UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
CURSO DE ENGENHARIA ELÉTRICA

LUÍS CARLOS MAGRIL JÚNIOR

SISTEMA DE MONITORAMENTO DE MÁQUINAS ELÉTRICAS VIA TCP/IP

TRABALHO DE CONCLUSÃO DE CURSO

CORNÉLIO PROCÓPIO
OUTUBRO/2015

LUÍS CARLOS MAGRIL JÚNIOR

SISTEMA DE MONITORAMENTO DE MÁQUINAS ELÉTRICAS VIA TCP/IP

Trabalho de Conclusão de Curso, do curso de Engenharia Elétrica da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Engenheiro Eletricista.

Orientador: Prof. Dr. Marcelo Favoretto Castoldi

Co-orientador: Prof. Dr. Alessandro Goedtel

CORNÉLIO PROCÓPIO

OUTUBRO/2015

Universidade Tecnológica Federal do Paraná
Campus Cornélio Procópio
Departamento de Engenharia Elétrica
Curso de Engenharia Elétrica

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS CORNÉLIO PROCÓPIO

FOLHA DE APROVAÇÃO

Luis Carlos Magril Junior

Sistema de monitoramento de máquinas elétricas via TCP/IP

Trabalho de conclusão de curso apresentado às 13:30hs do dia 18/11/2015 como requisito parcial para a obtenção do título de Engenheiro Eletricista no programa de Graduação em Engenharia Elétrica da Universidade Tecnológica Federal do Paraná. O candidato foi arguido pela Banca Avaliadora composta pelos professores abaixo assinados. Após deliberação, a Banca Avaliadora considerou o trabalho aprovado.

_____
Prof(a). Dr(a). Marcelo Favoretto Castoldi - Presidente (Orientador)

_____
Prof(a). Dr(a). Alessandro Goedtel - (Coorientador)

_____
Prof(a). Dr(a). Paulo Rogério Scalassara - (Membro)

_____
Prof(a). Dr(a). Wagner Endo - (Membro)

A folha de aprovação assinada encontra-se na coordenação do curso.

Aos meus pais.

AGRADECIMENTOS

Agradeço a toda minha família e principalmente aos meus pais, por terem proporcionado todo o suporte necessário para que eu pudesse concluir a graduação. Sou grato pelo incentivo e direcionamento dados durante toda a minha vida e, muito além disso, principalmente pelos exemplos de responsabilidade, trabalho e dedicação que são para mim.

À minha esposa por todo amor, carinho e compreensão durante os longos anos em que tivemos que estar afastados um do outro. Sem seu sorriso tranquilizante e suas palavras de incentivo, tudo teria sido mais difícil.

Agradeço ao Prof. Dr. Marcelo Favoretto Castoldi e ao Prof. Dr. Alessandro Goedtel pela orientação deste trabalho e pela confiança depositada em mim para execução do mesmo.

Por fim, gostaria de agradecer aos meus colegas de universidade, que com certeza facilitaram muito todos estes anos longe de casa e da minha família. Por todas as conversas, risadas e momentos de seriedade que passamos juntos. Estes anos de nossas vidas que compartilhamos certamente estarão para sempre em minha memória, assim como todos vocês.

RESUMO

MAGRIL JUNIOR, Luís Carlos. **Sistema de monitoramento de máquinas elétricas via TCP/IP**. 2015. 111f. Trabalho de Conclusão de Curso – Graduação em Engenharia Elétrica, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2015.

Neste trabalho é desenvolvido um sistema de monitoramento das grandezas elétricas de máquinas elétricas, que disponibiliza os dados em rede, via protocolo TCP/IP. Os dados de interesse são as formas de onda de tensão e corrente que realizam a alimentação das máquinas. Para realizar a aquisição destes sinais, foram desenvolvidas placas de sensoriamento e condicionamento dos sinais de tensão e corrente, em conjunto com um sistema de conversão analógico-digital que amostra e envia os dados para o usuário que os está requisitando na rede. Os sistemas são desenvolvidos de forma que possuam o menor custo de elaboração possível. Foi desenvolvida uma interface de controle e visualização dos dados no software Matlab, onde o usuário pode escolher a frequência de amostragem e os modos de operação disponíveis para aquisição dos dados. Os modos de operação que estão disponíveis ao usuário são os de aquisição em regime permanente, aquisição em regime transitório e atualização automática. O sistema desenvolvido pode ser utilizado de várias formas, podendo ser empregado como meio de aquisição dos sinais elétricos de máquinas para outros estudos. Porém, pode também ser empregado no ensino, onde o aluno pode observar o comportamento das grandezas elétricas das máquinas de maneira simples.

Palavras-Chave: Sistema de monitoramento. Máquinas Elétricas. TCP/IP. Aquisição de sinais. Condicionamento de sinais.

ABSTRACT

MAGRIL JUNIOR, Luís Carlos. **Sistema de monitoramento de máquinas elétricas via TCP/IP**. 2015. 111f. Trabalho de Conclusão de Curso – Graduação em Engenharia Elétrica, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2015.

In this work it is developed a monitoring system of electrical quantities of electrical machines, which can provide the data via TCP/IP protocol. The data of interest are the waveforms of current and voltage that feed the machines. To accomplish the acquisition of these data, sensing and conditioning boards were developed, in conjunction with an analog-to-digital conversion system that sample and sends the data to the user that is requesting in the network. The elaboration of the necessary systems were made in a way that it have the lowest cost possible. It developed a control and visualization interface of the data in the software Matlab, where the user can select the sample frequency and the operation modes available for the acquisition of the data. The operation modes that are available to the user are the acquisition in steady state, acquisition in transient state and automatic update. The system developed can be used in various forms as means of acquisition of the electrical signals for further studies. However, it can also be applied in the learning of electrical machines, where the student can observe the electrical behavior of the machines simply by using the system.

Key-Words: Monitoring System. Electrical Machines. TCP/IP. Signals Acquisiton. Signals Conditioning.

LISTA DE FIGURAS

Figura 1 - Diagrama de blocos de um sistema de monitoramento de dados. 16
Figura 2 - Transformador de potencial. 18
Figura 3 - Circuito divisor de tensão. 18
Figura 4 - Condicionamento com resistor shunt. 19
Figura 5 - Transformador de corrente. 20
Figura 6 - O modelo de referência OSI. 24
Figura 7 - Modelo de camadas TCP/IP. 26
Figura 8 - Modelo de transformador utilizado no protótipo e no sistema final como sensor de tensão. 29
Figura 9 – Transformador de corrente utilizado no trabalho. 30
Figura 10 – Placa de processamento Cerebot Mx7 empregada no trabalho. 32
Figura 11 – Exemplo gráfico do condicionamento de sinal necessário. 34
Figura 12 – Modelo do *Trimmer* utilizado no trabalho. 35
Figura 13 – Circuito de condicionamento desenvolvido. 36
Figura 14 - Forma de onda do secundário distorcida para tensões muito baixas aplicadas ao primário. 38
Figura 15 - Projeto da placa utilizada para ensaio do TC. 39
Figura 16 - Layout do circuito utilizado para ensaio do TC. 39
Figura 17 - Projeto em 3D da placa utilizada para ensaio do TC. 39
Figura 18 - Arranjo experimental para ensaio do TC. 40
Figura 19 - Circuito condicionador excitado com sinal de 60 Hz. 42
Figura 20 - Circuito condicionador excitado com sinal de 120 Hz. 42
Figura 21 - Resposta do circuito condicionador excitado com 240 Hz, com capacitor. 43
Figura 22 - Resposta do circuito condicionador excitado com 240 Hz, sem capacitor. 43
Figura 23 - Resposta do circuito condicionador excitado com 600 Hz, com capacitor. 44
Figura 24 - Resposta do circuito condicionador excitado com 600 Hz, sem capacitor. 44
Figura 25 – Projeto do circuito final de condicionamento de sinais. 46

Figura 26 - Layout da placa de aquisição e condicionamento de sinais. 46
Figura 27 – Projeto final em 3D da placa de condicionamento de sinais. 47
Figura 28 - Protótipo confeccionado da placa de condicionamento de sinais. 48
Figura 29 - Placa de condicionamento aplicada ao sistema final. 48
Figura 30 – Fluxograma da rotina desenvolvida para aquisição em regime permanente. 53
Figura 31 – Fluxograma da rotina desenvolvida para aquisição em regime transitório. 56
Figura 32 – Rotina desenvolvida final. 58
Figura 33 – Interface de inicialização. 60
Figura 34 – Interface de aquisição em regime permanente. 61
Figura 35 – Interface de aquisição em regime permanente em funcionamento. 62
Figura 36 – Interface de aquisição em regime transitório. 63
Figura 37 – Interface de aquisição em regime transitório em funcionamento. 64
Figura 38 – Interface de aquisição com atualização automática. 65
Figura 39 – Interface de aquisição com atualização automática em funcionamento. 66
Figura 40 – Interface que exibe o resumo das grandezas elétricas dos sinais. 67
Figura 41 - Interface que exibe o resumo das grandezas elétricas dos sinais em funcionamento. 68
Figura 42 - Interface para calibração do sistema de aquisição de sinais. 69
Figura 43 - Sistema de monitoramento de máquinas elétricas desenvolvido. 71
Figura 44 - Ligação do sistema de monitoramento ao MIT do LSI. 72
Figura 45 - Monitoramento das tensões do MIT em vazio, utilizando frequência de amostragem de 7680Hz com modo de operação em regime permanente. 73
Figura 46 - Monitoramento das correntes do MIT em vazio, utilizando frequência de amostragem de 7680Hz. 73
Figura 47 - Monitoramento das tensões do MIT em vazio, utilizando frequência de amostragem de 7680Hz, após calibração. 74
Figura 48 - Monitoramento das correntes do MIT em vazio, utilizando frequência de amostragem de 7680Hz, após calibração. 74
Figura 49 - Resumo do sinal de tensão da Fase A, para aquisição com 7680 Hz. 75
Figura 50 - Resumo do sinal de corrente da Fase A, para aquisição com 7680 Hz. 75
Figura 51 - Monitoramento das tensões do MIT em vazio, utilizando frequência de amostragem de 960 Hz. 76

Figura 52 - Monitoramento das correntes do MIT em vazio, utilizando frequência de amostragem de 960 Hz. 76
Figura 53 - Negativa da interface "Grandezas" para frequência de amostragem fora de sua faixa de operação. 77
Figura 54 - Modo de operação com atualização automática em funcionamento com frequência de amostragem de 7680 Hz e tempo de atualização de 3 segundos. 78
Figura 56 – Interface de regime transitório em funcionamento apresentando as correntes trifásicas no período transitório com frequência de amostragem de 1920 Hz. 79
Figura 57 - Tensão e corrente da fase A no período transitório com frequência de amostragem de 1920 Hz. 79
Figura 58 - Correntes trifásicas no período transitório com frequência de amostragem de 480 Hz. 80
Figura 59 - Correntes trifásicas no período transitório com frequência de amostragem de 480 Hz, com torque nominal aplicado ao eixo do MIT. 80

LISTA DE QUADROS

Quadro 1 – Dados elétricos do transformador utilizado no protótipo e no sistema final como sensor de tensão. 29
Quadro 2 – Dados elétricos dos transformadores utilizados no sistema final como sensor de tensão. 29
Quadro 3 – Dados elétricos do transformador de corrente utilizado no trabalho.....30
Quadro 4 – Materiais utilizados no condicionamento de sinais. 36
Quadro 5 - Valores observados no ensaio com o TP. 37
Quadro 6 - Valores observados no ensaio com o TC.....41
Quadro 7 – Materiais utilizados no sistema de condicionamento final. 45
Quadro 8 – Limites operativos da placa de aquisição 47
Quadro 9 – Grandezas envolvidas na amostragem dos sinais do trabalho. 51
Quadro 10 - Resumo das opções de frequências de amostragem disponibilizadas. 52

LISTA DE SIGLAS

A/D	Analógico-Digital
ANEEL	Agência Nacional de Energia Elétrica
CIPECA	Centro Integrado de Pesquisa em Controle e Automação
DARPA	*Defense Advanced Research Projects Agency*
FFT	*Fast Fourier Transform*
ICMP	*Internet Control Message Protocol*
LSI	Laboratório de Sistemas Inteligentes
IP	*Internet Protocol*
ISO	*International Standards Organization*
MIT	Motores de Indução Trifásico
OSI	*Open Systems Interconnection*
TC	Transformador de Corrente
TCP	*Transmission Control Protocol*
TP	Transformador de Potencial
UDP	*User Datagram Protocol*
UTFPR-CP	Universidade Tecnológia Federal do Paraná – Cornélio Procópio

SUMÁRIO

1 INTRODUÇÃO 12
1.1 IDENTIFICAÇÃO DO PROBLEMA 13
1.2 JUSTIFICATIVA 14
1.3 OBJETIVOS 14
1.3.1 Objetivo Geral 15
1.3.2 Objetivos Específicos 15
2 FUNDAMENTAÇÃO TEÓRICA 16
2.1 SISTEMA DE AQUISIÇÃO DE SINAIS 17
2.1.1 Sensor de Tensão 17
2.1.2 Sensor de Corrente 18
2.1.3 Tolerância dos Componentes 20
2.2 PROCESSAMENTO DIGITAL DOS SINAIS 21
2.2.1 Conversão Analógico-Digital 21
2.3 TRANSMISSÃO DOS DADOS 23
2.3.1 Comunicação em Rede 23
2.3.2 Comunicação via Protocolo TCP/IP 26
3 MÉTODO DE PESQUISA 28
3.1 SENSORES 28
3.1.1 Sensor de Tensão 28
3.1.2 Sensor de Corrente 30
3.2 MICROPROCESSADOR 31
3.3 CONDICIONAMENTO DE SINAIS 32
3.3.1 Projeto dos Protótipos de Aquisição 33
3.3.1 Ensaio dos Protótipos de Aquisição e Condicionamento de Sinais 37
3.3.2 Sistema de Aquisição e Condicionamento de Sinais Final 45
3.4 AMOSTRAGEM DOS SINAIS E TRANSMISSÃO DOS DADOS 49
3.4.1 Ambiente de Desenvolvimento 49
3.4.2 Parâmetros de Amostragem 49
3.4.2.1 Rotina de aquisição em regime permanente 52
3.4.2.2 Rotina de aquisição em regime transitório 55
3.4.2.3 Rotina final 57
3.5 INTERFACE DE CONTROLE E VISUALIZAÇÃO 59
3.5.1 Interface de Inicialização 59
3.5.2 Modo de Operação em Regime Permanente 61
3.5.3 Modo de Operação em Regime Transitório 63
3.5.5 Grandezas 66
4 RESULTADOS E DISCUSSÕES 71
4.1 MODO DE OPERAÇÃO EM REGIME PERMANENTE 72
4.2 MODO DE OPERAÇÃO COM ATUALIZAÇÃO AUTOMÁTICA. 77
4.3 MODO DE OPERAÇÃO EM REGIME TRANSITÓRIO 78
CONCLUSÃO 82
REFERÊNCIAS 85
APÊNDICE A – Rotina implementada na placa de processamento 87
APÊNDICE B – Código das interfaces desenvolvidas 91

1 INTRODUÇÃO

Segundo relatório divulgado pela ANEEL (2011), os motores elétricos são responsáveis por até 55% da energia consumida na indústria, sendo os motores de indução trifásicos (MITs) o principal meio de conversão eletromecânica utilizado, devido a sua construção simples, robustez e versatilidade. Os MITs são empregados nas mais diversas aplicações, desde o setor de agricultura até o setor de manufatura, desta forma é muito importante garantir seu funcionamento, evitando paradas indesejadas, falhas que geram custo e proporcionando sua operação de forma segura (WANG, LIU, *et al.*, 2011).

Quando ocorre a interrupção do funcionamento de um motor devido a uma falha inesperada, toda uma linha de produção pode ser afetada. Assim, é de grande importância realizar as manutenções necessárias para evitar este tipo de problema, otimizando o funcionamento e aumentando a vida útil das máquinas.

No modelo da manutenção preventiva, os procedimentos são realizados de forma programada, parando-se a linha de produção e verificando o estado de conservação máquina a máquina, o que pode ocasionar um desperdício de tempo que poderia ser empregado na produção, e de dinheiro, pois em certos casos peças componentes das máquinas são trocadas de forma precipitada.

Atualmente, busca-se mais eficiência com a manutenção preditiva, que tem por objetivo prever o momento em que uma falha irá ocorrer. Uma das formas aplicada como ferramenta para realizar esta previsão é o monitoramento das grandezas elétricas e mecânicas dos motores, que permite verificar o momento correto de se realizar a troca de algum componente, reduzindo o tempo de máquina parado e os custos de manutenção, pois as peças são substituídas apenas quando necessário. Este monitoramento busca informações nas formas de onda dos sinais elétricos das máquinas, sendo que por meio de comparação entre padrões é possível detectar uma variada gama de falhas, onde pode-se atuar antevendo um eventual problema (GONGORA, 2013).

Equipamentos que utilizam as informações contidas nas formas de onda dos sinais elétricos de máquinas e motores têm sido empregados na previsão de falhas de máquinas elétricas (GONGORA, 2013), na estimação da condição de operação e

grandezas mecânicas e também em sistemas supervisórios (RANIERI, 2007), que além de monitorar podem controlar o funcionamento destas máquinas.

1.1 IDENTIFICAÇÃO DO PROBLEMA

Nos sistemas aplicados às maquinas elétricas mencionados anteriormente, a principal informação utilizada é extraída dos sinais elétricos das fases que realizam a energização destas máquinas. Assim, em cada um destes sistemas existe um sistema de aquisição embarcado, que é responsável pela extração das informações dos sinais elétricos.

Percebe-se, então, que em um caso onde é necessário o emprego de vários sistemas aplicados às máquinas elétricas, existirá um número desnecessário de equipamentos de aquisição de sinais para que todos estes sistemas possam realizar suas funções, elevando o custo do processo como um todo.

Uma das opções existentes para contornar este problema é o uso de dispositivos dedicados à aquisição de sinais, como exemplo, pode-se citar a placa de aquisição de dados NI USB-6221 da *National Instruments* (SANTOS, 2012). Esta placa possui um valor a ser desembolsado de, aproximadamente, U$ 1595,00 (NATIONAL INTRUMENTS, 2014), e pode resolver o problema a priori. Entretanto, este sistema acaba disponibilizando os dados apenas localmente, necessitando de um sistema de transmissão de informações adicional para que outras aplicações possam utilizar as informações por ele fornecidas.

Deste modo, mostra-se que em um ambiente onde sejam necessárias as utilizações de dispositivos para monitoramento, detecção de falhas, estimação de parâmetros, entre outras aplicações, será dispendido um valor maior do que o necessário para que essas atividades sejam realizadas.

1.2 JUSTIFICATIVA

Mostra-se mais eficiente um sistema de monitoramento das grandezas elétricas de máquinas elétricas que possa disponibilizar as informações para todos os dispositivos que as necessitam, fazendo com que os custos de todos os equipamentos envolvidos com este processo sejam barateados.

Desta forma, este trabalho propõe uma nova estratégia no que tange o monitoramento de máquinas elétricas, propondo o projeto de um sistema que seja capaz de disponibilizar as grandezas das máquinas elétricas via rede, utilizando o protocolo *Transmission Control Protocol/Internet Protocol*, comumente chamado de TCP/IP, e que possua o menor custo de elaboração possível.

Pelo fato de disponibilizar remotamente as informações dos sinais amostrados, permite que vários sistemas possam utilizar as mesmas para dar continuidade em seus processos sem que tenham que estar fisicamente presentes no local.

Propõe-se, também, o desenvolvimento de uma interface de controle e visualização dos dados, onde o usuário poderá configurar os modos pelo quais deseja realizar a aquisição das informações e, onde possa realizar uma rápida análise do sinal, permitindo a extração de informações básicas de maneira ágil.

Ainda, o sistema de monitoramento de máquinas elétricas pode ser empregado no ensino, onde o aluno poderá verificar de forma simples através de experimentos, o comportamento da máquina em diferentes modos de operação, facilitando o entendimento dos conceitos e aumentando a motivação em relação ao assunto.

1.3 OBJETIVOS

Nesta seção são apresentados os objetivos gerais e específicos deste trabalho.

1.3.1 Objetivo Geral

Implementar um sistema de monitoramento das grandezas elétricas de máquinas elétricas que possa disponibilizar os dados obtidos na rede de comunicação local via protocolo TCP/IP.

1.3.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

- Utilizar os componentes necessários ao projeto que melhor se adaptarão com relação à aplicação e ao custo.
- Definir os elementos sensores que serão utilizados para medição das tensões e correntes.
- Definir o microprocessador que atenderá aos requisitos do trabalho.
- Projetar os condicionadores de sinais que serão empregados no projeto.
- Realizar ensaios em laboratório verificando o funcionamento dos sensores e condicionadores.
- Disponibilizar os dados oriundos das leituras em uma rede de comunicação local via protocolo TCP/IP.
- Desenvolver uma interface para visualização dos dados aquisitados.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção será realizada a explanação teórica dos conceitos envolvidos no desenvolvimento do trabalho, sendo abordados os princípios de um sistema de monitoramento de dados aplicados às máquinas elétricas. Este sistema é constituído por elementos e processos, tais como: sensores, condicionadores, processamento digital dos dados e a transmissão de dados em rede via protocolo TCP/IP. Assim pode-se dividir o sistema de monitoramento em três principais etapas:

- Aquisição e condicionamento dos sinais;
- Processamento digital dos sinais;
- Transmissão dos dados e visualização dos sinais;

Observa-se, na Figura 1, um sistema genérico de monitoramento de sinais, onde se pode notar as três etapas mencionadas anteriormente.

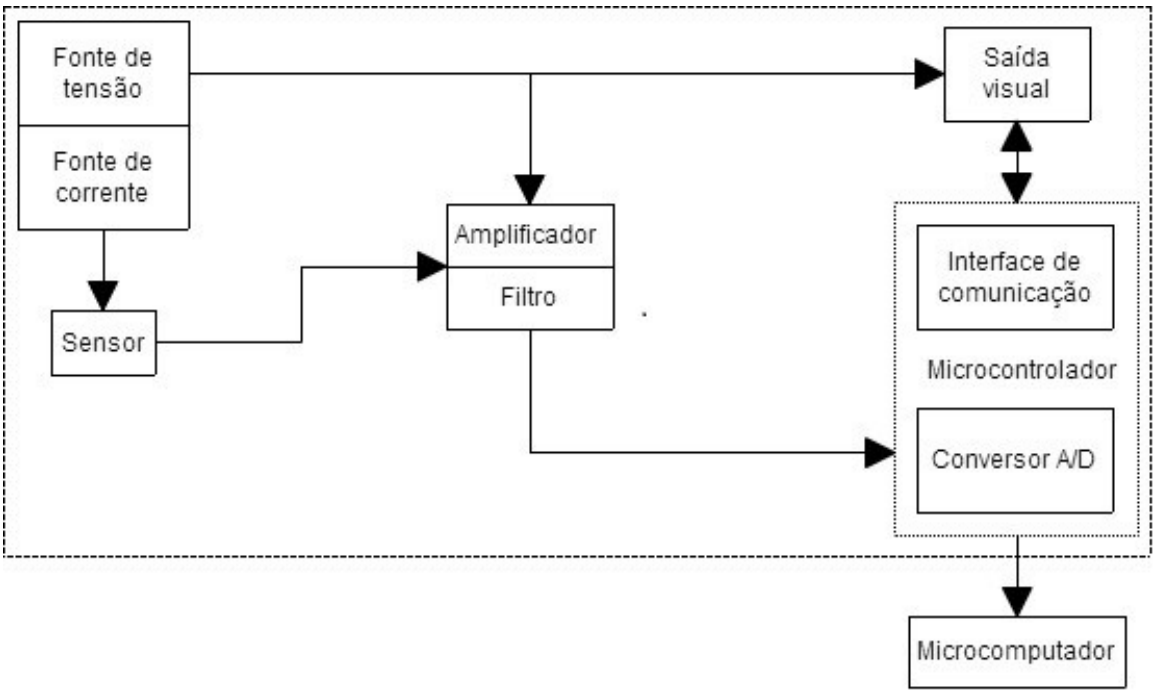

Figura 1 - Diagrama de blocos de um sistema de monitoramento de dados.
Fonte: Adaptado de Balbinot e Brusamarello (2011, p. 101).

2.1 SISTEMA DE AQUISIÇÃO DE SINAIS

Um sistema de aquisição de sinais tem por objetivo coletar os sinais provenientes de uma medição analógica para que sejam passíveis de serem manipulados por sistemas digitais, sendo formado por sensores, filtros e amplificadores. Este arranjo é necessário uma vez que os sistemas digitais trabalham com baixas tensões (BALBINOT e BRUSAMARELLO, 2011).

Neste trabalho será necessário realizar aquisição de seis grandezas analógicas, sendo elas os sinais referentes às tensões e correntes de alimentação do motor de indução trifásico. Assim, deve-se analisar a melhor forma de se proceder tendo em vista estes parâmetros de projeto.

2.1.1 Sensor de Tensão

Como os níveis de tensões no sistema elétrico não são adequados para serem enviados diretamente para um sistema de aquisição digital, faz-se necessário reduzir esta grandeza de forma a torná-la adequada aos níveis de tensão apropriados. Para isso, pode-se reduzir esses níveis para torná-los aceitáveis de duas principais maneiras, utilizando um circuito divisor de tensão, que geralmente é o mais utilizado no caso de a forma de onda da tensão ser contínua, ou utilizando um transformador de potencial (TP), que é o mais utilizado quando se trata de tensão alternada (THOMAZINI e ALBUQUERQUE, 2011). O TP é um transformador de instrumentação utilizado para reduzir as magnitudes de tensão e possibilitar a realização de medidas. Seu enrolamento primário é projetado para uma alta tensão, enquanto o secundário para baixa tensão, como pode ser observado na Figura 2. O objetivo deste equipamento é apresentar uma amostra fiel do sinal de tensão aplicado no primário a fim de que se possa realizar medidas precisas da tensão real utilizando-se a tensão de menor magnitude do secundário (CHAPMAN, 2013).

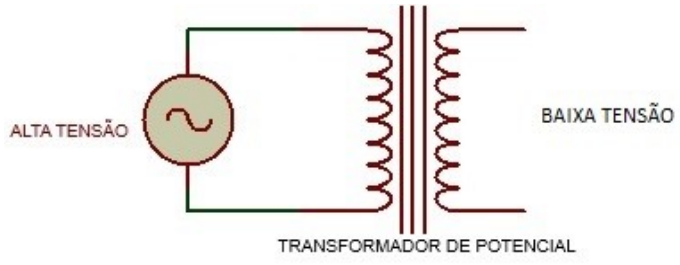

Figura 2 - Transformador de potencial.
Fonte: Autoria própria.

Na Figura 3, a seguir, é observado o circuito divisor de tensão. O princípio pelo qual este circuito pode condicionar o sinal de tensão é que, dada a associação de dois resistores em série no mesmo ramo, a queda de tensão sobre os resistores será proporcional aos valores de suas resistências, formalizando da seguinte maneira:

$$V_{R2} = \frac{R_2}{R_1 + R_2} V_{entrada} \quad (1)$$

desta forma, deve-se projetar os resistores R1 e R2 para que a queda de tensão sobre eles possua os valores desejados (ALEXANDER e SADIKU, 2008).

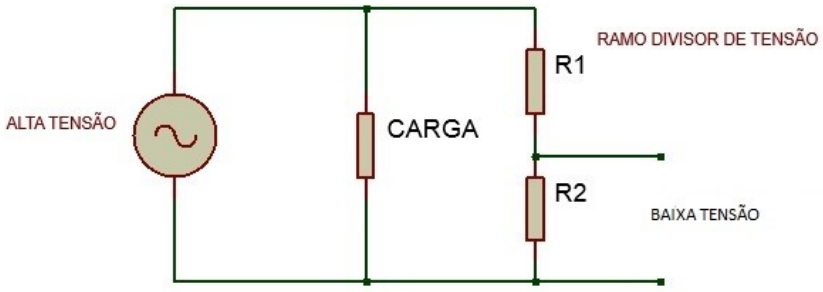

Figura 3 - Circuito divisor de tensão.
Fonte: Autoria própria.

2.1.2 Sensor de Corrente

Da mesma forma como discutido anteriormente, as magnitudes das correntes presentes na alimentação dos motores não são adequadas para serem enviadas diretamente ao sistema de aquisição. Com isso, para que se possa trabalhar com

estes sinais nos sistemas digitais é necessário realizar seu condicionamento. Desta forma, existem alguns sensores que podem ser aplicados, sendo os principais: o resistor shunt, o sensor de efeito Hall e o transformador de corrente (TC) (THOMAZINI e ALBUQUERQUE, 2011). O sensoriamento da corrente com o resistor shunt é baseada na lei de Ohm e pode ser observado na Figura 4.

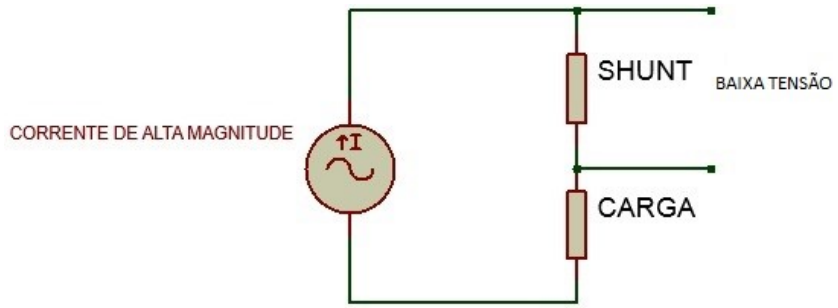

Figura 4 - Condicionamento com resistor shunt.
Fonte: Autoria própria.

