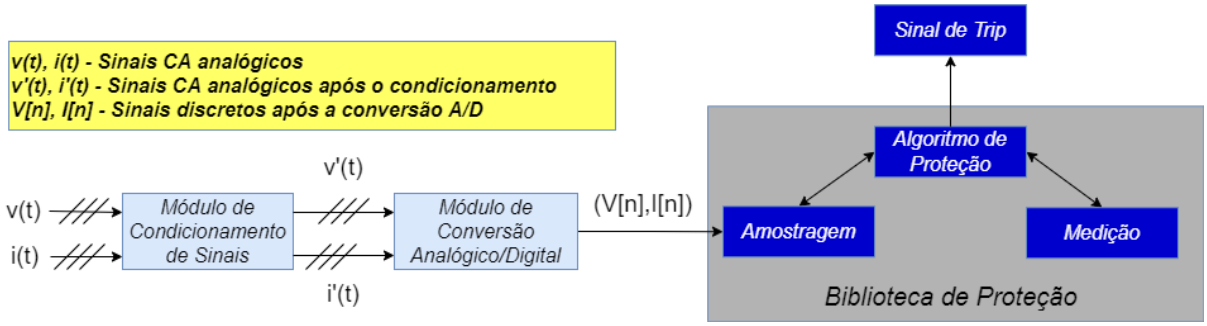
3.1 Considerações Iniciais

De modo a emular os principais subsistemas de um relé comercial para proteção, as seguintes funcionalidades foram implementadas no relé protótipo deste trabalho:

- Amostragem – obtenção de amostras no tempo discreto dos sinais analógicos de tensão e corrente de um sistema de potência;
- Medição – cálculo dos parâmetros elétricos dos sinais de entrada;
- Proteção – espaço onde as funções de proteção são programadas.

A Figura 4 apresenta o diagrama de blocos do relé protótipo. Cada bloco será detalhado nesta Seção do trabalho, desde os componentes utilizados, até quais dados de entrada o usuário desta plataforma deve inserir para que ela funcione corretamente (o diagrama elétrico do relé protótipo, bem como o custo estimado de cada componente é detalhado no Apêndice A deste trabalho).

Figura 4 – Diagrama de blocos para o relé protótipo



Fonte: Autoria própria.

3.2 Módulo de Condicionamento de Sinais

Este módulo consiste de três sensores de tensão e três sensores de corrente, os quais recebem sinais analógicos de tensão e corrente de uma fonte de energia, e os ajustam para valores menores de tensão (faixa de tensão de 0 a 5 V). Dessa forma, os sinais analógicos citados

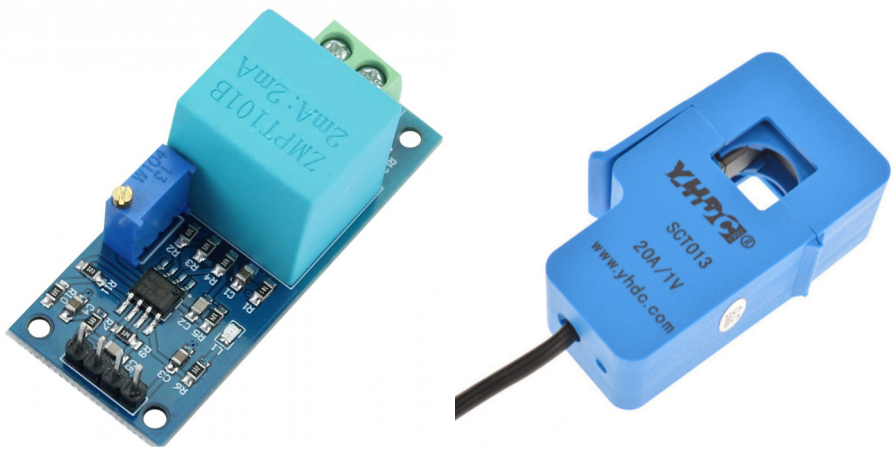
atenderão os parâmetros de entrada do Módulo do Conversão Analógico/Digital. Os principais dados técnicos dos sensores utilizados é apresentado na Tabela 2. Já a Figura 5 apresenta visualmente os sensores utilizados.

Tabela 2 – Principais parâmetros dos sensores do Módulo de Condicionamento de sinais

Dados Técnicos	**Sensor de Corrente**	**Sensor de Tensão**
Modelo	SCT – 013 – 20	ZMPT101B
Dados de Entrada	0 – 20 A (corrente alternada)	0 – 250 VAC
Dados de Saída	0 – 1 V (tensão alternada)	0 – 5 V (tensão alternada)
Observações	Sensor de Corrente não-invasivo	Ganho do sensor pode ser ajustado manualmente

Fonte: ETC (2021), YHDC (2021).

Figura 5 – Sensores de tensão e corrente utilizados no relé protótipo



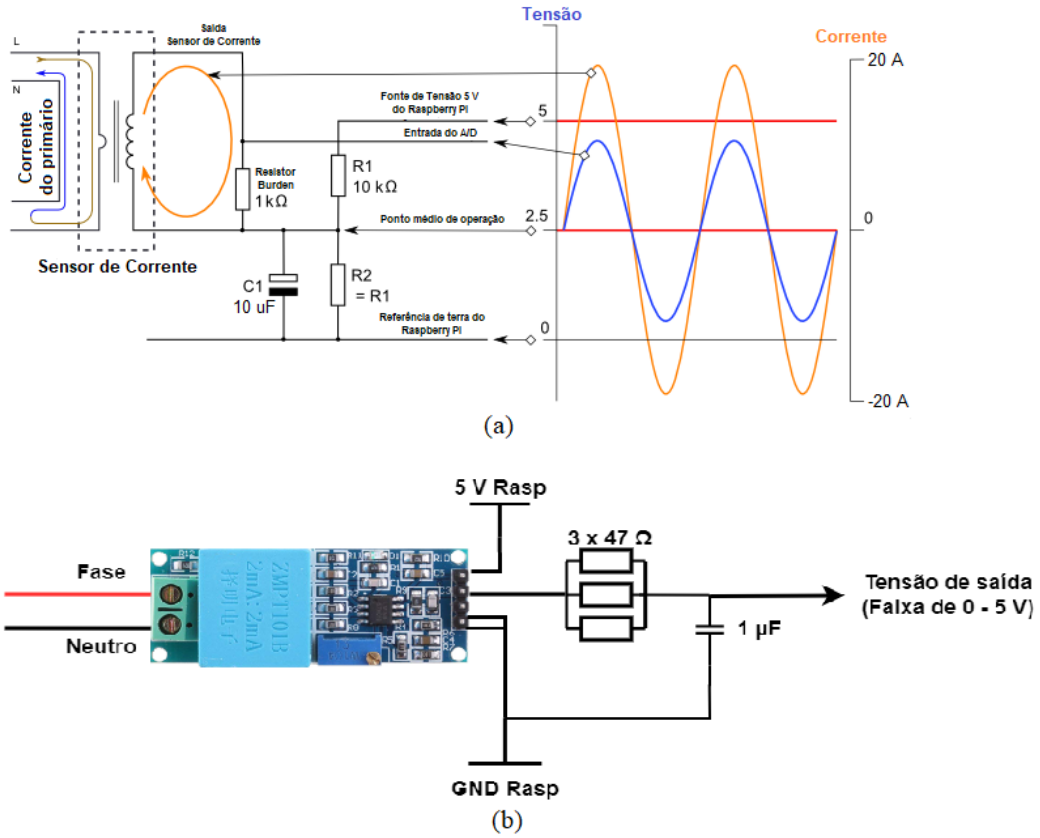
(a) Sensor de tensão ZMPT101B (b) Sensor de corrente SCT – 013 – 20

Fonte: ETC (2021), YHDC (2021).

Vale ressaltar que foi necessária uma adaptação no circuito interno dos sensores de corrente citados, uma vez que sua saída de tensão original (0 - 1 V) não era compatível com a faixa de tensão desejada (0 - 5 V). Com isso, o resistor *burden*[1] interno de cada um desses sensores foi removido. Além disso, utilizou-se um circuito de condicionamento na saída de cada sensor de corrente, com base no design apresentado por Learn Open Energy Monitor (n.d.). Quanto aos sensores de tensão, utilizou-se um filtro passa-baixa com uma frequência de corte em torno de 10 kHz, a fim de mitigar o impacto de ruídos vindos da fonte de energia, além de servir como um filtro *anti-aliasing*. As modificações citadas são apresentadas na Figura 6.

[1] Resistor que limita a tensão de saída do TC

Figura 6 – Circuitos do módulo de condicionamento (a) corrente (b) tensão



Fonte: Adaptado de Learn Open Energy Monitor (n.d.).

3.3 Módulo de Conversão A/D

Para fornecer sinais discretos ao *Raspberry Pi*, foi necessário o uso de um conversor A/D. Neste trabalho, foi utilizado o conversor A/D MCP3008, fabricado pela *Microchip*, devido a sua alta taxa amostral (até 200 ksps), além de sua interface serial SPI integrada, o que resulta em uma comunicação síncrona com o minicomputador (maior controle da amostragem dos sinais analógicos). Este conversor possui oito portas de entrada analógicas e uma resolução de 10 bits, calculada a partir de aproximações sucessivas (MICROCHIP, 2008). Dessas oito portas, seis foram utilizadas para receber os sinais de saída do módulo de condicionamento, como mostra a Figura 7. Uma vez que o *Raspberry Pi 3B+* opera em um nível lógico de tensão de 3,3 V, enquanto o conversor A/D opera em 5 V, um conversor lógico bidirecional foi utilizado para permitir a comunicação entre eles. A Figura 7 também mostra a conexão entre os dois equipamentos e os pinos utilizados. Por fim, destaca-se que o pino GPIO 21 do *Raspberry Pi 3B+* foi configurado como saída, para emitir o sinal de trip caso uma falta elétrica seja detectada.

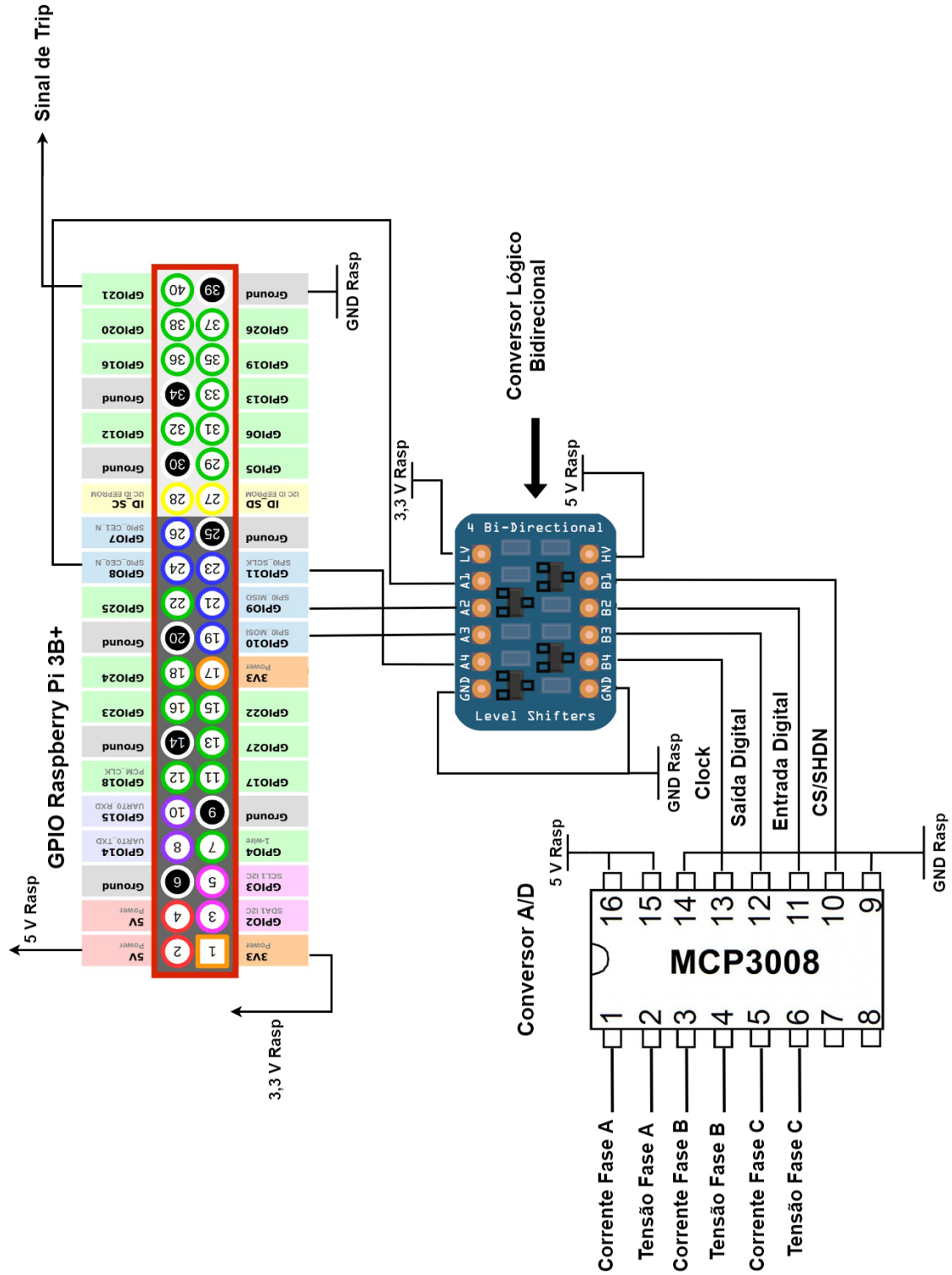


Figura 7 – Conexão entre o MCP3008 e o Raspberry Pi 3B+

Fonte: Autoria própria.

3.4 Biblioteca de Proteção

Programada dentro *Raspberry Pi 3B+*, a biblioteca de proteção é responsável pela amostragem dos sinais de tensão e corrente; medição dos parâmetros elétricos dos sinais de entrada, como o valor médio e eficaz (RMS); execução das funções de proteção selecionadas pelo usuário; e por salvar os dados amostrados em um arquivo de oscilografia (extensão *.csv*), o qual contempla dados pré-falta, falta e pós-falta. A exceção da função de oscilografia, todos os processos contidos no módulo de proteção são controlados pela biblioteca de tempo real descrita anteriormente neste trabalho, de modo a garantir precisão no tempo de atuação de cada processo. Com as funcionalidades citadas, um(a) usuário(a) do relé protótipo é capaz de usar essa biblioteca para executar um algortimo de proteção em *hardware* e em tempo real.

Além de contemplar inúmeros funções de proteção em um único dispositivo, relés numéricos também são capazes de armazenar dados de falta Durante os testes em laboratório do relé protótipo, foi constatado que ele não é capaz de gerar uma oscilografia dos sinais de entrada e executar uma função de proteção paralelamente, visto que isto prejudica sua atuação em tempo real. Esse comportamento se deve a estruturação da biblioteca em tempo real desenvolvida neste trabalho: ela lida com as tarefas em tempo real e as tarefas comuns do Raspberry Pi SO no mesmo nível (modelo do *kernel* único). Com isso, a execução de dois processos que exigem bastante poder de processamento do relé protótipo pode causar erro de falha de segmentação na biblioteca de proteção. Logo, o usuário desta plataforma deve escolher o modo de operação do relé protótipo antes de utilizá-lo.

