Universidade Federal de Ouro Preto
Instituto de Ciências Exatas e Aplicadas
Departamento de Engenharia Elétrica

Trabalho de Conclusão de Curso

DESENVOLVIMENTO DE UM SISTEMA SUPERVISÓRIO DE BAIXO CUSTO PARA SISTEMAS ELÉTRICOS

Alisson Fernandes Nunes

João Monlevade, MG
2018

Alisson Fernandes Nunes

DESENVOLVIMENTO DE UM SISTEMA SUPERVISÓRIO DE BAIXO CUSTO PARA SISTEMAS ELÉTRICOS

Trabalho de Conclusão de curso apresentado à Universidade Federal de Ouro Preto como parte dos requisitos para obtenção do Título de Bacharel em Engenharia Elétrica pelo Instituto de Ciências Exatas e Aplicadas da Universidade Federal de Ouro Preto.
Orientador: Prof. Víctor Costa da Silva Campos

Universidade Federal de Ouro Preto
João Monlevade
2018

MINISTÉRIO DA EDUCAÇÃO
Universidade Federal de Ouro Preto – UFOP
Instituto de Ciências Exatas e Aplicadas
Colegiado do Curso de Engenharia de Elétrica

UFOP

ATA DE DEFESA

Aos 18 dias do mês de dezembro de 2018, às 19 horas, no bloco B deste instituto, foi realizada a defesa de monografia pelo (a) formando (a) Alisson Fernandes Nunes, sendo a comissão examinadora constituída pelos professores: Marcelo Moreira Tiago, Márcio Feliciano Braga e Víctor Costa da Silva Campos.

O (a) candidato (a) apresentou a monografia intitulada: Desenvolvimento de um sistema supervisório de baixo custo para sistemas elétricos. A comissão examinadora deliberou, por unanimidade, pela aprovação do (a) candidato(a), com a nota média 7,5, de acordo com a tabela 1. Na forma regulamentar foi lavrada a presente ata que é assinada pelos membros da comissão examinadora e pelo (a) formando(a).

Tabela 1 – Notas de avaliação da banca examinadora

Banca Examinadora	Nota
Marcelo Moreira Tiago	7,5
Márcio Feliciano Braga	7,5
Víctor Costa da Silva Campos	7,5
Média	7,5

João Monlevade, 18 de dezembro de 2018.

Víctor Costa da Silva Campos
Professor(a) Orientador(a)

Alisson Fernandes Nunes
Aluno(a)

Marcelo Moreira Tiago
Professor(a) Convidado(a)

Márcio Feliciano Braga
Professor(a) Convidado(a)

Resumo

Este trabalho tem a finalidade de mostrar que o *hardware* Raspberry Pi é uma alternativa de baixo custo para a automação de pequenos sistemas elétricos, podendo até substituir os controladores lógicos programáveis, que juntamente com os sistemas supervisórios, são amplamente utilizados nas indústrias para armazenar dados e atuar sobre os processos de produção. Para provar a utilidade do Raspberry Pi, foi desenvolvido um sistema em pequena escala que realiza medição e cálculo de grandezas elétricas. Para o sistema desenvolvido, foi criado um supervisório capaz de fornecer ao usuário informações das grandezas elétricas envolvidas no processo de medição e cálculo. Nesta monografia, são descritos detalhes a respeito desse sistema, da plataforma de automação Codesys, do Raspberry Pi, do Arduino e das linguagens de programação aplicadas. A utilização do Raspberry Pi em grandes processos, com o objetivo de se alcançar níveis de controle e monitoramento maiores, é analisada para fins de aplicação em trabalhos futuros.

Palavras-chave: automação, supervisório, Raspberry Pi, CoDeSys, Arduino.

Abstract

This paper aims to demonstrate that the Raspberry Pi hardware is a low cost choice for small automations in electrical systems, being able to substitute programmabble logic controllers, that together with the supervisory systems, are widely used in the industry to store data and act on the production systems. In order to demonstrate the use of Raspberry Pi in this application, a system of small-scale was developed, that performs measurement and calculation of electrical quantities. A supervisory able to provide users information about the electrical quantities involved in the process of mesurement and calculation was built for this system. In this paper, the details of proposed system are described, as well as the Codesys automation plataform, Raspberry Pi, Arduino and the programming languages applied. The use of Raspberry Pi in extensive processes, aiming to achieve higher levels of control and monitoring, is analysed for the applicability of future research.