Quando uma corrente flui por um resistor de resistência conhecida, pode-se medir a queda de tensão ocasionada e desta forma calcular a corrente através deste resistor (THOMAZINI e ALBUQUERQUE, 2011). Formalizando, tem-se:

$$V_{shunt} = R_{shunt}.I \tag{2}$$

e

$$I = \frac{V_{shunt}}{R_{shunt}} \tag{3}$$

Outra forma de condicionamento da corrente é pelo sensor de efeito Hall, que fornece um valor de tensão proporcional em seus terminais de saída, variando de acordo com o modelo do sensor. Os sensores Hall são construídos com placas semicondutoras percorridas por correntes e, quando as correntes nestas placas não sofrem nenhuma influência externa, a diferença de potencial medida nas extremidades da placa é nula. Porém, quando um campo magnético externo cruza essas placas, a corrente sofre uma variação, e uma diferença de potencial pode ser medida nas extremidades da placa. Aproveitando-se deste fato, este efeito foi utilizado para que medidas pudessem ser realizadas com o emprego deste elemento (THOMAZINI e ALBUQUERQUE, 2011).

Um modo alternativo aos já citados é a utilização do TC, que assim como o TP, é um transformador de instrumentação com as mesmas finalidades, ou seja, a redução da magnitude do sinal do primário no secundário. O TC induz em seu secundário uma amostra de corrente que passa pelo primário, reduzindo a mesma à um nível seguro e que possa ser facilmente manipulada, como pode ser observado na Figura 5.

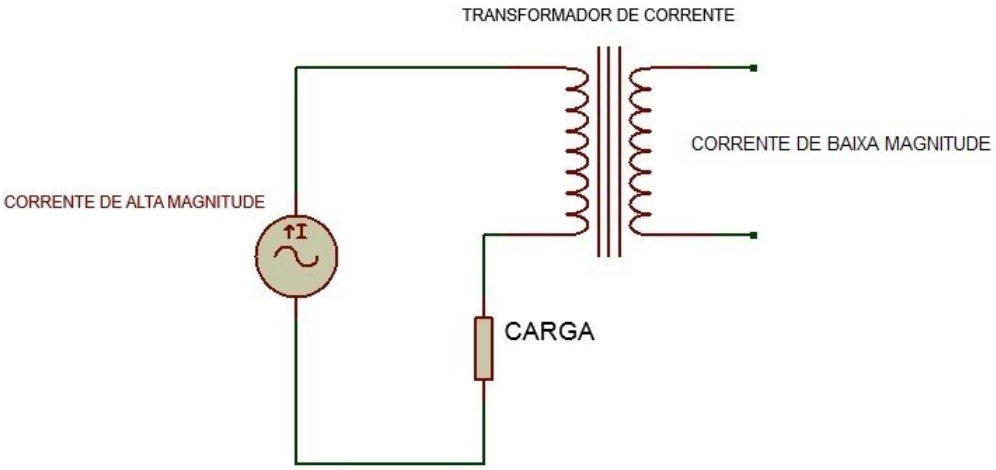

Figura 5 - Transformador de corrente.
Fonte: Autoria própria.

Geralmente, o TC consiste em um anel ferromagnético como primário, e o secundário é enrolado neste anel. É necessário que o secundário do transformador de corrente esteja sempre em curto circuito, pois se o mesmo permanecer aberto tensões extremamente elevadas poderão aparecer e causar danos aos elementos conectados no TC (CHAPMAN, 2013).

2.1.3 Tolerância dos Componentes

No projeto dos circuitos de condicionamento dos sinais, é comum o uso de componentes eletrônicos como resistores, capacitores e indutores. Estes elementos são fundamentais na eletrônica e são denominados como elementos passivos (ALEXANDER e SADIKU, 2008).

Como o projeto visa a medição de grandezas, é importante que esses componentes possuam valores reais o mais próximo possível do valor nominal. Isto deve ser frisado, pois existem tolerâncias dos valores atreladas e cada um destes elementos. Como exemplo, pode-se citar resistores que possuem a tolerância de 0,005%, e outros de 20% do valor nominal do componente (BALBINOT e BRUSAMARELLO, 2011).

Assim, é importante que os elementos empregados no projeto possuam uma tolerância condizente com a aplicação a ser efetuada.

2.2 PROCESSAMENTO DIGITAL DOS SINAIS

Uma vez que os sinais analógicos foram trabalhados e estão aptos a serem tratados por sistemas digitais, pela Figura 1 observa-se que posteriormente deve ocorrer uma conversão dos dados analógicos em digitais.

Nesta seção serão apresentados os conceitos teóricos que permitem que dados analógicos possam ser tratados por sistemas digitais.

2.2.1 Conversão Analógico-Digital

A maioria dos sinais de interesse prático são do tipo analógico, como o sinal de tensão de alimentação de um motor. Para que seja possível processar este sinal na forma digital, necessita-se realizar a conversão analógica para digital. Então, foram criados os chamados conversores analógico para digital, também conhecidos como *Analogic-Digital Converter* (ADC) ou conversores A/D (PROAKIS e MANOLAKIS, 1996).

Três processos podem conceituar a conversão A/D: a amostragem, a quantização e a codificação. O processo de amostragem é a conversão de um sinal contínuo no tempo em um sinal discreto, que é obtido tomando-se amostras do sinal contínuo com uma frequência determinada. A quantização acontece quando cada valor da amostra tomada do sinal contínuo no tempo recebe um valor finito

comparando-se com um valor de referência. Por último, a codificação acontece quando cada valor quantizado é representado por uma sequência binária (PROAKIS e MANOLAKIS, 1996).

Os conversores A/D assim como quaisquer outros instrumentos de medição estão sujeitos a erros intrínsecos de suas construções, sendo que um dos erros mais comuns na conversão analógica para digital é denominado erro de quantização. Este erro ocorre quando a resolução do conversor A/D é muito alta para o valor que se deseja medir. Este erro pode ser suavizado utilizando-se um conversor A/D que tenha o número de bits adequado aos parâmetros de projeto. Como se pode observar abaixo, em (4), a resolução do conversor é inversamente proporcional ao número de bits, sendo assim quanto maior for o número de bits, mais sensível será o conversor (TOCCI, WIDMER e MOSS, 2011):

$$resolução = \frac{V_{REF}}{2^N - 1} \tag{4}$$

em que V_{REF} é a tensão de referência do conversor A/D e N é o número de bits do mesmo.

Outro parâmetro que pode ocasionar erros na medição utilizando a conversão analógica para digital é o tempo de conversão, t_C. Este é o tempo que o conversor demora para realizar a conversão do valor analógico para o valor digital. Para que este erro não influencie as medidas, deve-se consultar o manual do conversor para determinar o tempo mínimo de conversão, pois este varia com a construção dos conversores (TOCCI, WIDMER e MOSS, 2011).

De modo que o sinal analógico seja bem representado, as amostras devem ser tomadas de maneira que seja possível a reconstrução fiel do sinal original. Sendo assim, é necessário conhecer algumas informações prévias do sinal analisado. A condição que assegura que as amostras convertidas representam bem o sinal original, acontece quando a frequência de amostragem F_s, possui o valor de duas vezes ou mais o valor da frequência máxima, $F_{máx}$, contida no sinal, ou seja:

$$F_s > 2F_{máx} \tag{5}$$

assim, F_s, é conhecida como taxa de Nyquist (PROAKIS e MANOLAKIS, 1996).

2.3 TRANSMISSÃO DOS DADOS

Assim que os sinais analógicos forem convertidos em dados digitais, estes estão prontos para serem manipulados e transportados por sistemas digitais, como pode-se observar na Figura 1.

Nesta seção serão apresentados os conceitos teóricos que permitem que dados digitalizados sejam transportados por uma rede via protocolo TCP/IP.

2.3.1 Comunicação em Rede

A comunicação em rede está presente em vários sistemas atualmente, como em bancos, lojas, fábricas e em muitos outros. As redes trazem muitas facilidades, transportando informações em tempo real e otimizando o tempo que era gasto por pessoas realizando estas mesmas tarefas (TORRES, 2001).

Um computador pode realizar a troca de informações com o outro sempre que os dois se comunicam na mesma linguagem, no caso das redes as linguagens são denominadas protocolos (TORRES, 2001). Em geral, os protocolos são organizados como uma pilha de camadas, colocadas uma sobre as outras. O objetivo de cada camada é realizar determinadas tarefas que poderão ser requisitadas pelas camadas superiores ou inferiores. Entre as camadas existe uma interface que define as operações e os serviços que a camada inferior tem a oferecer à camada que se encontra acima dela (TANENBAUM e WETHERALL, 2011).

Quando as redes de comunicação surgiram, cada fabricante utilizava um modo para realizar a comunicação entre seus produtos, ou seja, cada um possuía seu próprio protocolo, sendo assim, os fabricantes eram responsáveis por construírem todos os componentes do sistema de comunicação. Com a expansão das redes isso se tornou um problema, pois em certos casos não existia a compatibilidade entre os protocolos. A fim de resolver este empecilho, a *International Standards Organization* (ISO) criou um modelo de referência para que os fabricantes pudessem segui-lo, padronizando como a comunicação deveria ser realizada. Para este modelo deu-se o nome de *Open Systems Interconnection* (OSI) (TORRES, 2001).

O modelo OSI contém sete camadas de protocolo, e estas são organizadas de acordo com a Figura 6.

Camada	Funcionalidade
7	Aplicação
6	Apresentação
5	Sessão
4	Transporte
3	Rede
2	Enlace de dados (interface de hardware)
1	Conexão física do hardware

Figura 6 - O modelo de referência OSI.
Fonte: Adaptado de Comer (2006, p. 103).

Abaixo será descrito qual a função de cada camada segundo Tanenbaum e Wetherall (2011):

- Camada de aplicação: esta camada possui vários protocolos, sendo estes usados para realizar a interface entre a aplicação requisitada pelo usuário no computador e a transmissão de dados. Assim, quando um usuário abre o browser e envia o nome da página utilizando o protocolo de aplicação apropriado, o servidor então transmite a página para a aplicação.

- Camada de apresentação: nesta camada os dados são traduzidos para um formato que a comunicação entre as máquinas possa acontecer, normalmente faz a conversão dos dados de mais alto nível para mais baixo nível.

- Camada de sessão: esta camada faz o controle de diálogo, ou seja, gerencia quando um computador deve transmitir ou receber. Também permite que usuários em diferentes máquinas estabeleçam sessões de comunicação entre eles. É responsável, ainda, pela sincronização da comunicação entre as máquinas.

- Camada de transporte: basicamente, esta camada deve aceitar os dados da camada de sessão e dividi-los em unidades menores, enviando estes dados divididos para a camada de rede. É de responsabilidade desta camada garantir que os dados divididos sejam entregues corretamente ao receptor.

- Camada de rede: controla a operação da sub-rede, determinando a rota que os pacotes devem seguir para chegar até o destinatário. Se muitos pacotes forem necessários, a camada de rede também é responsável pelo controle de um eventual congestionamento.

- Camada de enlace de dados: esta camada é responsável por trabalhar os pacotes de dados para que sejam transferidos pela rede. Se o serviço for configurado como confiável, o receptor confirmará a recepção correta de cada pacote, enviando de volta um pacote de confirmação. Ainda, esta pode ser configurada para que limite o envio de dados de um transmissor rápido para um receptor lento, regulando desta maneira o tráfego de dados.

- Camada física: esta camada especifica a interconexão física entre os computadores da rede, devendo ser configurada de modo que assegure a transmissão correta dos dados. As configurações mais comuns nesta podem ser citadas como: quais os sinais elétricos que serão usados para representar o nível lógico dos bits, a quantidade de tempo que estes sinais devem durar para validar um nível lógico, se a transmissão pode ocorrer nos dois sentidos, a forma pela qual se dará início e fim na comunicação, quantos pinos o conector de rede terá, quais as funções destes pinos, entre outras questões físicas.

Em todas as camadas o fluxo de dados é bidirecional, ou seja, todas as ações que estas realizam como transmissoras, fazem o contrário quando são receptoras (TORRES, 2001).

2.3.2 Comunicação via Protocolo TCP/IP

Quando o governo americano percebeu a importância e o potencial da tecnologia de redes, financiaram a pesquisa que resultou na rede mundial de computadores. A pesquisa foi realizada pela *Defense Advanced Research Projects Agency* (DARPA), que possibilitou a invenção de um conjunto de padrões de rede que fazem com que os computadores se comuniquem. À tecnologia pesquisada foi dado o nome de TCP/IP *Internet Protocol Suite*, popularmente chamada de TCP/IP (COMER, 2006).

Os requisitos deste novo protocolo eram de que as conexões permanecessem intactas enquanto as máquinas de origem e de destino estivessem em funcionamento, mesmo que algum trecho da rede fosse comprometido. As aplicações visadas eram desde a transferência de arquivos até a transmissão de dados de voz em tempo real, mostrando, desta maneira, quanto o protocolo desenvolvido deveria ser flexível para atender a essas aplicações distintas (TANENBAUM e WETHERALL, 2011).

O TCP/IP assim como o modelo de referência OSI, possui camadas de protocolo, embora sejam diferentes. Na tecnologia TCP/IP existem 4 camadas, como exibe a Figura 7.

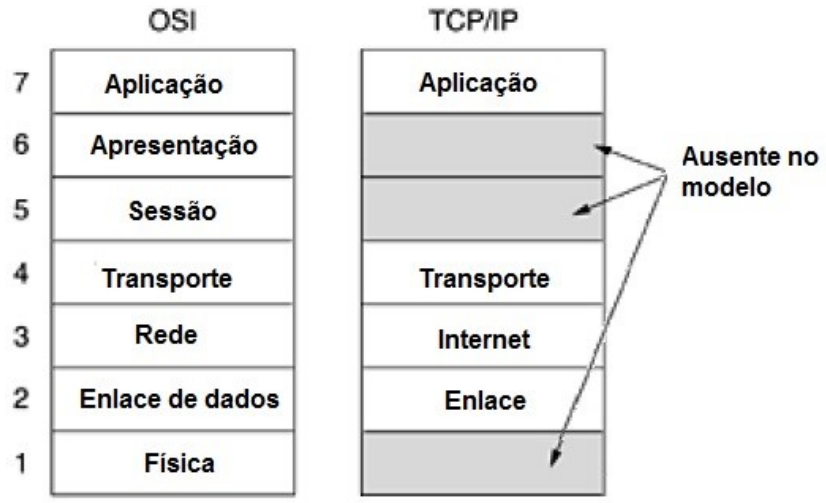

Figura 7 - Modelo de camadas TCP/IP.
Fonte: Adaptado de Tanenbaum e Wetherall (2011, p. 28).

Abaixo será descrito qual a função de cada camada segundo Tanenbaum e Wetherall (2011):

- Camada de aplicação: nesta camada estão os protocolos de nível mais alto. Assim estes devem ser utilizados para cada tipo específico de aplicação, como: protocolo de terminal virtual (TELNET), transferência de arquivos (FTP), correio eletrônico (SMTP), entre outros.

- Camada de transporte: a finalidade desta camada é garantir a comunicação entre o transmissor e o receptor. Assim, dois protocolos podem ser utilizados nesta camada. O primeiro é o *Transmission Control Protocol* (TCP), que é um protocolo que proverá uma comunicação confiável, pois controla o fluxo e o recebimento dos pacotes enviados. O segundo protocolo é o *User Datagram Protocol* (UDP), que é um protocolo dito não confiável, uma vez que não realiza o mesmo controle de recebimento dos pacotes enviados que o protocolo TCP faz. Este protocolo é usado quando se deseja uma transmissão mais rápida e sem a necessidade de confirmação de recebimento de todos os pacotes.

- Camada de internet: esta camada possui semelhanças com a camada de rede do modelo OSI, entretanto existem algumas diferenças. Esta é responsável por fazer com que os dados injetados nas redes cheguem corretamente aos seus destinos. Define um formato de pacote oficial e um protocolo chamado *Internet Protocol* (IP), mais um protocolo que o acompanha, chamado *Internet Control Message Protocol* (ICMP). Suas ações são de entregar os pacotes IP onde foram requisitados.

- Camada de enlace: descreve como linhas seriais e Ethernet devem cumprir os requisitos desta camada de interconexão com serviços não orientados a conexões. Desta forma, esta não é uma camada propriamente dita, mas uma interface entre hosts e os enlaces de transmissão.

O modelo descrito anteriormente ficou conhecido por TCP/IP, pois os protocolos TCP e IP são os mais comuns nas redes, sendo implementados em camadas específicas do protocolo, entretanto dependendo do tipo e característica das aplicações outros protocolos mais apropriados devem ser aplicados em substituição do TCP/IP (TORRES, 2001).

3 MÉTODO DE PESQUISA

Nesta seção são apresentados os materiais, metodologia e sistemas utilizados e desenvolvidos para elaboração do trabalho.

3.1 SENSORES

Sendo um dos objetivos específicos do trabalho o emprego dos componentes que melhor se adaptem aos sistemas com relação à aplicação e aos custos, foram levantadas as informações dos sensores que poderiam ser aplicados.

3.1.1 Sensor de Tensão

É necessário que o sensor possibilite a amostragem do sinal de tensão original, para isso poderia ser empregado apenas um circuito divisor de tensão. Porém, neste caso, a eficiência energética do sistema seria prejudicada devido às perdas decorrentes do circuito resistivo. Observou-se que a melhor escolha seria a utilização de um transformador de baixa potência como um TP, para reduzir a tensão aplicada à máquina e possibilitar a amostragem do sinal após seu condicionamento, visto que uma amostra fiel do sinal de tensão aplicado ao primário do transformador é induzida em seu secundário. Além das características já citadas, a escolha do TP como sensor proporciona um elemento de simples funcionamento, baixo custo e fácil substituição no caso de avarias.

Dois modelos diferentes de transformadores foram aplicados ao sistema de sensoriamento dos sinais de tensão, isto porque primeiramente foi adquirido um modelo para montagem de um protótipo para realização de ensaios, buscando verificar a aplicabilidade ao trabalho. Após a confirmação do funcionamento do sistema, buscou-se adquirir o mesmo modelo de transformador utilizado no protótipo, porém não foi possível encontrar.

Um dos modelos utilizados está apresentado na Figura 8, sendo seus dados elétricos apresentados no Quadro 1. Este transformador foi utilizado para a realização do protótipo e possui uma unidade aplicada ao sistema final.

Figura 8 - Modelo de transformador utilizado no protótipo e no sistema final como sensor de tensão.
Fonte: Eletrodex.

Grandeza	*Valor*	*Unidade*
Tensão Primário	127/220	V
Tensão Secundária	3 + 0	V
Potência	0,9	VA

Quadro 1 – Dados elétricos do transformador utilizado no protótipo e no sistema final como sensor de tensão.
Fonte: Fabricante.

Outro modelo aplicado é fisicamente construído como o transformador anterior, apresentado na Figura 8, sendo seus dados elétricos apresentados no Quadro 2. Este transformador foi utilizado na composição do sistema final e possui duas unidades aplicadas ao trabalho.

Grandeza	*Valor*	*Unidade*
Tensão Primário	127/220	V
Tensão Secundária	3 + 3	V
Potência	1,8	VA

Quadro 2 – Dados elétricos dos transformadores utilizados no sistema final como sensor de tensão.
Fonte: Fabricante.

3.1.2 Sensor de Corrente

Levando em consideração as mesmas premissas aplicadas na subseção anterior, para escolha do sensor de tensão, optou-se pela utilização de TC para realizar o sensoriamento da corrente elétrica. O transformador de corrente proporciona uma amostra reduzida fiel do sinal de corrente aplicado ao primário em seu secundário, sendo suas características semelhantes às dos transformadores utilizados para o sensoriamento de tensão, tendo um simples funcionamento, fácil substituição no caso de avarias e baixo custo. O transformador de corrente aplicado ao trabalho está apresentado na Figura 9 e suas características podem ser observadas no Quadro 3.

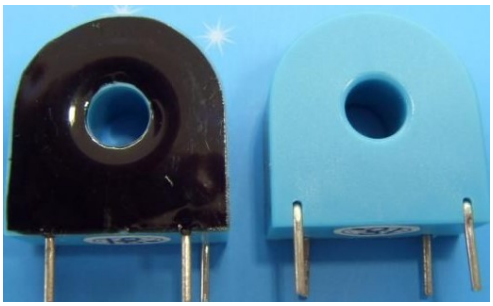

Figura 9 – Transformador de corrente utilizado no trabalho.
Fonte: Fabricante.

Grandeza	***Valor***	***Unidade***
Corrente Primária	20	A
Corrente Secundária	20	mA

Quadro 3 – Dados elétricos do transformador de corrente utilizado no trabalho.
Fonte: Fabricante.

Após a definição dos sensores foi necessário verificar o modo pelo qual os sinais de tensão e corrente poderiam ser amostrados, para isso foi necessário o uso de um microprocessador.

3.2 MICROPROCESSADOR

Para realizar a pesquisa do microprocessador que seria empregado no projeto, foram levantadas as características mínimas que o mesmo deveria possuir para que pudesse atender aos requisitos do trabalho. Estas características estão listadas a seguir:

- Transmissão de dados via protocolo TCP/IP;
- Capacidade de amostragem de sinais acima de 1200 Hz, visando que a amostragem dos sinais permita a aquisição da 10ª ordem harmônica, considerando a frequência fundamental de 60 Hz;
- Mínimo de seis entradas analógicas para conversão digital, visando as máquinas trifásicas;

Após o levantamento das características, inicialmente foi considerado a montagem de um sistema de processamento de sinais, que seria composto de um microprocessador e todos os outros elementos necessários para o funcionamento do mesmo. Entretanto, observou-se ser mais vantajoso fazer o uso de uma placa de processamento já pronta, visto que a montagem desta não contempla os objetivos propostos.

A placa adquirida e utilizada no trabalho foi o modelo Cerebot Mx7 do fabricante Digilent, que pode ser observada na Figura 10. As principais características desta placa relacionadas ao trabalho estão listadas a seguir:

- Microprocessador de 32 bits (PIC32MX795F512L);
- Clock configurável de até 80 MHz;
- 16 canais de conversão analógico-digital de 10 bits;
- Controlador ethernet com entrada RJ45;
- Alimentação pode ser realizada de variadas formas;

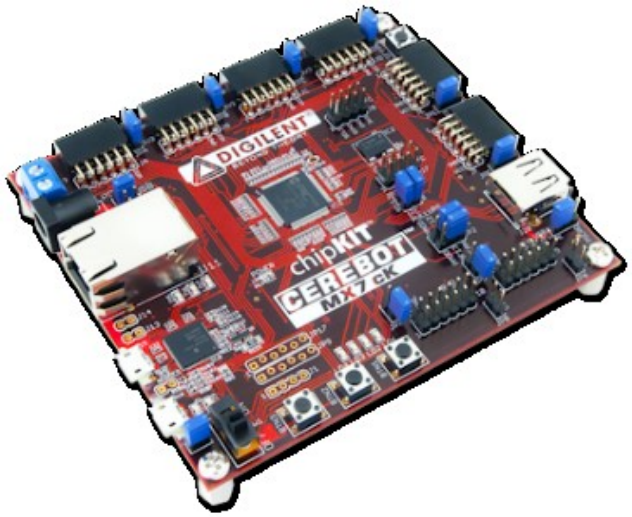

Figura 10 – Placa de processamento Cerebot Mx7 empregada no trabalho.
Fonte: Digilent

Após a definição do microprocessador foi possível projetar os condicionadores de sinais necessários para amostrar os sinais disponibilizados pelos sensores, uma vez que os requisitos elétricos da placa de processamento já são conhecidos.

3.3 CONDICIONAMENTO DE SINAIS

O condicionamento de sinais é necessário, pois geralmente os sinais provenientes dos sensores não podem ser aplicados diretamente sobre os pinos dos microprocessadores, o que pode causar mal funcionamento e danos a estes elementos. No caso do microprocessador PIC32MX795F512L, que compõe a placa Cerebot Mx7 utilizada, a tensão máxima que pode ser aplicada a qualquer pino de entrada ou saída é 3,6 V. Como o módulo do conversor A/D será utilizado para realizar a amostragem dos sinais de tensão e corrente, deve-se condicionar estes sinais para que estejam dentro dos limites elétricos e de operação, fazendo com que o sistema funcione de forma correta e não cause avariais.

O módulo do conversor A/D disponibiliza duas entradas para a tensão de referência, sendo esta tensão utilizada no processo de quantização, que ocorre na conversão dos sinais analógicos para digitais. No trabalho foram utilizadas as referências internas de tensão, desta forma, foi empregado como limite superior de

referência a tensão de alimentação do microprocessador, que é de 3,3 V, e como limite inferior foi utilizado o pino de referência do microprocessador, parametrizado em 0 V. Como já citado, o conversor A/D utilizado no trabalho possui 10 bits, sendo assim, sua saída é dada no intervalo de valores de 0 a 1023. Com os valores de referência utilizados cada bit corresponderá 3,223 mV, de acordo com (4). Assim, para um valor de tensão de 1,65 V, que seria exatamente metade do valor de referência, o conversor A/D retornaria um valor lido de 512.

Após verificado o intervalo de operação do conversor A/D, foi necessário verificar quais seriam as alterações que deveriam ser aplicadas nos sinais originados pelos sensores para que pudessem ser amostrados. Nas próximas subseções serão apresentados os cálculos e projetos dos condicionadores de sinais para os sinais de tensão e corrente necessários.

3.3.1 Projeto dos Protótipos de Aquisição

Como mencionado anteriormente, deve-se condicionar o sinal para que os valores de tensão a serem amostrados pelo conversor A/D fiquem dentro do intervalo de 0 a 3,3 V.

Para o caso dos sinais de tensão, os TPs empregados para realizar o sensoriamento, possuem saída de 3 V quando a tensão aplicada ao primário é de 220 V. Desta forma, amplitude da tensão no secundário é dada por:

$$V_{2M} = \sqrt{2}\, V_2 \tag{6}$$

onde,

V_{2M}, é o valor de pico do secundário do TP;

V_2, é o valor RMS do secundário do TP;

Assim, por (6), o valor máximo que o sinal do TP terá no secundário em condições normais será de 4,243 V. Portanto, o sinal do secundário do transformador deve sofrer uma redução de sua amplitude e um *offset* para que não aplique valores

negativos nos pinos do conversor A/D. A Figura 11 exemplifica como o sinal original deve ser alterado para que o condicionamento seja realizado.

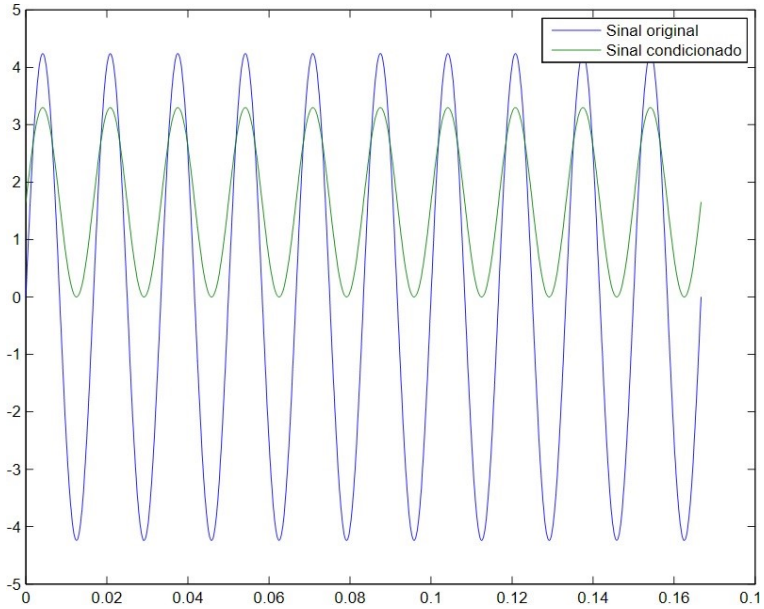

Figura 11 – Exemplo gráfico do condicionamento de sinal necessário.
Fonte: Autoria própria.

O *offset* a ser aplicado deve estar na metade do intervalo de referência do conversor, que é dado por:

$$Offset = \frac{V_{ref+} - V_{ref-}}{2} \tag{7}$$

onde:

V_{ref+}, é o valor de referência superior escolhido para o conversor;

V_{ref-}, é o valor de referência inferior escolhido para o conversor;

Pelos valores já citados e com base em (7), o valor necessário para o *offset* é de, 1,65 V. Para que a saída do sensor de tensão somada ao *offset* esteja dentro do intervalo de 0 a 3,3 V é preciso que a amplitude da tensão no secundário seja de 1,65 V. Assim, para realizar esta redução foi utilizado um *trimmer*, como o apresentado na Figura 12. A implementação do *trimmer* ao invés de um circuito resistivo utilizando resistores se mostra mais viável, devido às variações significativas de valores que os secundários dos TPs e TCs apresentam. Com o emprego dos *trimmers* pode-se adaptar o sistema de condicionamento a estas variações, realizando ensaios simples em laboratório, utilizando um osciloscópio e calibrando o valor do *trimmer*, principalmente no caso da substituição do equipamento sensor.

Então, conectando o secundário do TP ao *trimmer* e variando seu valor, é possível ajustar o sinal de modo que o valor máximo seja de 1,65 V.

Figura 12 – Modelo do *Trimmer* utilizado no trabalho.
Fonte: Eletrodex

A mesma metodologia descrita até o momento foi utilizada para realizar o condicionamento dos sinais de corrente. A grande mudança ocorre pelo fato de o TC realizar uma redução na corrente, sendo que esta corrente reduzida deve ser convertida em uma tensão proporcional. Para isso, foi adicionado um resistor entre os terminais do secundário de cada TC, fazendo com que ocorra uma diferença de potencial sobre o mesmo, conforme (2), que será utilizada para amostrar a corrente. Desta forma, assim como o *trimmer* foi utilizado para a redução do sinal de tensão, para o sinal de corrente o valor do mesmo é ajustado para que a diferença de potencial no secundário do TC tenha uma amplitude de 1,65 V, que somada ao *offset* já mencionado, fará com que o sinal de corrente seja condicionado a estar dentro do intervalo de 0 a 3,3 V.