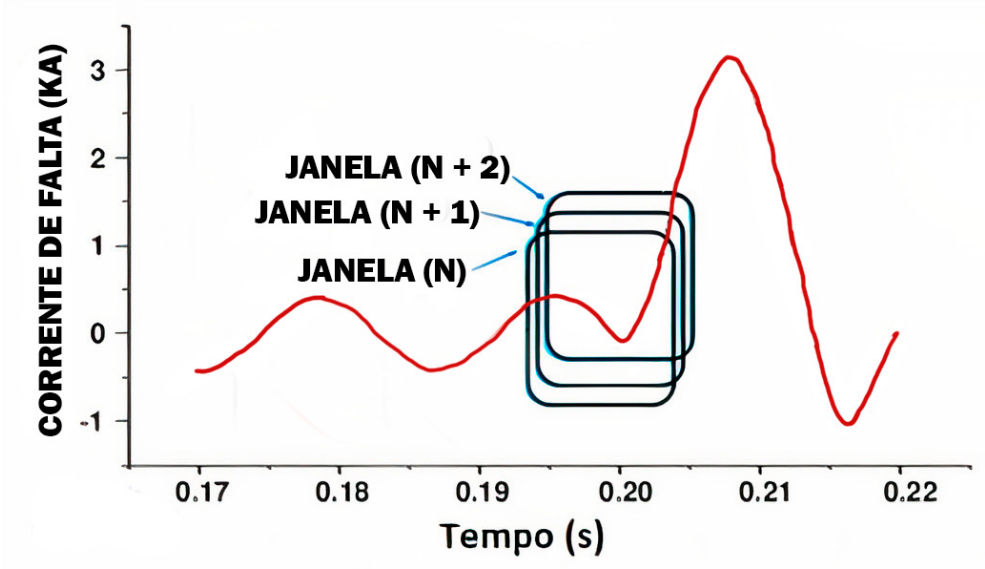
Esta biblioteca foi criado em linguagem C/C++ com base no código fornecido por Ryan (2020). Para executá-lo apropriadamente, quatro parâmetros de entrada são necessários, como apresentado a seguir:

- A quantidade de períodos (ciclos) de um sinal CA, para gerar a janela de dados;
- Frequência de amostragem desejada em kHz;
- Número de sinais de entrada, limitado ao máximo de oito (mesmo número de entradas do conversor A/D adotado neste trabalho);
- Diretório do caminho de pastas dentro do Raspberry Pi SO onde o arquivo *.csv* dos dados amostrados será salvo, juntamente com o nome do arquivo.

O termo *janela de dados* refere-se ao conjunto de amostras que serão utilizadas nas etapas de medição e proteção do relé protótipo. Cada vez que uma nova amostra é requisitada pelo algoritmo de proteção, a janela de dados é atualizada, descartando a amostra mais antiga e

adicionando a mais recente dentro dela. Esse processo é ilustrado na Figura 8.

Figura 8 – Processo de atualização da janela de dados



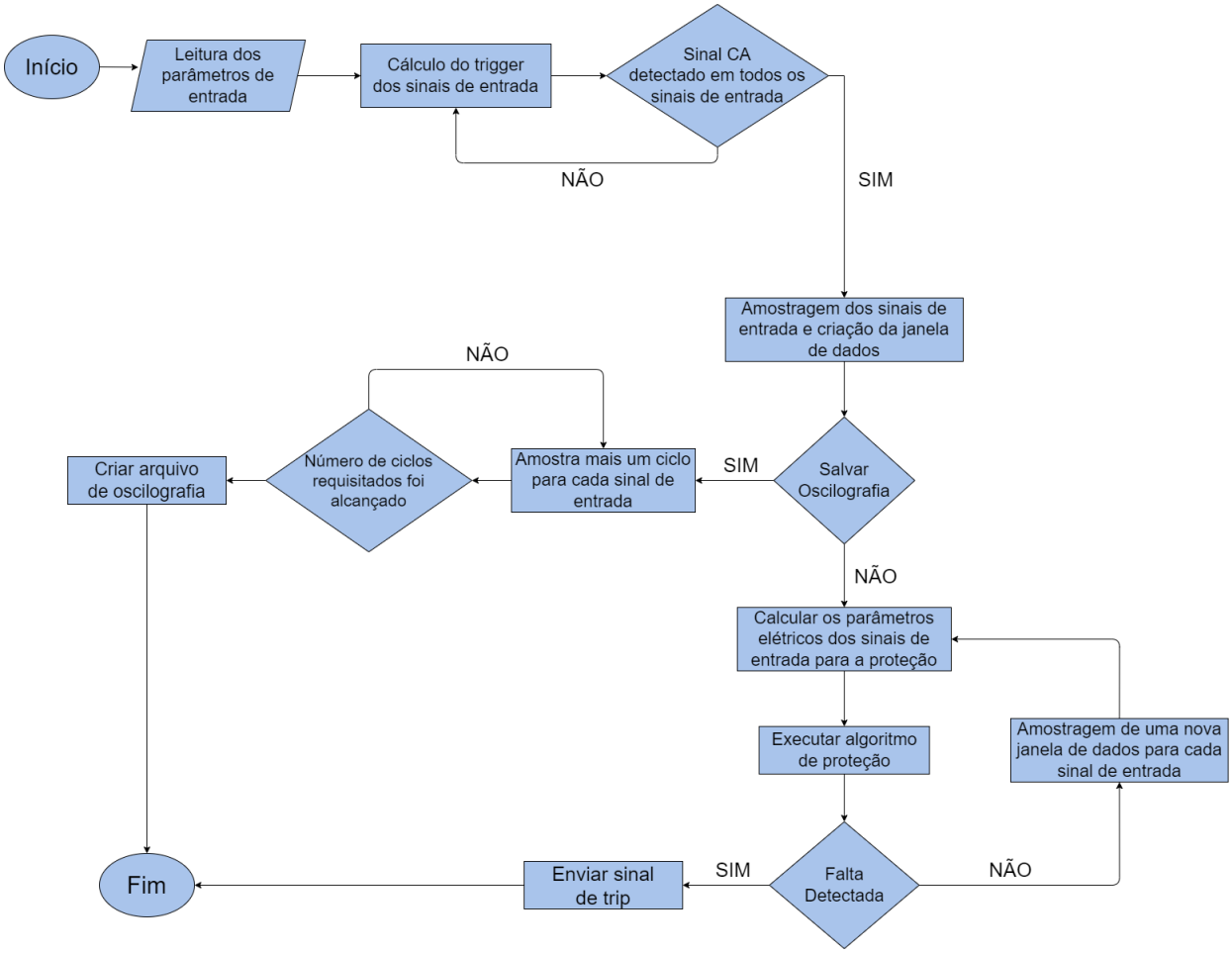
Fonte: Adaptado de Wang e Dinavahi (2016).

O relé protótipo possui dois modos de operação: amostragem (capaz de salvar vários ciclos dos sinais de entrada em um arquivo de oscilografia) ou proteção (realiza a amostragem de meio período ou período completo dos sinais de entrada e envia esses dados para as funções de medição e proteção). Cabe ao usuário do relé protótipo escolher o modo de operação. Além disso, antes de algum dos modos de operação ter sido escolhido, a biblioteca de proteção possui uma função para detectar sinais CA em sua entrada, através do cálculo do *trigger* de cada um deles. A Figura 9 apresenta um fluxograma que resume os modos de operação do relé protótipo. O Apêndice B deste trabalho apresenta mais detalhes de como operar o relé protótipo dentro do ambiente do *Raspberry Pi*.

3.5 Considerações Finais

Este capítulo detalhou quais os componentes eletrônicos utilizados para construir o relé protótipo, bem como a biblioteca de proteção desenvolvida para amostrar os sinais de tensão e corrente de um sistema de potência e realizar o cálculo dos parâmetros elétricos necessários para então executar um algoritmo de proteção. Uma vez que o relé protótipo está apto a operar, será necessário criar sinais de falta que irão alimentá-lo, bem como elaborar funções de proteção que testarão seu poder computacional.

Figura 9 – Fluxograma dos modos de operação do relé protótipo



Fonte: Autoria própria

4 SIMULAÇÃO DE EVENTOS ELÉTRICOS E FUNÇÕES DE PROTEÇÃO DESENVOLVIDAS

Este capítulo apresenta o alimentador em média tensão modelado no *software* ATP/ATPDraw, no qual as faltas trifásicas e transitórios foram simuladas. Além disso, são apresentadas as funções de proteção já programadas no relé protótipo, incluindo o algoritmo de proteção baseado em redes neurais desenvolvido para testar a precisão do relé protótipo em executar funções de proteção mais complexas.

4.1 Considerações Iniciais

Para testar o desempenho do relé protótipo em detectar faltas elétricas e enviar respostas em um tempo preciso e pré-determinado, foram simuladas faltas trifásicas simétricas em um alimentador de média tensão modelado no *software* ATP/ATPDraw. Esse tipo de falta elétrica foi escolhido pelo fato de ser mais facilmente detectável por relés de proteção, visto que somente o cálculo de parâmetros elétricos simples, como o valor eficaz (RMS) de uma forma de onda, é suficiente para tal finalidade (falta de sobrecorrente). Dessa forma, o foco deste trabalho foi direcionado para o desenvolvimento do *hardware* e *software* do relé protótipo.

A precisão do relé protótipo também será testada ao executar algoritmos de proteção, sejam simples ou complexos. Para detectar faltas trifásicas, foram programadas as funções de sobrecorrente e subtensão. Além disso, um algoritmo de proteção com base em redes neurais foi desenvolvido, capaz de diferenciar faltas trifásicas de transitórios gerados pela entrada e saída de cargas de alta demanda, também simulados no *software* ATP/ATPDraw. Os arquivos *.lis* gerados pelas simulações foram, então, enviados a uma mala de teste de relés para alimentar o relé protótipo.

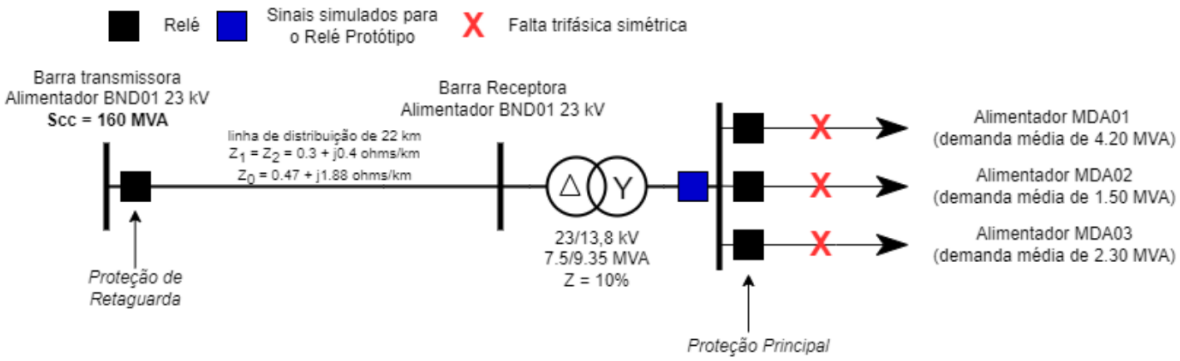
O *software* ATP/ATPDraw foi escolhido para realizar as simulações devido a sua capacidade em gerar transitórios bem definidos, como consequência de eventos elétricos dos mais diversos. Ele possui vários componentes elétricos (e.g. geradores, transformadores, linhas de transmissão e distribuição, chaves, etc.) pré-definidos, que podem ser utilizados para simular os mais diversos sistemas de potência (ATPDRAW, 2012).

4.2 Alimentador de média tensão modelado e sinais simulados

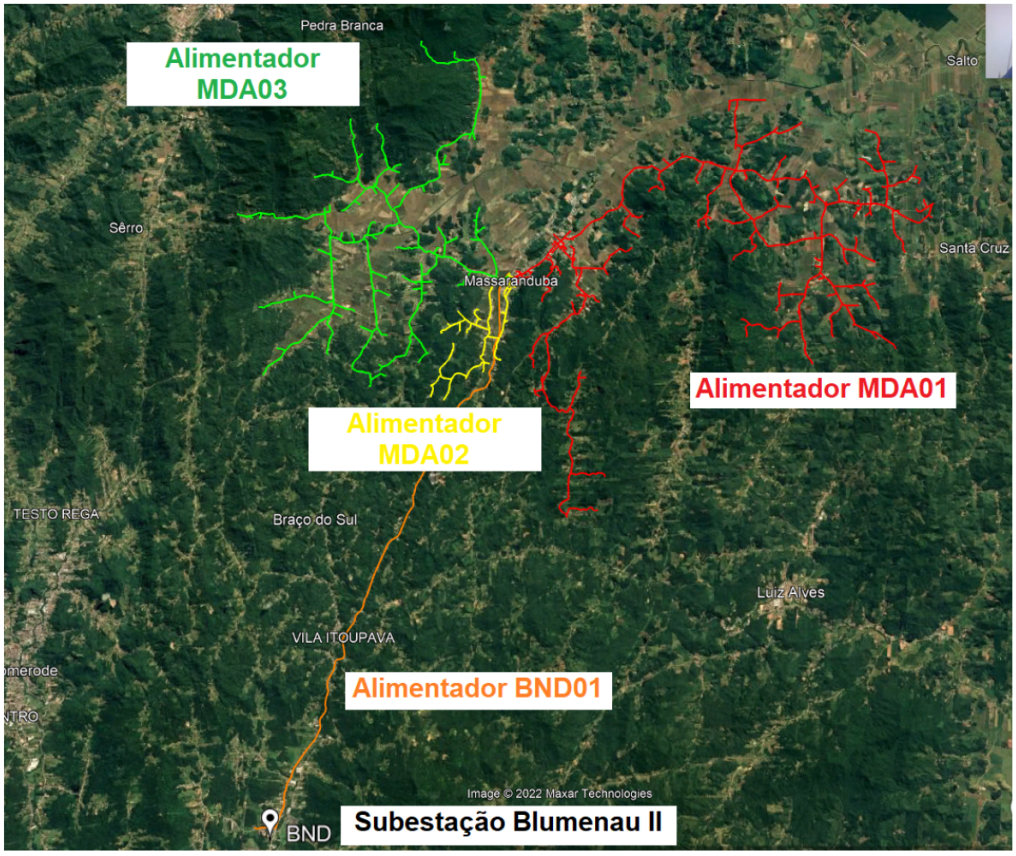
Foi escolhido para modelagem no *software* ATP/ATPDraw um dos alimentadores de distribuição em média tensão da Subestação Blumenau II, localizado no município de Blumenau – SC. Representado pelo código BND01, esse alimentador opera no nível de tensão 23 kV, frequência fundamental de 60 Hz e fator de potência 0,92. Ele é um alimentador expresso (sem cargas elétricas conectadas a ele) de 22 km, utilizando cabo de rede nua 4/0 CA, o qual conecta-se a uma subestação abaixadora 23/13,8 kV, da qual três alimentadores de distribuição (MDA01, MDA02 e MDA03), fornecem energia elétrica para a cidade de Massaranduba - SC, cuja demanda média diária de energia é estimada em 8 MVA. O sistema elétrico descrito é de propriedade da concessionária *Celesc Distribuição.* Atualmente, a proteção de retaguarda dos relés de proteção dos alimentadores em nível de tensão 13,8 kV é exercida pelo relé de proteção em 23 kV do alimentador BND01, o qual pode demorar demasiadamente para operar devido à longa distância. A Figura 10 apresenta o diagrama unifilar do sistema elétrico descrito, além de sua extensão geográfica e sua modelagem no *software* ATP/ATPDraw.