Keywords: automation, supervisory, Raspberry Pi, CoDeSys, Arduino.

Lista de ilustrações

Lista de abreviaturas e siglas

CLP	Controlador Lógico Programável ou do inglês PLC (*Programmable Logic Controller*)
ERP	*Enterprise Resource Planning*
HDMI	*High-Definition Multimedia Interface*
IHM	Interface Homem Máquina ou do inglês HMI (*Human-Machine Interface*)
ANEEL	Agência Nacional de Energia Elétrica
I/O	Input/Output
MES	*Manufacturing Execution System*
NOOBS	*New Out Of the Box Software*
PID	Proporcional Integral Derivativo
PRODIST	Procedimentos de Distribuição de Energia Elétrica no Sistema Elétrico Nacional
RTU	*Remote Terminal Unit*
PIMS	*Plant Information Management System*
RAM	*Random Access Memory*
RMS	*Root mean square*
SCADA	*Supervisory Control and Data Acquisition*
SD	Seguro Digital
SFC	*Sequential Function Chart*
SO	Sistema Operacional

TC	Transformador de Corrente
USB	*Universal Serial Bus*

Sumário

Capítulo 1

APRESENTAÇÃO

Este capítulo tem por finalidade introduzir e apresentar as motivações, o objetivo e estrutura desta monografia, cujo foco é o desenvolvimento de um sistema supervisório de baixo custo para sistemas elétricos.

1.1 Introdução

Nas décadas de 70 e 80, com o progresso da eletrônica e o surgimento dos microprocessadores, o computador se tornou uma peça fundamental nos diversos segmentos industriais, dando origem aos primeiros sistemas supervisórios.

Esses ambientes permitem que o usuário visualize o processo a ser controlado e supervisionado de forma gráfica, por meio de telas sinópticas que representam cada bloco funcional do processo associado.

A criação de um sistema de supervisão envolve custos distribuídos em três fases principais (Queiroz; Cury, 2002):

- Aquisição de licença do sistema supervisório a ser utilizado;
- Aquisição de licença dos *drivers* de comunicação utilizados para integrar o supervisório com os equipamentos utilizados no processo;
- Custo de engenharia para o desenvolvimento das aplicações para o sistema supervisório.

O computador Raspberry Pi integra de forma econômica e reduzida recursos de um computador e de um controlador em um só dispositivo. Segundo Andrade, Soma e Nakamura (2016), o Raspberry Pi pode fazer parte de um sistema de controle e monitoramento de máquinas destacando-se por sua versatilidade de recursos, representando a inclusão de outros meios no gerenciamento de controle e monitoramento de um equipamento industrial.

Esse trabalho apresenta o desenvolvimento de um sistema supervisório de baixo custo, utilizando o computador Raspberry Pi, aplicado a um sistema de medição e cálculo de grandezas elétricas.

1.2 Motivação

Para um avanço na agilidade da produção, qualidade na manutenção e ajustes, previsibilidade de produtos, controle de máquinas e redução de trabalhos repetitivos, os empresários têm investido na automação de suas indústrias. No entanto, a implantação de um projeto de automação pode representar um custo elevado, principalmente para as empresas de pequeno e médio porte.

Diante das menções acima, é possível concluir que torna-se cada dia mais viável o desenvolvimento de um sistema de supervisão com baixo custo para fins de automação industrial.

1.3 Objetivo

Este trabalho tem o objetivo de demonstrar que o Raspberry Pi é uma alternativa de baixo custo para a automação de pequenos sistemas elétricos, podendo até substituir os controladores industriais.

Propor a implementação de um sistema supervisório capaz de monitorar o consumo de energia elétrica de uma determinada carga monofásica em uma residência ou indústria.