Definido a forma pela qual ocorreria o condicionamento dos sinais, foi projetado o circuito utilizado, que pode ser observado na Figura 13.

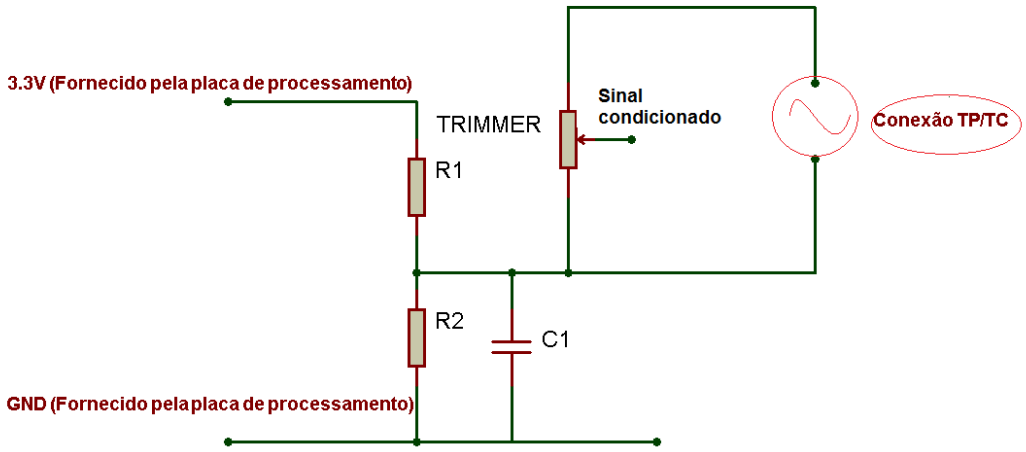

Figura 13 – Circuito de condicionamento desenvolvido.
Fonte: Autoria própria.

Este circuito pode ser dividido em dois blocos, o primeiro é composto pelos resistores R1 e R2, sendo responsável pelo *offset* do sinal. O valor de tensão de 3,3 V é fornecido em uma das saídas da placa de processamento e, este é reduzido com o emprego deste divisor resistivo, sendo que os valores de R1 e R2 são iguais. Assim, de acordo com a (1), o valor da divisão de tensão será de 1,65 V. Importante ressaltar o uso de resistores com precisão da ordem de 1% para realizar o *offset* necessário, uma vez que o sistema deve ter uma alta exatidão para que as medidas correspondam aos valores mais próximos possíveis dos reais.

O segundo bloco é composto pelo *trimmer* e por um capacitor. O *trimmer* é responsável por controlar a amplitude do sinal proveniente dos sensores, já o capacitor é utilizado para atenuar os ruídos de alta frequência presentes nos sinais. A junção destes dois blocos fornece as alterações necessárias aos sinais para que possam ser amostrados pela placa de processamento.

Os dispositivos utilizados para realizar o condicionamento dos sinais de tensão e de corrente estão apresentados no Quadro 4, sendo estes os necessários para realizar o condicionamento de apenas um sinal.

Protótipo do condicionamento de sinais	
Material	***Quantidade***
Capacitor	1
Resistor	2
Trimmer	1

Quadro 4 – Materiais utilizados no condicionamento de sinais.
Fonte: Autoria própria.

Após o projeto dos protótipos dos sistemas de sensoriamento e condicionamento de sinais, foi necessário verificar a funcionalidade dos mesmos realizando ensaios em laboratório.

3.3.1 Ensaio dos Protótipos de Aquisição e Condicionamento de Sinais

Para validação dos protótipos projetados foram realizados ensaios com os mesmos, sendo verificados os sensores de tensão, de corrente e o circuito condicionador de sinal.

Para verificar a aplicabilidade do sensor de tensão, foi necessário analisar como o mesmo responde aos diferentes valores de alimentação que uma máquina elétrica pode receber. Assim, utilizou-se uma fonte variável de tensão alternada para variar a tensão na entrada do TP, utilizando um osciloscópio e um multímetro para realizar as medições.

O Quadro 5, abaixo, mostra os valores que foram aplicados no primário do sensor, V1, e os valores que foram observados no secundário, V2. Ainda, calculou-se a relação de transformação, a, dada por (8), para cada par de valores medidos.

$$a = \frac{V1}{V2} \tag{8}$$

V1 [V]	V2 [V]	a
28,0	0,33	83,8
54,7	0,97	56,3
82,4	1,49	55,3
109,0	1,96	55,6
135,0	2,51	53,8
187,0	3,43	54,5
206,0	4,35	47,4

Quadro 5 - Valores observados no ensaio com o TP.
Fonte: Autoria própria.

Pode-se observar que o sensor utilizado não corresponde com uma variação totalmente linear quando a tensão no primário é variada. Isto é prejudicial para o

sistema, visto que para diferentes tensões de alimentação das máquinas o sistema deve ser calibrado para compensar a não-linearidade do sensor. Ainda, foi observado que para tensões muito baixas, se comparadas ao valor nominal do transformador utilizado, o sinal de tensão no secundário sofre grandes alterações na sua forma de onda, conforme ilustra a Figura 14, observada na realização do ensaio.

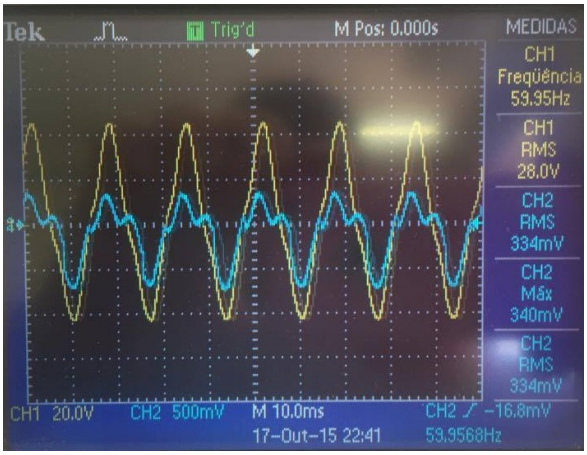

Figura 14 - Forma de onda do secundário distorcida para tensões muito baixas aplicadas ao primário.
Fonte: Autoria própria.

Realizado o ensaio do sensor de tensão, foi necessário verificar a aplicabilidade do sensor de corrente, analisando como o mesmo responde aos diferentes valores de corrente que sua faixa de operação permite.

A fim de possibilitar o ensaio do sensor de corrente, foi desenvolvida uma placa para acomodar o TC. O circuito projetado para ensaio do TC pode ser verificado na Figura 15, o layout da placa na Figura 16, e o projeto 3D na Figura 17.

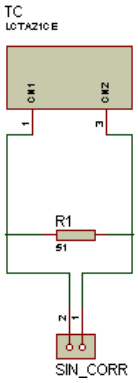

Figura 15 - Projeto da placa utilizada para ensaio do TC.
Fonte: Autoria própria.

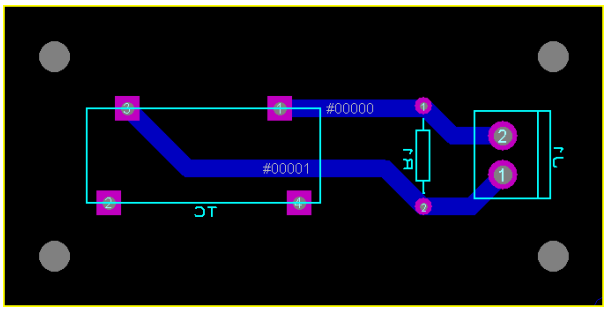

Figura 16 - Layout do circuito utilizado para ensaio do TC.
Fonte: Autoria própria.

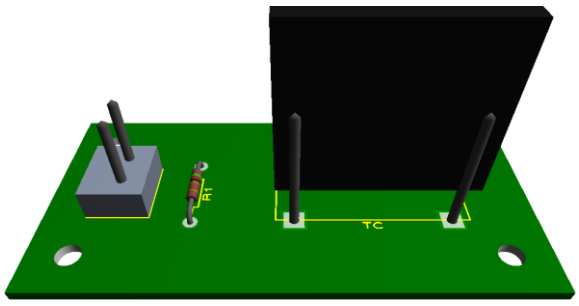

Figura 17 - Projeto em 3D da placa utilizada para ensaio do TC.
Fonte: Autoria própria.

Para avaliação, foi conectado um resistor de 51 Ω em seu secundário, convertendo o sinal de corrente em tensão. Então, utilizou-se três lâmpadas de 220V e 200W conectadas em paralelo para simulação de cargas, por meio de um amperímetro e um osciloscópio as medidas foram tomadas. A Figura 18 ilustra o esquema de ligação adotado para a realização do ensaio do TC.

Para realizar a conversão do valor de tensão no secundário do TC, para o valor proporcional de corrente no primário, deve-se proceder como em (9):

$$I_1 = \frac{V_{TC}}{R_{TC}}.1000 \quad (9)$$

onde:

I_1, é o valor de corrente no primário do TC;

V_{TC}, é o valor da tensão gerada nos terminais da placa do TC;

R_{TC}, é o valor do resistor conectado no secundário do TC, neste caso 51 Ω;

e a constante 1000, é a relação de transformação do TC;

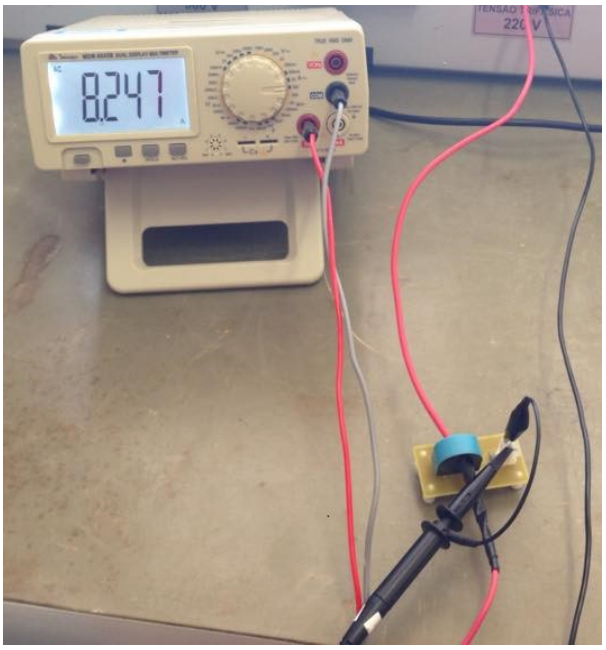

Figura 18 - Arranjo experimental para ensaio do TC.
Fonte: Autoria própria.

Primeiramente, foi ligado uma lâmpada em 127V e medido a corrente da mesma com um amperímetro, também foi medida a tensão que o protótipo do sensor de corrente retornava quando o mesmo era excitado. O mesmo procedimento foi repetido utilizando duas e três lâmpadas. Então, a tensão de alimentação das lâmpadas foi alterada para 220V e o procedimento mencionado foi repetido. No Quadro 6 se observa os valores obtidos com o ensaio realizado.

Lâmpadas	Alimentação [V]	Corrente Real [A]	Tensão TC [V]	Corrente Primário TC [A]	Erro [%]
1	127	0,646	0,038	0,745	15,3%
2	127	1,301	0,072	1,411	8,5%
3	127	1,965	0,100	1,961	-0,2%
1	220	0,850	0,050	0,980	15,3%
2	220	1,766	0,100	1,960	11,0%
3	220	2,655	0,150	2,941	10,8%

Quadro 6 - Valores observados no ensaio com o TC.
Fonte: Autoria própria.

Com o ensaio, foi observado um erro médio de, aproximadamente, 10% em relação as medidas realizadas com o amperímetro. Também foi possível perceber que o TC não sofre não-linearidades como observadas no ensaio do TP.

Por fim, realizou-se os ensaios com o condicionador de sinais projetado, montando o circuito de condicionamento em um *protoboard*. Buscou-se verificar se o mesmo realizaria o condicionamento necessário e, se poderia ser aplicado em altas frequências. Para isso, foi utilizado um gerador de sinais simulando os sinais que estariam sendo transmitidos pelos sensores. Como características, o sinal utilizado tinha forma de onda senoidal com amplitude de 1,65 V. A frequência do mesmo foi variada com alguns valores, como, 60, 120, 180, 240 e 600 Hz.

Além da verificação do condicionamento, buscou-se observar como o uso do capacitor para atenuação dos ruídos de alta frequência é importante. Para isso, este foi eliminado do circuito para verificação em certos momentos.

Na Figura 19 se pode observar a resposta do circuito condicionador quando excitado com um sinal de 60 Hz. Já na Figura 20, é mostrado a resposta do circuito quando é aplicado um sinal de 120 Hz.

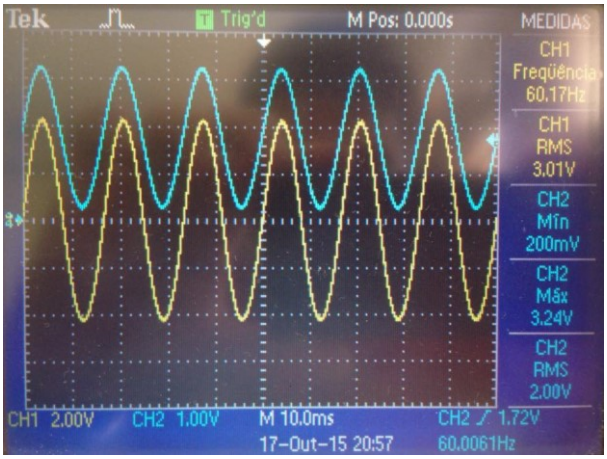

Figura 19 - Circuito condicionador excitado com sinal de 60 Hz.
Fonte: Autoria própria.

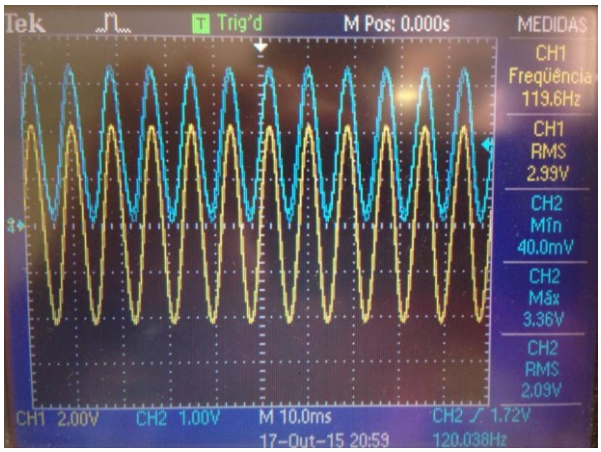

Figura 20 - Circuito condicionador excitado com sinal de 120 Hz.
Fonte: Autoria própria.

Em ambos os casos o condicionamento é realizado de maneira satisfatória, fazendo com que o sinal permaneça dentro da faixa de operação do conversor A/D, de 0 a 3,3 V. Na Figura 21 se pode observar a resposta do circuito condicionador para um sinal com frequência de 240 Hz. Já na Figura 22 foi utilizado o mesmo sinal, porém neste caso o capacitor foi removido do circuito de condicionamento.

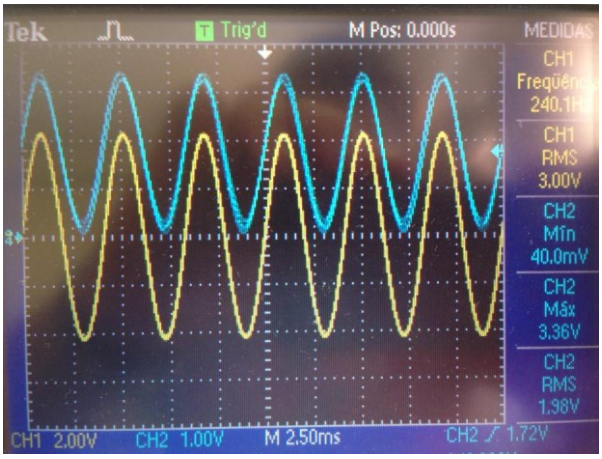

Figura 21 - Resposta do circuito condicionador excitado com 240 Hz, com capacitor.
Fonte: Autoria própria.

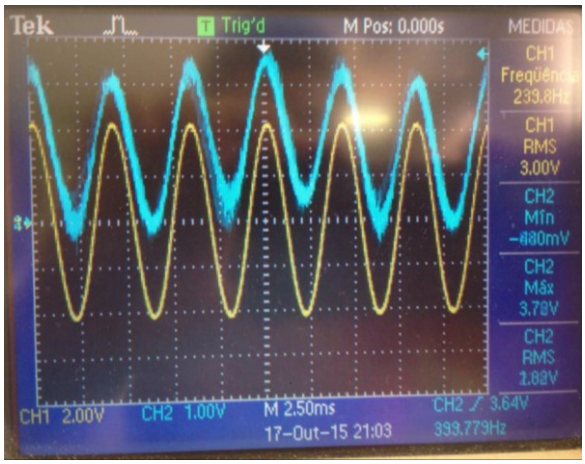

Figura 22 - Resposta do circuito condicionador excitado com 240 Hz, sem capacitor.
Fonte: Autoria própria.

Pode-se observar que o sinal é condicionado de forma satisfatória, entretanto quando o capacitor foi removido do circuito o nível de ruído no sinal condicionado aumentou consideravelmente, evidenciando a importância do uso deste elemento. O mesmo comportamento foi observado nas imagens da Figura 23 e da Figura 24, onde foi utilizado um sinal de 600 Hz para excitação do circuito.

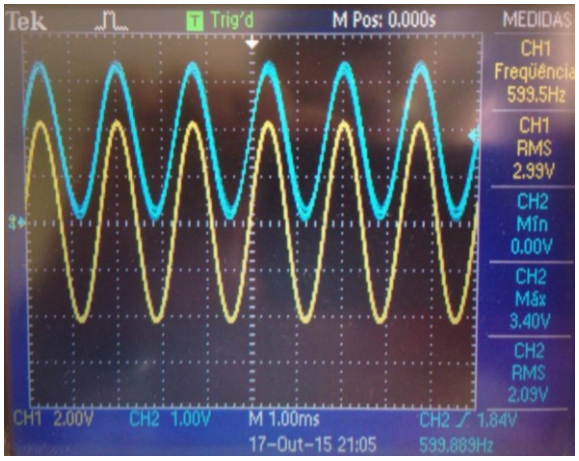

Figura 23 - Resposta do circuito condicionador excitado com 600 Hz, com capacitor.
Fonte: Autoria própria.

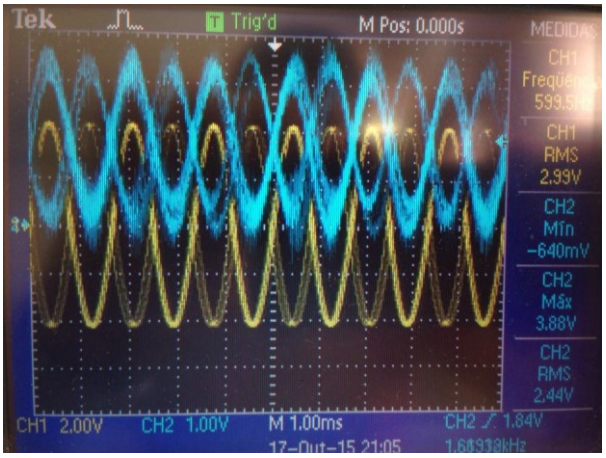

Figura 24 - Resposta do circuito condicionador excitado com 600 Hz, sem capacitor.
Fonte: Autoria própria.

Com o ensaio dos sensores e condicionador de sinais, foi possível verificar que os mesmos podem ser aplicados ao trabalho. Não foram observadas características negativas no circuito condicionador de sinal. No TC foi observado um erro relativamente alto se comparado ao multímetro, da ordem de 10%, que será aceito neste trabalho. Já no caso do TP, concluiu-se que será necessário definir valores padrão para utilização do mesmo, e que devem ser previstos fatores de compensação, devido a não-linearidade do transformador empregado.

3.3.2 Sistema de Aquisição e Condicionamento de Sinais Final

Após a validação por meio dos ensaios mencionados anteriormente, foi necessário desenvolver o sistema final de aquisição. Para isso, levou-se em consideração as características das máquinas trifásicas, que são alimentadas por três fases. Foi então projetado um sistema capaz de realizar a aquisição de seis sinais analógicos, sendo três de tensão e três de corrente. Os materiais utilizados podem ser observados no Quadro 7, sendo o circuito empregado exibido na

Figura 25 e o layout projetado na Figura 26. Para projeto e montagem do sistema final, foi utilizado o *software* Proteus.

Condicionamento de Sinais			
Material	***Quantidade***	***Valor***	***Unidade***
Barra de pinos	1	6 x 2	-
Capacitor	6	100	ηF
Resistor	12	200	kΩ
Transformador de corrente	3	-	-
Transformador de Potencial	3	-	-
Trimmer	3	10	kΩ
Trimmer	3	100	Ω

Quadro 7 – Materiais utilizados no sistema de condicionamento final.
Fonte: Autoria Própria.

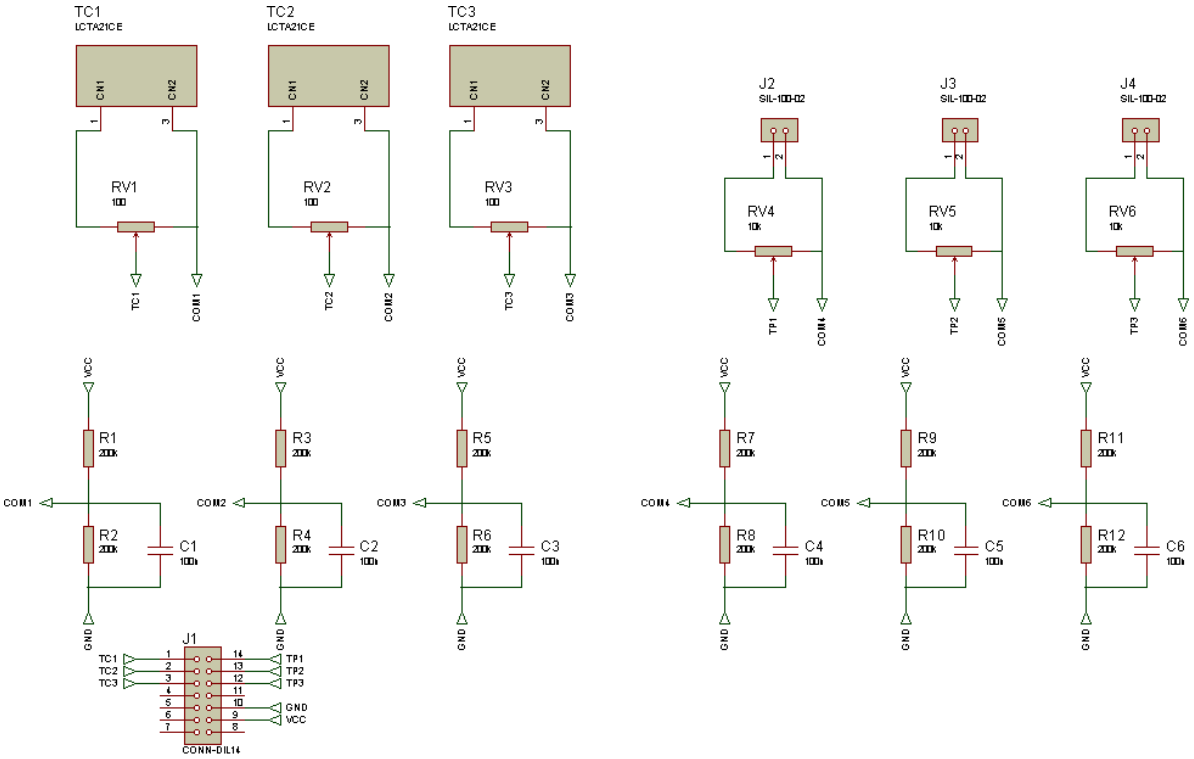

Figura 25 – Projeto do circuito final de condicionamento de sinais.
Fonte: Autoria própria.

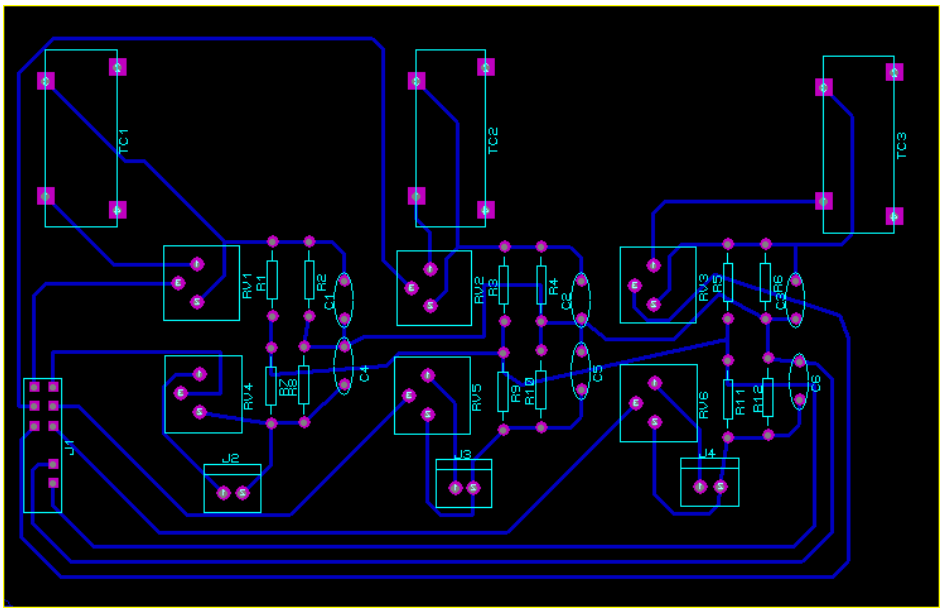

Figura 26 - Layout da placa de aquisição e condicionamento de sinais.
Fonte: Autoria própria.

O projeto 3D da placa de aquisição e condicionamento de sinais pode ser observado na Figura 27.

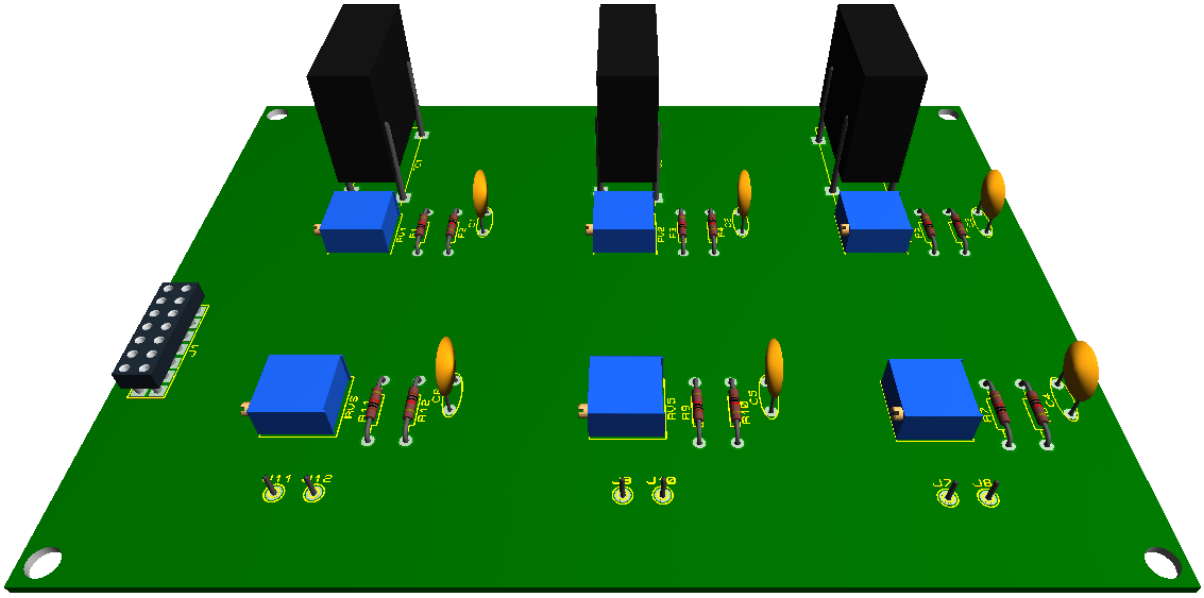

Figura 27 – Projeto final em 3D da placa de condicionamento de sinais.
Fonte: Autoria própria.

Os limites operativos da placa de aquisição e condicionamentos de sinais podem ser verificados no Quadro 8.

Limites operativos da placa de aquisição		
Grandeza	***Valor***	***Unidade***
Tensão	220	V
Corrente	20	A

Quadro 8 – Limites operativos da placa de aquisição
Fonte: Autoria própria.

Após o projeto da placa de aquisição e condicionamento, um protótipo foi confeccionado para uma verificação mais assertiva de seu funcionamento e *design*, este pode ser observado na Figura 28.