Faltas trifásicas simétricas e transitórios de entrada e saída de carga foram simuladas ao longo dos alimentadores de distribuição em 13,8 kV. Utilizanto como referência a barra de baixa tensão do transformador abaixador, os eventos elétricos citados foram simulados nas distâncias de 2 km, 4 km, 6 km e 8 km ao longo dos alimentadores em 13,8 kV. Distâncias maiores não foram utilizadas pois o transitório gerado não era significativo. Além disso, a posição angular na qual as faltas e alterações de carga ocorreram também foram variadas; nas posições 0°, 30°, 90° (referente a fase A da tensão trifásica na barra de 13,8 kV). Particularmente, no caso das simulações de entrada e saída de carga, foi utilizada uma carga trifásica fictícia de 2,5 MW (representada por uma resistência de R = 8,103 ohms no *software* ATP/ATPDraw) para gerar transitórios significativos nesse evento elétrico. Essa carga foi escolhida pelo fato de ser a máxima carga que obrigatoriamente deve ser atendida por sistemas de distribuição em média tensão no Brasil (ANEEL, 2021). Os componentes empregados para modelar o alimentador BND01, bem como suas principais configurações, são apresentados na Tabela 3.

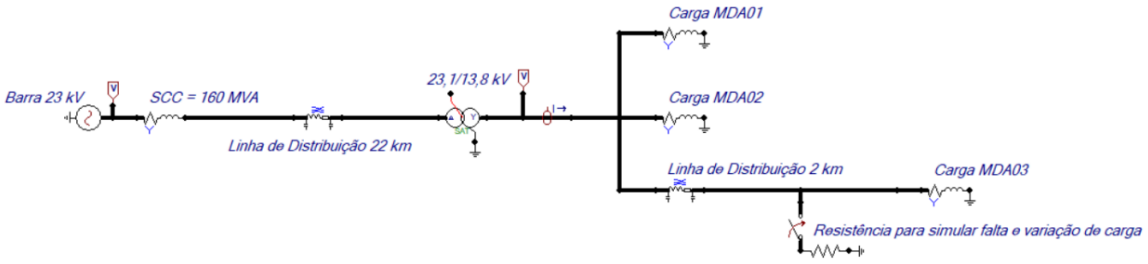
Figura 10 – Alimentador BND01 da Subestação Blumenau II



(a) Diagrama Unifilar



(b) Extensão Geográfica



(c) Modelagem no ATP

Fonte: Autoria própria

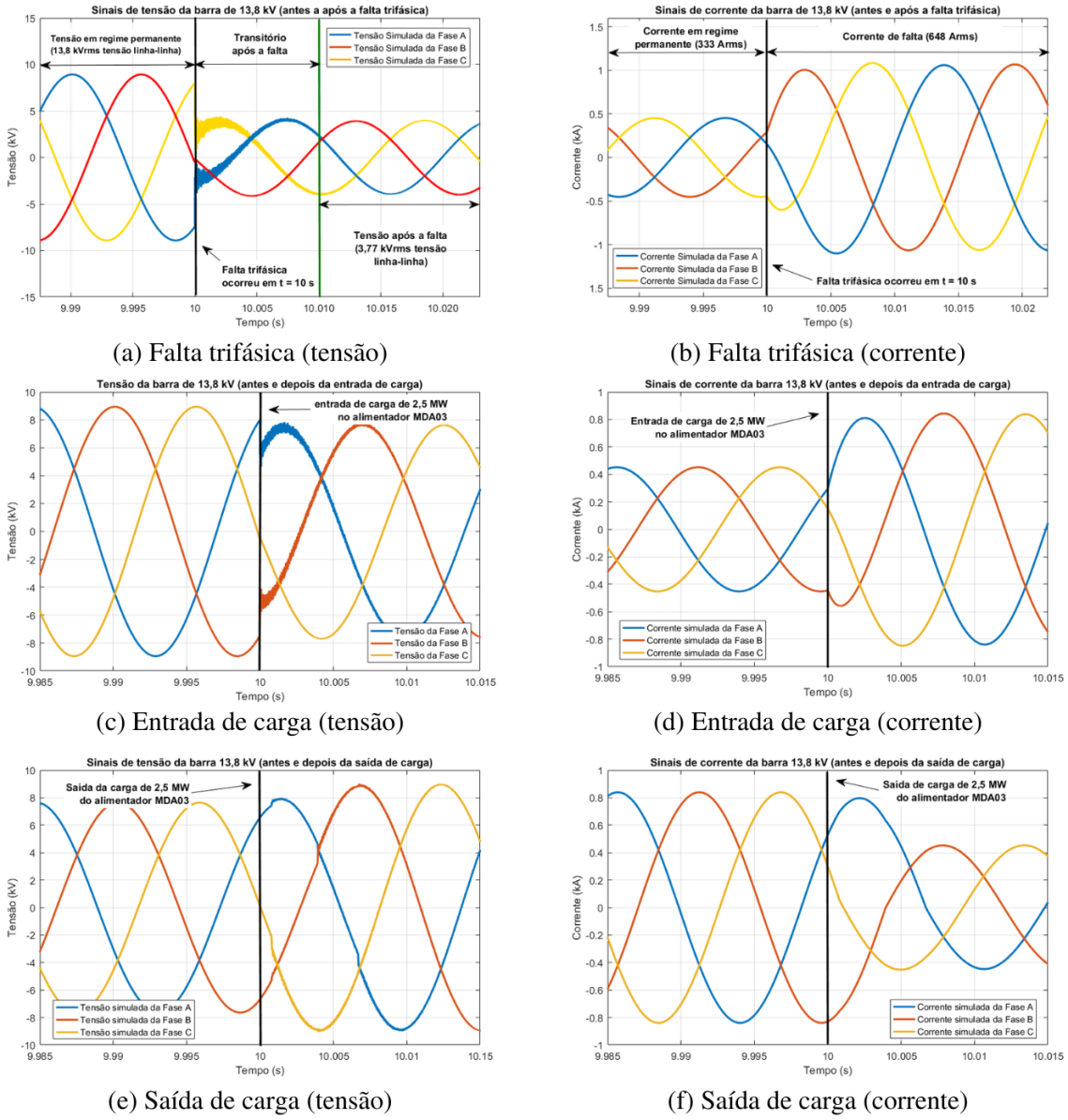
Tabela 3 – Componentes utilizados para modelar o alimentador BND01 no software ATP/ATP-Draw

Componente ATP	**Configuração dos parâmetros dos componentes**
Fonte CA (ACSOURCE)	Fonte trifásica aterrada, Tensão de linha 23 kV RMS Ângulos por fase de 0°, 120° e 240°, frequência de 60 Hz
Potência de curto-ciruito de 169 MVA (RLCY3)	As resistências de todas as fases são iguais a 3,04 ohms As indutâncias de todas as fases são iguais a 3,43 mH As capacitâncias de todas as fases são nulas
Linha de Distribuição cabo 4/0 CA (LINEPI3S)	Resistência de sequência positiva igual a = 7,79 e-4 ohms/m Indutância de sequência positiva igual a = 50,85 e-3 mH/m Resistência de sequência zero igual a = 6,13 e-4 ohms/m Indutância de sequência zero igual a = 11,37 e-3 mH/m Todas as capacitâncias são nulas Comprimento da rede = 22.000 metros
Transformador abaixador 23/13,8 kV (SATTRAFO)	Transformação Delta/Estrela aterrado, defasagem angular de 30° Tensão primária de 23 kV /Tensão secundária de 7,967 kV Resistência do primário de 9,58 ohms/Resistência do secundário de 1,67 ohms Indutância do primário de 28,93 mH/Indutância do secundário de 5,06 mH
Carga de 4,20 MVA do alimentador MDA01 (RLCY3)	Resistência de todas as fases igual a 29,34 ohms Indutância de todas as fases igual a 33,16 mH As capacitâncias de todas as fases são nulas
Carga de 1,50 MVA do alimentador MDA02 (RLCY3)	Resistência de todas as fases igual a 104,91 ohms Indutância de todas as fases igual a 118,55 mH As capacitâncias de todas as fases são nulas
Carga de 2,30 MVA do alimentador MDA03 (RLCY3)	Resistência de todas as fases igual a 81,03 ohms Indutância de todas as fases igual a 84,24 mH As capacitâncias de todas as fases são nulas
Chave para simular as faltas trifásicas e transitórios (TSWITCH)	Fechamento da chave = 0,05 s; Abertura da Chave = 1 s (para faltas trifásicas) Fechamento da chave = -1 s; Abertura da Chave = 0,05 s (para transitórios de carga) Número de fases = 3
Resistência entre a chave e o terra para simular as faltas e transitórios (RESISTOR)	Resistência = 1 ohm (para faltas trifásicas) ou 8,103 ohms (para transitórios de carga)
Configuração da Simulação	Simulação no domínio do tempo) Δt = 1e-8, tempo máximo de simulação = 0,1 s (para transitórios de carga)

Fonte: Autoria própria

Para gerar um sinal transitório bem definido durante as faltas e variações de carga, uma resolução de Δt = 1e-8 segundos foi usada na simulação das faltas. O tempo de simulação foi definido em 0,1 segundos. A Figura 11 apresenta um exemplo de um conjunto de sinais simulados de tensão e corrente do barramaneto de 13,8 kV, para uma falta trifásica, aumento de carga, redução de carga no alimentador MDA03, a 2 km do barramento de 13,8 kV, na posição 90° em relação a tensão da fase A.

Figura 11 – Formas de onda simuladas no software ATP/ATPDraw



(a) Falta trifásica (tensão)

(b) Falta trifásica (corrente)

(c) Entrada de carga (tensão)

(d) Entrada de carga (corrente)

(e) Saída de carga (tensão)

(f) Saída de carga (corrente)

Fonte: Autoria própria.

Para alimentar o relé protótipo com os sinais de falta gerados pelo ATP/ATPDraw, utilizou-se uma mala de teste de relés hexafásica da *Conprove Engenharia*, modelo CE – 6006 (Figura 12). Esse equipamento é capaz de fornecer sinais de tensão e corrente de alta precisão, emulando, por exemplo, sinais em regime permanente, sinais de sobrecorrente, sobre e subtensão, dentre outros. Ela também fornece relatórios de testes, a partir dos quais as características de fábrica de um relé digital podem ser comparadas com seu desempenho no momento do teste, e indicam se os resultados correspondem aos fornecido pelo fabricante. Através de um Processador de Sinal Digital (DSP), essa mala de testes pode criar várias formas de onda, atingindo frequências de até 3 kHz (valor verificado após testes em laboratório). Por fim, este dispositivo pode reproduzir sinais gerados a partir de simulações do programa ATP/ATPDraw (extensão *.lis*) (CONPROVE ENGENHARIA, 2021). O modo de conexão da mala de teste com os sensores de tensão e corrente do relé protótipo são apresentados no Apêndice A. A mala de testes será responsável por medir o tempo de atuação do relé protótipo, em relação ao tempo em que a falta elétrica gerada por ele ocorre, através do sinal de *trip* enviado pelo relé protótipo e coletado por suas entradas binárias.

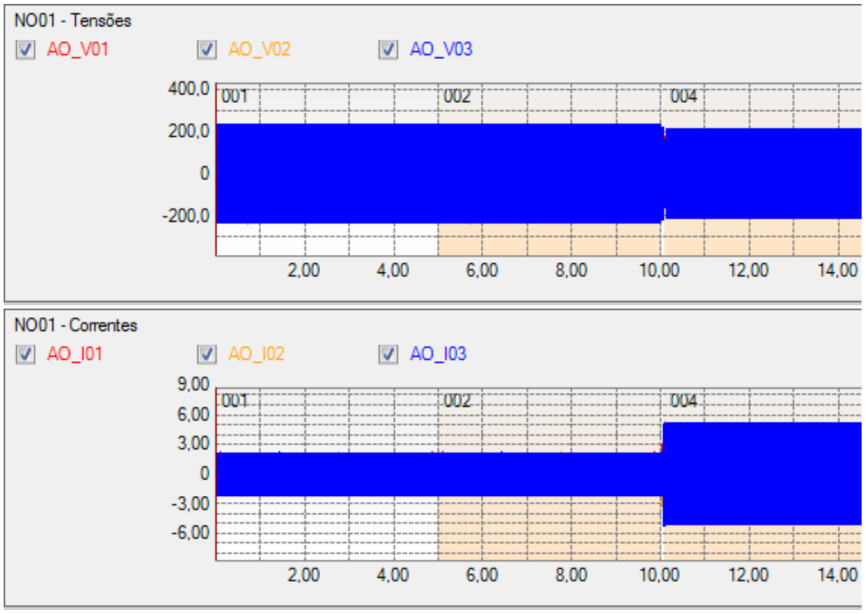
A amplitude dos sinais simulados é reduzida pela mala de teste em relação a grandeza em quilovolts dos sinais simulados, a fim de se enquadrar na faixa de operação de entrada dos relés de proteção comerciais mais comuns. Para este trabalho de Mestrado, a tensão e corrente máxima a ser fornecida pela mala de testes foi de 220 Vrms e 5 Arms, respectivamente (esses valores estão dentro da faixa de operação dos sensores de tensão e corrente do relé protótipo).

Figura 12 – Mala de teste de relés da Conprove Engenharia (modelo CE -6006)

Fonte: Conprove Engenharia (2021).

Vale ressaltar que, na Figura 11, a falta elétrica e as variações de carga ocorrem em t = 10 s, o qual é um tempo maior que o tempo de simulação dos eventos elétricos (0,1 s). Isso se deve a inserção de arquivos de simulação de formas de onda em regime permanente que representam o estado das tensões e correntes antes e após os eventos elétricos citados. A janela de tempo total dos sinais elétricos que são enviadas ao relé protótipo é de 15 s, como mostra a Figura 13. Elas são postas em sequência dentro do ambiente do configuração da mala de testes. Adotou-se essa abordagem de envio de sinais de falta para o relé protótipo de modo a verificar se ele é capaz de atuar devidamente na presença de faltas elétricas e não atuar quando na presença de sinais em regime permanente.

Figura 13 – Sequência das formas de onda dentro da mala de testes de relés



Sequências 001 e 002 - Formas de onda em regime permanente antes da falta
Sequência 003 - Falta elétrica
Sequência 004 - Forma de onda em regime permanente após a falta.