1.4 Estrutura do Trabalho

Essa monografia é composta por seis capítulos. O primeiro foi dedicado a apresentar alguns argumentos que justificam o uso de sistemas supervisórios em ambientes industriais, independente da complexidade que eles tenham. No segundo capítulo, são apresentados conceitos sobre sistemas de automação, com foco nos sistemas supervisórios, que são parte do objetivo da realização da presente monografia. No Capítulo 3, são apresentados os *softwares*, *hardwares* e a linguagem de programação aplicados no desenvolvimento do protótipo. No quarto capítulo, o protótipo, que consiste em um sistema de medição e cálculo de grandezas elétricas é apresentado com detalhes. No quinto capítulo, são mostrados resultados que comprovam a eficiência do projeto desenvolvido. No sexto e último capítulo, são expostas as conclusões do autor e propostas de melhorias que podem ser feitas, para fins de aplicação do modelo de sistema supervisório em trabalhos futuros.

Capítulo 2

SISTEMAS DE AUTOMAÇÃO

Este capítulo apresenta alguns conceitos sobre a hierarquia dos sistemas de automação industrial, com foco nos sistemas supervisórios.

2.1 Hierarquia dos Sistemas de Automação

A hierarquia, ou pirâmide da automação, apresenta quatro níveis (ver Figura 1), que representam desde os equipamentos e dispositivos de campo até o gerenciamento corporativo da empresa.

No primeiro nível, a base da pirâmide, estão os equipamentos de aquisição de dados, de atuação e os de controle. Os equipamentos de aquisição de dados são os sensores utilizados para medição das variáveis do processo. Os equipamentos de atuação tratam de instrumentos que, ao receberem um comando, atuam no processo de forma a variar o fluxo da malha de controle. Como exemplo, pode-se citar uma válvula, que quando recebe um comando pode alterar a quantidade de líquido que entra em um tanque. Os equipamentos de controle são instrumentos capazes de processar uma lógica previamente definida, baseada em uma ou mais variáveis de entrada (variáveis medidas) e um valor de *setpoint*, e emitir um sinal de controle para que um equipamento de atuação interfira na malha.

No segundo nível, estão os equipamentos e *softwares* destinados a supervisão dos processos executados na planta, ou seja, o Sistema de Supervisão e Aquisição de Dados (SCADA). Neste nível, o objetivo é permitir que o usuário seja capaz de monitorar e atuar sobre a malha de controle sem que seja necessária a atuação em campo. Neste nível também há suporte de banco de dados para que seja possível armazenar as informações do processo ao longo do tempo (Coelho, 2009).

No terceiro nível, tem-se o Sistema para Execução na Manufatura (MES), que corresponde a um sistema usado para o gerenciamento das atividades de produção, de forma a se estabelecer uma ligação entre o planejamento e o chão de fábrica. Como

exemplos de funcionalidades de um MES, pode-se citar:

- Ordem para reposição de materiais;
- Análise de desempenho da produção;
- Controle estatístico do processo;
- Apuração de custos;
- Melhor visibilidade do chão de fábrica.

No quarto nível, o topo da pirâmide, concentra-se a etapa de Planejamento de Recursos da Empresa (ERP). Em um processo automatizado, esse nível é responsável por integrar todos os dados da produção, como exemplos: contabilidade, recursos humanos e manutenção. O objetivo é integrar os setores da empresa, sendo uma arquitetura de ligação entre todas as funções dos diferentes departamentos.

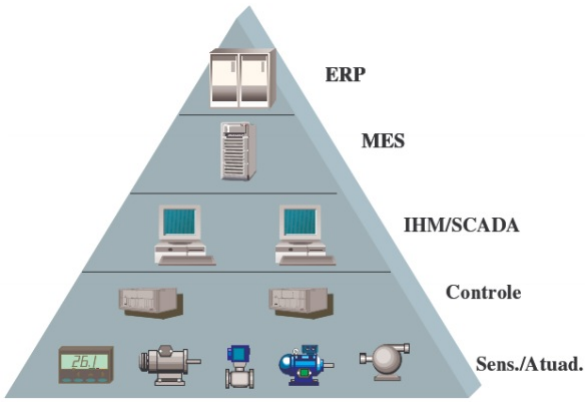

Figura 1 – Hierarquia de Sistemas de Automação.