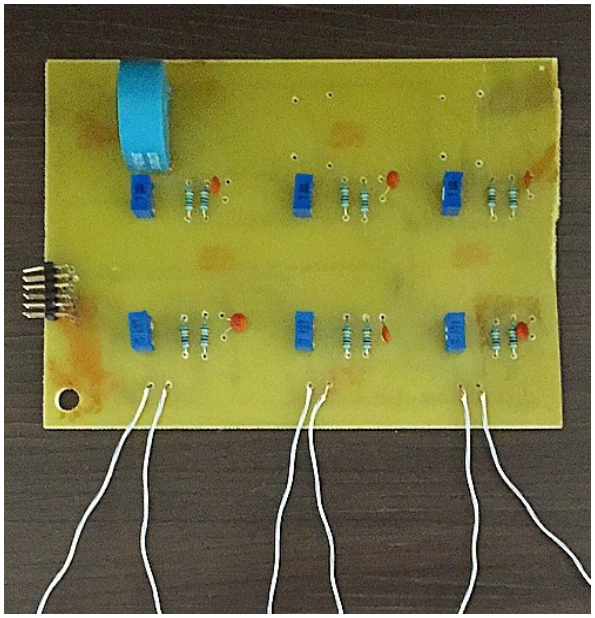

Figura 28 - Protótipo confeccionado da placa de condicionamento de sinais.
Fonte: Autoria própria.

Depois de efetuar as correções de algumas imperfeições do protótipo, a placa de aquisição e condicionamento final foi elaborada, Figura 29, sendo esta aplicada ao sistema final de monitoramento de máquinas elétricas.

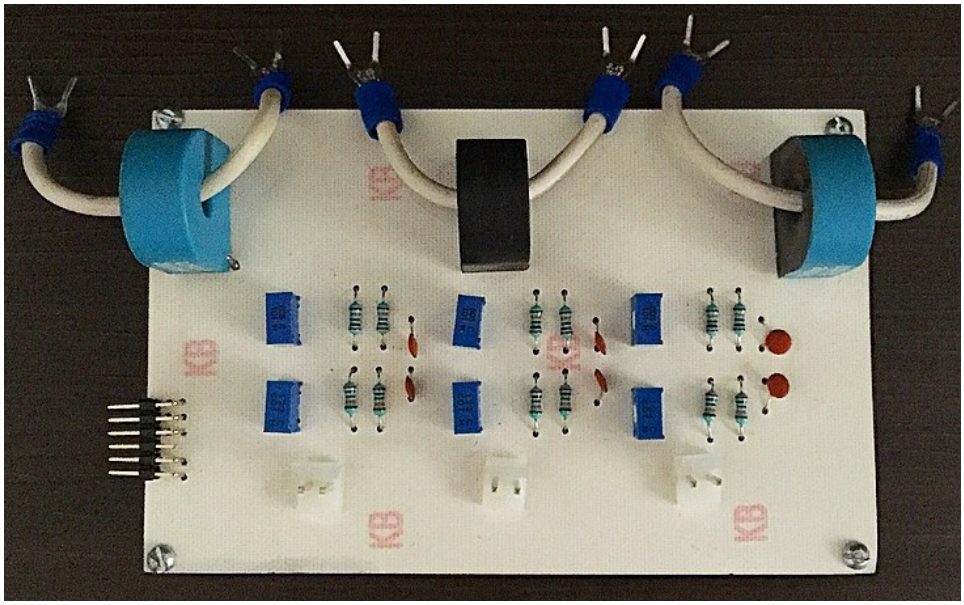

Figura 29 - Placa de condicionamento aplicada ao sistema final.
Fonte: Autoria própria.

Com a realização dos testes dos protótipos e da montagem da placa final de aquisição e condicionamento de sinais, foi necessário desenvolver a rotina

implementada no microprocessador para realizar a amostragem, e fazer com que os dados oriundos desta possam ser enviados pela rede via protocolo TCP/IP. Este assunto será discutido na próxima seção.

3.4 AMOSTRAGEM DOS SINAIS E TRANSMISSÃO DOS DADOS

3.4.1 Ambiente de Desenvolvimento

A placa de processamento Cerebot Mx7 é uma placa de desenvolvimento que pode ser programada com o software MPLAB da Microchip e também com o ambiente de desenvolvimento MPIDE disponibilizado pela chipKIT em conjunto com a fabricante da placa, Digilent. Neste trabalho optou-se por utilizar o software MPIDE para realizar o desenvolvimento e programação da rotina implementada na placa de processamento.

O software MPIDE disponibiliza funções que realizam tarefas complexas e que são implementadas de forma extremamente simples se comparadas com o uso do MPLAB. Para que a rotina seja desenvolvida em seu ambiente de programação é necessário programar um arquivo disponibilizado pelo fabricante, denominado *Bootloader,* este arquivo faz com que a placa seja previamente configurada para implementação das rotinas disponibilizadas pelo software MPIDE, minimizando a complexidade do trabalho executado pelo usuário. Outra vantagem importante para é o fato de o MPIDE ser totalmente gratuito.

3.4.2 Parâmetros de Amostragem

As premissas utilizadas para desenvolvimento da rotina implementada no microprocessador estão listadas abaixo:

- Frequência de amostragem alta suficiente para análise do sinal até 10ª ordem harmônica;

- Amostragem de no mínimo 10 períodos;
- A amostragem deve permitir a manipulação matemática dos dados, a fim de se extrair informações como: espectro harmônico, valor RMS, distorção harmônica, entre outros;
- Para todas as premissas acimas considera-se a frequência fundamental de 60 Hz;
- Amostragem considerando a máquina em regime permanente;
- Amostragem considerando a máquina em regime transitório;

Para realizar a amostragem de um sinal, deve-se considerar a frequência mínima de amostragem como sendo o dobro da frequência máxima de interesse, segundo a regra de Nyquist, conforme (5). Desta forma, deve-se amostrar um sinal de 60 Hz a no mínimo 1200 Hz para que seja possível realizar a análise do sinal considerando-se a 10ª harmônica. Para que haja 10 períodos de um sinal de 60 Hz, o tempo mínimo de amostragem deve ser de 166,67 ms.

Tomando como referência o software Matlab para realizar a manipulação matemática dos dados amostrados, cada período deve conter um número de amostras que seja uma potência de dois, desta forma, a função *Fast Fourier Transform* (FFT) pode ser aplicada de maneira correta para análise dos sinais (Matlab Documentation, 2015). O número de amostras, n, pode ser calculado por (9):

$$n = \frac{T_{min}}{T_{amostragem}} \tag{10}$$

onde:

T_{min}, é o tempo mínimo que se deseja amostrar o sinal;

$T_{amostragem}$, é o tempo que levado para coletar uma amostra do sinal;

Considerando o período e frequência de amostragem já citados, haveriam 200 amostras do sinal. Como este número não é uma potência de dois, a frequência e período de amostragem não satisfazem as premissas supracitadas. Assim, estes parâmetros foram determinados com as seguintes considerações:

- Amostragem de no mínimo 16 períodos do sinal, pois é a próxima potência de dois depois de 10 períodos (quantidade mínima considerada nas premissas);

- 2048 amostras por sinal é a quantidade máxima em potência de dois que as funções utilizadas no código conseguem manipular;

A partir destas considerações, o número de amostras por período é 128, dado por (11):

$$np = \frac{nm}{nT} \tag{11}$$

onde:

np, número de amostras por período;

nm, número máximo de amostras por sinal;

nT, número de períodos;

A frequência de amostragem pode ser calculada por (12):

$$F_{amostragem} = \frac{T_{fundamental}}{np} \tag{12}$$

onde,

$F_{amostragem}$, é a frequência de amostragem;

$T_{fundamental}$, é o período da frequência fundamental do sinal;

np, é o número de amostras por período;

Assim, considerando as grandezas já mencionadas, a frequência de amostragem resultante é 7680 Hz, garantindo que todas as premissas referentes aos parâmetros de amostragem sejam respeitadas. Um resumo das grandezas envolvidas na amostragem dos sinais pode ser observado no Quadro 9.

Parâmetros de Amostragem		
Grandeza	***Valor***	***Unidade***
Frequência Fundamental	60	Hz
Período fundamental	16,7	ms
Número de Períodos	16	-
Tempo de aquisição	266,7	ms
Número máximo de amostras	2048	-
Amostras por período	128	-
Frequência de Amostragem	7680	Hz

Quadro 9 – Grandezas envolvidas na amostragem dos sinais do trabalho.
Fonte: Autoria própria.

Apesar deste trabalho possuir requisitos mínimos, pensando nas variadas aplicações que o mesmo pode possibilitar, optou-se por permitir que o usuário possa realizar modificações na frequência de amostragem. Desta forma, o Quadro 10 apresenta as opções disponíveis:

Amostras por período	Frequência Amostragem [Hz]	Número de Períodos 60 Hz	Tempo total de amostragem [s]
8	480	256	4,267
16	960	128	2,133
32	1920	64	1,067
64	3840	32	0,533
128	7680	16	0,267
256	15360	8	0,133
512	30720	4	0,067

Quadro 10 - Resumo das opções de frequências de amostragem disponibilizadas.
Fonte: Autoria própria.

3.4.2.1 Rotina de aquisição em regime permanente

Com os parâmetros de amostragem determinados, iniciou-se o desenvolvimento da rotina considerando primeiramente a aquisição em regime permanente. O fluxograma da rotina desenvolvida é apresentado na Figura 30.

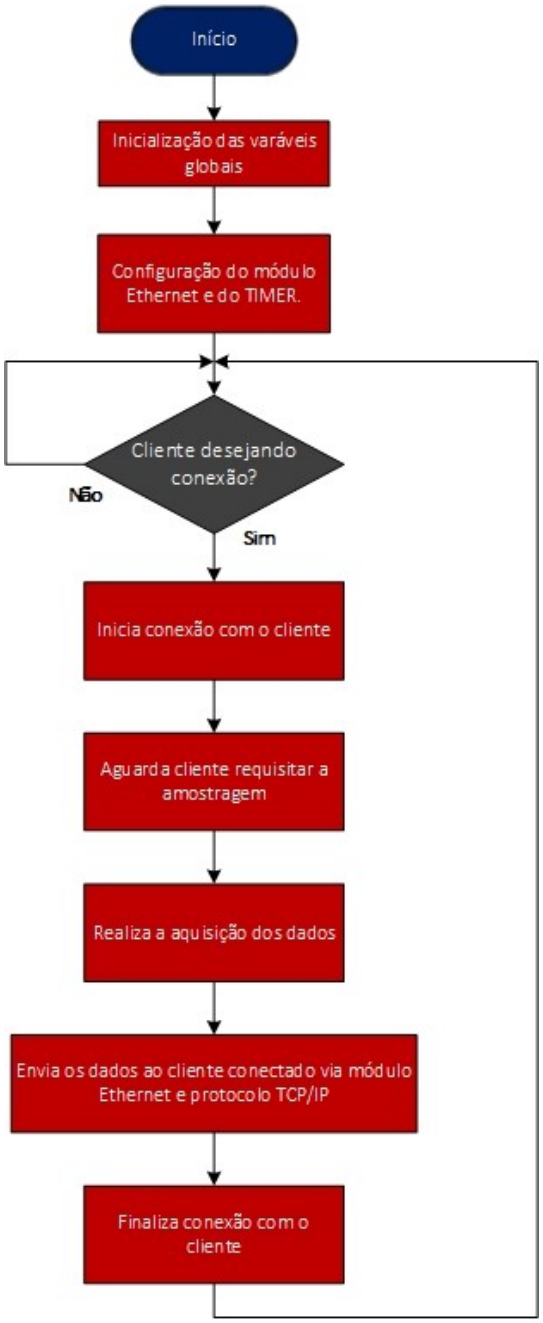

Figura 30 – Fluxograma da rotina desenvolvida para aquisição em regime permanente.
Fonte: Autoria própria.

Quando a placa de processamento é ligada, o microprocessador realiza a inicialização das variáveis globais, e neste ponto é reservado memória para as variáveis que auxiliam as operações no decorrer da rotina e também as variáveis em que serão armazenados os valores das amostras.

A próxima operação é a inicialização do módulo *Ethernet* e do *Timer*. O módulo *Ethernet* é configurado para possibilitar transmissão dos dados amostrados via protocolo TCP/IP pela rede. As informações necessárias para realizar a

configuração deste módulo são os endereços MAC, IP e a porta de comunicação. O endereço MAC, é um endereço físico associado à placa de processamento. Neste trabalho, o endereço MAC foi ajustado para ser igual ao do controlador *Ethernet* presente no microprocessador. O IP escolhido deve ser configurado conforme a rede local em que a placa estará conectada, desta forma, este valor pode sofrer alterações de rede para rede. Para o caso em que for necessário realizar a mudança do número do IP, o novo valor deve ser alterado na rotina e gravado novamente no microprocessador. Assim como ocorre com o IP, acontece com a porta de comunicação a ser configurada. Então, deve-se verificar com o administrador da rede o número IP e a porta de comunicação que deverão ser utilizadas para o correto funcionamento do sistema. Uma vez que os valores para estas variáveis estão definidos, as funções de inicialização do módulo *Ethernet* configuram o controlador e inicia o módulo, possibilitando a troca de informações com outros dispositivos conectados na rede.

No próximo processo ocorre a configuração do módulo *Timer*, sendo este utilizado para auxiliar no processo de aquisição das amostras. O *Timer* permite que rotinas que dependam de intervalos críticos de tempo possam ser executadas, que é o caso da amostragem dos sinais. Se por acaso os intervalos entre as amostras não forem bem definidos, as informações a serem extraídas destes dados podem ser prejudicadas.

Uma vez que as configurações iniciais são realizadas, o módulo *Ethernet* verifica a existência de clientes desejando realizar a conexão com a placa de processamento. Quando um cliente se conecta à placa, o sistema aguarda até que o mesmo informe de qual maneira deseja proceder na aquisição em regime permanente. É neste ponto que o cliente deve informar qual frequência de amostragem deseja utilizar. Então, o módulo *Timer* é configurado para atender à solicitação do cliente. As amostras começam a serem tomadas pelo conversor A/D, e são salvas na memória ram da placa de processamento.

Assim que o processo de amostragem é finalizado, as amostras salvas na memória são transmitidas pela rede via protocolo TCP/IP, por meio das funções do módulo *Ethernet*. Quando a transmissão dos dados é concluída, a conexão com o cliente é finalizada e o sistema volta a verificar se existem clientes desejando uma nova conexão.

3.4.2.2 Rotina de aquisição em regime transitório

Uma vez verificado o funcionamento do código em regime permanente, desenvolveu-se a rotina para aquisição em regime transitório. Na Figura 31 pode-se observar o fluxograma da rotina desenvolvida. Esta, em muito se parece com o código feito para aquisição em regime permanente, sendo a única exceção, a inclusão do bloco que ilustra o processo de detecção de energização da máquina.

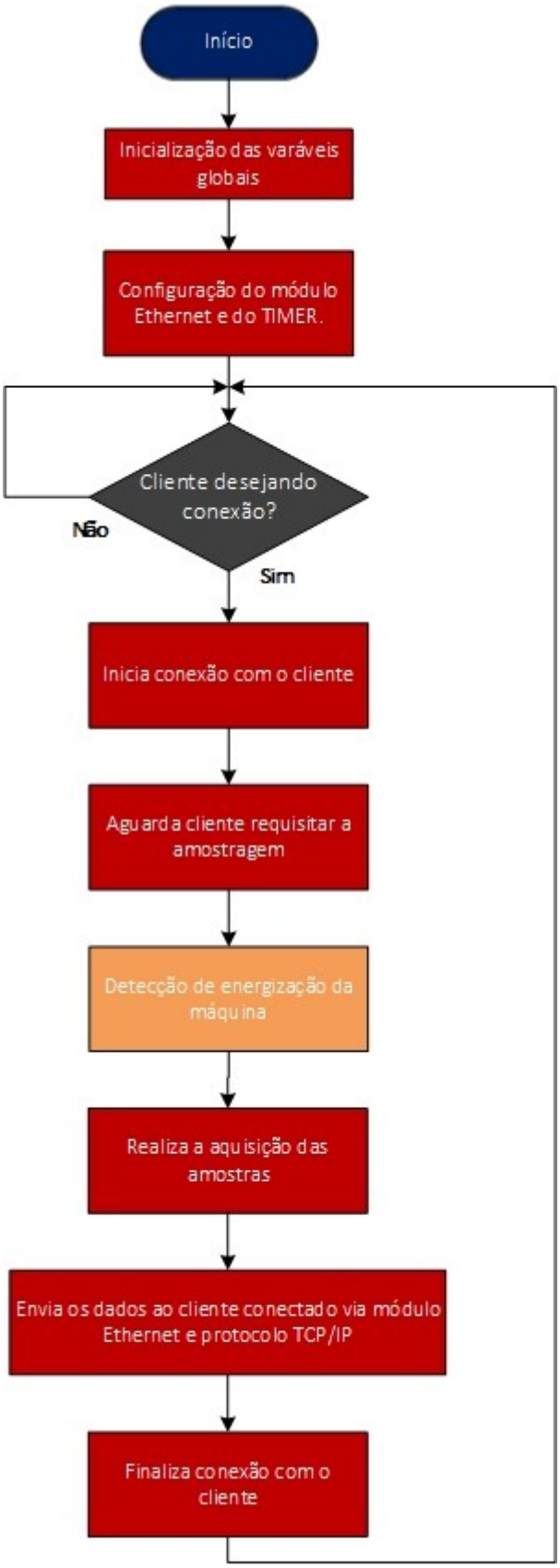

Figura 31 – Fluxograma da rotina desenvolvida para aquisição em regime transitório.
Fonte: Autoria própria.

Até o novo processo inserido na rotina, o funcionamento acontece da mesma maneira como descrito anteriormente. O processo de detecção de energização da máquina foi inserido para que possa ser possível realizar a aquisição do regime transitório decorrente da energização da máquina. Para isso, monitora-se a corrente da Fase A, quando o valor instantâneo de corrente ultrapassa o intervalo de -2,77 a 2,77 A, o sistema considera que a máquina foi energizada e, então, inicia-se o processo de amostragem como descrito na aquisição em regime permanente. Inicialmente seria utilizado um intervalo menor para realizar o monitoramento da energização da máquina, porém não foi possível proceder com a utilização do mesmo após a realização de alguns testes.

O monitoramento da corrente da Fase A é realizado tomando-se amostras a uma frequência extremamente elevada (a nível do *clock* do microprocessador), desta forma, o sistema consegue detectar e iniciar o processo de amostragem sem perder dados do período transitório.

3.4.2.3 Rotina final

Com as duas rotinas desenvolvidas validadas, elaborou-se o código final implementado no microprocessador, sendo que neste código as rotinas para aquisição em regime permanente e em regime transitório foram incorporadas em um único código, cujo fluxograma pode ser observado na Figura 32. O código implementado no sistema pode ser consultado no APÊNDICE A.

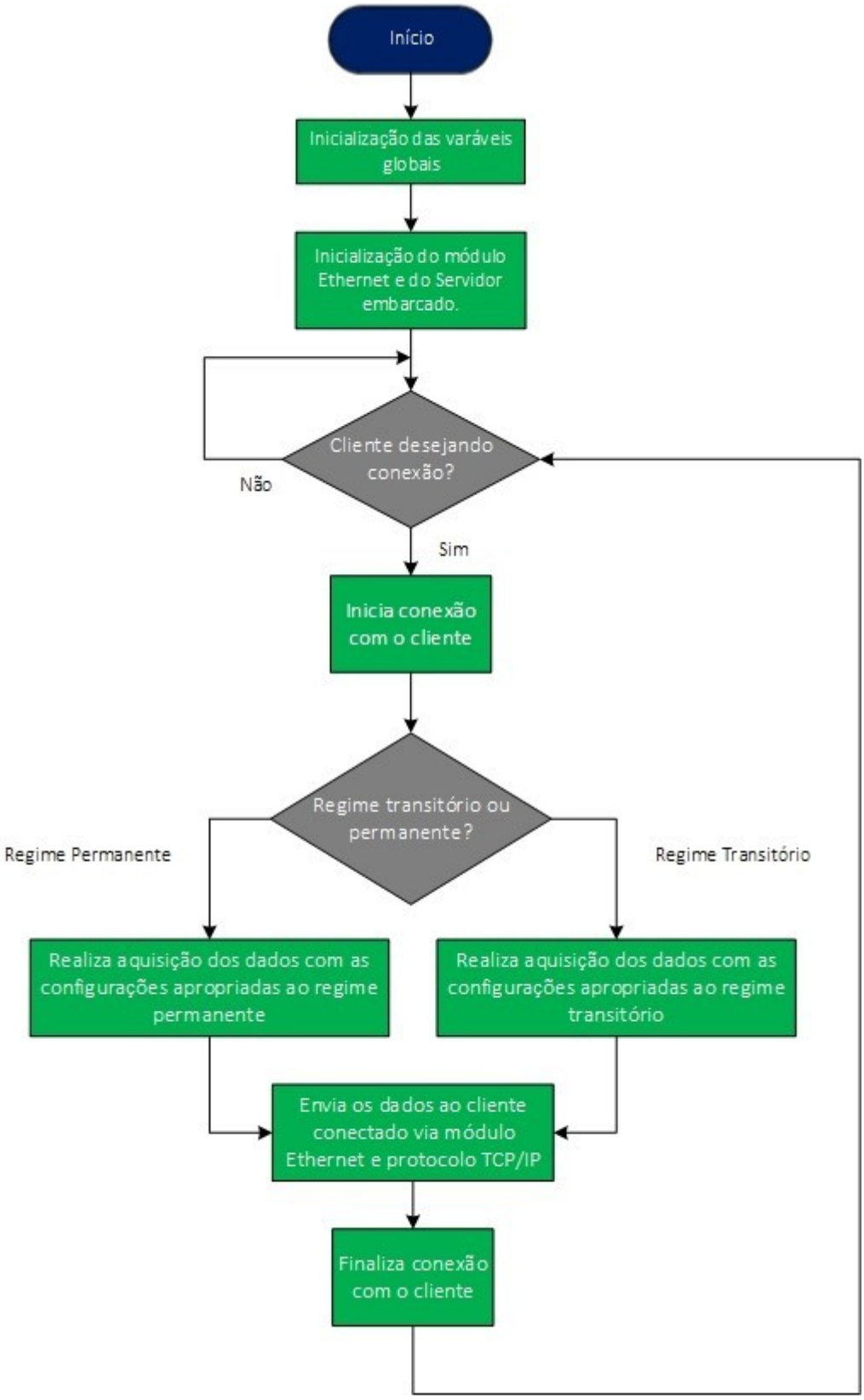

Figura 32 – Rotina desenvolvida final.
Fonte: Autoria própria.

Com o código devidamente em funcionamento, foi dado início ao desenvolvimento da interface de controle e visualização dos dados, que é utilizada em

um computador para controlar a aquisição realizada pela placa de processamento e visualizar os sinais amostrados.

3.5 INTERFACE DE CONTROLE E VISUALIZAÇÃO

Para realizar o desenvolvimento da interface de controle e visualização, optou-se pelo uso do software Matlab. A escolha foi baseada na grande gama de funções que o software possui para trabalhar os dados que são enviados pela placa de processamento e também pelo seu corrente uso no ensino universitário.

Foram desenvolvidas no total seis janelas para realizar a interface entre o usuário e as rotinas de aquisição, sendo elas:

- Interface de inicialização;
- Interface de aquisição em regime permanente;
- Interface de aquisição em regime transitório;
- Interface de aquisição com atualização automática;
- Interface para visualização das grandezas;
- Interface de calibração do sistema de aquisição;

3.5.1 Interface de Inicialização

A interface de inicialização é responsável por realizar as configurações iniciais do modo de operação do sistema, sendo seu layout observado na Figura 33. O usuário deve escolher dentre as opções disponíveis.

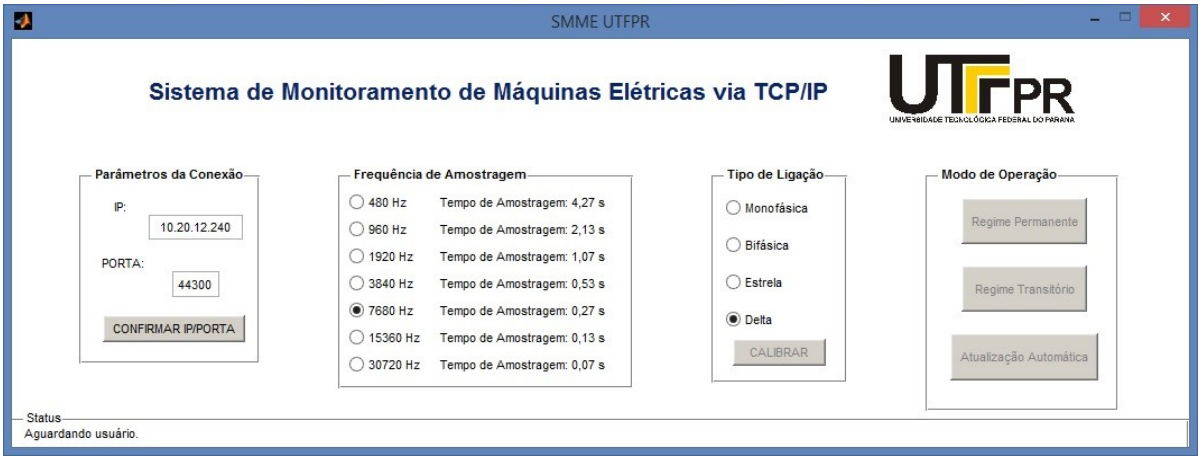

Figura 33 – Interface de inicialização.
Fonte: Autoria própria.

Neste ponto, o usuário deve confirmar o IP e a porta configurados no código da placa de processamento, sendo que em uma eventual rede onde existem mais de um sistema de aquisição conectados, pode-se optar em qual deles deseja-se conectar. Uma vez confirmado os valores, o usuário deve pressionar o botão “CONFIRMAR IP/PORTA”. Quando pressionado, o sistema realiza um teste de conexão com placa com os respectivos valores de IP e porta informados e, se tudo ocorrer bem, uma mensagem será exibida para o usuário no campo “Status”, sendo os botões do modo de operação e calibração ativados. Por outro lado, no caso de uma negativa no teste de conexão, será mostrada uma mensagem informando ao usuário a ocorrência de uma falha no teste, sendo requisitado uma verificação nas conexões e, neste caso, os botões do modo de operação e calibração permanecerão desativados.

A próxima configuração diz respeito à frequência de amostragem, sendo que o usuário pode selecionar um valor dentre sete disponíveis, logo após a frequência é informado o tempo total de amostragem do sinal que será possível alcançar utilizando determinada frequência.

Após selecionar a frequência de amostragem, o usuário deve escolher qual o tipo de ligação será realizado na alimentação da máquina. Esta ligação deve ser feita exatamente igual nos sensores de tensão do sistema, isto porque para cada modo existente de escolha, foi verificado em laboratório o valor necessário de compensação nos valores amostrados de tensão, que é aplicado de acordo com cada tipo de ligação.

Então, seleciona-se o modo de operação pelo qual será realizada a amostragem dos sinais, sendo que existem três modos disponíveis, regime permanente, regime transitório e atualização automática.

3.5.2 Modo de Operação em Regime Permanente

O usuário deve optar pelo modo de operação em regime permanente quando desejar obter informações da máquina em funcionamento contínuo. O layout desta interface pode ser observado na Figura 34.

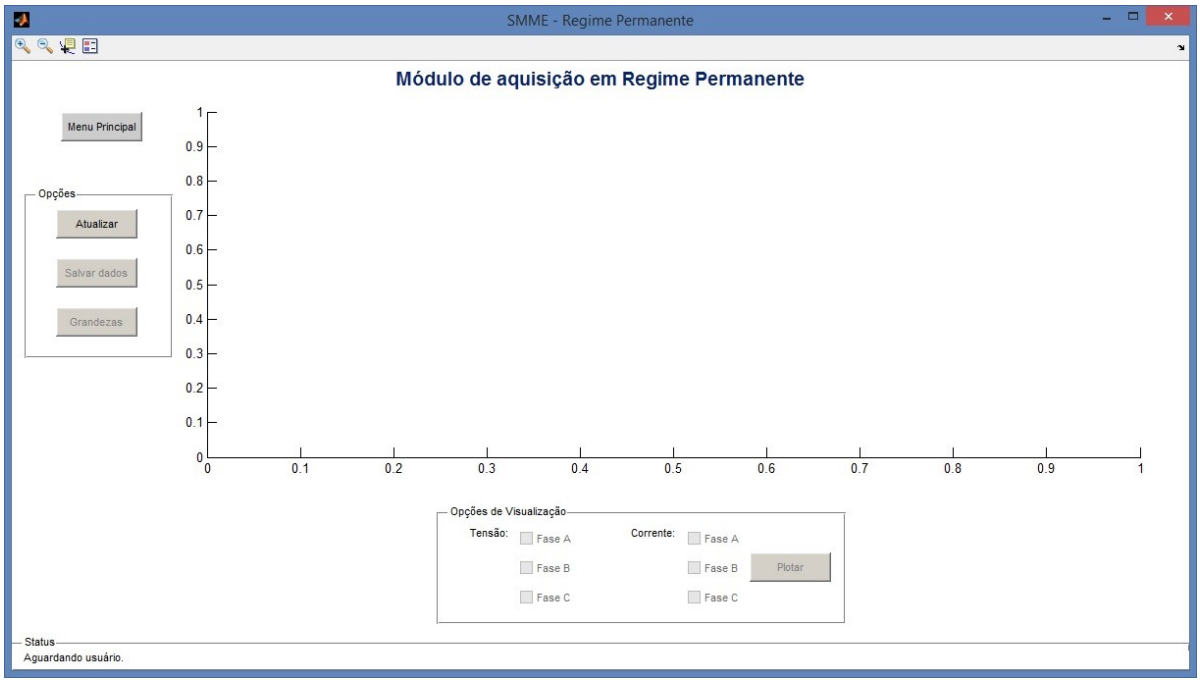

Figura 34 – Interface de aquisição em regime permanente.
Fonte: Autoria Própria.