Fonte: Autoria própria

4.3 Funções de proteção programadas e modelagem da rede neural

Um dos principais aspectos do relé protótipo deste trabalho é permitir que seus usuários programem seus próprios algoritmos de proteção, de modo a verificar seu tempo de resposta em tempo real e em *hardware* frente a faltas geradas externamente. Para testar a capacidade do relé protótipo em executar funções de proteção em tempo real, foram programadas as seguintes funções:

- Proteção de sobrecorrente de tempo definido - os valores eficazes (RMS) de cada sinal de entrada (tensão e corrente) é calculado pela função de medição da

biblioteca de proteção, por um período de meio ciclo, e comparados com o valor de *pick-up* definido pelo usuário. A falta elétrica é detectada quando o valor RMS das tensões e correntes forem menores e maiores que seus respectivos valores de *pick-up*. O sinal de *trip* é enviado caso a falta se mantenha por um período específico de tempo, cujo valor também é definido pelo usuário;

- Algortimo da rede neural – essa rede neural é capaz de diferenciar faltas trifásicas de variações de carga elétrica, com base no valor RMS dos sinais de entrada (tensão e corrente). Como mostra a Figura 11, os transitórios criados por uma falta trifásica e o aumento de carga são semelhantes. Logo, uma função de proteção de sobrecorrente comum não seria capaz de diferenciar tais sinais, o que faz o uso de uma rede neural necessário. Essa função terá um tempo pré-determinado pelo usuário para decidir se os sinais de entrada representam um sinal em regime permanente, uma falta elétrica ou um transitório. O sinal de *trip* só será enviado quando o tempo de decisão se encerrar e a falta elétrica tiver sido detectada.

A função de proteção de sobrecorrente de tempo definido foi escolhida para testar o relé protótipo pelo fato de ser uma das funções de proteção mais simples de serem programadas, visto que somente é necessário o valor RMS dos sinais de entrada para identificar uma falta trifásica. O Algoritmo 1 apresenta mais detalhes de como essa função foi codificada dentro do *Raspberry Pi 3B+*.

Algoritmo 1: Proteção de sobrecorrente de tempo definido

Entrada: tempo_definido, Vpickup, Ipickup

Saída: Trip

Enquanto Sempre **Faça**

// V(k) e I(k) representam o valor RMS calculado da k-ésima janela de dados

Se I(k) $\geq$ Ipickup **E** V(k) $\leq$ Vpickup **Então**

Retorna Verdadeiro

Enquanto tempo $<$ tempo_definido **Faça**

Se I(k) $\geq$ Ipickup **E** V(k) $\leq$ Vpickup **Então**

Retorna Verdadeiro

Senão

Retorna Falso

Fim Enquanto *// encerra ambas funções Enquanto*

Fim Se

Fim Enquanto

Retorna Trip

Fim Enquanto *// encerra Enquanto do loop infinito*

Senão

Retorna Falso

Fim Se

$k = k + 1$ *// atualiza a janela de dados*

Repete

Quanto a rede neural desenvolvida, ela foi criada para verificar se o relé protótipo é capaz de lidar com funções de proteção mais complexas. Vale ressaltar que ela não é uma nova função de proteção; ela somente testa o tempo de atuação do relé protótipo.

A rede neural desenvolvida é do tipo *feed-foward*, onde a conexão entre os neurônios parte de uma camada inicial para uma oculta (ou ocultas), até chegar à camada de saída, sem um caminho de volta. Ela utiliza neurônios do tipo *perceptron* em cada camada, geralmente utilizados para classificar os dados de entrada em categorias (DA SILVA, 2016). Logo, a rede neural desenvolvida também é do tipo *percetron* de várias camadas (*multilayer perceptron* ou MLP). Ela foi programada no relé protótipo através da biblioteca em linguagem C/C++ disponibilizada por Winkle (2020), a qual utiliza um algoritmo de *backpropagation* para calcular os pesos de cada neurônio e treinar da rede neural.

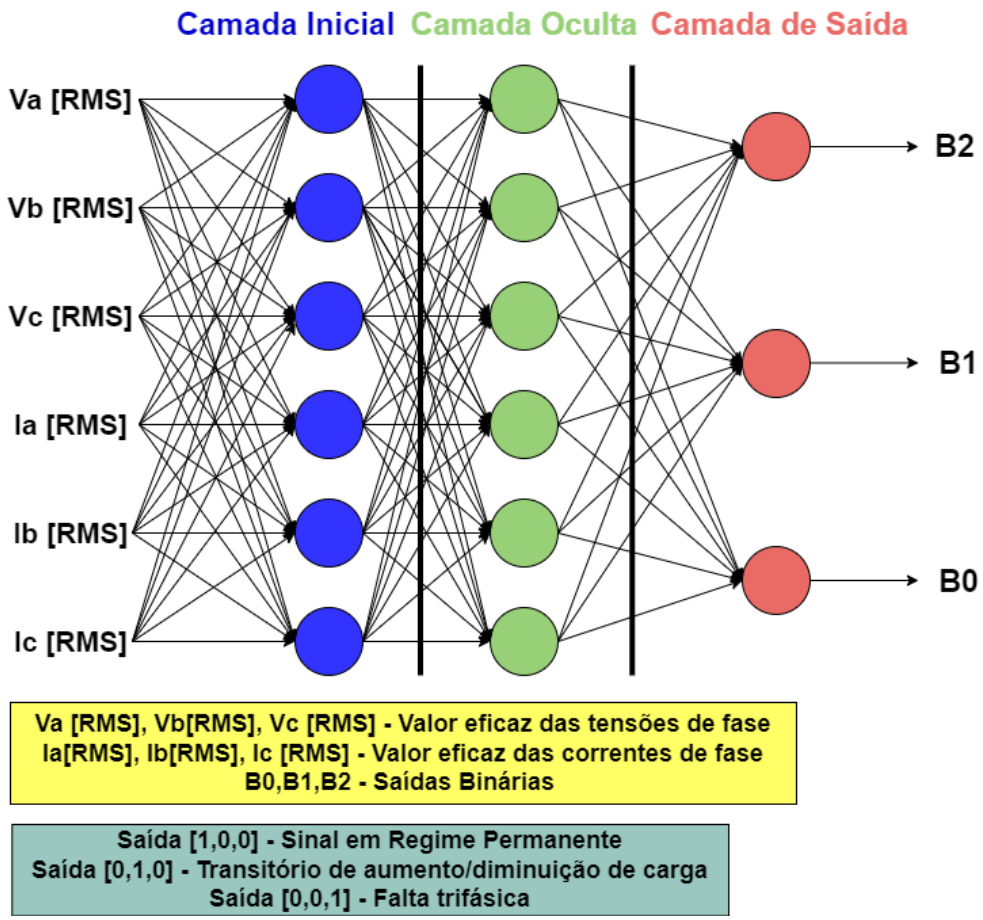
A rede desenvolvida possui seis neurônios na camada inicial (equivalente ao número de valores eficazes das tensões e correntes de um alimentador trifásico), uma camada oculta com seis neurônios, e três neurônios na camada de saída. As saídas representam valores binários (B2, B1, B0), os quais combinados indicam se o sinal de entrada está em regime permanente, é um sinal de falta trifásica, ou se uma variação de carga ocorreu. O relé protótipo só envia o sinal de *trip* para a mala de teste quando uma falta trifásica for detectada, como mostra o Algoritmo 2. Mesmo quando a falta é detectada antes do tempo definido pelo usuário, a rede neural continua a ser executada, de modo a confirmar se a falta se mantém ou não.

Algoritmo 2: Detecção de falta trifásica pela rede neural desenvolvida

Entrada: tempo_definido
Saída: Trip
Enquanto Sempre **Faça**
//V(k) e I(k) representam o valor RMS calculado da k-ésima janela de dados
//B2, B1 e B0 represetam os valores binários que coletam a resposta da rede neural
{B2;B1;B0} = Rede_neural(V(k),I(k))
Se B0 $>$ B1 **E** B0 $>$ B2 **Então**
Retorna Verdadeiro
Enquanto tempo $<$ tempo_definido **Faça**
{B2;B1;B0} = Rede_neural(V(k),I(k))
Se B0 $>$ B1 **E** B0 $>$ B2 **Então**
Retorna Verdadeiro
Se (B1 $>$ B2 **E** B1 $>$ B2) **OU** (B2 $>$ B0 **E** B2 $>$ B1)
Retorna Falso
Fim Enquanto // *encerra ambas funções Enquanto*
Fim Se
Fim Enquanto
Retorna Trip
Fim Enquanto // *encerra Enquanto do loop infinito*
Se (B1 $>$ B2 **E** B1 $>$ B2) **OU** (B2 $>$ B0 **E** B2 $>$ B1)
Retorna Falso
Fim Se
$k = k+1$ // *atualiza a janela de dados*
Repete

Para treinar a rede neural, foi utilizado um conjunto de dados de 50 mil amostras, obtidas pelo relé protótipo a uma frequência amostral de 3 kHz, o que resulta em uma média de 50 amostras por período dos sinais de entrada CA em 60 Hz. O valor RMS de cada sinal de entrada foi calculado por período, contemplando valores de tensão e corrente em regime permanente, durante a falta trifásica e durante as variações de carga, simulados nos alimentadores modelados nesta seção do trabalho. No total, mil valores RMS foram calculados para cada sinal de entrada, dos quais 100 representam o estado em regime permanente; 300 equivalem as faltas trifásicas; e os últimos 300 represetam as variações de carga (150 para aumento de carga; 150 para redução de carga). Desse conjunto de dados, 80% foi separado para o treinamento da rede neural, enquanto 20% foi utilizado nos testes da rede. A Figura 14 mostra graficamente a rede neural desenvolvida. Toda a criação e treinamento da rede neural foi realizada dentro do *Raspberry Pi 3B+*.

Figura 14 – Rede neural desenvolvida para detectar faltas trifásicas



Fonte: Autoria própria

4.4 Considerações Finais

Este capítulo retratou o sistema elétrico modelado no *software* ATP/ATPDraw, o qual foi utilizado para gerar os sinais de falta que alimentarão o relé protótipo. Também foram apresentados os algoritmos de proteção criados para testar seu desempenho em tempo real, desde uma função de sobrecorrente simples, até uma função complexa baseada em redes neurais.

Para avaliar a viabilidade do uso do relé protótipo, os modos de operação como amostrador de sinais e relé de proteção serão testados em seguida.

5 TESTES NO RELÉ PROTÓTIPO E RESULTADOS

Esta Seção descreve os testes em laboratório realizados no relé protótipo para avaliar seu tempo de atuação em tempo real frente a faltas elétricas, tendo seu desempenho comparado a um relé comercial. Além disso, a precisão da amostragem do relé protótipo em reproduzir os sinais de entrada também foi avaliada, bem como o impacto de mudanças em sua frequência amostral no tempo de atuação das funções de proteção. Por fim, o tempo de atuação do relé protótipo ao executar a rede neural desenvolvida foi mensurado de modo a avaliar seu desempenho em executar algoritmos complexos.

5.1 Considerações Iniciais

Os Capítulos 3 e 4 deste trabalho apresentaram as principais funcionalidades e configurações do relé protótipo, além das funções de proteção pré-programadas nele. Para verificar se relé protótipo efetivamente executa suas principais atribuições adequadamente, os seguintes testes foram realizados nele em laboratório:

- Teste da operação como amostrador de sinais: verificação da semelhança entre os sinais amostrados pelo relé protótipo em relação aos sinais criados pela mala de teste de relés;
- Teste de operação como relé de proteção: o tempo de atuação do relé protótipo é comparado com um relé comercial, ambos submetidos a mesma falta trifásica. Também é avaliado como a variação da frequência de amostragem do relé protótipo afeta seu tempo de atuação;
- Teste operação para precisão no tempo de atuação do relé protótipo: o tempo de atuação do relé protótipo será medido ao executar a rede neural desenvolvida, além de mensurar como alterações na frequência de amostragem afetam esse parâmetro. Não só a exatidão da rede neural em identificar o estado dos sinais de entrada será avaliada, como também a precisão do relé protótipo em manter o tempo de atuação próximo ao pré-determinado.

O relé comercial adotado neste trabalho para servir de referência para o relé protótipo é o MiCOM P142 da *Schneider Electric*, utilizado na proteção de alimentadores de distribuição. O Relé P142 possui uma taxa de amostragem fixa de 24 amostras por ciclo, o que equivale a uma frequência de amostragem de 1,44 kHz para um sinal de 60 Hz. Ele realiza a amostragem

dos sinais de tensão e corrente a cada meio período (SCHNEIDER ELECTRIC, 2020). A Figura 15 apresenta o relé comercial utilizado.

Figura 15 – Relé comercial utiliza nos testes do relé protótipo (P142 da Schneider Electric)

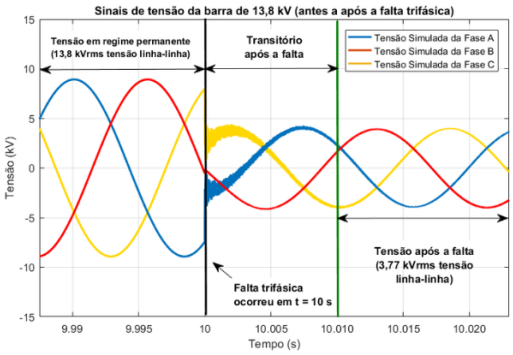
Fonte: Schneider Electric (2020).

As próximas subseções irão detalhar como os testes descritos foram realizados, bem como os resultados obtidos e discussões relevantes.

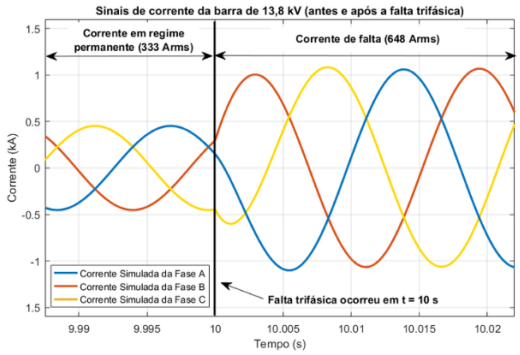
5.2 Teste de operação como amostrador de sinais

A precisão no processo de amostragem de um relé de proteção digital é de suma importância, uma vez que a representação de um sinal discretizado mais próxima do sinal analógico real implica em uma melhor atuação dos algoritmos de proteção para detectar faltas elétricas. Para verificar se o relé protótipo deste trabalho é capaz de entregar um sinal amostrado semelhante ao sinal gerado pela mala de teste de relés, utilizou-se um osciloscópio modelo THS3024 da *Tektronix Inc.*, o qual pode atingir uma taxa de amostragem de 5 Gsps, além de uma largura de banda de 200 MHz (TEKTRONIX, 2022). Os sinais amostrados pelo relé protótipo (na frequência amostral de 5 kHz) foram comparados com os sinais amostrados pelo osciloscópio citado, como ilustra a Figura 16. Utilizou-se como sinal de referência a falta trifásica simulada no alimentador modelado no ATP/ATPDraw, como apresentado na Subseção 4.2 deste trabalho. As pontas de prova do osciloscópio foram conectadas na saída do sensores do Módulo de Condicionamento de Sinais do relé protótipo (Subseção 3.2). Dessa forma, ambos os equipamentos enxergam os mesmos sinais de entrada.