Fonte: Retirado de Coelho (2009).

2.2 Sistemas Supervisórios

Sistemas supervisórios permitem que informações de um processo produtivo ou instalação física sejam monitoradas e rastreadas. As informações são coletadas por meio de equipamentos de aquisição de dados, manipuladas, analisadas, armazenadas e posteriormente apresentadas ao usuário. Um sistema supervisório normalmente recebe o nome de Sistema de Supervisão e Aquisição de Dados e tem o objetivo de facilitar a interação do usuário com o chão de fábrica.

Os primeiros SCADAs foram implementados para informar periodicamente o estado corrente do processo industrial, monitorando sinais representativos de medidas e dispositivos, por meio de um painel de lâmpadas.

Quando o monitoramento e o controle de um processo são feitos por meio de um sistema supervisório, segundo Coelho (2009), o processamento das variáveis de campo é mais rápido e eficiente. Um incidente no processo pode ser rapidamente detectado e mudanças nos *setpoints* são imediatamente providenciadas pelo sistema, estabilizando a situação.

Os SCADAs identificam todas as variáveis numéricas ou alfanuméricas envolvidas na aplicação por meio de *tags*, podendo executar funções computacionais (operações matemáticas e lógicas) ou representar pontos de entrada/saída (I/O) de dados do processo que está sendo controlado (Jurizato; Pereira et al., 2003). É com base nos valores das *tags* que os dados coletados são apresentados ao usuário.

2.2.1 Funções de um Sistema SCADA

Os sistemas de supervisão de processos industriais coletam dados do processo, por meio de remotas, os formatam e os apresentam ao operador de diversas formas. Assim, tem-se uma interface de alto nível entre operador e processo, em que o operador tem informações em tempo real de todos os eventos.

Segundo Coelho (2009), os SCADAs desempenham as seguintes funções:

- Supervisão: englobam todas as funções de monitoramento do processo, como elaboração de gráficos e relatórios de comportamento e tendência de variáveis analógicas e digitais.
- Operação: incluem ligar e desligar equipamentos e sequências de equipamentos, operação de malhas PID, mudança de modo de operação de equipamentos, etc.
- Controle: alguns sistemas de supervisão possuem uma linguagem que permite definir diretamente ações de controle, sem depender de um nível intermediário, representado por remotas inteligentes. Todas as operações de entrada e saída são executadas diretamente por meio de cartões I/O ligados diretamente ao barramento do micro, ou por remotas mais simples.

Algumas funções de um SCADA são detalhadas nas seções 2.2.1.1 a 2.2.1.3.

2.2.1.1 Apresentação Gráfica

Os sinóticos (fluxogramas de operação da planta) representam uma área do processo com um certo nível de detalhe (Queiroz; Cury, 2002). Para se obter uma visão mais detalhada de uma determinada área, pode-se recorrer a um novo sinótico, a um sinótico de hierarquia inferior (sub-sinótico), ou a uma visão de uma outra camada do mesmo sinótico.

No sistema gráfico (exemplo mostrado na Figura 2), o desenho é formado livremente pela combinação de entidades geométricas fundamentais como retas, retângulos, elipses e círculos, texto rasterizado e vetorial, arcos e *splines*. Após definidos, os símbolos são armazenados em uma biblioteca.

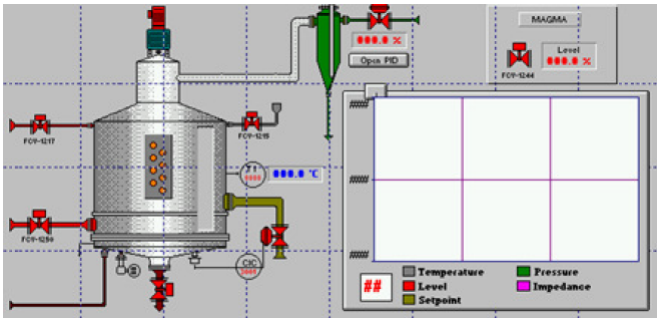

Figura 2 – Exemplo de uma apresentação gráfica.

Fonte: Retirado de Queiroz e Cury (2002).