Neste modo de operação, o usuário determina o momento em que deseja realizar a aquisição, pressionando o botão “Atualizar”. A interface comunica-se com a placa de processamento via protocolo TCP/IP, informando a requisição de amostragem em regime permanente com a frequência de amostragem escolhida. Assim que a placa de processamento verifica a requisição do usuário, a amostragem é iniciada, sendo que ao seu final os dados são transmitidos à interface. O usuário

deve acompanhar o processo pelo campo “Status” e, caso tudo ocorra de forma satisfatória, uma mensagem o informará, sendo que as opções no campo “Opções de Visualização” serão liberados para escolha do usuário. Neste campo podem ser escolhidos os sinais que serão plotados no gráfico quando o botão “Plotar” for pressionado. A Figura 35 exibe a interface em funcionamento com dois sinais plotados, como exemplo.

Além das opções de visualização serem liberadas com a aquisição dos sinais, os botões de “Salvar dados” e “Grandezas” também são liberados. Quando o botão de “Salvar dados” é pressionado, uma janela é aberta para escolha do caminho que se deseja salvar os dados aquisitados. O botão “Grandezas” será tratado em uma seção específica a frente.

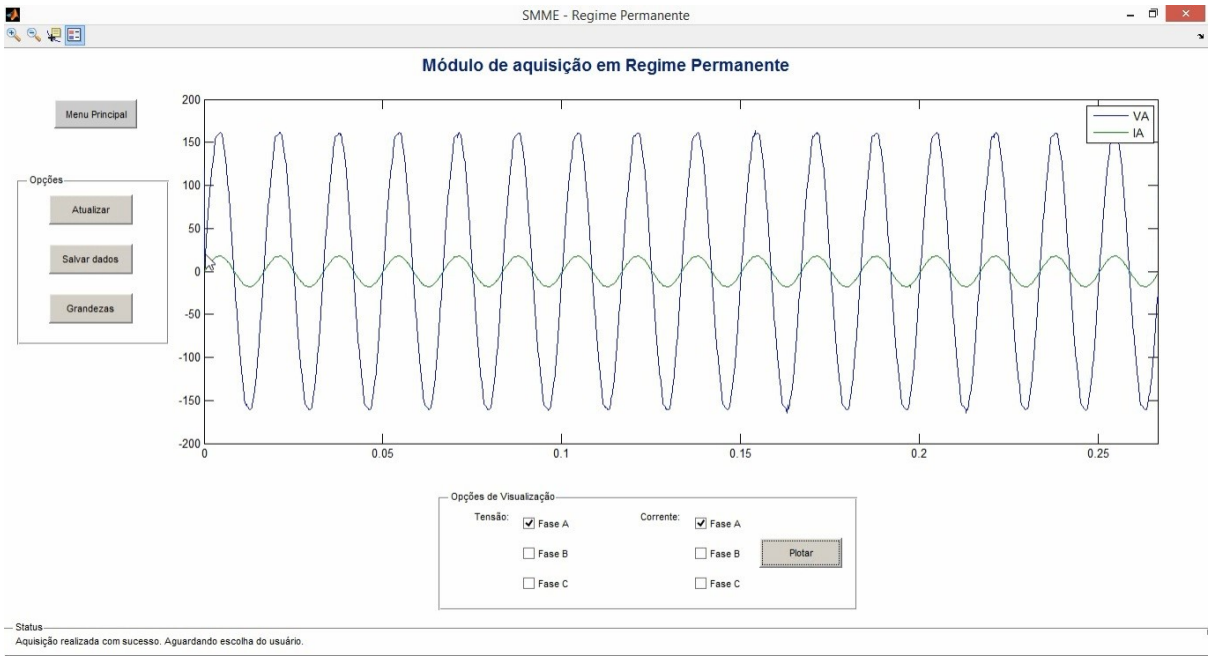

Figura 35 – Interface de aquisição em regime permanente em funcionamento.
Fonte: Autoria própria.

No caso de se pressionar o botão “Atualizar” e o sistema retornar uma negativa no campo “Status”, as opções mencionadas no parágrafo anterior não serão liberadas, o usuário deverá verificar as conexões e tentar novamente. O botão “Menu Principal” é utilizado quando o usuário deseja acessar a interface de inicialização novamente.

3.5.3 Modo de Operação em Regime Transitório

O usuário deve optar pelo modo de operação em regime transitório, quando desejar obter informações dos sinais elétricos da máquina em no período transiente. O layout desta interface pode ser observado na Figura 36.

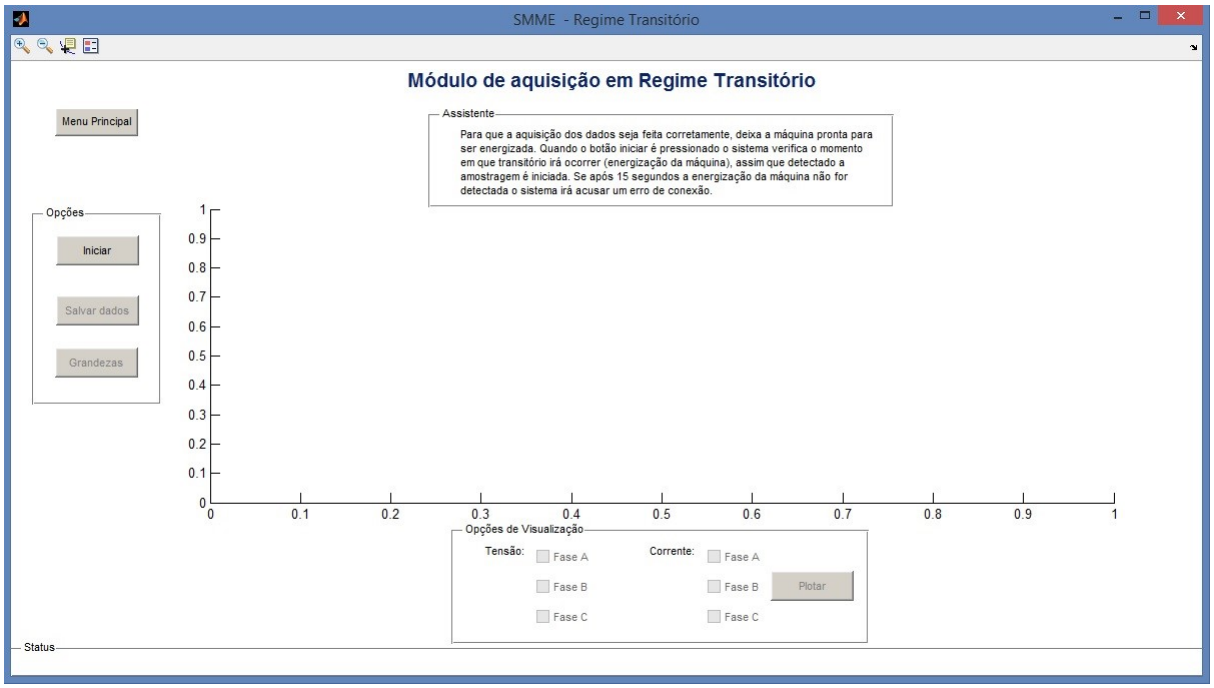

Figura 36 – Interface de aquisição em regime transitório.
Fonte: Autoria própria.

Neste modo de operação o usuário determina o momento em que deseja iniciar a monitorização do transitório pela placa de processamento, pressionando o botão “Iniciar”. Algumas instruções são exibidas na tela assim que a interface é aberta com o intuito de auxiliar o usuário para o correto funcionamento deste modo de operação.

Iniciando a aquisição, a interface comunica-se com a placa de processamento via protocolo TCP/IP, informando a requisição de monitorização do transitório e também informando a frequência de amostragem escolhida para realizar a aquisição. Assim que a placa de processamento verifica a requisição do usuário, o sistema começa a amostrar os valores da Fase A do sistema de aquisição, e quando o valor instantâneo da corrente deixa o intervalo de -2,77 a 2,77 A, a amostragem é iniciada,

sendo que ao seu final os dados são transmitidos à interface. Caso a máquina não seja energizada dentro de 15 segundos, o sistema irá informar um erro de conexão ao usuário, pois não foi detectado a variação de corrente necessária para que o sistema inicie a conexão. Este mesmo informe será exibido ao usuário no caso de o sistema não conseguir realizar a conexão com a interface.

O usuário deve acompanhar o processo pelo campo “Status”, caso tudo ocorra de forma satisfatória uma mensagem o informará, sendo que as alternativas no campo “Opções de Visualização” serão liberados para escolha do usuário. A plotagem dos sinais amostrados procede da mesma forma como mencionado na interface anterior. A Figura 37 exibe a interface em funcionamento com dois sinais plotados. Além das opções de visualização serem liberadas com a aquisição dos sinais, os botões de “Salvar dados” e “Grandezas” também são liberados. Quando o botão de “Salvar dados” é pressionado, uma janela se abre para escolha do caminho em que se deseja salvar os dados aquisitados. O botão “Grandezas” será tratado em uma seção específica a frente.

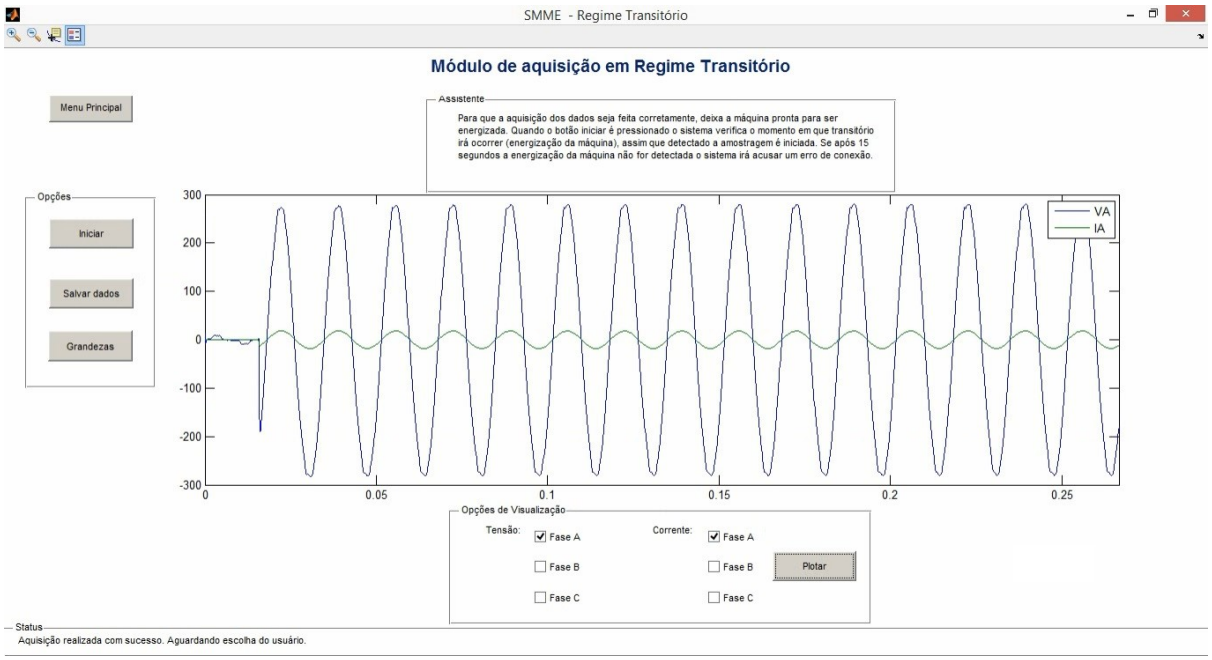

Figura 37 – Interface de aquisição em regime transitório em funcionamento.
Fonte: Autoria própria.

O botão “Menu Principal” é utilizado quando o usuário deseja acessar a interface de inicialização novamente.

3.5.4 Modo de Operação com Atualização Automática

O usuário deve optar pelo modo de operação com Atualização Automática quando desejar obter informações dos sinais elétricos da máquina de forma contínua. O layout desta interface pode ser observado na Figura 38.

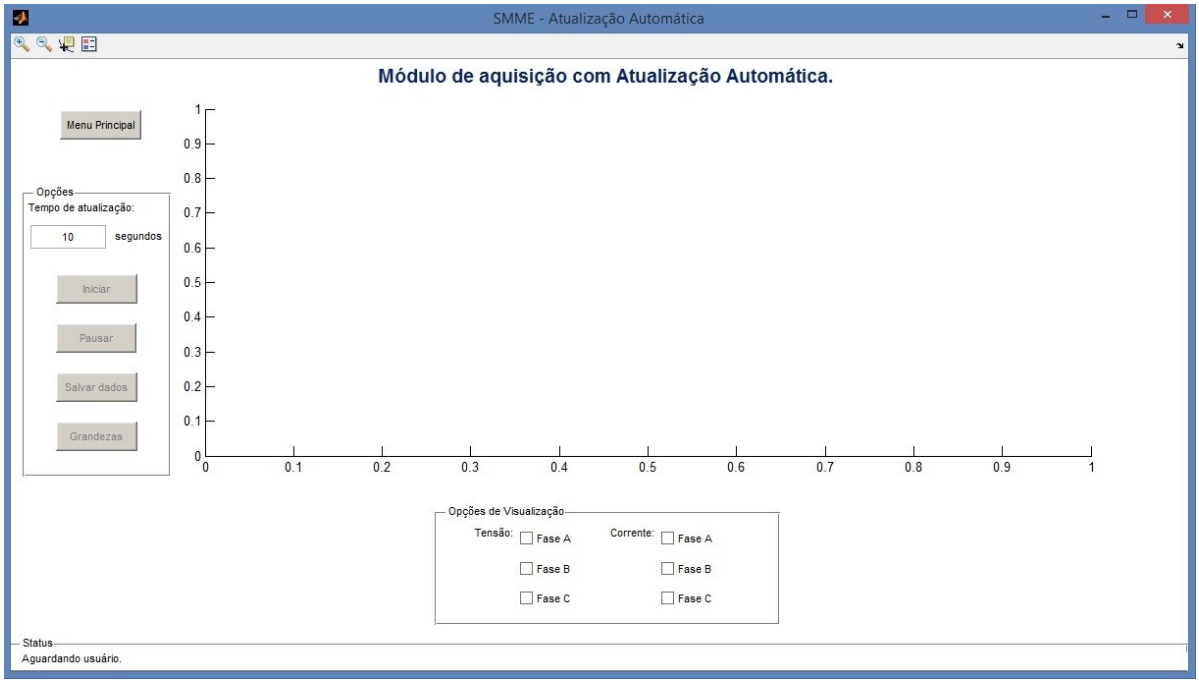

Figura 38 – Interface de aquisição com atualização automática.
Fonte: Autoria própria.

Neste modo de operação o usuário deve selecionar pelo menos um sinal para ser plotado e, quando este sinal é selecionado, o botão “Iniciar” é liberado. O tempo de atualização requerido deve ser informado, sendo especificado por padrão o tempo de dez segundos. Quando o botão “Iniciar” é pressionado, a interface comunica-se com a placa de processamento via protocolo TCP/IP, informando frequência de amostragem escolhida e iniciando a amostragem. O usuário deve acompanhar o processo pelo campo “Status” e, caso tudo ocorra de forma satisfatória, uma mensagem o informará, e mostrará o tempo restante, em segundos, para a próxima aquisição.

A plotagem dos sinais amostrados acontece de forma automática a cada atualização, caso for desejável incluir ao retirar algum sinal do gráfico na próxima atualização, basta escolher no campo “Opções de Visualização”. A Figura 39 mostra

sinais sendo plotados durante o uso desta interface. Quando é dado início ao processo de atualização automática, o botão “Pausar” é liberado, assim, quando o mesmo é pressionado, o processo é interrompido, porém a atualização em andamento é finalizada. Quando o processo é interrompido, os botões “Salvar dados” e “Grandezas” são liberados e, quando o botão de “Salvar dados” é pressionado, uma janela se abre para escolha do caminho em que se deseja salvar os dados aquisitados. A interface que se abre quando o botão “Grandezas” é pressionado, será tratado na próxima seção.

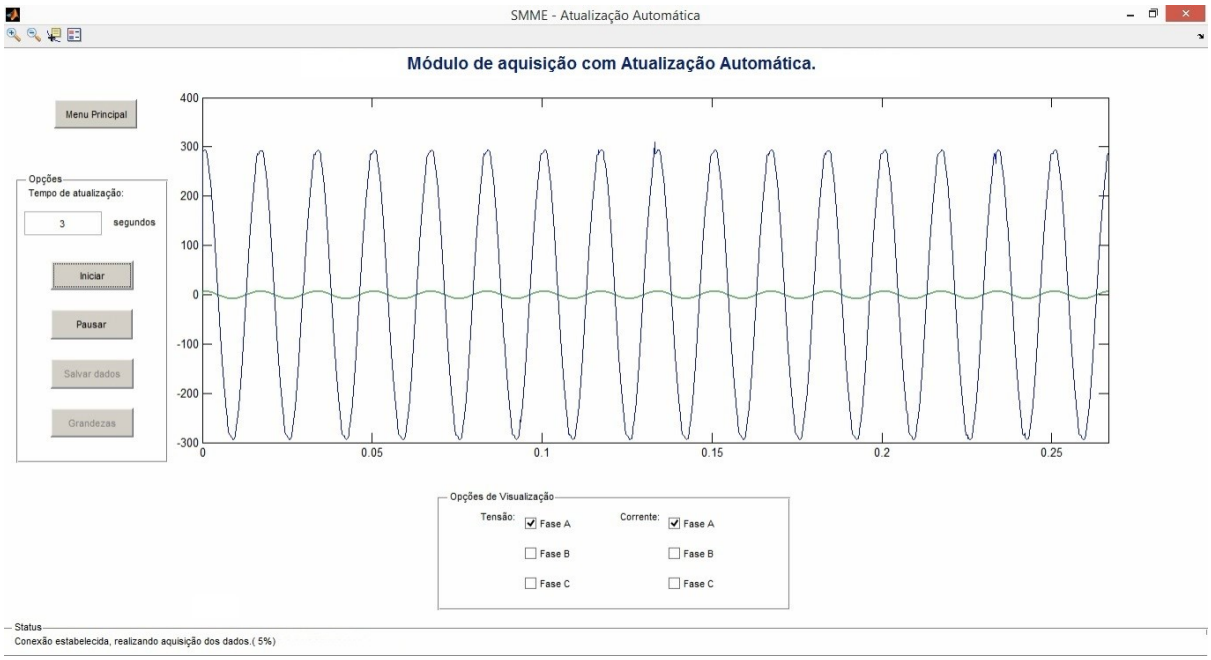

Figura 39 – Interface de aquisição com atualização automática em funcionamento.
Fonte: Autoria própria.

O botão “Menu Principal” é utilizado quando o usuário deseja acessar a interface de inicialização novamente.

3.5.5 Grandezas

Em todos os modos de operação existe um botão denominado “Grandezas”, e quando dados aquisitados são enviados à interface este botão é liberado. Esta aplicação foi criada para apresentar um breve conjunto de informações sobre o sinal

recém aquisitado. Na Figura 40 pode-se observar o layout desenvolvido para esta interface.

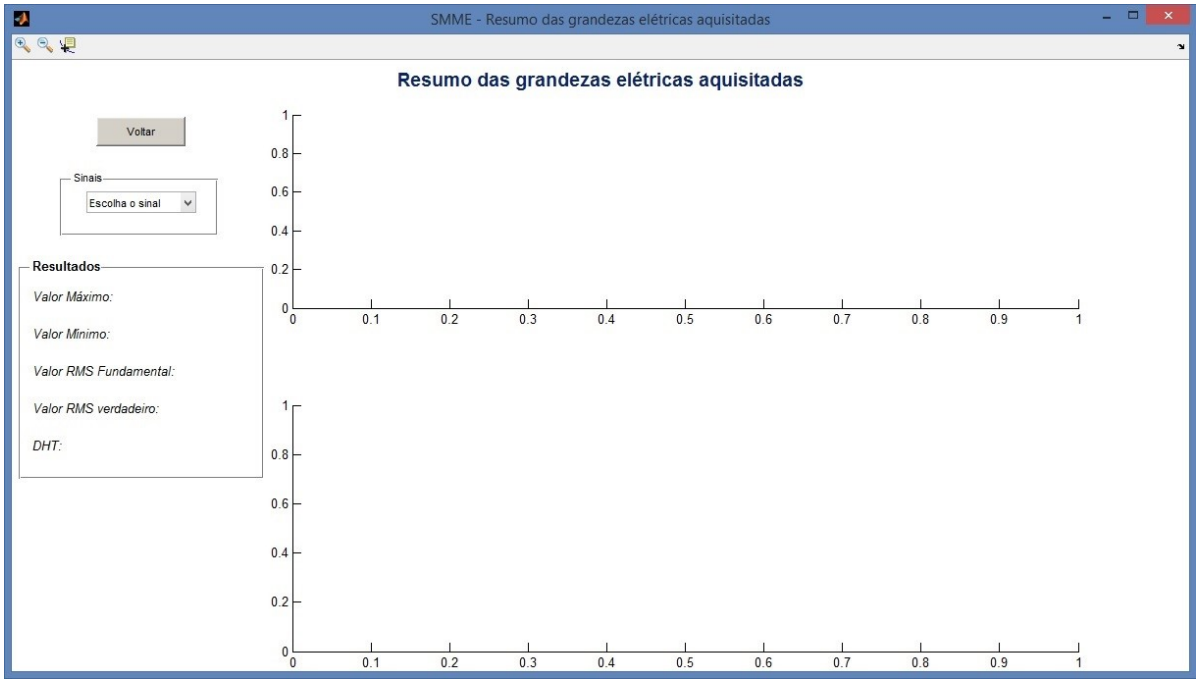

Figura 40 – Interface que exibe o resumo das grandezas elétricas dos sinais.
Fonte: Autoria própria.

No campo “Sinais”, existem seis opções de sinais para escolha, que são as três fases de tensão e as três fases de corrente da máquina. Quando um destes sinais é selecionado, o sinal e o espectro harmônico são plotados e outras informações como, valor máximo, mínimo, RMS fundamental, RMS verdadeiro e a distorção harmônica total (DHT) são exibidos no campo “Resultados”. A Figura 41, mostra um caso onde um sinal foi selecionado e os dados estão disponíveis para visualização.

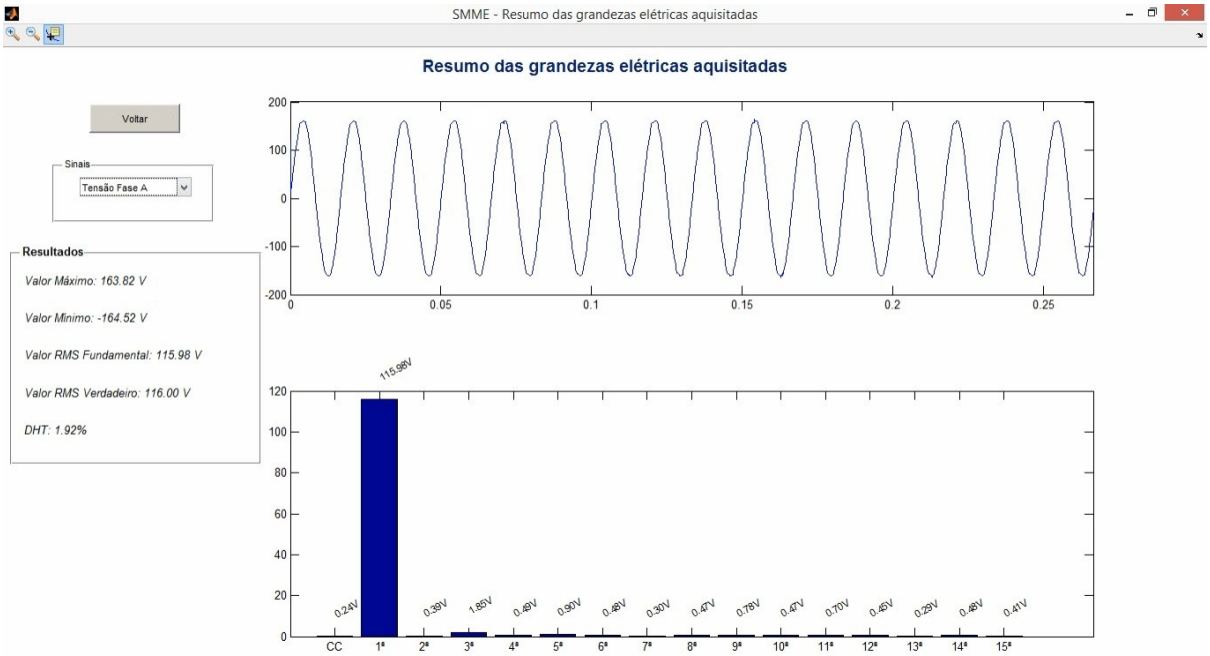

Figura 41 - Interface que exibe o resumo das grandezas elétricas dos sinais em funcionamento. Fonte: Autoria própria.

A interface “Grandezas” possui uma restrição operativa para frequências de amostragem igual ou inferior a 960 Hz, devido à quantidade de amostras disponíveis por período. Outro problema ocorre quando a frequência fundamental é diferente de 60 Hz, sendo que nestes casos os valores calculados não corresponderão aos valores reais. Desta forma esta aplicação deve ser utilizada com cautela, respeitando seus limites operativos.

3.5.6 Calibração

Na interface de inicialização está presente o botão que aciona a aplicação desenvolvida para calibrar o sistema de aquisição. Esta aplicação foi criada com o intuito de realizar um ajuste fino nos dados coletados, para que possa representar o máximo possível da realidade, a interface inicial pode ser observada na Figura 42.

Figura 42 - Interface para calibração do sistema de aquisição de sinais.
Fonte: Autoria própria.

Apesar do sistema de aquisição contar com os *trimmers* para ajustes do mesmo, a interface de calibração fornece um meio para realização de um ajuste fino, mais exato. Também é possível realizar a compensação necessária nos valores aquisitados com o transformador de potencial, devido a sua característica de não-linearidade.

Para seu funcionamento, é necessário informar o modo pelo qual o sistema de alimentação está conectado à máquina, então seleciona-se as grandezas que se deseja realizar a calibração. Posteriormente, é necessário informar os valores RMS de tensão e corrente medidos com um equipamento calibrado. Com base nos valores aquisitados e nos valores informados, o sistema irá calcular a compensação necessária que deve realizar para que os dados amostrados estejam o mais próximo dos reais. A compensação é realizada multiplicando-se os valores amostrados por uma constante de compensação, sendo esta calculada da seguinte maneira:

$$Constante\ de\ compensação = \frac{V_{real}\sqrt{2}}{\overline{D_{máx}}\frac{V_{ref+}}{1024} - \frac{V_{ref+}}{2}}$$

onde:

V_{real}: valor RMS da grandeza medido com instrumento calibrado;

$\overline{D_{máx}}$: valor médio dos valores digitais máximo coletados;

V_{ref+}: valor de referência superior escolhido para configuração do conversor A/D;

Para realização desta compensação, é necessário que os sistemas utilizados para fazer as medidas dos valores RMS de tensão e corrente, estejam calibrados e, desta forma, informando os valores reais. Ainda, é importante que os de tensão e corrente que estarão sendo utilizados para realizar esta calibração possuam poucas variações.

Os códigos desenvolvidos e utilizados podem ser consultados no APÊNDICE B.

4 RESULTADOS E DISCUSSÕES

Nesta seção serão apresentados os principais resultados obtidos com a elaboração e funcionamento do sistema de monitoramento de máquina elétricas via TCP/IP desenvolvido.

Para montagem do sistema final, foi adquirido um suporte em acrílico para alocar os sensores e a placa de condicionamento. Ainda, foi previsto um espaço adicional para acomodação da placa de processamento. Na Figura 43 pode ser observado o sistema final confeccionado.

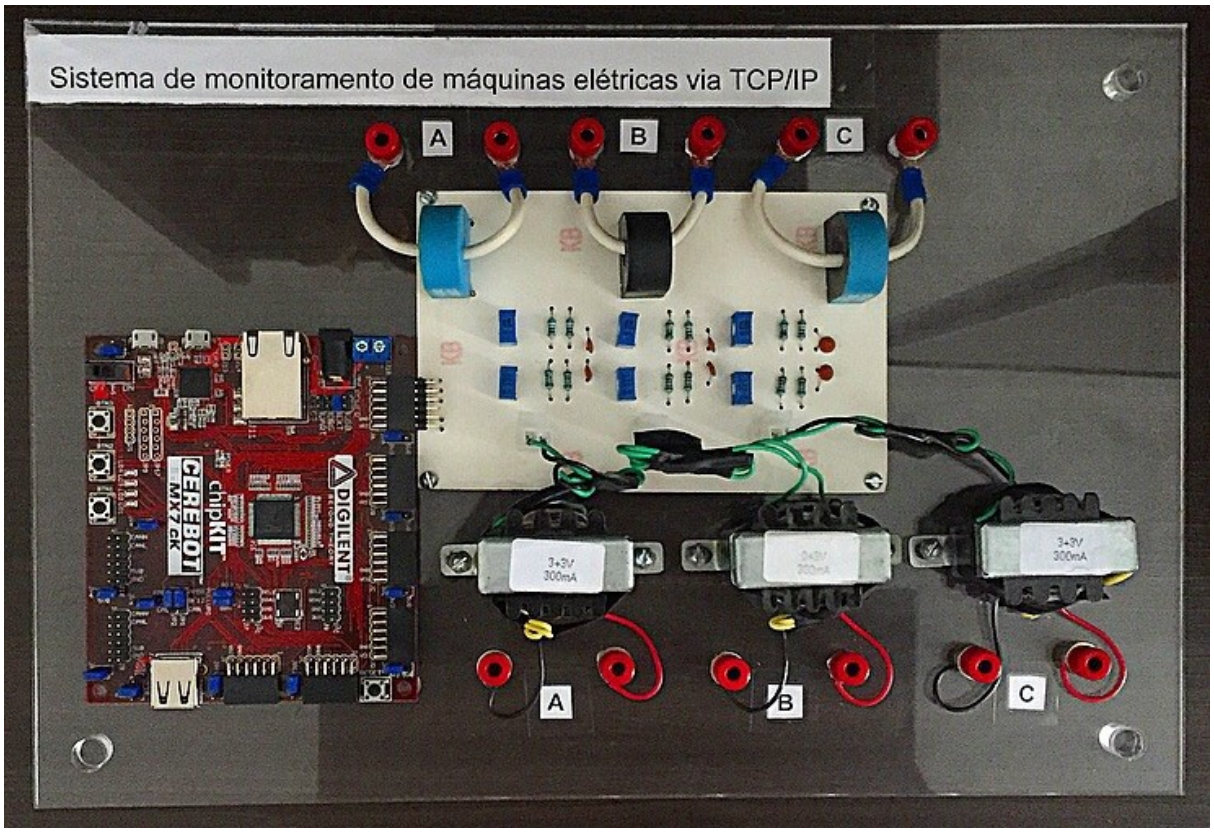

Figura 43 - Sistema de monitoramento de máquinas elétricas desenvolvido.
Fonte: Autoria própria.