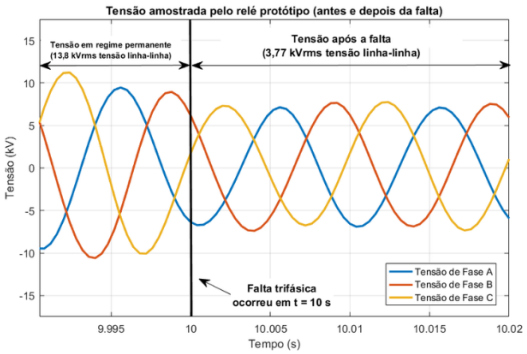
Figura 16 – Comparação dos sinais simulados e amostrados(ATP, relé protótipo e osciloscópio)



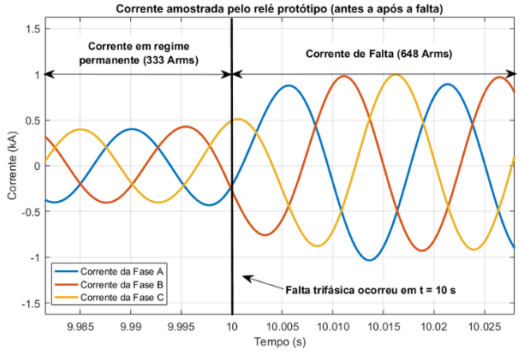
(a) Falta trifásica do ATP (tensão)



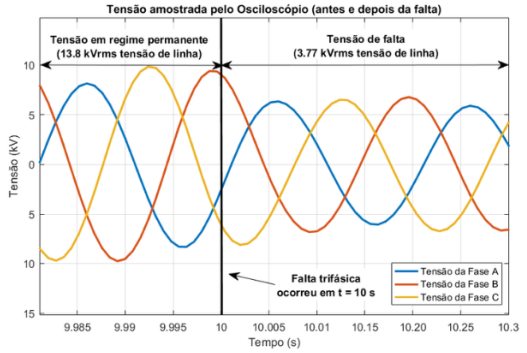
(b) Falta trifásica do ATP (corrente)



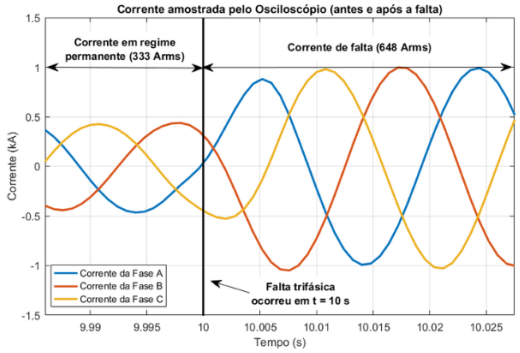
(c) Falta amostrada pelo relé protótipo (tensão)



(d) Falta amostrada pelo relé protótipo (corrente)



(e) Falta amostrada pelo osciloscópio (tensão)



(f) Falta amostrada pelo osciloscópio (corrente)

Fonte: Autoria própria.

Comparando as formas de onda da Figura 16, as formas de onda da tensão e corrente amostradas pelo relé protótipo e pelo osciloscópio possuem escopos semelhantes. Além disso, nenhum dos equipamentos foi capaz de detectar o elemento transitório nos sinais de falta gerados pelo ATP/ATPDraw. Isso implica que o transitório contido nas formas de onda produzidos pelo software ATP/ATPDraw possuem uma frequência maior que 3 kHz, cuja valor a mala de testes da Conprove não é capaz de reproduzir (CONPROVE ENGENHARIA, 2021). Portanto, o processo de amostragem do relé protótipo funciona adequadamente, pois ele representa com precisão os sinais em regime permanente e de falta gerados pela mala de testes.

5.3 Teste de operação como relé de proteção

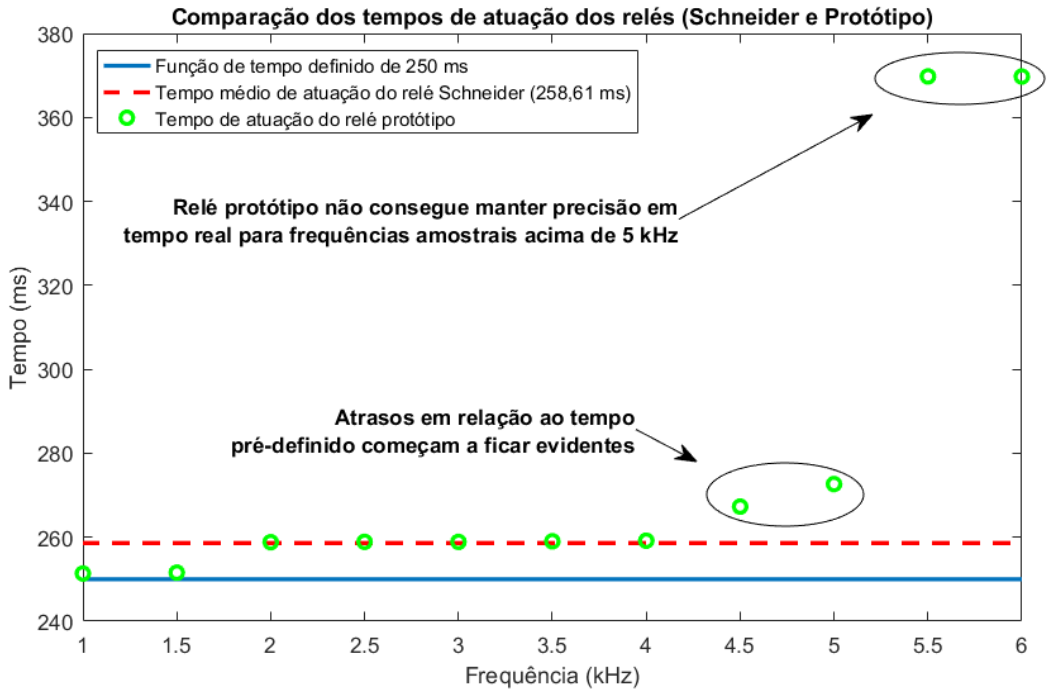
Para testar o modo de proteção do relé protótipo, tanto ele quanto o relé P142 foram submetidos às mesmas mesmas faltas trifásicas apresentadas na Figura 11. Além de gerar os sinais de falta, a mala de testes também foi utilizada para aferir o tempo de atuação de ambos os relés (tempo entre a ocorrência da falta e o envio do sinal de *trip* para a mala de testes), a fim de verificar se a plataforma desenvolvida possui um tempo de resposta próximo de um relé comercial. O modo de conexão do relé P142 e do relé protótipo na mala de testes é apresentado no Apêndice A

Tanto o relé protótipo quanto o relé P142 foram configurados com a função de sobrecorrente de tempo definido. Utilizando como referência a escala de operação da mala de testes, o valor da tensão e corrente em regime permanente dos sinais de entrada foi definido como 220 Vrms e 3 Arms, respectivamente. Já os valores de *pick-up*, foram definidos como 110 Vrms e 5 Arms. Uma vez a falta detectada, ambos os relés foram configurados para enviar o sinal de *trip* caso a falta se mantenha por mais de 250 ms (função de proteção de sobrecorrente de tempo definido). Este valor específico de tempo de disparo foi escolhido para o teste da função de sobrecorrente pois o relé P142 já possuía este valor pré-ajustado em suas configurações (SCHNEIDER ELECTRIC, 2020). Esse valor de tempo de atuação é atribuído ao parâmetro $tempo\_definido$ apresentado no Algoritmo 1, enquanto $Vpickup = 110$ e $Ipickup = 5$;

Além do tempo de atuação, este teste também contempla o impacto da frequência amostral nesse parâmetro. O relé protótipo utilizará a frequência amostral inicial de 1,44 kHz, para uma comparação direta com o relé P142. Em seguida, uma série de frequências de amostragem, iniciando em 1 kHz e aumentada em 0,5 kHz por vez, será testada no relé protótipo até que ele não mais possa entregar uma resposta em tempo real adequada (atrasos no tempo

de atuação significativamente maiores que 250 ms). Em cada frequência amostral, o tempo de atuação do relé protótipo foi medido várias vezes, de modo a ter uma grande quantidade de dados para o cálculo mais preciso desse parâmetro. A Figura 17 apresenta graficamente a comparação nos tempos de atuação de ambos os relés em relação a frequência amostral do relé protótipo.

Figura 17 – Comparação dos tempos de atuação do relé protótipo e comercial após a falta trifásica



Fonte: Autoria própria.

Pela Figura 17, observa-se que o relé protótipo possui um tempo de atuação muito próximo ao do relé P142, em relação a frequência de amostragem inicial de 1,44 kHz. À medida que ela aumenta, o tempo de atuação apresenta pequenos atrasos em relação ao tempo definido de 250 ms, até o ponto em que o relé protótipo não mais consegue manter uma resposta em tempo real para frequências de amostragem acima de 5 kHz. De acordo com Schneider Electric (2020), o relé P142 tem um tempo médio de atraso de 20 ms quando a função de proteção de sobrecorrente é selecionada. Assim o tempo de atuação do relé comercial foi medidos após o valor de *pick-up* da falta ter sido identificado, ao invés do momento da ocorrência da falta (referência utilizada para o relé protótipo). Esse atraso foi deduzido na apresenação da Figura 17, a qual mostra somente o tempo de atuação da proteção de sobrecorrente. A Tabela 4 compila os atrasos no tempo de atuação de ambos os relés.

Tabela 4 – Atrasos no tempo de atuação dos relés (acima de 250 ms)

	Atrasos (ms)			
Relés	**Média**	**Mínima**	**Máxima**	**Variância**
Relé P142	8,62	8,05	9,12	0,13
Relé protótipo (frequência amostral fixa de 1,44 kHz)	1,08	0,63	1,57	0,11
Relé protótipo (todas as frequências amostrais medidas)	13,04	1,37	32,67	126,45
Número de amostras por relé: 10				

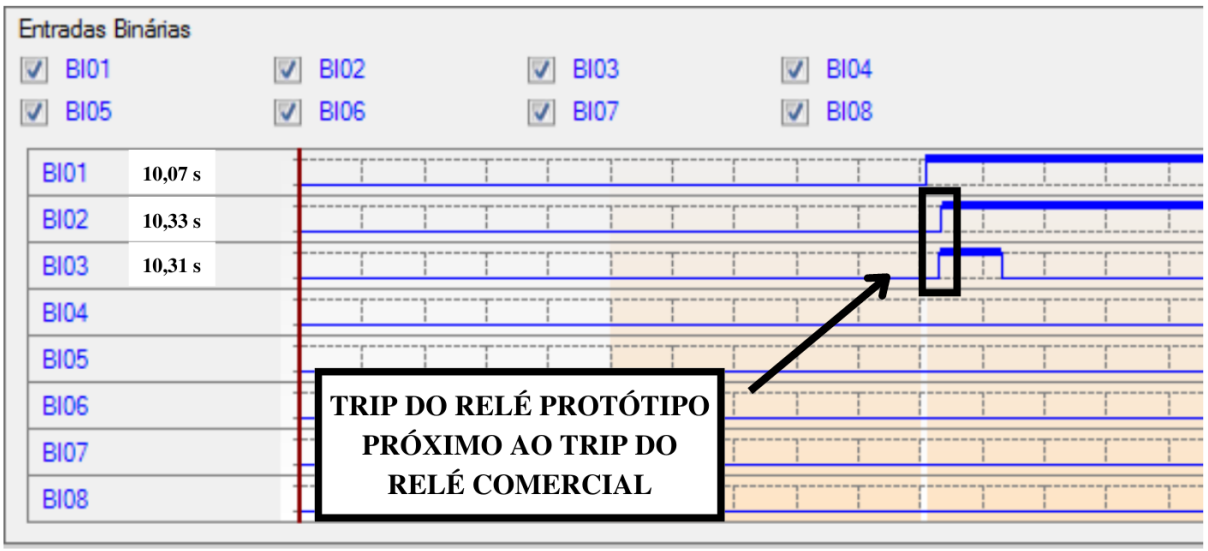
Fonte: Autoria própria.

A Tabela 4 mostra que, na frequência de amostragem de 1,44 kHz, o relé protótipo não só tem um tempo de resposta mais rápido à falta trifásica que o relé comercial, como também tem uma variância próxima a dele, o que mostra que a biblioteca em tempo real utilizada neste trabalho é capaz de manter a precisão em seu tempo de resposta. Ao incluir as medições de todas as frequências de amostragem testadas, a variância dos dados aumenta drasticamente, o que indica que frequências de amostragem mais altas possuem um impacto significativo no tempo de atuação em tempo real do relé protótipo. Devido a limitação de frequência de 5 kHz, o uso deste relé protótipo para teste de algoritmos de proteção para faltas em corrente contínua (CC) não é recomendado, visto que a detecção desse tipo de falta requer instrumentos não convencionais, capazes de fornecer altas frequências de amostragem, de dezenas de kHz a poucos MHz (WANG *et al.*, 2017).

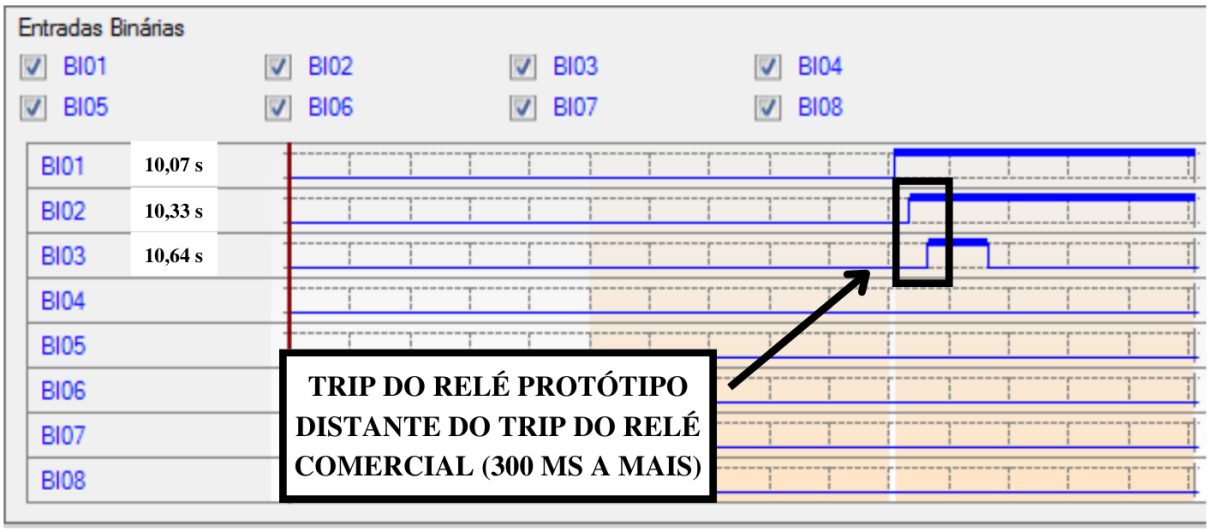
A origem dos atrasos maiores no tempo de atuação da proteção no relé protótipo (a medida que a frequência amostral aumenta) se deve pela lógica de funcionamento da biblioteca em tempo real desenvolvida neste trabalho. Como citado na Subseção 2.3, a biblioteca em tempo real lida com as tarefas em tempo real e comuns linearmente. Uma vez que a amostragem dos sinais de entrada sempre vem antes das etapas de medição e proteção, a cada iteração em que a função de sobrecorrente é executada (tempo médio por iteração de 150 μs), pequenos atrasos vão se acumulando a medida que a janela de dados é atualizada e um novo valor RMS de cada sinal de entrada é calculado, resultando no atraso final no envio do sinal de *trip*. Logo, o relé protótipo pode ser considerado um *sistema em tempo real do tipo leve* (*soft*), capaz de executar um algoritmo de proteção com um tempo de atuação próximo a um relé comercial, porém seu uso não é recomendado para sistemas de proteção reais, os quais requerem um sistema em tempo real crítico (*hard*). Com base na representação visual das entradas binárias da mala de teste, a precisão e perda da característica em tempo real do relé protótipo de acordo com a frequência amostral utilizada é apresentada na Figura 18.

Figura 18 – Representação visual da atuação em tempo real do relé protótipo

Legenda:
BI01 - Entrada binária que recebe o sinal de pick-up do relé comercial
BI02 - Entrada binária que recebe o sinal de trip do relé comercial
BI03 - Entrada binária que recebe do sinal de trip do relé protótipo



(a) Precisão no tempo real (frequência de amostragem = 1,44 kHz)



(b) Perda no tempo real (frequência de amostragem = 5,5 kHz)

Fonte: Autoria própria.

5.4 Teste da precisão do relé protótipo

Para testar a rede neural detalhada na Subseção 4.3, alimentou-se o relé protótipo com os sinais de falta e variação de carga apresentados na Figura 11. Pela escala de valores da mala de testes, os valores de referência de tensão e corrente para cada evento elétrico que a rede neural é capaz de detectar são:

- Regime permanente: 220 Vrms/3 Arms;
- Falta elétrica: mudança de 220 Vrms/3 Arms para 110 Vrms/5 Arms;
- Entrada de carga de 2,5 MW no alimentador: mudança de 220 Vrms/3 Arms para 200 Vrms/5 Arms;
- Saída de carga de 2,5 MW no alimentador: mudança de 220 Vrms/3 Arms para 230 Vrms/1 Arms.

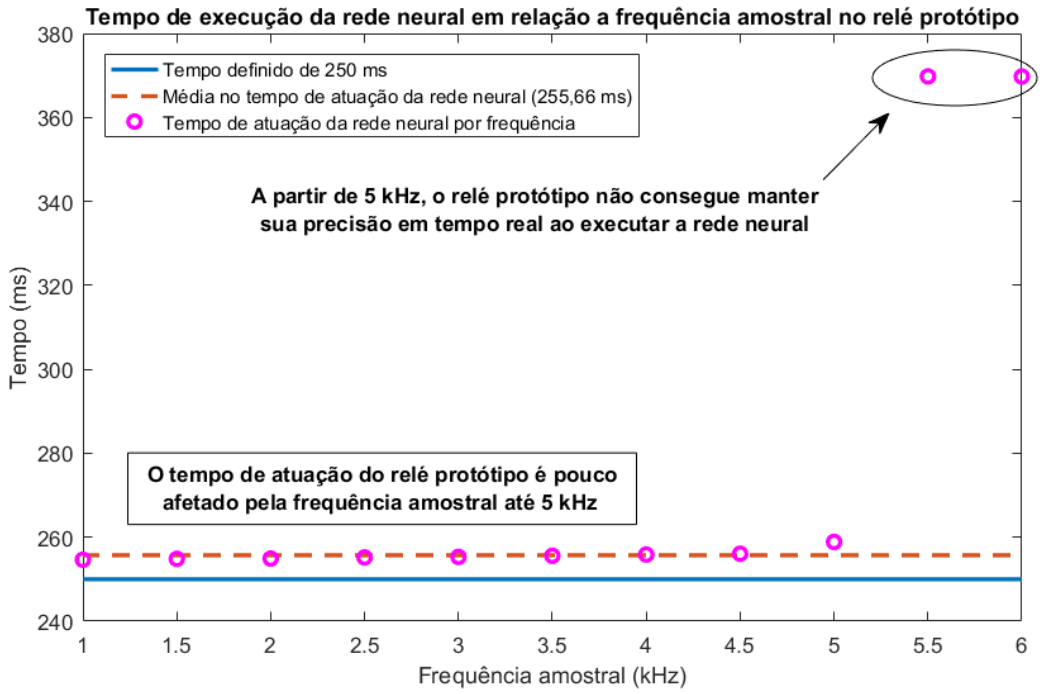
Assim, como no teste do modo de proteção do relé protótipo, o tempo de atuação para a rede neural desenvolvida tomar a decisão se o sinal de entrada representa um dos três eventos elétricos citados foi mantido em 250 ms, mensurando possíveis atrasos. A frequência de amostragem do relé protótipo foi ajustada no valor inicial de 1 kHz e aumentada gradativamente (passos de 0,5 kHz) até que sua precisão em tempo real fosse perdida. A Figura 19 apresenta o tempo de atuação da rede neural por frequência testada.

Pela Figura 19, entre as frequências de amostragem de 1 kHz a 5 kHz, houve pouca variação no valor do tempo de atuação do relé protótipo ao executar a rede neural desenvolvida, concentrando-se ao longo da média de 255,66 ms. Para frequências acima de 5 kHz, semelhante aos teste anterior, o relé protótipo não foi capaz de manter o tempo de atuação determinado para a rede neural. O fato da frequência amostral pouco impactar no tempo de atuação da rede neural desenvolvida é um indicativo de que o peso desse algortimo no tempo de processamento do *Raspberry Pi* é maior do que o gasto na amostragem da janela de dados. Logo, os atrasos gerados a cada iteração da rede neural (tempo médio por iteração de 150 μs) possuem uma contribuição mais significativa no atraso final do sinal de *trip* do que a função de amostragem. Compilando os atrasos medidos entre as frequências amostrais de 1 kHz a 5 kHz, o atraso total do relé protótipo ao executar a rede neural desenvolvida é apresentada na Tabela 5.

A maior variância no tempo de atuação do relé protótipo em relação ao teste com a função de sobrecorrente é um reflexo da complexidade da rede neural executada. Quanto a precisão e perda da característica em tempo real do relé protótipo para este teste; sua representação visual é semelhante a apresentada na Figura 18, uma vez que os atrasos acima da frequência de 5

kHz produziram resultados semelhantes.

Figura 19 – Tempo de atuação da rede neural no relé protótipo de acordo com a frequência de amostragem



Fonte: Autoria própria.

Tabela 5 – Atrasos no tempo de execução da rede neural desenvolvida (acima de 250 ms)

	Atrasos (ms)			
Rede Neural	**Média**	**Mínima**	**Máxima**	**Variância**
Relé protótipo (todas as frequências amostrais medidas até 5 kHz)	5,66	4,65	8,88	1,47
Número de testes da rede neural: 10				

Fonte: Autoria própria.

Durante os testes da rede neural, a precisão da rede neural desenvolvida também foi avaliada, sendo esse valor cerca de 90%, pois apenas um dos dez testes apresentados na Tabela 5 apresentou um resultado diferente do esperado (resultado falso negativo ao detectar um aumento de carga ao invés da falta trifásica).

6 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho apresentou uma nova plataforma que pode ser utilizada como um relé universal, capaz de testar novas funções de proteção em tempo real. Dentre suas características, destaca-se sua acessibilidade (componentes facilmente encontrados no mercado), não requerer programação em baixo nível ou profundo conhecimento de sua arquitetura. Embora o componente principal deste relé protótipo, um *Raspberry Pi 3B+*, não tenha sido originalmente criado para executar sistemas em tempo real, como sistemas de proteção, este minicoputador é capaz de fornecer esse tipo de resposta através de configurações no *kernel* de seu Sistema Operacional.

Combinando o *Raspberry Pi 3B+* com sensores de tensão e corrente CA, e um conversor A/D, mostrou-se que a plataforma desenvolvida é capaz de responder a faltas elétricas trifásicas de modo semelhante à um relé comercial. O relé protótipo foi capaz de fornecer respostas em tempo real em frequências de amostragem de até 5 kHz, além de executar uma rede neural com precisão em seu tempo de atuação, o que mostra sua capacidade em lidar com funções de proteção mais complexas. Contudo, ele não se enquadra como um sistema em tempo real crítico, atendendo mais as características de um sistema em tempo real leve. Mesmo com as limitações apresentadas, a plataforma desenvolvida pode ser aplicada em ambiente educacionais no ensino de proteção de sistemas elétricos; dando oportunidade aos usuários de entenderem como as funções básicas de proteção operam de forma detalhada, além de criarem suas próprias funções de proteção e expandir a biblioteca do relé protótipo. Além disso, a evolução da arquitetura e poder de processamento dos minicomputadores podem levar a evolução desse relé protótipo, melhorando sua robustez para lidar com sistemas em tempo real críticos.

Mais testes serão necessários para avaliar se mais funcionalidades podem ser adicionadas a esta plataforma sem comprometer sua resposta em tempo real, como transformadas de Fourier ou Wavelet; outras funções de proteção para sistemas de potência CA, como proteção de distância, direcional, diferencial ou até mesmo novos algoritmos de proteção já desenvolvidos na literatura.

6.1 Apresentação de Artigos e Periódicos

Ao longo dessa pesquisa, dois trabalhos científicos foram publicados. O resumo bibliográfico e os testes em tempo real apresentados na Seção 2 deste trabalho foi o foco do artigo apresentado no Simpósio Brasileiro de Automação Inteligente (SBAI) no ano de 2019, na cidade

de Ouro Preto – MG (LOPES; BERTHO JUNIOR, 2019a), intitulado *Estudo de tecnologias alternativas para validação de algoritmos de proteção.* Este mesmo artigo foi adaptado para ser publicado em um capítulo da revista *Produção do Conhecimento na Engenharia Elétrica 2* (Qualis L1), publicada pela Atena Editora, no ano de 2019 (LOPES; BERTHO JUNIOR, 2019b). Além disso, o artigo *Development of a low-cost relay prototype for real-time power protection functions* foi submetido e aprovado para publicação na revista *Learning and NonLinear Models* (Qualis B5), da Sociedade Brasileira de Inteligência Computacional, ao tempo de escrita desta dissertação.

REFERÊNCIAS

ANEEL. **Resolução normativa Aneel nº 1.000, de 7 de dezembro de 2021**. [S.l.]: ANEEL, 2021.

AQUEL, A. **Introduction to raspberry pi 3 B+**. 2018. The engineering projects. Disponível em: <https://www.theengineeringprojects.com/2018/07/introduction-to-raspberry-pi-3-b-plus.html>. Acesso em: 04 dez. 2021.

ATPDRAW. **Welcome to the web page of ATPDraw**. 2012. ATP Draw. Disponível em: <https://www.atpdraw.net/index.php>. Acesso em: 04 dez. 2021.

BUTTAZZO, G. C. **Hard real-time computing systems: predictable scheduling algorithms and applications**. [S.l.]: Springer, 2013. v. 2.

CHALAS, K. **Evaluation of Real-time operating systems for FGC controls**. [S.l.]: CERN, 2015.

CONPROVE ENGENHARIA. **CE – 6006**. 2021. Conprove. Disponível em: <https://conprove.com/produto/02-ce-6006-testador-universal-hexafasico-e-analisador-de-energia-microprocessado-com-protocolo-iec-61850-mala-de-testes-de-reles-hexafasica/>. Acesso em: 04 dez. 2021.

DA SILVA, I. N. **Redes neurais artificiais para engenharia e ciências aplicadas: fundamentos teóricos e aspectos práticos**. [S.l.]: Artliber, 2016. v. 3.

DANTAS, D. T.; PELLINI, E. L.; MANASSERO JUNIOR, G. Energy and reactive power differential protection hardware-in-the-loop validation for transformer application. **The Journal of Engineering**, v. 2018, n. 15, p. 1160–1164, 2018.

ETC. **ZMPT101B(ZMPT107) voltage transformer operating guide**. [S.l.]: ETC, 2021.

GERUM, P. **Home**. 2022. Xenomai. Disponível em: <https://source.denx.de/Xenomai/xenomai/-/wikis/home>. Acesso em: 04 ago. 2022.

HERNANDEZ, M. E. *et al.* Embedded real-time simulation platform for power distribution systems. **IEEE Access**, v. 6, p. 6243–6256, 2017. Disponível em: <https://ieeexplore.ieee.org/abstract/document/8214096>. Acesso em: 05 mai. 2022.

JOHANSSON, G. **Real-time linux testbench on raspberry pi 3 using xenomai**. Monografia (Especialização) — School of electrical engineering and computer science, Stockholm, Sweden, 2018.

JOHN, A. *et al.* Automation of 11 kv substation using raspberry pi. **International Conference on Circuit, Power and Computing Technologies (ICCPCT)**, 2017. Disponível em: <https://ieeexplore.ieee.org/abstract/document/8074264>. Acesso em: 05 mai. 2022.

JOY, U. B. *et al.* Microcontroller based feeder protection system from various fault conditions in distribution line. **International Conference on Innovations in Science, Engineering and Technology (ICISET)**, IEEE, p. 89–94, 2022.

KUFFEL, R.; FORSYTH, P.; PETERS, C. The role and importance of real time digital simulation in the development and testing of power system control and protection equipment. **IFAC-PapersOnLine**, Elsevier, v. 49, n. 27, p. 178–182, 2016.

LEARN OPEN ENERGY MONITOR. **CT Sensors - Interfacing with an Arduino**. n.d. Learn Open Energy Monitor. Disponível em: <https://learn.openenergymonitor.org/electricity-monitoring/ct-sensors/interface-with-arduino>. Acesso em: 04 dez. 2021.

LECCESE, F. *et al.* A new power quality instrument based on raspberry-pi. **Electronics**, v. 5, n. 4, 2016. Disponível em: <https://www.mdpi.com/2079-9292/5/4/64>. Acesso em: 05 mai. 2022.

LOPES, P. R.; BERTHO JUNIOR, R. Estudo de tecnologias alternativas para validação de algoritmos de proteção. In: SOCIEDADE BRASILEIRA DE AUTOMÁTICA, 14., 2019, Ouro Preto, MG. **Anais eletrônicos...** Campinas, SP: Simpósio Brasileiro de Automação Inteligente, 2019. Disponível em: <https://proceedings.science/sbai-2019/papers/estudo-de-tecnologiasalternativas-para-validacao-de-algoritmos-de-protecao>. Acesso em: 12 out. 2021.

LOPES, P. R.; BERTHO JUNIOR, R. Estudo de tecnologias alternativas para validação de algoritmos de proteção. In: HOLZMANN, H. A.; DALLAMUTA, J.; GRANZA, M. H. **A Produção do Conhecimento na Engenharia Elétrica 2**. Ponta Grossa, PR: Atena Editora, 2019. p. 147–158.

MARTINS, C. A. P. **Desenvolvimento de hardware para aquisição de sinais e processamento de algoritmos de proteção e controle**. Dissertação (Mestrado) — Universidade Estadual de Campinas, 2017.

MCKENNEY, P. **A realtime preemption overview**. 2005. LWN.Net. Disponível em: <https://lwn.net/Articles/146861/>. Acesso em: 04 ago. 2022.

MICROCHIP. **MCP3004/3008**. [S.l.]: Microchip, 2008.

MITRA, S.; CHATTOPADHYAY, P. Design and implementation of flexible numerical overcurrent relay on fpga. **International Journal of Electrical Power and Energy Systems**, v. 104, p. 797–806, 2019.

MOLLOY, D. **Exploring Raspberry Pi: Interfacing to the Real World with Embedded Linux**. [S.l.]: Wiley, 2016. v. 1.

MONARO, R. M. **Lógica fuzzy aplicada na melhoria da proteção digital de geradores síncronos**. Tese (Doutorado) — Universidade de São Paulo, 2013.

MONARO, R. M. *et al.* Sistema integrado para desenvolvimento e execução em tempo real de algoritmos de proteção de sistemas elétricos. **Sba: Controle e Automação**, Sociedade Brasileira de Automatica, v. 23, n. 2, p. 202–215, 2012.