Após a elaboração do sistema final, foi necessário realizar testes com situações reais de monitoramento de máquinas elétricas, onde, buscou-se avaliar o trabalho desenvolvido aplicando todas as características que o sistema possui. Para isso, foi utilizado o Laboratório de Sistemas Inteligentes (LSI), do Centro Integrado de Pesquisa e Automação (CIPECA), da Universidade Tecnológica Federal do Paraná campus Cornélio Procópio (UTFPR-CP). Onde foram utilizados um MI e um gerador

de corrente contínua, cuja função era emular uma carga. A conexão do sistema de monitoramento foi realizada na rede de comunicação do laboratório e os resultados foram avaliados com o auxílio de equipamentos de medição presentes no mesmo. O arranjo experimental pode ser observado na Figura 44.

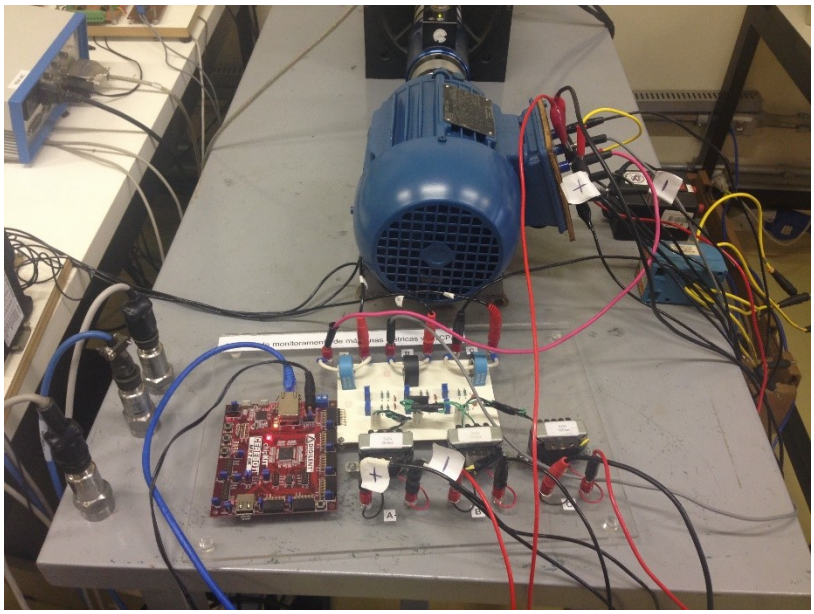

Figura 44 - Ligação do sistema de monitoramento ao MIT do LSI.
Fonte: Autoria Própria.

Como cada rede de computadores possui suas particularidades tanto em relação a sua construção como em sua configuração, foi necessário o suporte da Coordenadoria de Gestão de Tecnologia da Informação (COGETI) da UTFPR-CP, para configuração de um IP fixo para a placa. Com a criação deste, a COGETI informa em quais locais serão possíveis realizar a comunicação com a placa, sendo que devido as características da rede, a comunicação não acontece em quaisquer pontos da universidade. Após realizadas as configurações necessárias e os resultados dos testes de comunicações terem sido satisfatórios, deu-se início ao monitoramento.

4.1 MODO DE OPERAÇÃO EM REGIME PERMANENTE

Para verificação do funcionamento do sistema desenvolvido, foi avaliado primeiramente o modo de operação em regime permanente. Para isso o MIT foi ligado em vazio, sendo as primeiras aquisições tomadas utilizando-se a frequência de

amostragem de 7680 Hz. Na Figura 45 e na Figura 46, pode-se observar o monitoramento das formas de onda trifásicas de tensão e corrente respectivamente.

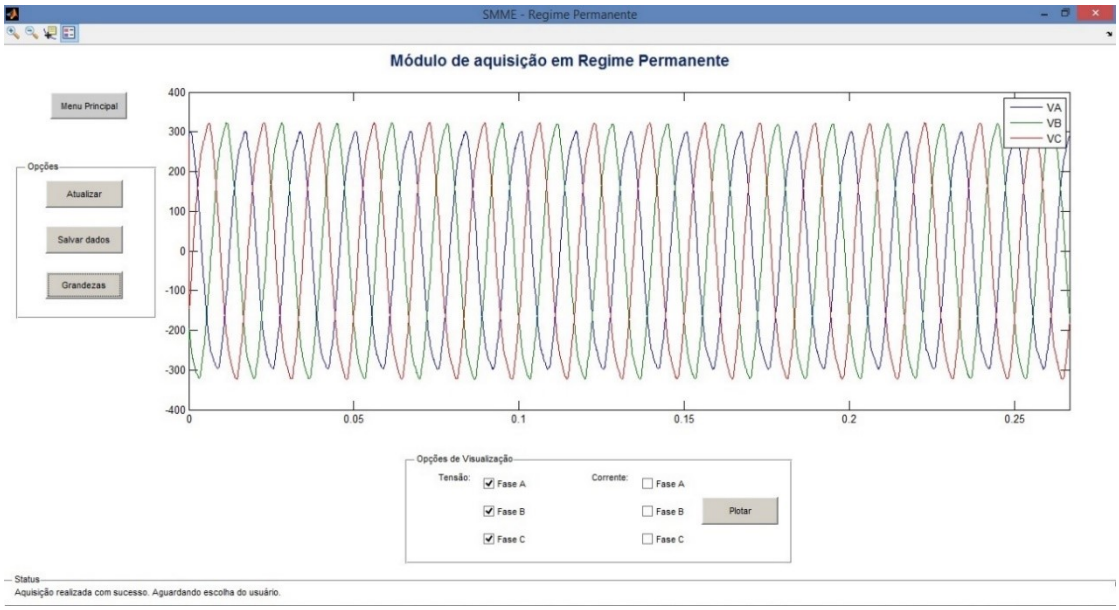

Figura 45 - Monitoramento das tensões do MIT em vazio, utilizando frequência de amostragem de 7680Hz com modo de operação em regime permanente.
Fonte: Autoria própria.

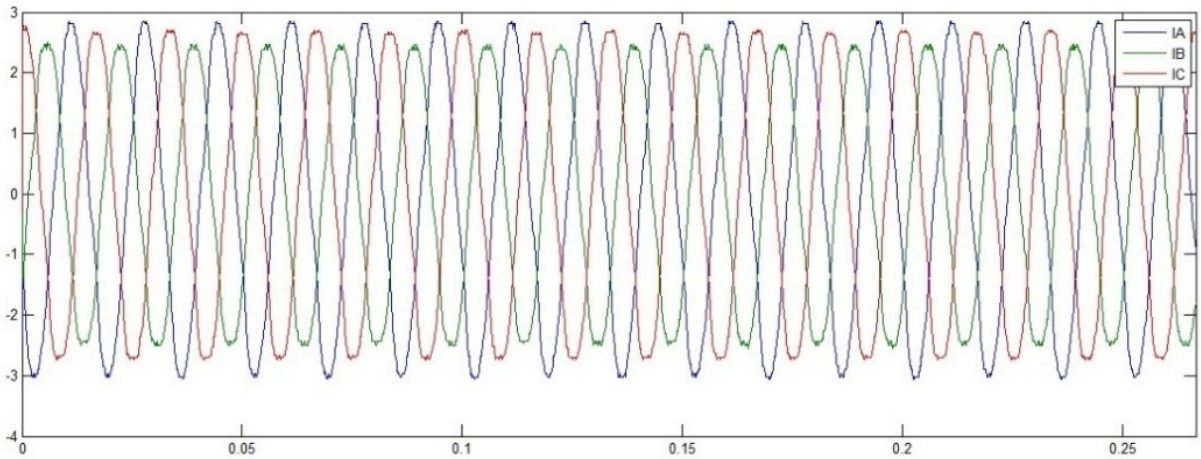

Figura 46 - Monitoramento das correntes do MIT em vazio, utilizando frequência de amostragem de 7680Hz.
Fonte: Autoria própria.

Em ambos os casos se percebe um desbalanceamento entre as fases que não foi observado nos instrumentos auxiliares presentes no laboratório. Desta forma, foi necessário utilizar o assistente para calibração do sistema de aquisição.

Após o uso do assistente para calibração, novas aquisições foram realizadas da mesma forma, sendo os resultados exibidos na Figura 47 e na Figura 48.

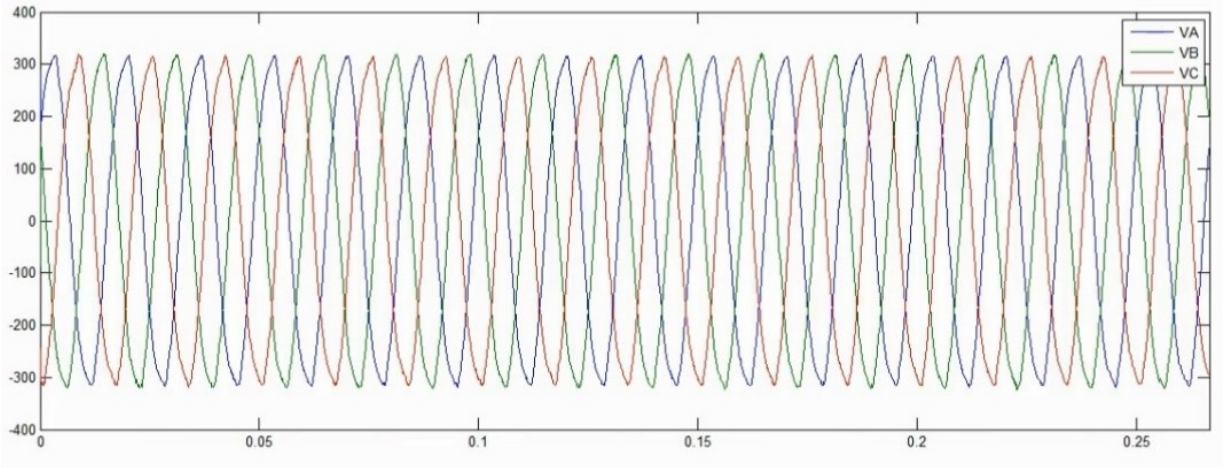

Figura 47 - Monitoramento das tensões do MIT em vazio, utilizando frequência de amostragem de 7680Hz, após calibração.
Fonte: Autoria própria.

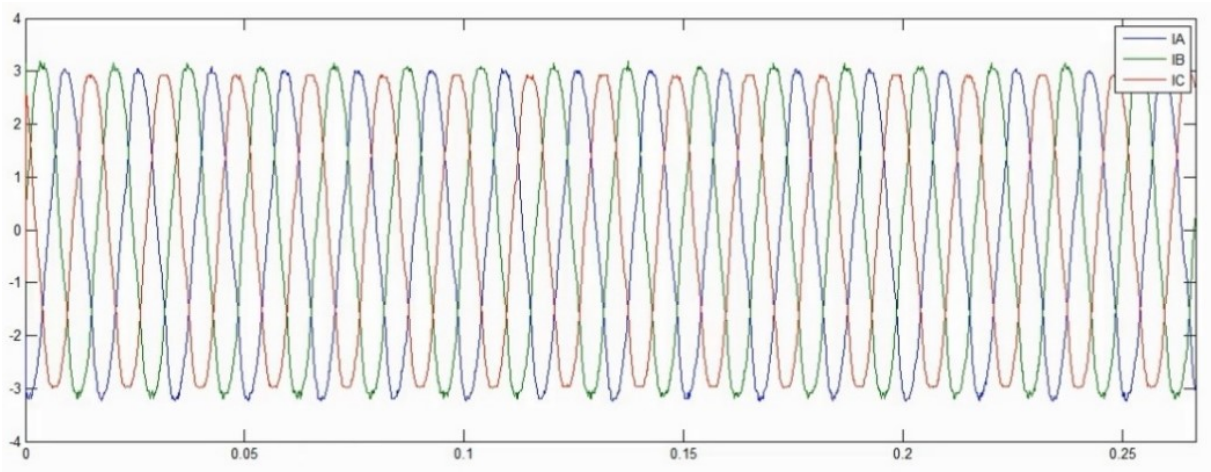

Figura 48 - Monitoramento das correntes do MIT em vazio, utilizando frequência de amostragem de 7680Hz, após calibração.
Fonte: Autoria própria.

Em ambos os casos foi possível perceber a melhora do desbalanceamento entre os sinais, sendo que desta forma os valores se aproximaram dos resultados obtidos com os instrumentos utilizados no laboratório.

Na Figura 49 e na Figura 50 pode ser observado o funcionamento da interface “Grandezas”, sendo possível visualizar os resumos para os sinais de tensão e de corrente da fase A, respectivamente.

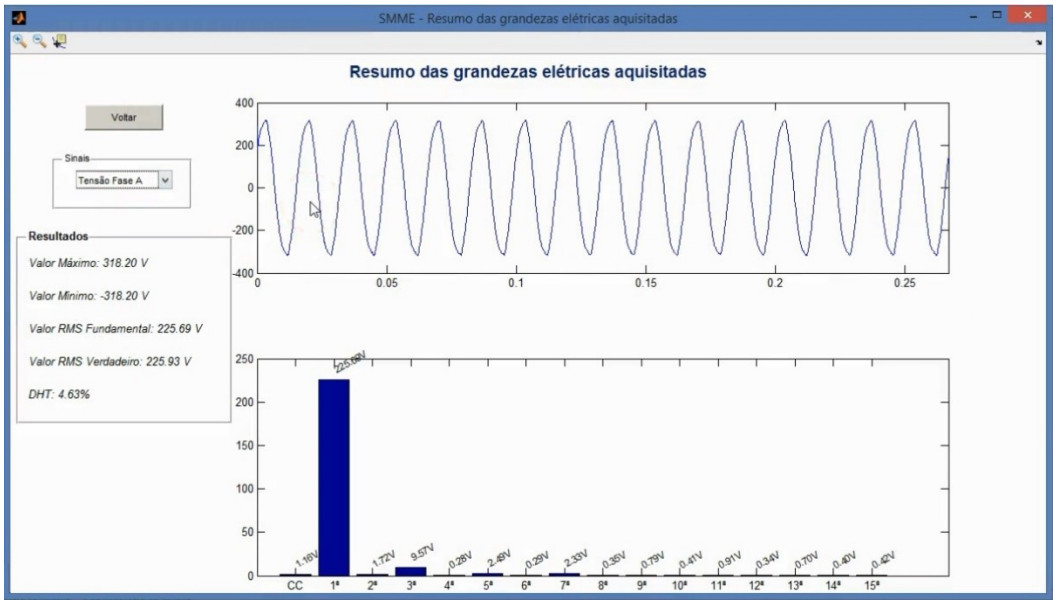

Figura 49 - Resumo do sinal de tensão da Fase A, para aquisição com 7680 Hz.
Fonte: Autoria própria.

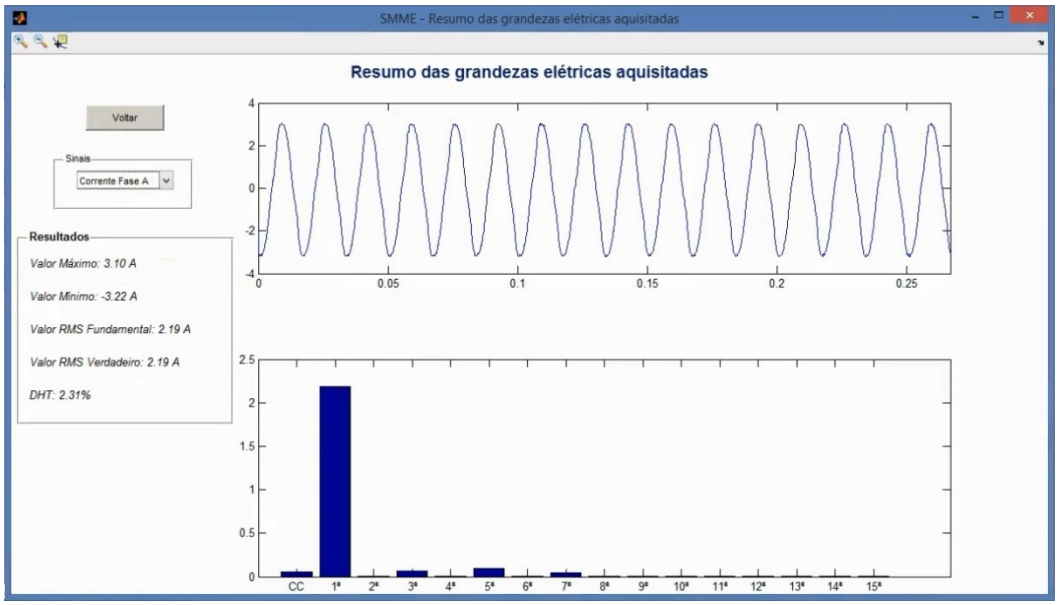

Figura 50 - Resumo do sinal de corrente da Fase A, para aquisição com 7680 Hz.
Fonte: Autoria própria.

Em ambos os casos os resultados obtidos foram satisfatórios se comparados com os valores medidos pelos equipamentos auxiliares do laboratório.

Ainda, foram realizadas novas aquisições utilizando a frequência de amostragem de 960 Hz. Na Figura 51 e na Figura 52, se pode observar o monitoramento das formas de onda trifásicas de tensão e corrente respectivamente.

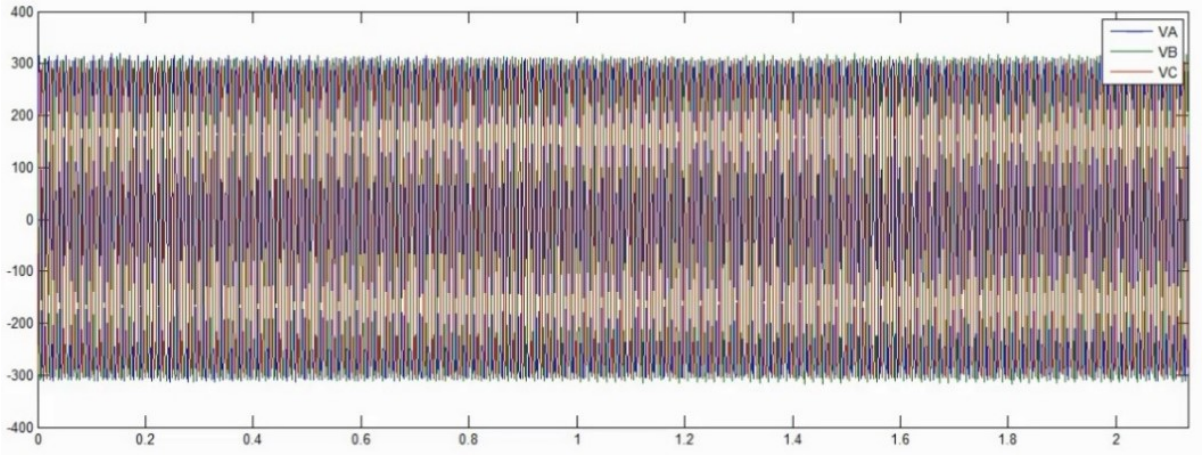

Figura 51 - Monitoramento das tensões do MIT em vazio, utilizando frequência de amostragem de 960 Hz.
Fonte: Autoria própria.

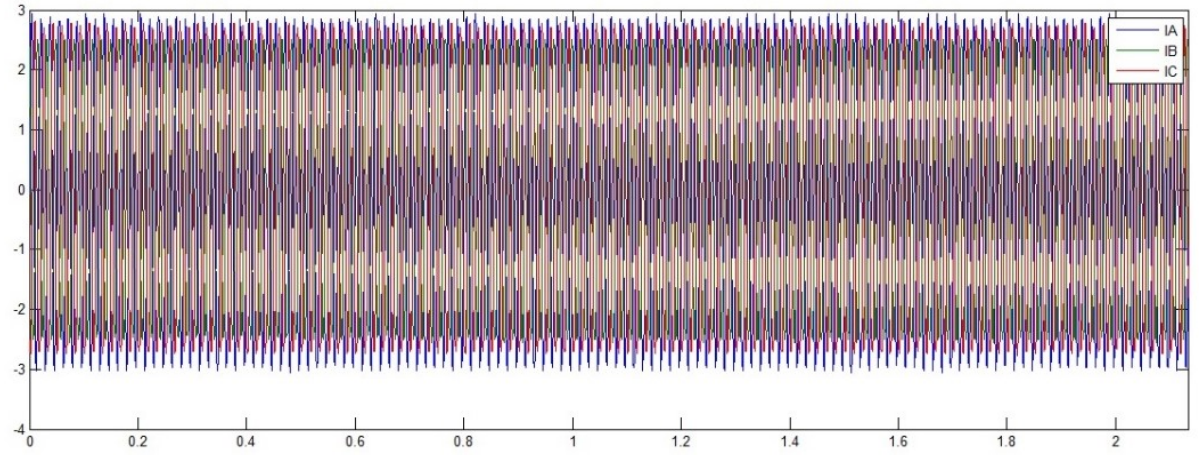

Figura 52 - Monitoramento das correntes do MIT em vazio, utilizando frequência de amostragem de 960 Hz.
Fonte: Autoria própria.

Em ambos os casos os resultados obtidos foram satisfatórios. Tentou-se acessar a interface “Grandezas”, porém como já mencionado anteriormente no texto, esta opção está restrita a uma faixa de frequências de amostragens que não contempla a frequência de 960 Hz. Desta forma, a interface retornou uma negativa, como pode ser observado na Figura 53.

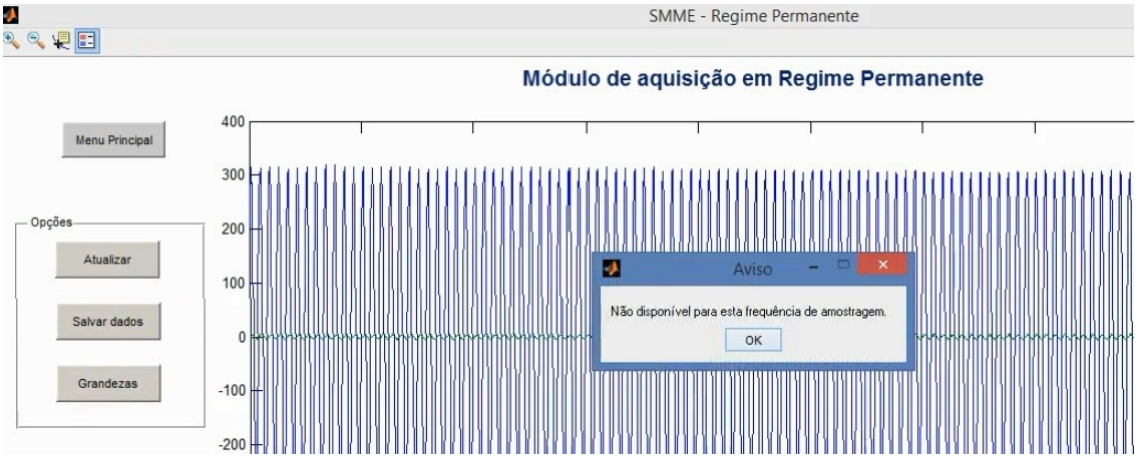

Figura 53 - Negativa da interface "Grandezas" para frequência de amostragem fora de sua faixa de operação.
Fonte: Autoria própria.

Com as aplicações realizadas, foi possível verificar o funcionamento satisfatório do modo de operação em regime permanente, da interface “Grandezas” e do assistente de calibração elaborados.

4.2 MODO DE OPERAÇÃO COM ATUALIZAÇÃO AUTOMÁTICA

Após o teste com o modo de operação em regime permanente, foi realizado a avaliação do modo de operação com atualização automática. Para isso o MIT foi ligado em vazio, sendo as primeiras aquisições tomadas utilizando-se a frequência de amostragem de 7680 Hz e o tempo de atualização foi ajustado para três segundos. Na Figura 54, se pode observar o monitoramento das formas de onda trifásicas de tensão e corrente utilizando o modo de operação com atualização automática.

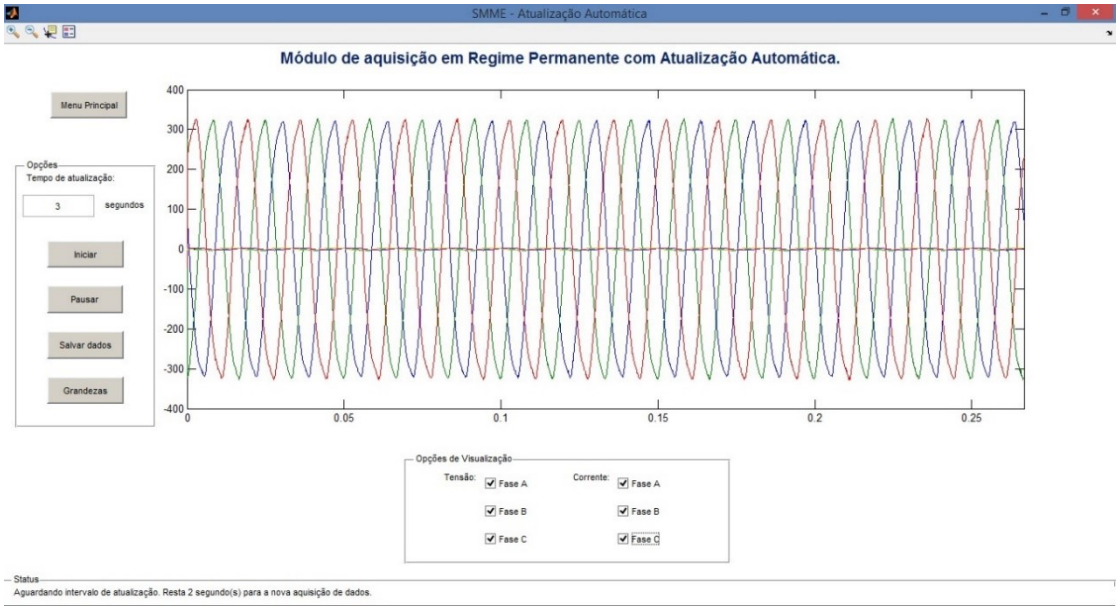

Figura 54 - Modo de operação com atualização automática em funcionamento com frequência de amostragem de 7680 Hz e tempo de atualização de 3 segundos.
Fonte: Autoria própria.

Realizando o monitoramento nestas condições, as funções deste modo de operação obtiveram um funcionamento satisfatório.

4.3 MODO DE OPERAÇÃO EM REGIME TRANSITÓRIO

Após a avaliação do modo de operação com atualização automática, foi realizado o teste com a interface desenvolvida para aquisição dos sinais das máquinas elétricas em regime transitório. Para isso o MIT foi ligado em vazio e posteriormente com carga nominal. Como já mencionado, para que a aquisição seja realizada, a placa de processamento monitora os valores de corrente da fase A, detectando o momento em que a máquina é energizada. Quando estes valores deixam o intervalo predefinido, a aquisição é iniciada.

Primeiramente o MIT foi acionado sem carga no eixo, sendo que a frequência de amostragem utilizada para realizar o monitoramento foi de 1920 Hz. Na Figura 55 se pode observar as correntes trifásicas no período transitório.

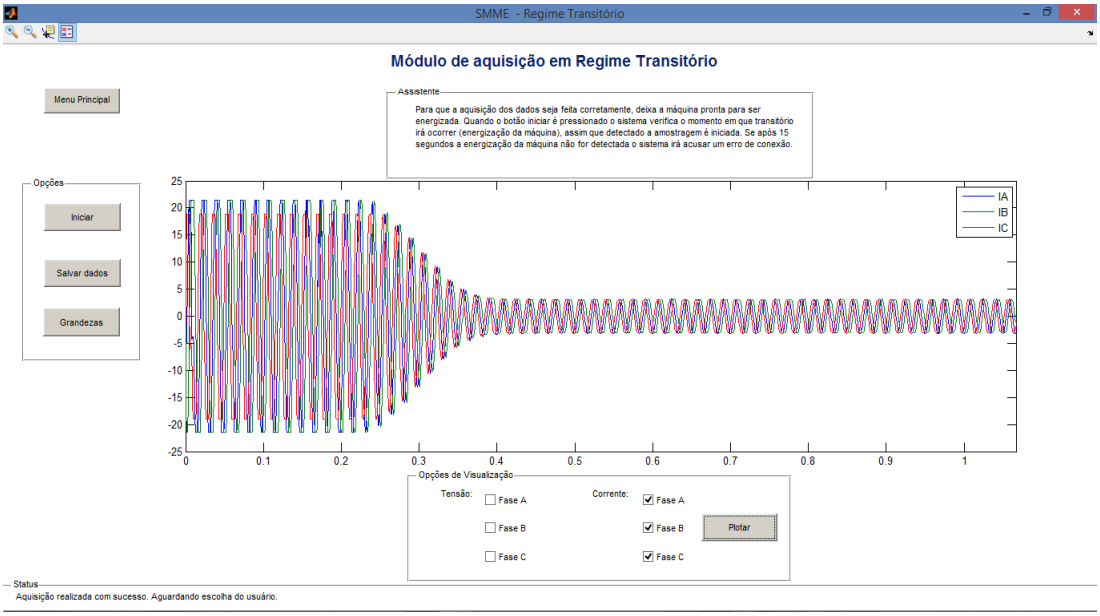

Figura 55 – Interface de regime transitório em funcionamento apresentando as correntes trifásicas no período transitório com frequência de amostragem de 1920 Hz.
Fonte: Autoria própria.

Observa-se, que os valores de corrente saturaram no início da aquisição, isto ocorre pelo fato destes valores ultrapassarem os valores nominais do sistema. Porém, nota-se uma falta de calibração do sistema de aquisição de corrente, visto que o sensor possui faixa de operação até 20 A RMS, que possuiria um valor de pico maior que os valores que saturaram.

Já na Figura 56 se observa a tensão e a corrente da fase A para o mesmo período de aquisição. Também a possível observar a variação que o sinal de tensão sofre no período transiente.

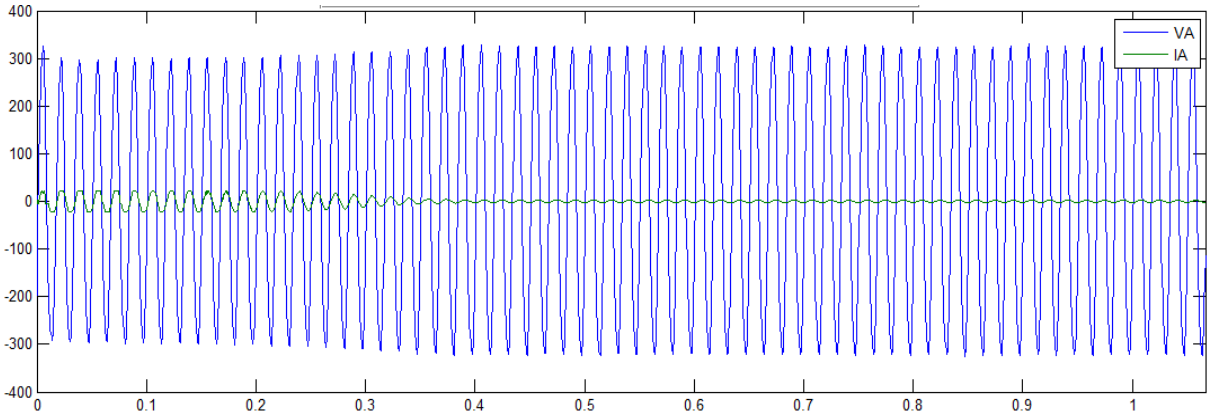

Figura 56 - Tensão e corrente da fase A no período transitório com frequência de amostragem de 1920 Hz.
Fonte: Autoria própria.

A frequência de amostragem foi alterada para 480 Hz, sendo as formas de onda resultantes das correntes exibidas na Figura 57.

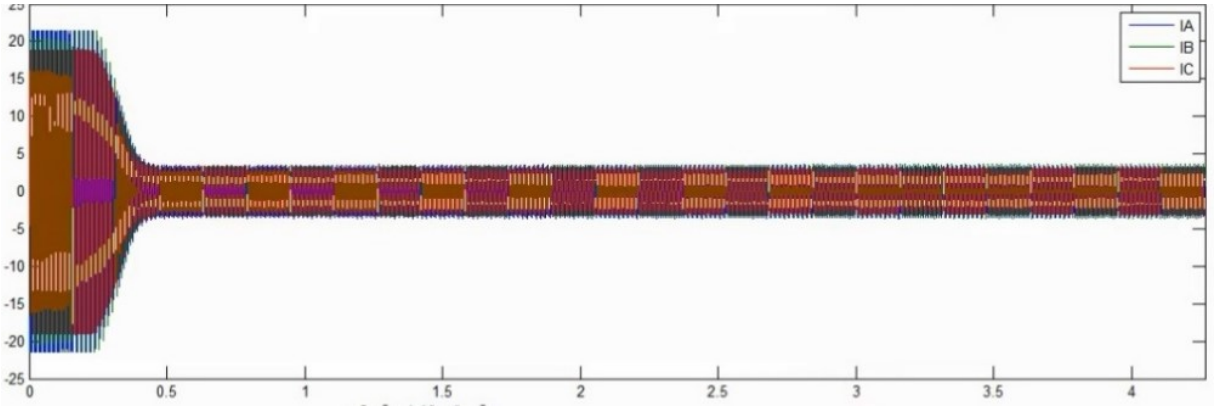

Figura 57 - Correntes trifásicas no período transitório com frequência de amostragem de 480 Hz.
Fonte: Autoria própria.

Observa-se que o período transiente da máquina termina antes de 0,5 segundos, sendo que as amplitudes das correntes não passam de 3 A quando a máquina atinge o regime permanente.

Posteriormente, foi aplicado ao eixo do MIT uma carga com a grandeza de seu torque nominal. A Figura 58 mostra as formas de onda de corrente que foram monitoradas.

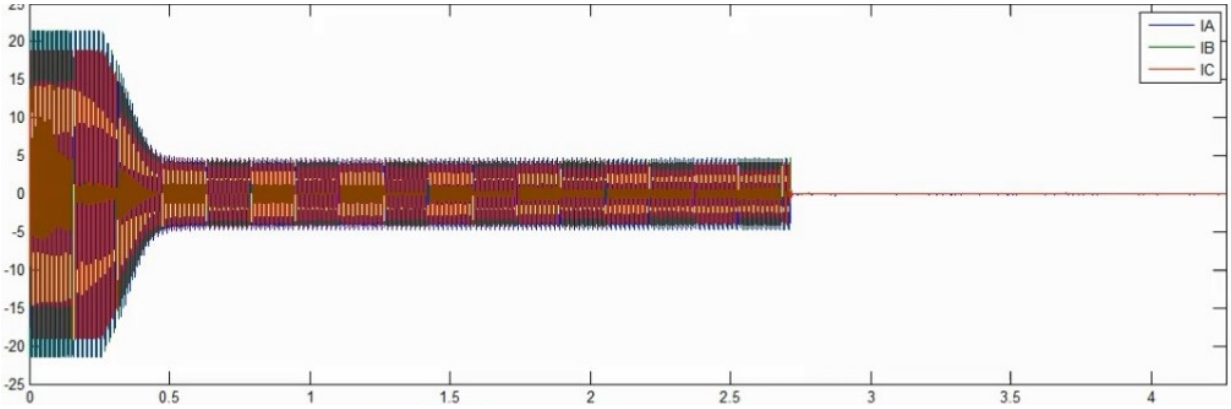

Figura 58 - Correntes trifásicas no período transitório com frequência de amostragem de 480 Hz, com torque nominal aplicado ao eixo do MIT.
Fonte: Autoria própria.

Observa-se que o período transiente da máquina termina aproximadamente em 0,5 segundos, sendo que as amplitudes das correntes chegam a 5 A quando a máquina atinge o regime permanente.

Comparando-se a Figura 57 e a Figura 58, é possível identificarmos através da grandeza da forma de onda e pela duração do período transitório, a diferença de carga existente quando a máquina está em vazio e com carga nominal no eixo.

Com a finalização do teste da interface de regime transitório, verificou-se que todos os sistemas funcionaram conforme projetado, de forma satisfatória e conforme suas limitações.

CONCLUSÃO

Neste trabalho foi apresentado o projeto e montagem de um sistema de monitoramento de máquinas elétricas via TCP/IP. Os parâmetros de interesse para realização do monitoramento são os sinais de tensão e corrente que realizam a energização das máquinas. Através das informações contidas nestes sinais é possível obter uma série de informações que são importantes para realizar a análise do estado de conservação físico das partes elétricas e mecânicas das máquinas, bem como realizar o controle e supervisão das mesmas.

Para montagem de todo o sistema, foram utilizados os materiais que melhor se adaptaram à aplicação com relação ao custo e as características necessárias ao trabalho.

Para que fosse possível efetuar o sensoriamento dos sinais de tensão, foram utilizados três transformadores de baixa potência para diminuir a magnitude destes sinais, resultando em uma amostra fiel reduzida do sinal original. Com o mesmo intuito, utilizou-se três TCs para aquisição dos sinais de corrente. Os dois tipos de equipamentos mencionados se mostraram adequados a aplicação após a realização de ensaios com os mesmos, ainda, verificou-se que possuem um baixo custo de aquisição, sendo adequados ao trabalho.

Para realizar a amostragem dos sinais, foi necessário a aplicação de um conversor A/D, sendo que este elemento estava incluso em uma placa de processamento adquirida para também efetuar a transmissão via protocolo TCP/IP dos dados amostrados, o modelo utilizado foi o Cerebot Mx7, do fabricante Digilent.

O conversor A/D é configurado para realizar a amostragem dos sinais dentro de uma faixa limitada de operação. Devido a isto, foi necessário desenvolver um circuito capaz de realizar o condicionamento dos sinais vindos dos sensores, para que estes fossem adequados a faixa de operação necessária. O circuito desenvolvido baseou-se na aplicação de elementos simples, tais como resistores, capacitores e *trimmers*, sendo este capaz de se adaptar diferenças que por ventura possam ocorrer nos valores dos sensores, compensando desvios para mais ou para menos nos sinais.

Com o condicionamento dos sinais, foi possível realizar a amostragem com o conversor A/D da placa de processamento. Assim, foi necessário definir as configurações pelas quais os sinais seriam aquisitados, entre estas configurações

estão o número de amostras e a frequência de amostragem. O número de amostras configurado foi de 2048 amostras por sinal, devido a limitação impostas pelas funções utilizadas no código implementado na placa de processamento. Já para a frequência de amostragem as opções disponíveis são de 480, 960, 1920, 3840, 7680, 15360 e 30720 Hz.

Com os dados aquisitados e armazenados, é preciso que os mesmos sejam transmitidos pela rede via protocolo TCP/IP. Para isso foi necessário realizar as configurações de rede na placa de processamento. Os dados configurados foram o endereço MAC, o IP e a porta pela qual a comunicação ocorrerá. Dependendo das características da rede em que o sistema será aplicado, os valores devem ser alterados no código gravado na placa de processamento para o correto funcionamento.

Para efetuar a visualização e controle das aquisições, foram elaboradas interfaces no *software* Matlab. Esta interface é responsável por realizar o controle das aquisições feitas pela placa de processamento, por plotar os sinais aquisitados pelo sistema, pela visualização de dados resumidos dos sinais e pela calibração do sistema de aquisição. Ainda, foram desenvolvidos três modos de operação, sendo eles, o de operação em regime permanente, em regime transitório e com atualização automática.

O modo de operação em regime permanente, foi desenvolvido para realizar a aquisição dos sinais quando a máquina está trabalhando de forma contínua e tem pouca ou nenhuma variação em seu funcionamento. O usuário do sistema determina quando deseja realizar a aquisição dos sinais, e então, a placa de processamento transmite os dados pela rede.

O modo de operação em regime transitório, foi desenvolvido para realizar a aquisição dos sinais quando a máquina sofre bruscas variações em seu funcionamento, geralmente nas energizações e em mudanças bruscas de carga. Este modo foi desenvolvido para amostrar os dados do período transitório decorrente da energização das máquinas. Para isso, o usuário determina o momento em que deseja realizar a aquisição, porém a amostragem só tem início quando o sistema detecta a energização da máquina, iniciando a aquisição e posteriormente transmissão dos dados.

No modo de operação com atualização automática, a realização da aquisição dos sinais acontece de acordo com o intervalo de tempo, em segundos, que é

escolhido pelo usuário. Assim, quando os dados são atualizados os mesmos são plotados para visualização, e quando o tempo de atualização é esgotado, uma nova aquisição é realizada.

Com o desenvolvimento dos sistemas mencionados anteriormente, os objetivos do trabalho foram atingidos de forma satisfatória, sendo possível realizar o monitoramento das grandezas elétricas de máquinas elétricas via protocolo TCP/IP. A grande desvantagem do trabalho decorre dos sensores de tensão e corrente empregados. O sensor de corrente é pouco sensível para correntes com valores baixos, e o sensor de tensão possui uma característica de não-linearidade, o que pode causar um erro se forem utilizados valores de tensão para alimentação das máquinas diferentes dos previstos.

Para trabalhos futuros, sugere-se melhorar a interface desenvolvida no *software* Matlab, agregando ferramentas mais específicas, como por exemplo, um módulo que possa utilizar os dados aquisitados para verificar as falhas presentes na máquina monitorada. Outra sugestão é o uso de um sistema de sensores e condicionadores de sinais com características superiores aos implementados no trabalho, visto as limitações que os elementos empregados possuem. Ainda, poderia ser acrescentado no modo de operação com atualização automática uma opção para salvar os dados aquisitados, de modo a montar um banco de dados sobre o funcionamento da máquina.

REFERÊNCIAS

AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA. **Relatório de compatibilidade dos equipamentos de baixa tensão existentes no mercado aos diversos níveis de tensão nominal de distribuição secundária**, 2011. Disponivel em: <http://www.aneel.gov.br/>. Acesso em: 17 Outubro 2014.

ALEXANDER, C. K.; SADIKU, M. N. O. **Fundamentos de Circuitos Elétricos**. 3. ed. Porto Alegre: AMGH, 2008.

BALBINOT, A.; BRUSAMARELLO, V. J. **Instrumentação e Fundamentos de Medidas**. 2ª. ed. Rio de Janeiro: LTC, 2011.

CHAPMAN, S. J. **Fundamentos de Máquinas Elétricas**. 5. ed. Porto Alegre: AMGH, 2013.

COMER, D. E. **Interligação de redes com TCP/IP**. 5. ed. Rio de Janeiro: Elsevier, v. 1, 2006.

DIGILENT INC. **chipKIT Cerebot Mx7**, 2015. Disponivel em: <https://www.digilentinc.com/>. Acesso em: 21 Janeiro 2015.

ELETRODEX. **Eletrodex Eletrônica**, 2015. Disponivel em: <http://www.eletrodex.com.br/>. Acesso em: 28 Janeiro 2015.

GONGORA, W. S. **Uma abordagem neural no diagnóstico de falhas em rolamentos de motores de indução trifásicos**. Universidade Tecnológica Federal do Paraná. Cornélio Procópio, p. 97. 2013.

MATLAB Documentation. **Mathworks**, 2015. Disponivel em: <http://www.mathworks.com/help/matlab/>. Acesso em: 15 maio 2015.

NATIONAL INTRUMENTS. **Placa de aquisição NI USB-6221**, 2014. Disponivel em: <http://sine.ni.com/nips/cds/view/p/lang/pt/nid/203481>. Acesso em: 2 Novembro 2014.

PROAKIS, J. G.; MANOLAKIS, D. G. **Digital Signal Processing:** Principles, Algorithms, and Applications. 3. ed. Upper Saddle River: Prentice-Hall, 1996.

RANIERI, F. **Sistema supervisório de parâmetros de máquinas elétricas via TCP/IP e painel eletrônico de mensagens**. Universidade de São Paulo. São Carlos. 2007.

SANTOS, T. H. **Estimador neural de velocidade aplicado a um driver de controle escalar do motor de indução trifasico**. Universidade Tecnológica Federal do Paraná. Cornélio Procópio. 2012.

TANENBAUM, A. S.; WETHERALL, D. **Redes de computadores**. 5. ed. São Paulo: Pearson Prentice Hall, 2011.

THOMAZINI, D.; ALBUQUERQUE, P. U. B. **Sensores Industriais:** fundamentos e aplicações. 8. ed. São Paulo: Érica, 2011.

TOCCI, R. J.; WIDMER, N. S.; MOSS, G. L. **Sistemas Digitais:** princípios e aplicações. 11. ed. São Paulo: Pearson Prentice Hall, 2011.

TORRES, G. **Redes de Computadores Curso Completo**. Edição Especial. ed. Rio de Janeiro: Axcel Books, 2001.

/

WANG, C. et al. An on-line distributed induction motor monitoring system based-on ARM and CAN bus. **Sixth International Symposium on Parallel Computing in Electrical Engineering**, 2011.

APÊNDICE A – Rotina implementada na placa de processamento

```
%//****-->BIBLIOTECA UTILIZADA<---****//
#include <chipKITEthernet.h>
#include <Timer.h>
#include <sys/attribs.h>

%//****-->VARIÁVEIS DO SISTEMA<---****//
byte mac[] = {
  0x00, 0x18, 0x3e, 0x01, 0x83, 0x32 };
byte ip[] = {
  10,20,12,240 };
int c = 0;
Server server(44300);
volatile bool samplesReady = false;
volatile uint16_t t = 0;
unsigned short int V1[2047], V2[2047], V3[2047], I1[2047], I2[2047],
I3[2047];
Timer4 timer;

void __attribute__((interrupt)) isr()
{
  V1[t] = analogRead(0);
  V2[t] = analogRead(1);
  V3[t] = analogRead(2);
  I1[t] = analogRead(4);
  I2[t] = analogRead(5);
  I3[t] = analogRead(6);
  t++;
  if(t>2047)
  {
    timer.stop();
    samplesReady = true;
    t=0;
  }
  clearIntFlag(_TIMER_4_IRQ);
}


%//****-->INICIALIZAÇÕES INICIAIS<---****//
void setup()
{
  pinMode(0,INPUT);
  pinMode(1,INPUT);
  pinMode(2,INPUT);
  pinMode(4,INPUT);
  pinMode(5,INPUT);
  pinMode(6,INPUT);
  timer.setFrequency(7680);
  timer.attachInterrupt(isr);
  Ethernet.begin(mac, ip);
  server.begin();
}

void loop()
{
  Client client = server.available();
```

```
if (client)
{
  while (client.connected())
  {
    if (client.available())
    {
      c = client.read();
      if (c == 11)
      {
        timer.setFrequency(480);
        timer.start();
        while(!samplesReady){
        };
        enviar_dados(client);
      }
      else if (c == 12)
      {
        timer.setFrequency(960);
        timer.start();
        while(!samplesReady){
        };
        enviar_dados(client);
      }
      else if (c == 13)
      {
        timer.setFrequency(1920);
        timer.start();
        while(!samplesReady){
        };
        enviar_dados(client);
      }
      else if (c == 14)
      {
        timer.setFrequency(3840);
        timer.start();
        while(!samplesReady){
        };
        enviar_dados(client);
      }
      else if (c == 15)
      {
        timer.setFrequency(7680);
        timer.start();
        while(!samplesReady){
        };
        enviar_dados(client);
      }
      else if (c == 16)
      {
        timer.setFrequency(15360);
        timer.start();
        while(!samplesReady){
        };
        enviar_dados(client);
      }
      else if (c == 17)
      {
        timer.setFrequency(30720);
        timer.start();
        while(!samplesReady){
        };
```

```
  enviar_dados(client);
}
else if (c == 21)
{
  analogRead(4);
  delay(1);
  while(analogRead(4) >= 462 && analogRead(4) <= 562){
  };
  timer.setFrequency(480);
  timer.start();
  while(!samplesReady){
  };
  enviar_dados(client);
}
else if (c == 22)
{
  analogRead(4);
  delay(1);
  while(analogRead(4) >= 462 && analogRead(4) <= 562){
  };
  timer.setFrequency(960);
  timer.start();
  while(!samplesReady){
  };
  enviar_dados(client);
}
else if (c == 23)
{
  analogRead(4);
  delay(1);
  while(analogRead(4) >= 462 && analogRead(4) <= 562){
  };
  timer.setFrequency(1920);
  timer.start();
  while(!samplesReady){
  };
  enviar_dados(client);
}
else if (c == 24)
{
  analogRead(4);
  delay(1);
  while(analogRead(4) >= 462 && analogRead(4) <= 562){
  };
  timer.start();
  timer.setFrequency(3840);
  while(!samplesReady){
  };
  enviar_dados(client);
}
else if (c == 25)
{
  while(analogRead(4) >= 462 && analogRead(4) <= 562){
  };
  timer.setFrequency(7680);
  timer.start();
  while(!samplesReady){
  };
  enviar_dados(client);
}
else if (c == 26)
```

```
        {
          analogRead(4);
          delay(1);
          while(analogRead(4) >= 462 && analogRead(4) <= 562){
          };
          timer.setFrequency(15360);
          timer.start();
          while(!samplesReady){
          };
          enviar_dados(client);
        }
        else if (c == 27)
        {
          analogRead(4);
          delay(1);
          while(analogRead(4) >= 462 && analogRead(4) <= 562){
          };
          timer.setFrequency(30720);
          timer.start();
          while(!samplesReady){
          };
          enviar_dados(client);
        }
        else if (c == 9)
        {
          teste_conexao(client);
        }
        c=0;
        break;
      }
    }
  }
  client.stop();
}

void teste_conexao (Client client)
{
  client.print(1);
  client.println();
}

void enviar_dados (Client client)
{
  char buffer[100];
  for(int j = 0; j < 2048; j++)
  {
    sprintf (buffer, "%d %d %d %d %d %d", V1[j], V2[j], V3[j], I1[j],
I2[j], I3[j]);
    client.print(buffer);
    client.println();
    delay(1);
  }
}
```

APÊNDICE B – Código das interfaces desenvolvidas

Interface de inicialização

```
% --- FUNÇÃO PARA INICIALIZAÇÃO DA INTERFACE -> NÃO EDITAR <-
function varargout = INTERFACE_NOVO(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @INTERFACE_NOVO_OpeningFcn, ...
                   'gui_OutputFcn',  @INTERFACE_NOVO_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function INTERFACE_NOVO_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
global FREQAMOS np const1 const2 const3 const4 const5 const6
FREQAMOS = 5;
np = 16;
load('CLBC.txt');
const1 = CLBC(1,4);
const2 = CLBC(2,4);
const3 = CLBC(3,4);
const4 = CLBC(4,4);
const5 = CLBC(5,4);
const6 = CLBC(6,4);
guidata(hObject, handles);
guidata(handles.output,handles);

function varargout = INTERFACE_NOVO_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
guidata(handles.output,handles);

function ip_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function porta_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function painel_status_CreateFcn(hObject, eventdata, handles)

function ip_DeleteFcn(hObject, eventdata, handles)

function a_CreateFcn(hObject, eventdata, handles)

function ip_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles);

function porta_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles);

function texto_status_CreateFcn(hObject, eventdata, handles)
handles = guihandles(gcbo);
set(handles.texto_status,'String','Aguardando usuário.');
guidata(hObject, handles);

% --- Executa o teste de conexão com o IP e a PORTA informados.
function confirmar_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
set(handles.texto_status,'String','Relizando teste de conexão com o módulo
de comunicação.');
pause(0.5)
global IP PORTA
IP = get(handles.ip,'String');
PORTA = str2num(get(handles.porta,'String'));
try
set(handles.texto_status,'String','Conectando-se ao módulo de
comunicação.');
pause(0.5)
cerebot = tcpip(IP, PORTA, 'Timeout', 3);
fopen(cerebot);
flushinput(cerebot);
fwrite(cerebot,9);
a = fscanf(cerebot, '%f')';
fclose(cerebot);
cerebot=0;
if a == 1
set(handles.reg_permanente,'enable','on');
set(handles.reg_transitorio,'enable','on');
set(handles.atu_auto,'enable','on');
set(handles.calibracao,'enable','on');
set(handles.texto_status,'String','Comunicação realizada com sucesso.
Selecione um módulo de aquisição disponível.');
pause(0.5)
else
    set(handles.texto_status,'String','Não foi possível estabelecer conexão
com o módulo de comunicação. Verifique as conexões.');
    pause(0.5)
end
catch
    set(handles.texto_status,'String','Não foi possível estabelecer conexão
com o módulo de comunicação. Verifique as conexões.');
    pause(0.5)
end

% --- Executa o módulo de aquisição em regime permanente.
```

```
function reg_permanente_Callback(hObject, eventdata, handles)
REGIME_PERMANENTE;
delete(handles.PRINCIPAL);

% --- Executa o módulo de aquisição em regime transitório.
function reg_transitorio_Callback(hObject, eventdata, handles)
REGIME_TRANSITORIO;
delete(handles.PRINCIPAL);

% --- Executa o módulo de aquisição em atualização automática.
function atu_auto_Callback(hObject, eventdata, handles)
REGIME_AUTOMATICO;
delete(handles.PRINCIPAL);


function Freq_SelectionChangeFcn(hObject, eventdata, handles)
handles = guihandles(gcbo);
global FREQAMOS np
get(get(handles.Freq,'SelectedObject'),'Tag');
switch get(get(handles.Freq,'SelectedObject'),'Tag')
    case 'a'
        FREQAMOS = 1;
        np = 256;
    case 'b'
        FREQAMOS = 2;
        np = 128;
    case 'c'
        FREQAMOS = 3;
        np = 64;
    case 'd'
        FREQAMOS = 4;
        np = 32;
    case 'e'
        FREQAMOS = 5;
        np = 16;
    case 'f'
        FREQAMOS = 6;
        np = 8;
    case 'g'
        FREQAMOS = 7;
        np = 4;
end
guidata(hObject, handles);

function Lig_SelectionChangeFcn(hObject, eventdata, handles)
handles = guihandles(gcbo);
global const1 const2 const3 const4 const5 const6
get(get(handles.Lig,'SelectedObject'),'Tag');
switch get(get(handles.Lig,'SelectedObject'),'Tag')
    case 'h'
        const1 = CLBC(1,1);
        const2 = CLBC(2,1);
        const3 = CLBC(3,1);
        const4 = CLBC(4,1);
        const5 = CLBC(5,1);
        const6 = CLBC(6,1);
    case 'i'
        const1 = CLBC(1,2);
        const2 = CLBC(2,2);
        const3 = CLBC(3,2);
```

```
        const4 = CLBC(4,2);
        const5 = CLBC(5,2);
        const6 = CLBC(6,2);
    case 'g'
        const1 = CLBC(1,3);
        const2 = CLBC(2,3);
        const3 = CLBC(3,3);
        const4 = CLBC(4,3);
        const5 = CLBC(5,3);
        const6 = CLBC(6,3);
    case 'j'
        const1 = CLBC(1,4);
        const2 = CLBC(2,4);
        const3 = CLBC(3,4);
        const4 = CLBC(4,5);
        const5 = CLBC(5,4);
        const6 = CLBC(6,4);
end
guidata(hObject, handles);

function axes3_CreateFcn(hObject, eventdata, handles)
axes(hObject)
imshow('logo UTFPR.jpg');

function calibracao_Callback(hObject, eventdata, handles)
CALIBRACAO_
delete(handles.PRINCIPAL);
```

Interface de calibração

```
function varargout = CALIBRACAO_(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @CALIBRACAO__OpeningFcn, ...
                   'gui_OutputFcn',  @CALIBRACAO__OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function CALIBRACAO__OpeningFcn(hObject, eventdata, handles, varargin)

global DADOS CAL
CAL = 1;
DADOS = zeros(2048,6);
```

```
handles.output = hObject;

guidata(hObject, handles);

function varargout = CALIBRACAO__OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;

function pushbutton2_Callback(hObject, eventdata, handles)
INTERFACE_NOVO
close(CALIBRACAO_);

function checkbox1_Callback(hObject, eventdata, handles)

function checkbox3_Callback(hObject, eventdata, handles)

function Lig_SelectionChangeFcn(hObject, eventdata, handles)
handles = guihandles(gcbo);
global CAL
get(get(handles.Lig,'SelectedObject'),'Tag');
switch get(get(handles.Lig,'SelectedObject'),'Tag')
    case 'h'
        CAL = 1;

    case 'i'
        CAL = 2;

    case 'g'
        CAL = 4;

    case 'j'
        CAL = 3;

end
guidata(hObject, handles);

function calibrar_Callback(hObject, eventdata, handles)
CLBC = load('CLBC.txt');
handles = guihandles(gcbo);
global IP PORTA DADOS CAL
try
    set(handles.texto_status,'String','Calibrando...');
    drawnow;
    cerebot = tcpip(IP,PORTA);
    fopen(cerebot);
    fwrite(cerebot,15);
    i=1;
    while (i<=2048)
        valores = fscanf(cerebot, '%d');
        DADOS(i,:) = valores;
        i = i + 1;
    end
    fclose(cerebot);
    cerebot=0;
    DADOS = sort(DADOS);
    V1m = sum(DADOS(2033:2048,1))/16;
    V2m = sum(DADOS(2033:2048,2))/16;
    V3m = sum(DADOS(2033:2048,3))/16;
    I1m = sum(DADOS(2033:2048,4))/16;
```

```
    I2m = sum(DADOS(2033:2048,5))/16;
    I3m = sum(DADOS(2033:2048,6))/16;
    V1rms = str2num(get(handles.text7,'String'));
    V2rms = str2num(get(handles.text8,'String'));
    V3rms = str2num(get(handles.text9,'String'));
    I1rms = str2num(get(handles.text10,'String'));
    I2rms = str2num(get(handles.text11,'String'));
    I3rms = str2num(get(handles.text12,'String'));
    if get(handles.checkbox1,'Value') == 1
        CALIB(1,1) = (V1rms*sqrt(2))/(V1m*(3.3/1024)-(3.3/2));
        CALIB(2,1) = (V2rms*sqrt(2))/(V2m*(3.3/1024)-(3.3/2));
        CALIB(3,1) = (V3rms*sqrt(2))/(V3m*(3.3/1024)-(3.3/2));
        CLBC(1:3,CAL) = CALIB(1:3,1);
    end
    if get(handles.checkbox3,'Value') == 1
        CALIB(4,1) = (I1rms*sqrt(2))/(I1m*(3.3/1024)-(3.3/2));
        CALIB(5,1) = (I2rms*sqrt(2))/(I2m*(3.3/1024)-(3.3/2));
        CALIB(6,1) = (I3rms*sqrt(2))/(I3m*(3.3/1024)-(3.3/2));
        CLBC(4:6,CAL) = CALIB(4:6,1);
    end
    save ( 'CLBC.txt','CLBC', '-ascii');
    set(handles.texto_status,'String','As opções escolhidas foram
calibradas.');
catch
    set(handles.texto_status,'String','Não foi possível estabelecer conexão
com o módulo de comunicação. Verifique as conexões.');
    pause(0.5)
end
guidata(hObject, handles)

function texto_status_CreateFcn(hObject, eventdata, handles)
handles = guihandles(gcbo);
set(handles.texto_status,'String','Aguardando usuário.');
guidata(hObject, handles);
```