ORDONEZ, E.; PENTEADO, C.; DA SILVA, A. **Microcontroladores e FPGAs: aplicações em automação**. [S.l.]: Novatec Editora, 2005. v. 1.

PELLINI, E. L. *et al.* Custom distribution feeder recloser ied with high impedance protection function. **International Conference and Exhibition on Electricity Distribution**, v. 22, p. 1–4, 2013.

RASPBERRY PI FOUNDATION. **Raspberry Pi Foundation - About Us**. 2021. Raspberry Pi. Disponível em: <https://www.raspberrypi.org/about/>. Acesso em: 04 dez. 2021.

ROBERTS, T. **Solution: Dedicating one core to a real-time process**. 2013. Raspberry Pi Forums. Disponível em: <https://forums.raspberrypi.com/viewtopic.php?t=228727>. Acesso em: 04 dez. 2021.

RYAN, W. **GitHub - wryan67/VoltageCatcher: Capture voltage using Raspberry Pi and mcp3008**. 2020. GitHub. Disponível em: <https://github.com/wryan67/VoltageCatcher>. Acesso em: 04 dez. 2021.

SCHNEIDER ELECTRIC. **Easergy MiCOM P14x Feeder Management Relay**. [S.l.]: Schneider Electric, 2020.

TAM, H. **GitHub - thanhtam-h/rpi23-rt: Preempt-rt patched kernel 4.9.80 for raspberry pi 2, 3 (include 3b+)**. 2018. GitHub. Disponível em: <https://github.com/thanhtam-h/rpi23-rt>. Acesso em: 04 dez. 2021.

TEKTRONIX. **THS3000 Handheld Oscilloscope**. 2022. Tektronix. Disponível em: <https://www.tek.com/en/products/oscilloscopes/ths3000-handheld-oscilloscope>. Acesso em: 05 mar. 2022.

TENSORFLOW. **TensorFlow Lite for microcontrollers**. 2021. Disponível em: <https://www.tensorflow.org/lite/microcontrollers>. Acesso em: 01 jul. 2022.

WALTER, J.; FAKIH, M.; GRUTTNER, K. Hardware-based real-time simulation on the raspberry pi. **Workshop on High performance and Real-time Embedded Systems**, 2013. Disponível em: <https://pdfs.semanticscholar.org/52dd/916ea279d21a0d4335f80e181314a975fa70.pdf?_ga=2.200705198.1482294817.1567108814-132626642.1531676532>. Acesso em: 05 mai. 2022.

WANG, M. *et al.* A review on ac and dc protection equipment and technologies: Towards multivendor solution. In: CIGRÈ WINNIPEG 2017 COLLOQUIUM, Winnipeg, Canada. [S.l.], 2017.

WANG, Y.; DINAVAHI, V. Real-time digital multi-function protection system on reconfigurable hardware. **IET Generation, Transmission and Distribution**, v. 10, n. 10, p. 2295–2305, 2016.

WEI, M.; CHEN, Z. Distribution system protection with communication technologies. **IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society**, 2010.

WINKLE, L. V. **GitHub - codeplea/genann: simple neural network library in ANSI C**. 2020. GitHub. Disponível em: <https://github.com/codeplea/genann>. Acesso em: 04 dez. 2021.

YHDC. **Split core current transformer**. [S.l.]: YHDC, 2021.

APÊNDICE A – CIRCUITO ESQUEMÁTICO DO RELÉ PROTÓTIPO E CUSTOS RELACIONADOS

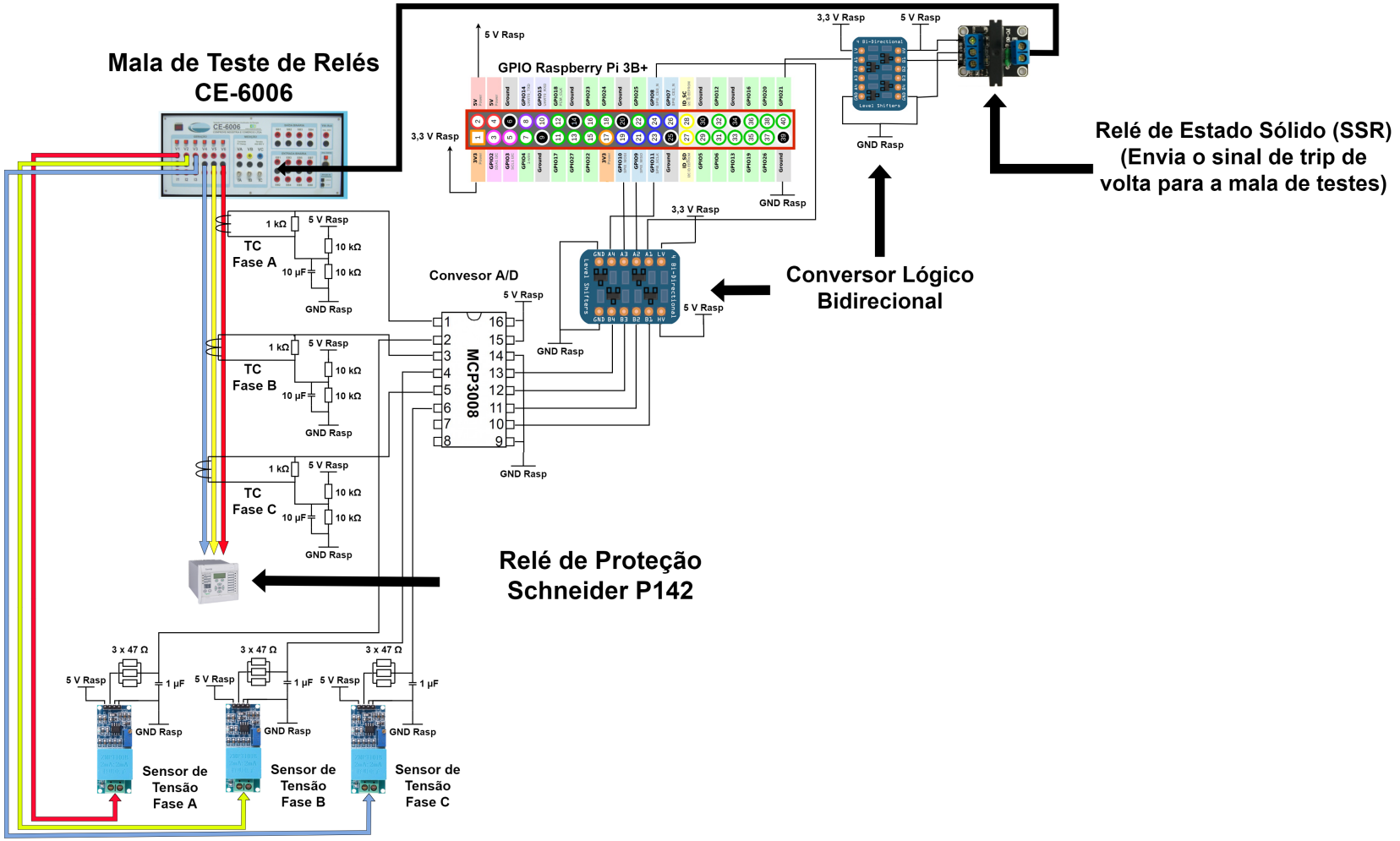
A Tabela 6 lista todos os componentes elétricos e eletrônicos utilizados na construção do relé protótipo, bem como seu custo médio de mercado (valores na data 04/06/2022). Os preços apresentados foram obtidos em várias lojas *online* de componentes eletrônicos. Não foram incluídos os preços do Relé Schneider P142 ou da Mala de testes da Conprove Engenharia, visto que eles foram utilizados somente para testar o relé protótipo. Caso um usuário dessa plataforma queira replicar o circuito apresentado neste trabalho, tanto o relé comercial quanto a fonte de sinais que alimenta o relé protótipo podem ser adaptadas para outros equipamentos conforme a necessidade do usuário. Quanto ao circuito esquemático do relé protótipo, este é apresentado na Figura 20.

Tabela 6 – Componentes eletrônicos utilizados no relé protótipo e seu custo

Componente	**Quantidade**	**Custo unitário (R$)**	**Custo Total (R$)**
Módulo Sensor de Tensão AC 0 a 250V ZMPT101B	3	31,26	93,78
Sensor de Corrente Não Invasivo 20A SCT-013	3	76,90	230,70
Protoboard 400 Pontos	1	17,00	17,00
Kit com 140 Jumpers em U para Protoboard	1	23,66	23,66
Resistor 47R 5% (1/4W)	9	0,06	0,54
Resistor 1K 5% (1/2W)	3	0,14	0,42
Resistor 10K 5% (1/2W)	6	0,35	2,10
Capacitor Eletrolítico 1µF/50V	3	0,23	0,69
Capacitor Eletrolítico 10µF/100V	3	0,23	0,69
MCP3008 Conversor Analógico/Digital	1	34,90	34,90
Conversor de Nível Lógico 3.3V-5V Bidirecional - 4 Canais	2	6,56	13,12
Módulo Relé de Estado Sólido SSR 5V	1	20,45	20,45
Raspberry Pi 3 Model B+	1	431,12	431,12
Total Investido: R$ 869,17			

Fonte: Autoria própria.

Figura 20 – Circuito esquemático do relé protótipo



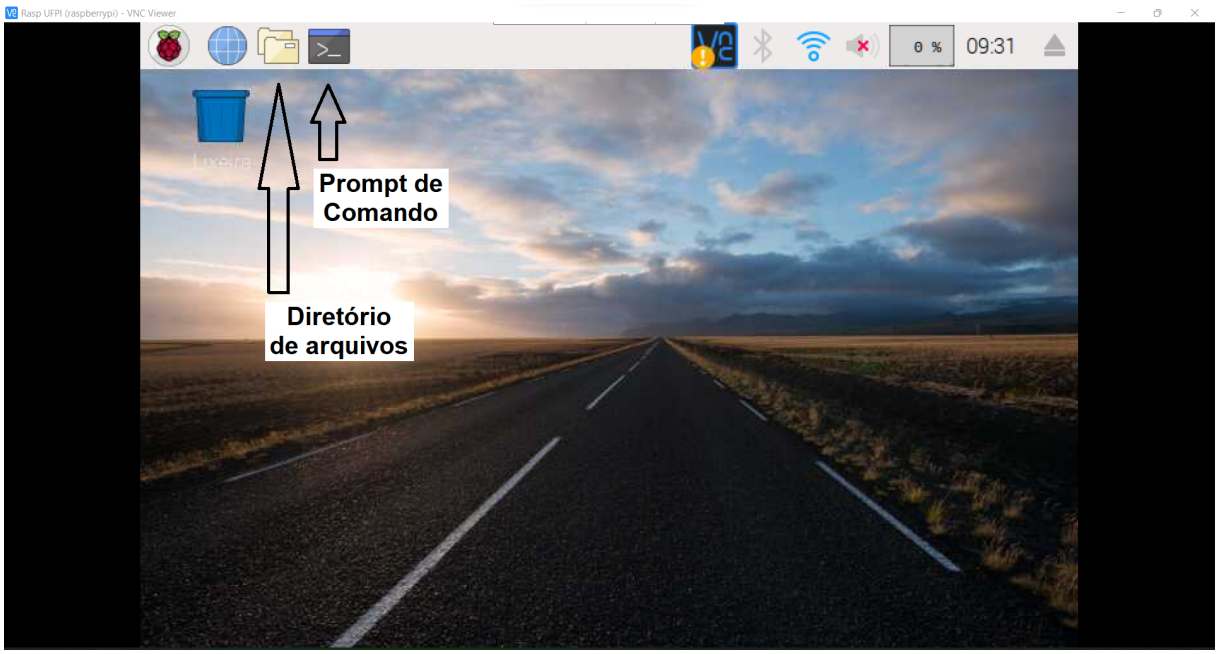
Fonte: Autoria própria

APÊNDICE B – MODO DE USO DO RELÉ PROTÓTIPO

A biblioteca de proteção desenvolvida neste trabalho pode ser encontrada neste link[1], contidos em uma pasta intitulada *Universal Relay Library*. Recomenda-se que, para o funcionamento correto dessa biblioteca, o usuário adquira o modelo 3B ou 3B+ do *Raspberry Pi*, visto que, para outras versões mais recentes desse minicomputador, adaptações mais profundas na biblioteca podem ser necessárias.

Uma vez instalado o Raspberry Pi OS e o *patch* Preempt RT (seguindo as instruções em Raspberry Pi Foundation (2021) e Tam (2018), respectivamente), seja por acesso direto ou remoto ao minicomputador, o usuário terá acesso ao seguinte ambiente interno do *Raspberry Pi*, como mostra a Figura 21.

Figura 21 – Tela inicial do Raspberry Pi 3B+



Fonte: Autoria própria.

Dentre os arquivos baixados anteriormente (eles podem ser alocados em qualquer diretório a escolha do usuário), os arquivos de referência da biblioteca de proteção estão contidos na pasta “SRC”; cujo nome é uma abreviação para *System Control Register*. A relação de arquivos dentro dessa pasta e suas funcionalidades são explanadas a seguir:

- *Main.cpp*: arquivo principal da biblioteca, a qual concentra as funções de amostragem, medição e proteção;

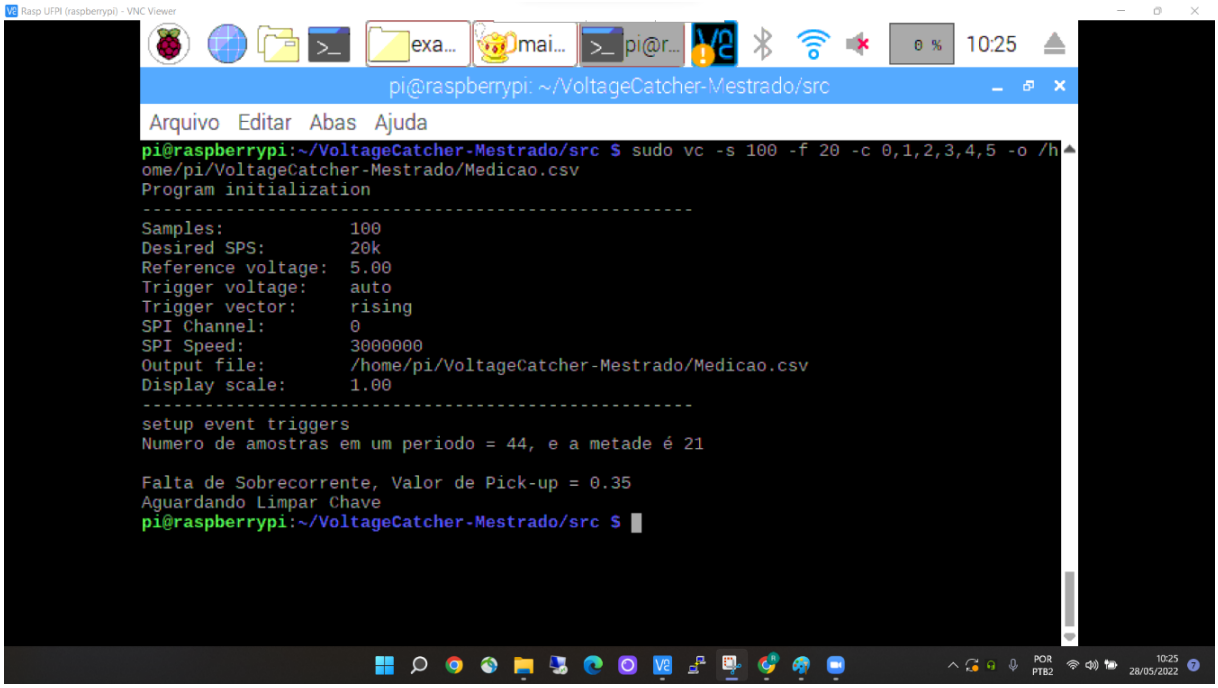
[1] <https://drive.google.com/file/d/1OpTRuVKFBkF_9bo69tIHYCia7Baj2xpi/view?usp=sharing>

- *Options.cpp*: arquivo secundário responsável pelas variáveis que recebem os parâmetros de entrada o relé protótipo (apresentado na Subseção 3.4 deste trabalho);
- *Realtime.cpp*: arquivo secundário que concentra as funções de tempo real, tanto para priorização de funções, quanto para escolher qual dos *cores* do *Raspberry Pi* será utilizado na priorização;
- *Genann.c*: arquivo secundário que concentra as funções para criação, treinamento e execução de redes neurais.