Interface de aquisição em regime permanente

```
% --- FUNÇÃO PARA INICIALIZAÇÃO DA INTERFACE -> NÃO EDITAR <-
function varargout = REGIME_PERMANENTE(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @REGIME_PERMANENTE_OpeningFcn, ...
                   'gui_OutputFcn',  @REGIME_PERMANENTE_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
```

```
function REGIME_PERMANENTE_OpeningFcn(hObject, eventdata, handles,
varargin)
handles.output = hObject;
guidata(hObject, handles);

function varargout = REGIME_PERMANENTE_OutputFcn(hObject, eventdata,
handles)
varargout{1} = handles.output;
set(gcf, 'units','normalized','outerposition',[0 0 1 1]);

function TA_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function TB_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function TC_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function CA_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function CB_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function CC_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

% --- Realiza a aquisição dos dados.
function Atualizar_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
global IP PORTA DADOS FREQAMOS const1 const2 const3 const4 const5 const6
DADOS = zeros(2048,6);
try
    set(handles.Status,'String','Conectando-se ao módulo de comunicação.');
    drawnow;
    cerebot = tcpip(IP,PORTA);
    fopen(cerebot);
catch
    set(handles.Status,'String','Não foi possíve estabelecer conexão com o
módulo de comunicação. Verifique as conexões e tente novamente.');
    drawnow;
    return
end
set(handles.Status,'String','Conexão estabelecida, realizando aquisição dos
dados.');
drawnow;
try
    start = FREQAMOS + 10;
    fwrite(cerebot,start);
    i=1;
    while (i<=2048)
        valores = fscanf(cerebot, '%d');
```

```
        DADOS(i,:) = valores;
        i = i + 1;
        p = (i/2048)*100;
        P = sprintf('Conexão estabelecida, realizando aquisição dos dados.(
%.f%%)', p);
        set(handles.Status,'String',P);
        drawnow;
    end
    fclose(cerebot);
    cerebot=0;
    DADOS = DADOS*(3.33/1024)-(3.33/2);
    DADOS(:,1) = DADOS(:,1)*const1;
    DADOS(:,2) = DADOS(:,2)*const2;
    DADOS(:,3) = DADOS(:,3)*const3;
    DADOS(:,4) = DADOS(:,4)*const4;
    DADOS(:,5) = DADOS(:,5)*const5;
    DADOS(:,6) = DADOS(:,6)*const6;
    set(handles.Status,'String','Aquisição realizada com sucesso.
Aguardando escolha do usuário.');
    set(handles.Salvar,'enable','on');
    set(handles.Plotar,'enable','on');
    set(handles.Grandezas,'enable','on');
    set(handles.TA,'enable','on');
    set(handles.TB,'enable','on');
    set(handles.TC,'enable','on');
    set(handles.CA,'enable','on');
    set(handles.CB,'enable','on');
    set(handles.CC,'enable','on');
catch
    set(handles.Status,'String','Aquisição dos dados mal sucedida.
Verifique as conexões e tente novamente.');
end
guidata(hObject, handles)

% --- Voltar ao Menu Principal
function Menu_Principal_Callback(hObject, eventdata, handles)
INTERFACE_NOVO;
delete(handles.PERMANENTE);

% --- Escolha dos sinais para plotar o gráfico.
function Plotar_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
global DADOS np
dados_plot = zeros(2048,6);
if get(handles.TA,'Value') == 1
    V1 = DADOS(:,1);
    a = 'VA';
else
    V1 = [];
    a = [];
end
if get(handles.TB,'Value') == 1
    V2 = DADOS(:,2);
    b = 'VB';
else
    V2 = [];
    b = [];
end
if get(handles.TC,'Value') == 1
    V3 = DADOS(:,3);
    c = 'VC';
```

```
else
    V3 = [];
    c = [];
end
if get(handles.CA,'Value') == 1
    I1 = DADOS(:,4);
    d = 'IA';
else
    I1 = [];
    d = [];
end
if get(handles.CB,'Value') == 1
    I2 = DADOS(:,5);
    e = 'IB';
else
    I2 = [];
    e = [];
end
if get(handles.CC,'Value') == 1
    I3 = DADOS(:,6);
    f = 'IC';
else
    I3 = [];
    f = [];
end
dados_plot = [V1 V2 V3 I1 I2 I3];
plot(linspace(0,(np/60),2048),dados_plot);
xlim([0 (np/60)]);
legend([a;b;c;d;e;f]);
guidata(hObject, handles)

% --- Abre a interface de Grandezas
function Grandezas_Callback(hObject, eventdata, handles)
global FREQAMOS
if FREQAMOS == 1 || FREQAMOS ==2
    msgbox('Não disponível para esta frequência de amostragem.','Aviso');
else
    CALCULOS;
end

% --- Salva os dados aquisitados.
function Salvar_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
global DADOS
uisave('DADOS', 'SMME001.mat')
guidata(hObject, handles)
```

Interface de aquisição em regime transitório

```
% --- FUNÇÃO PARA INICIALIZAÇÃO DA INTERFACE -> NÃO EDITAR <-
function varargout = REGIME_TRANSITORIO(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @REGIME_TRANSITORIO_OpeningFcn, ...
                   'gui_OutputFcn',  @REGIME_TRANSITORIO_OutputFcn, ...
```

```
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function REGIME_TRANSITORIO_OpeningFcn(hObject, eventdata, handles,
varargin)
set(handles.Status,'String','Aguardando usuário.');
handles.output = hObject;
guidata(hObject, handles);
function varargout = REGIME_TRANSITORIO_OutputFcn(hObject, eventdata,
handles)
varargout{1} = handles.output;
set(gcf, 'units','normalized','outerposition',[0 0 1 1]);

function TA_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function TB_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function TC_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function CA_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function CB_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function CC_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

% --- Realiza a aquisição dos dados.
function Iniciar_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
global DADOS PORTA IP FREQAMOS const1 const2 const3 const4 const5 const6
DADOS = zeros(2048,6);
try
    set(handles.Status,'String','Conectando-se ao módulo de comunicação.');
    drawnow;
    cerebot = tcpip(IP, PORTA, 'Timeout', 15);
    fopen(cerebot);
    catch
    set(handles.Status,'String','Não foi possível estabelecer conexão com o
módulo de comunicação. Verifique as conexões e tente novamente.');
    drawnow;
    return
```

```
end
set(handles.Status,'String','Conexão estabelecida, siga as instruções
apresentadas pelo assistente.');
drawnow;
try
    start = FREQAMOS + 20;
    fwrite(cerebot,start);
    i=1;
    while (i<=2048)
        valores = fscanf(cerebot, '%d');
        DADOS(i,:) = valores;
        i = i + 1;
        p = (i/2048)*100;
        P = sprintf('Conexão estabelecida, realizando aquisição dos dados.(
%.f%%)', p);
        set(handles.Status,'String',P);
        drawnow;
    end
    fclose(cerebot);
    cerebot=0;
    DADOS = DADOS*(3.27/1024)-(3.27/2);
    DADOS(:,1) = DADOS(:,1)*const1;
    DADOS(:,2) = DADOS(:,2)*const2;
    DADOS(:,3) = DADOS(:,3)*const3;
    DADOS(:,4) = DADOS(:,4)*const4;
    DADOS(:,5) = DADOS(:,5)*const5;
    DADOS(:,6) = DADOS(:,6)*const6;
    set(handles.Status,'String','Aquisição realizada com sucesso.
Aguardando escolha do usuário.');
    set(handles.Salvar,'enable','on');
    set(handles.Plotar,'enable','on');
    set(handles.Grandezas,'enable','on');
    set(handles.TA,'enable','on');
    set(handles.TB,'enable','on');
    set(handles.TC,'enable','on');
    set(handles.CA,'enable','on');
    set(handles.CB,'enable','on');
    set(handles.CC,'enable','on');
catch
    set(handles.Status,'String','Aquisição dos dados mal sucedida.
Verifique as conexões e tente novamente.');
end
guidata(hObject, handles)

% --- Escolha dos sinais para plotar o gráfico.
function Plotar_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
global DADOS np
dados_plot = zeros(2048,6);
if get(handles.TA,'Value') == 1
    V1 = DADOS(:,1);
    a = 'VA';
else
    V1 = [];
    a = [];
end
if get(handles.TB,'Value') == 1
    V2 = DADOS(:,2);
    b = 'VB';
else
    V2 = [];
```

```
    b = [];
end
if get(handles.TC,'Value') == 1
    V3 = DADOS(:,3);
    c = 'VC';
else
    V3 = [];
    c = [];
end
if get(handles.CA,'Value') == 1
    I1 = DADOS(:,4);
    d = 'IA';
else
    I1 = [];
    d = [];
end
if get(handles.CB,'Value') == 1
    I2 = DADOS(:,5);
    e = 'IB';
else
    I2 = [];
    e = [];
end
if get(handles.CC,'Value') == 1
    I3 = DADOS(:,6);
    f = 'IC';
else
    I3 = [];
    f = [];
end
dados_plot = [V1 V2 V3 I1 I2 I3];
plot(linspace(0,(np/60),2048),dados_plot);
xlim([0 (np/60)]);
legend([a;b;c;d;e;f]);
guidata(hObject, handles)

% --- Salva os dados aquisitados.
function Salvar_Callback(hObject, eventdata, handles)
global DADOS
uisave('DADOS', 'SMME001.mat')

% --- Voltar ao Menu Principal
function Menu_Callback(hObject, eventdata, handles)
INTERFACE_NOVO;
delete(REGIME_TRANSITORIO);


% --- Executes on button press in Grandezas.
function Grandezas_Callback(hObject, eventdata, handles)
global FREQAMOS
if FREQAMOS == 1 || FREQAMOS ==2
    msgbox('Não disponível para esta frequência de amostragem.','Aviso');
else
    CALCULOS;
end
```

Interface de aquisição com atualização automática

```
% --- FUNÇÃO PARA INICIALIZAÇÃO DA INTERFACE -> NÃO EDITAR <-
function varargout = REGIME_AUTOMATICO(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @REGIME_AUTOMATICO_OpeningFcn, ...
                   'gui_OutputFcn',  @REGIME_AUTOMATICO_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function REGIME_AUTOMATICO_OpeningFcn(hObject, eventdata, handles,
varargin)
handles.output = hObject;
guidata(hObject, handles);

function varargout = REGIME_AUTOMATICO_OutputFcn(hObject, eventdata,
handles)
set(handles.Status,'String','Entre com o tempo desejado, escolha as fases
de tensões e correntes desejadas e inicie a atualização automática.');
varargout{1} = handles.output;
set(gcf, 'units','normalized','outerposition',[0 0 1 1]);

function edit3_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function TA_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
set(handles.Iniciar,'enable','on');
guidata(hObject, handles)

function TB_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
set(handles.Iniciar,'enable','on');
guidata(hObject, handles)

function TC_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
set(handles.Iniciar,'enable','on');
guidata(hObject, handles)

function CA_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
set(handles.Iniciar,'enable','on');
guidata(hObject, handles)
```

```
function CB_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
set(handles.Iniciar,'enable','on');
guidata(hObject, handles)

function CC_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
set(handles.Iniciar,'enable','on');
guidata(hObject, handles)

function edit3_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function Status_CreateFcn(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

% --- Executado quando botão iniciar é ativado.
function Iniciar_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
global DADOS TA PORTA IP A FREQAMOS np const1 const2 const3 const4 const5
const6
DADOS = zeros(2048,6);
TA = get(handles.edit3,'String');
TA = str2num(TA);
A=0;
cerebot = tcpip(IP,PORTA);
while(1)
    try
        fopen(cerebot);
    catch
        set(handles.Status,'String','Não foi possíve estabelecer conexão
com o módulo de comunicação. Verifique as conexões e tente novamente.');
        drawnow;
        break
    end
    set(handles.Status,'String','Conexão estabelecida, realizando aquisição
dos dados.');
    drawnow;
    try
        start = FREQAMOS + 10;
        fwrite(cerebot,start);
        i=1;
        while (i<=2048)
            valores = fscanf(cerebot, '%d');
            DADOS(i,:) = valores;
            i = i + 1;
            p = (i/2048)*100;
            P = sprintf('Conexão estabelecida, realizando aquisição dos
dados.( %.f%%)', p);
            set(handles.Status,'String',P);
            drawnow;
        end
        fclose(cerebot);
        DADOS = DADOS*(3.27/1024)-(3.27/2);
        DADOS(:,1) = DADOS(:,1)*const1;
        DADOS(:,2) = DADOS(:,2)*const2;
        DADOS(:,3) = DADOS(:,3)*const3;
```

```
        DADOS(:,4) = DADOS(:,4)*const4;
        DADOS(:,5) = DADOS(:,5)*const5;
        DADOS(:,6) = DADOS(:,6)*const6;
        set(handles.Status,'String','Aquisição realizada com sucesso.');
        set(handles.Pausar,'enable','on');
        dados_plot = zeros(2048,6);
        if get(handles.TA,'Value') == 1
            V1 = DADOS(:,1);
        else
            V1 = [];
        end
        if get(handles.TB,'Value') == 1
            V2 = DADOS(:,2);
        else
            V2 = [];
        end
        if get(handles.TC,'Value') == 1
            V3 = DADOS(:,3);
        else
            V3 = [];
        end
        if get(handles.CA,'Value') == 1
            I1 = DADOS(:,4);
        else
            I1 = [];
        end
        if get(handles.CB,'Value') == 1
            I2 = DADOS(:,5);
        else
            I2 = [];
        end
        if get(handles.CC,'Value') == 1
            I3 = DADOS(:,6);
        else
            I3 = [];
        end
        dados_plot = [V1 V2 V3 I1 I2 I3];
        plot(linspace(0,(np/60),2048),dados_plot);
        xlim([0 (np/60)]);
        ps = 0;
        while( ps < TA)
            restante = TA - ps;
            P = sprintf('Aguardando intervalo de atualização. Resta %.f
segundo(s) para a nova aquisição de dados.', restante);
            set(handles.Status,'String',P);
            drawnow;
            pause(1);
            ps = ps + 1;
        if A==1;
            break;
        end
        end
        if A==1;
            break;
        end
    catch
        set(handles.Status,'String','Aquisição dos dados mal sucedida.
Verifique as conexões e tente novamente.');
        break
    end
end
```

```
cerebot = 0;

% --- Voltar ao Menu Principal
function Menu_Callback(hObject, eventdata, handles)
INTERFACE_NOVO;
delete(handles.AUTOMATICO);

% ---- Pausa a atualização automática
function Pausar_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
global A
A=1;
set(handles.Pausar,'enable','off');
set(handles.Iniciar,'enable','on');
set(handles.Salvar,'enable','on');
set(handles.Grandezas,'enable','on');
guidata(hObject, handles)
set(handles.Status,'String','Aguardando usuário.');
return

% --- Salvar dados
function Salvar_Callback(hObject, eventdata, handles)
global DADOS
uisave('DADOS', 'SMME001.mat')


% --- Executes on button press in Grandezas.
function Grandezas_Callback(hObject, eventdata, handles)
global FREQAMOS
if FREQAMOS == 1 || FREQAMOS ==2
    msgbox('Não disponível para esta frequência de amostragem.','Aviso');
else
    CALCULOS;
end
```

Interface de grandezas

```
function varargout = CALCULOS(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @CALCULOS_OpeningFcn, ...
                   'gui_OutputFcn',  @CALCULOS_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
```

```
end

function CALCULOS_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
global DADOS agreg ang Max Min RMS RMS1 DHT FREQAMOS
switch FREQAMOS
    case 3
        T = 32;
        P = 2048/T;

    case 4
        T = 64;
        P = 2048/T;

    case 5
        T = 128;
        P = 2048/T;

    case 6
        T = 256;
        P = 2048/T;

    case 7
        T = 512;
        P = 2048/T;
end
t_ini = 1;
t_fim = T;
for i = 1 : 1 : P
    F1 = fft(DADOS(t_ini:t_fim,1),T);
    F2 = fft(DADOS(t_ini:t_fim,2),T);
    F3 = fft(DADOS(t_ini:t_fim,3),T);
    F4 = fft(DADOS(t_ini:t_fim,4),T);
    F5 = fft(DADOS(t_ini:t_fim,5),T);
    F6 = fft(DADOS(t_ini:t_fim,6),T);
    fcc1 = F1(1)/T;
    fcc2 = F2(1)/T;
    fcc3 = F3(1)/T;
    fcc4 = F4(1)/T;
    fcc5 = F5(1)/T;
    fcc6 = F6(1)/T;
    fh1 = 2*F1(2:end)/T;
    fh2 = 2*F2(2:end)/T;
    fh3 = 2*F3(2:end)/T;
    fh4 = 2*F4(2:end)/T;
    fh5 = 2*F5(2:end)/T;
    fh6 = 2*F6(2:end)/T;
    fhrms1 = abs(fh1(1:16)/sqrt(2));
    fhrms2 = abs(fh2(1:16)/sqrt(2));
    fhrms3 = abs(fh3(1:16)/sqrt(2));
    fhrms4 = abs(fh4(1:16)/sqrt(2));
    fhrms5 = abs(fh5(1:16)/sqrt(2));
    fhrms6 = abs(fh6(1:16)/sqrt(2));
    frms1 =[fcc1;fhrms1];
    frms2 =[fcc2;fhrms2];
    frms3 =[fcc3;fhrms3];
    frms4 =[fcc4;fhrms4];
    frms5 =[fcc5;fhrms5];
    frms6 =[fcc6;fhrms6];
    fhh1 = [fcc1;fh1];
```

```
    fhh2 = [fcc2;fh2];
    fhh3 = [fcc3;fh3];
    fhh4 = [fcc4;fh4];
    fhh5 = [fcc5;fh5];
    fhh6 = [fcc6;fh6];
    mod1(:,i) = fhh1(1:16);
    mod2(:,i) = fhh2(1:16);
    mod3(:,i) = fhh3(1:16);
    mod4(:,i) = fhh4(1:16);
    mod5(:,i) = fhh5(1:16);
    mod6(:,i) = fhh6(1:16);
    t_ini = t_fim+1;
    t_fim = t_fim+T;
end
agreg1 = zeros(16,1);
agreg2 = zeros(16,1);
agreg3 = zeros(16,1);
agreg4 = zeros(16,1);
agreg5 = zeros(16,1);
agreg6 = zeros(16,1);

for i = 1 : 1 : P
    agreg1 = agreg1 + mod1(:,i).^2;
    agreg2 = agreg2 + mod2(:,i).^2;
    agreg3 = agreg3 + mod3(:,i).^2;
    agreg4 = agreg4 + mod4(:,i).^2;
    agreg5 = agreg5 + mod5(:,i).^2;
    agreg6 = agreg6 + mod6(:,i).^2;
end
agreg1 = sqrt(agreg1/(2*length(DADOS)/T));
agreg2 = sqrt(agreg2/(2*length(DADOS)/T));
agreg3 = sqrt(agreg3/(2*length(DADOS)/T));
agreg4 = sqrt(agreg4/(2*length(DADOS)/T));
agreg5 = sqrt(agreg5/(2*length(DADOS)/T));
agreg6 = sqrt(agreg6/(2*length(DADOS)/T));

%ANGULOS
ang1 = rad2deg(angle(agreg1));
ang2 = rad2deg(angle(agreg2));
ang3 = rad2deg(angle(agreg3));
ang4 = rad2deg(angle(agreg4));
ang5 = rad2deg(angle(agreg5));
ang6 = rad2deg(angle(agreg6));

agreg1 = abs(agreg1);
agreg2 = abs(agreg2);
agreg3 = abs(agreg3);
agreg4 = abs(agreg4);
agreg5 = abs(agreg5);
agreg6 = abs(agreg6);

%DISTORÇÃO HARMÔNICA
dht1=0;
dht2=0;
dht3=0;
dht4=0;
dht5=0;
dht6=0;
for i = 3 : 1 : 16
    dht1 = dht1 + agreg1(i)^2;
    dht2 = dht2 + agreg1(i)^2;
```

```
    dht3 = dht3 + agreg1(i)^2;
    dht4 = dht4 + agreg1(i)^2;
    dht5 = dht5 + agreg1(i)^2;
    dht6 = dht6 + agreg1(i)^2;
end
dht1 = (sqrt(dht1)/agreg1(2))*100;
dht2 = (sqrt(dht2)/agreg2(2))*100;
dht3 = (sqrt(dht3)/agreg3(2))*100;
dht4 = (sqrt(dht4)/agreg4(2))*100;
dht5 = (sqrt(dht5)/agreg5(2))*100;
dht6 = (sqrt(dht6)/agreg6(2))*100;

%Valor RMS verdadeiro
VRMS1=0;
VRMS2=0;
VRMS3=0;
IRMS1=0;
IRMS2=0;
IRMS3=0;
for i = 2 : 1 : 16
    VRMS1 = agreg1(i)^2 + VRMS1;
    VRMS2 = agreg2(i)^2 + VRMS2;
    VRMS3 = agreg3(i)^2 + VRMS3;
    IRMS1 = agreg4(i)^2 + IRMS1;
    IRMS2 = agreg5(i)^2 + IRMS2;
    IRMS3 = agreg6(i)^2 + IRMS3;
end
VRMS1 = sqrt(VRMS1);
VRMS2 = sqrt(VRMS2);
VRMS3 = sqrt(VRMS3);
IRMS1 = sqrt(IRMS1);
IRMS2 = sqrt(IRMS2);
IRMS3 = sqrt(IRMS3);
agreg = [agreg1 agreg2 agreg3 agreg4 agreg5 agreg6];
ang = [ang1 ang2 ang3 ang4 ang5 ang6];
Max = max(DADOS);
Min = min(DADOS);
RMS1 = [agreg1(2) agreg2(2) agreg3(2) agreg4(2) agreg5(2) agreg6(2)];
RMS = [VRMS1 VRMS2 VRMS3 IRMS1 IRMS2 IRMS3];
DHT = [dht1 dht2 dht3 dht4 dht5 dht6];

guidata(hObject, handles);

function varargout = CALCULOS_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;


% --- Executes on button press in Grandezas.
function Grandezas_Callback(hObject, eventdata, handles)
GRANDEZAS


% --- Plota a forma de onda e os harmonicos referentes ao sinal
selecionado.
function lista_Callback(hObject, eventdata, handles)
handles = guihandles(gcbo);
global DADOS agreg Max Min RMS1 RMS DHT
lista = get(handles.lista,'Value');
if lista == 2
    handles=guidata(hObject);
```

```
        plot(linspace(0,(16/60),2048),DADOS(:,1),'parent', handles.sin);
        axes(handles.harmonico)
        harmonica = 1;
        sigla = 'V';
        Maximo = sprintf('Valor Máximo: %.2f V', Max(1));
        Minimo = sprintf('Valor Mínimo: %.2f V', Min(1));
        RMSF = sprintf('Valor RMS Fundamental: %.2f V', RMS1(1));
        RMSR = sprintf('Valor RMS Verdadeiro: %.2f V', RMS(1));
        DHTS = sprintf('DHT: %.2f%% ', DHT(1));
        set(handles.vmax,'String',Maximo);
        set(handles.vmin,'String',Minimo);
        set(handles.vrmsf,'String',RMSF);
        set(handles.vrms,'String',RMSR);
        set(handles.tdhs,'String',DHTS);
        guidata(hObject, handles)
    elseif lista == 3
        handles=guidata(hObject);
        plot(linspace(0,(16/60),2048),DADOS(:,2),'parent', handles.sin);
        axes(handles.harmonico)
        harmonica = 2;
        sigla = 'V';
        Maximo = sprintf('Valor Máximo: %.2f V', Max(2));
        Minimo = sprintf('Valor Mínimo: %.2f V', Min(2));
        RMSF = sprintf('Valor RMS Fundamental: %.2f V', RMS1(2));
        RMSR = sprintf('Valor RMS Verdadeiro: %.2f V', RMS(2));
        DHTS = sprintf('DHT: %.2f%% ', DHT(2));
        set(handles.vmax,'String',Maximo);
        set(handles.vmin,'String',Minimo);
        set(handles.vrmsf,'String',RMSF);
        set(handles.vrms,'String',RMSR);
        set(handles.tdhs,'String',DHTS);
        guidata(hObject, handles)
    elseif lista == 4
        handles=guidata(hObject);
        plot(linspace(0,(16/60),2048),DADOS(:,3),'parent', handles.sin);
        axes(handles.harmonico)
        harmonica = 3;
        sigla = 'V';
        Maximo = sprintf('Valor Máximo: %.2f V', Max(3));
        Minimo = sprintf('Valor Mínimo: %.2f V', Min(3));
        RMSF = sprintf('Valor RMS Fundamental: %.2f V', RMS1(3));
        RMSR = sprintf('Valor RMS Verdadeiro: %.2f V', RMS(3));
        DHTS = sprintf('DHT: %.2f%% ', DHT(3));
        set(handles.vmax,'String',Maximo);
        set(handles.vmin,'String',Minimo);
        set(handles.vrmsf,'String',RMSF);
        set(handles.vrms,'String',RMSR);
        set(handles.tdhs,'String',DHTS);
        guidata(hObject, handles)
    elseif lista == 5
        handles=guidata(hObject);
        plot(linspace(0,(16/60),2048),DADOS(:,4),'parent', handles.sin);
        axes(handles.harmonico)
        harmonica = 4;
        sigla = 'A';
        Maximo = sprintf('Valor Máximo: %.2f A', Max(4));
        Minimo = sprintf('Valor Mínimo: %.2f A', Min(4));
        RMSF = sprintf('Valor RMS Fundamental: %.2f A', RMS1(4));
        RMSR = sprintf('Valor RMS Verdadeiro: %.2f A', RMS(4));
        DHTS = sprintf('DHT: %.2f%% ', DHT(4));
        set(handles.vmax,'String',Maximo);
```

```
    set(handles.vmin,'String',Minimo);
    set(handles.vrmsf,'String',RMSF);
    set(handles.vrms,'String',RMSR);
    set(handles.tdhs,'String',DHTS);
    guidata(hObject, handles)
elseif lista == 6
    handles=guidata(hObject);
    plot(linspace(0,(16/60),2048),DADOS(:,5),'parent', handles.sin);
    axes(handles.harmonico)
    harmonica = 5;
    sigla = 'A';
    Maximo = sprintf('Valor Máximo: %.2f A', Max(5));
    Minimo = sprintf('Valor Mínimo: %.2f A', Min(5));
    RMSF = sprintf('Valor RMS Fundamental: %.2f A', RMS1(5));
    RMSR = sprintf('Valor RMS Verdadeiro: %.2f A', RMS(5));
    DHTS = sprintf('DHT: %.2f%% ', DHT(5));
    set(handles.vmax,'String',Maximo);
    set(handles.vmin,'String',Minimo);
    set(handles.vrmsf,'String',RMSF);
    set(handles.vrms,'String',RMSR);
    set(handles.tdhs,'String',DHTS);
    guidata(hObject, handles)
elseif lista == 7
    handles=guidata(hObject);
    plot(linspace(0,(16/60),2048),DADOS(:,6),'parent', handles.sin);
    ylabel = ('Corrente Fase C [A]');
    axes(handles.harmonico)
    ylabel = ('Amplitude [A]');
    harmonica = 6;
    sigla = 'A';
    Maximo = sprintf('Valor Máximo: %.2f A', Max(6));
    Minimo = sprintf('Valor Mínimo: %.2f A', Min(6));
    RMSF = sprintf('Valor RMS Fundamental: %.2f A', RMS1(6));
    RMSR = sprintf('Valor RMS Verdadeiro: %.2f A', RMS(6));
    DHTS = sprintf('DHT: %.2f%% ', DHT(6));
    set(handles.vmax,'String',Maximo);
    set(handles.vmin,'String',Minimo);
    set(handles.vrmsf,'String',RMSF);
    set(handles.vrms,'String',RMSR);
    set(handles.tdhs,'String',DHTS);
    guidata(hObject, handles)
elseif lista == 1
    return
end
axes(handles.harmonico)
bar([1:16],agreg(1:16,harmonica));
set(gca,'XTick',1:16);
textox =
{'CC','1ª','2ª','3ª','4ª','5ª','6ª','7ª','8ª','9ª','10ª','11ª','12ª','13ª',
'14ª','15ª'};
set(gca,'XTickLabel', textox);
xt = [1:16];
yt = agreg(1:16,harmonica)+ 10;
for k = 1:1:16
    ytxt = [num2str(agreg(k,harmonica),'%.2f'), sigla];
    text(xt(k),yt(k),ytxt,'rotation',30,'fontsize',8,'fontweight',
'normal');
end
axes(handles.sin)
xlim([0 (16/60)]);
guidata(hObject, handles)
```

```
function lista_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function sin_CreateFcn(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function harmonico_CreateFcn(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function vmax_CreateFcn(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)

function vmin_CreateFcn(hObject, eventdata, handles)
handles = guihandles(gcbo);
guidata(hObject, handles)


% --- Executes on button press in Voltar.
function Voltar_Callback(hObject, eventdata, handles)
delete(handles.CALCULOS);
```