Juntamente com os arquivos de trabalho *.c* e *.cpp*, também há os arquivos de biblioteca *.h* equivalente para cada arquivo. Devido ao uso de funções C/C++ em conjunto no arquivo *Main.cpp*, foi utilizado um *Makefile* para compilar todos as bibliotecas citadas em um único arquivo resultante.

Para execução das funções de proteção dentro do prompt de comando do *Raspberry Pi*, o usuário deve declarar os parâmetros de entrada apresentados neste trabalho como mostra a Figura 22. A seguir, é detalhado o que cada elemento apresentado representa. A resposta fornecida pelo *Raspberry Pi* vale tanto para a falta de sobrecorrente quanto para a rede neural.

Figura 22 – Declaração dos parâmetros de entrada dentro do Raspberry Pi 3B+



Fonte: Autoria própria.

- *sudo*: comando para acessar as funções do Raspberry Pi como administrador;

- *vc*: referência ao código *Voltage Catcher* criado por Ryan (2020), código no qual este trabalho se baseou para criar as bibliotecas do relé protótipo;
- *-s*: comando para registrar quantas amostras iniciais o conversor A/D do relé protótipo deve ler para identificar se os sinais de entrada são do tipo corrente alternada (CA). Esse comando é opcional para o usuário, portanto não é um parâmetro de entrada. Para o exemplo da Figura 22, 100 medidas de tensão e corrente de cada fase foram amostradas;
- *-f*: declaração da frequência amostral desejada pelo usuário, na escala de grandeza em kHz. Nos testes em laboratório, notou-se que há uma discrepância entre a frequência desejada e a realmente executada pelo conversor A/D. No exemplo da Figura 22, o valor de 20 kHz correspondeu somente a 2,6 kHz na amostragem real (como mostra o texto referenciando o número de amostras mais adiante). Os dados apresentando neste trabalho referente as limitações de frequência amostral do relé protótipo fazem referência ao valor real de amostragem;
- *-c*: indicação de quais canais do conversor A/D serão amostrados. Na Figura 22, os canais de 0 a 5 (pinos 1 a 6 do MCP3008) foram amostrados, representando as tensões e correntes trifásicas;
- *-o*: diretório onde o arquivo de oscilografia será salvo. No exemplo da Figura 22, o arquivo foi nomeado como *Medicao.csv*, e será salvo no mesmo diretório onde está a pasta SRC.

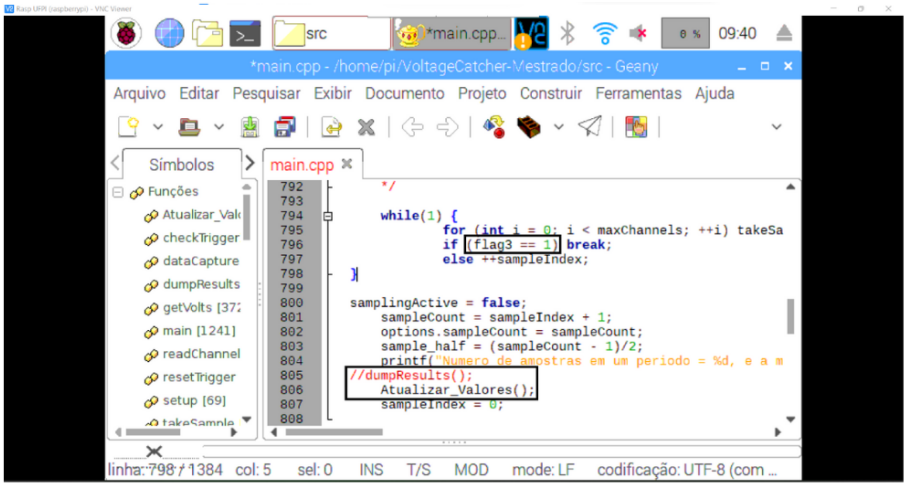
Como resposta aos parâmetros de entrada apresentados, o *Raspberry Pi* retorna o número de amostras que por ciclo, além do número de amostras por meio ciclo (como mostra a Figura 22). Para a função de proteção de sobrecorrente, ele também retorna o valor eficaz (RMS) da corrente de falta (o valor baixo de corrente de falta é resultado da subtração do valor médio do sinal, de modo que o sinal amostrado tenha cruzamento no zero). Por fim, o texto na Figura 18 indicando "Aguardando Limpar Chave"refere-se a mudança de estado do relé de estado sólido de LIGADO para DESLIGADO, após o sinal de *trip* ter sido enviado para a mala de testes de relés.

No estágio atual de desenvolvimento das bibliotecas do relé de protótipo, algumas de suas funcionalidades apresentadas neste trabalho ainda não estão disponíveis dentro do espaço do *prompt* de comando do *Raspberry Pi*, sendo necessário alterações dentro do código do arquivo *Main.cpp* para que elas sejam utilizadas pelo usuário. Uma delas é a opção de mudar o modo de operação do relé protótipo, entre seus modos de amostragem e proteção. Para isso, é necessário

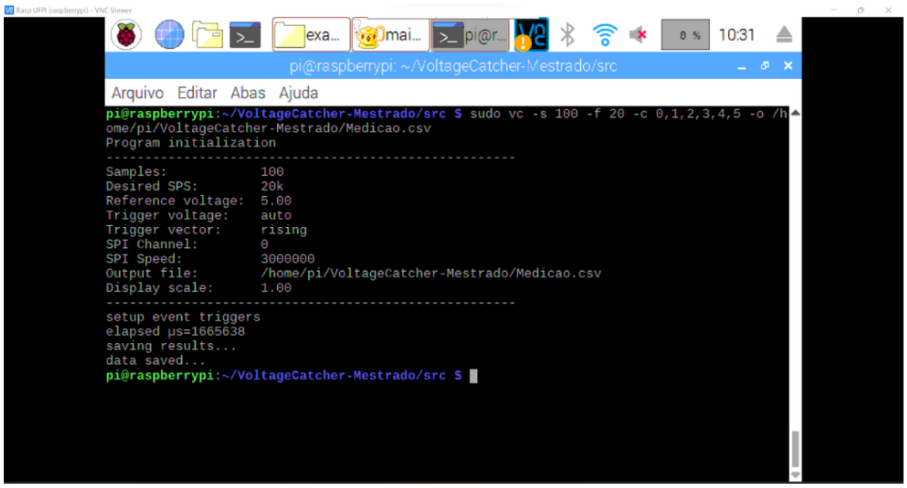
editar o valor condicional da variável *flag3*. Na Figura 22, o valor da variável *flag3* é atualmente 1, além do comando *dumpResults()* estar comentado, o que indica o modo de operação do relé protótipo como relé de proteção (a função *Atualiza_Valores()* contém as funções que atualizam a janela de dados amostrada para a proteção, além de calcular o valor médio e eficaz dos sinais de entrada e executar a função de proteção). Para que o modo de amostragem seja ativado, o usuário deve inserir no valor *flag3* o número de janela de dados que ele deseja amostrar e salvar no arquivo de oscilografia (valor maior que 1). Além disso, o comentário da função *dumpResults()* deverá ser removido (a função *Atualiza_Valores()* deverá ser comentada, consequentemente). Caso o modo de amostragem seja escolhido, a resposta do *Raspberry Pi* é apresentada na Figura 23, assim como o modelo do arquivo de oscilografia.

Em relação as funções de proteção, o relé protótipo possui as funções de sobrecorrente, subtensão e a rede neural desenvolvida neste trabalho. Além disso, ele também contém funções para o cálculo do valor médio e eficaz dos sinais amostrados. Caberá ao usuário criar não só novas funções matemáticas dentro do relé protótipo, como também os novos algoritmos de proteção. Eles deverão ser declarados dentro da função *Atualiza_Valores()*, como mostra a Figura 24. Outra alternativa é o desenvolvimento do algoritmo em um arquivo *.c* separado, o que pode ser chamado pela *Main.cpp*, como foi o caso da rede neural desenvolvida. Na função *Atualiza_Valores()*, o usuário também deve escolher o valor do tempo definido que sua função de proteção deve ocorrer, na grandeza de microssegundos, como também mostra a Figura 24.

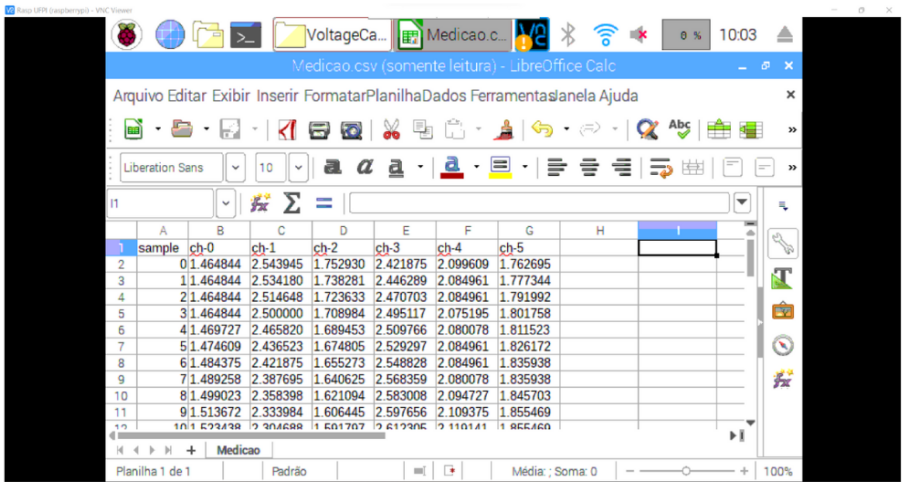
Figura 23 – Passo a passo para ativar o modo de amostragem do relé protótipo



(a) Alterar valor condicional da variável flag3



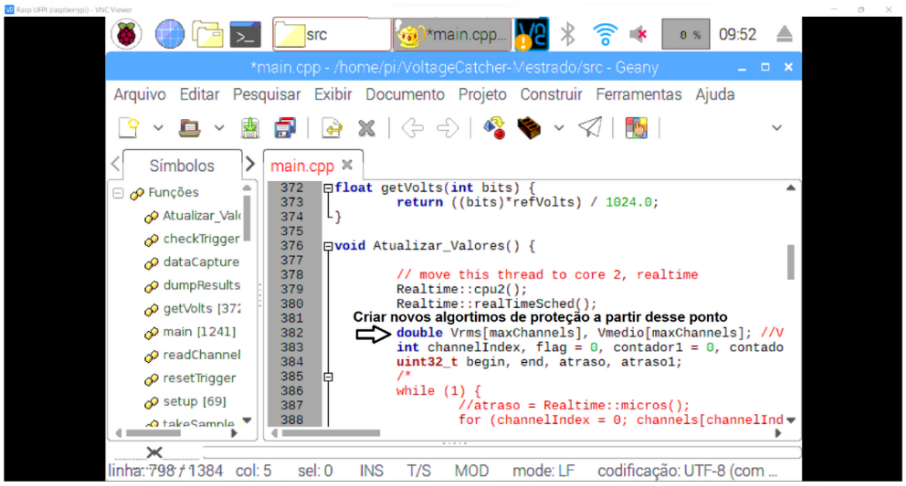
(b) Resposta do relé protótipo



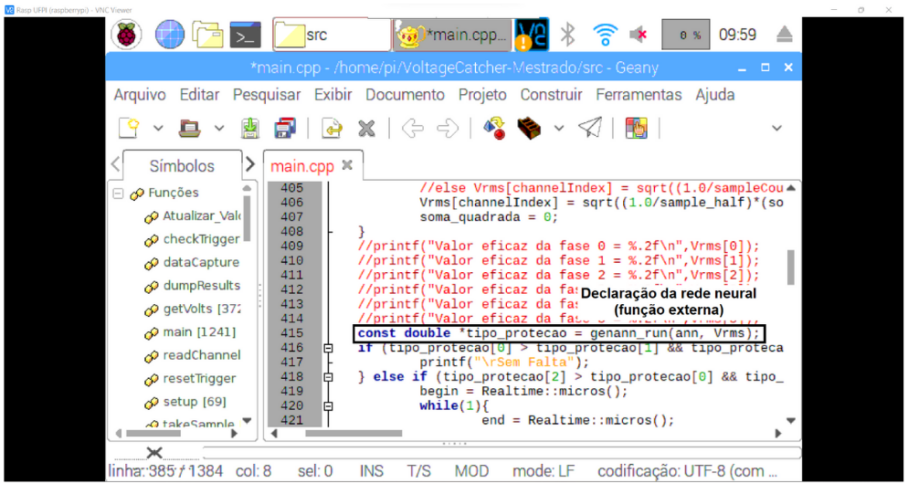
(c) Layout do arquivo de oscilografia

Fonte: Autoria própria.

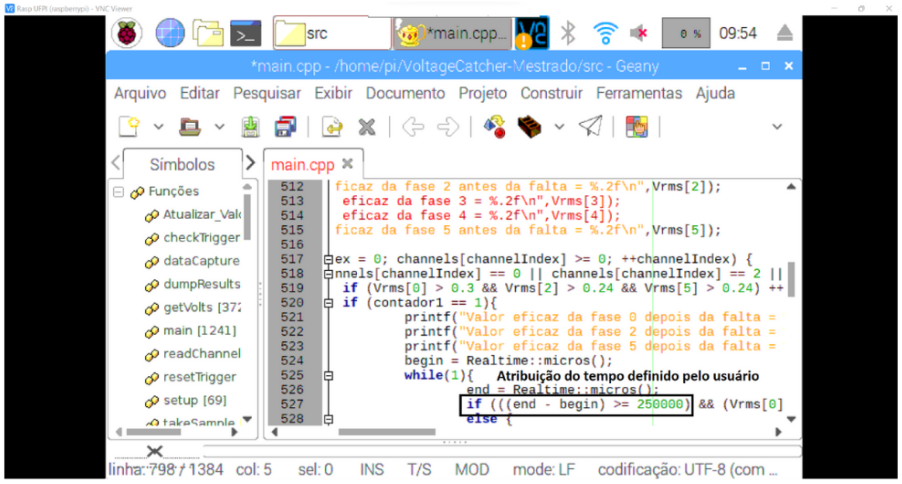
Figura 24 – Espaço para declaração as funções de proteção dentro do relé protótipo



(a) Função de proteção interna



(b) Função de proteção externa



(c) Atribuição do tempo definido

Fonte: Autoria própria.