O construtor de sinóticos é a ferramenta que permite ao usuário criar novos sinóticos. Alguns construtores são editores gráficos que definem duas estruturas de dados básicas: uma para a máscara e a outra para os campos dinâmicos.

2.2.1.2 Gerenciador de Alarmes

Presente em todos os SCADAs, o gerenciador de alarmes recebe os eventos excepcionais do processo e os registra, identificando:

- Data e hora do evento;
- Variável alarmada;
- Valor no momento do alarme;
- Descrição do evento;
- Data e hora de normalização do evento;
- Status do evento.

Os eventos são armazenados em um *buffer* ou memória e são mantidos em tela, a partir do mais recente. Após esse período, o arquivo pode ser transferido para outro computador de maior capacidade de armazenamento ou simplesmente descartado.

2.2.1.3 Relatórios de Monitoramento

Segundo Jurizato, Pereira et al. (2003), uma das principais funções de um SCADA é sua capacidade de armazenar dados e gerar relatórios de produção ao final de um dia, mês, semana ou turno, conforme exemplo apresentado na Figura 3. Relatórios de produção apresentam quanto uma determinada planta produziu, quanto consumiu de insumos, de energia e constituem o principal relatório de interesse gerencial.

Por outro lado, os relatórios de monitoramento de equipamentos são de interesse principalmente dos responsáveis pela manutenção. Eles dizem quando cada equipamento parou, porque parou e por quanto tempo ficou parado, por exemplo.

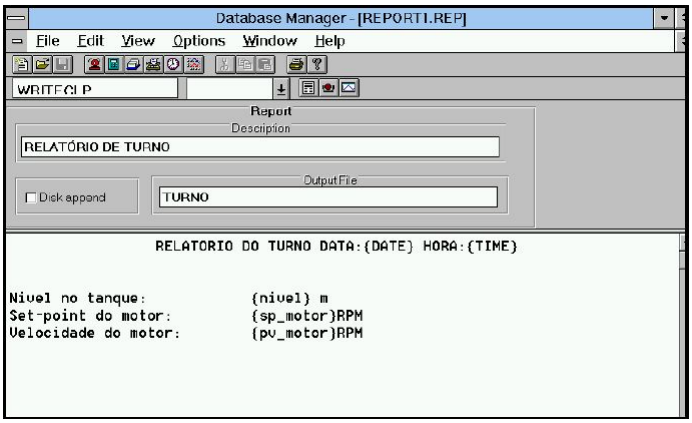

Figura 3 – Exemplo de tela de configuração de relatórios.

Fonte: Retirado de Jurizato, Pereira et al. (2003).

2.2.2 Estações de um Sistema SCADA

Segundo Jurizato, Pereira et al. (2003), as estações SCADA normalmente são classificadas como:

- Estação Local e Remota: Uma estação corresponde a qualquer computador que esteja rodando um *software* supervisório. Enquanto a estação local é a que se opera ou configura o *software*, a estação remota é acessada por meio de um *link* de comunicação.
- Estação Independente ou *Stand Alone*: Desempenha todas funções de um sistema de supervisão, porém não conectada a uma rede de comunicação. Um exemplo é apresentado na Figura 4.
- Estação Servidora de Base de Dados (servidor SCADA): Estação que executa a função de aquisição de dados.
- Estação de Monitoramento e Operação: É uma estação que permite que o operador monitore o processo, altere os seus parâmetros, reconheça alarmes e outras tarefas

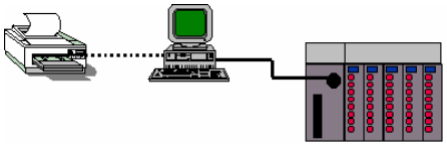

Figura 4 – Exemplo de uma estação *Stand Alone*.

Fonte: Retirado de Jurizato, Pereira et al. (2003).

de operação do processo, mas não permite alterar a configuração de telas e base de dados.

- Estação de Gerência: É uma estação que permite a gerentes, supervisores e outras pessoas autorizadas o acesso aos dados de processo em forma de relatórios, gráficos e telas, porém, não permite que sejam feitos reconhecimentos de alarmes ou alteração de parâmetros do processo.
- Estação de Engenharia: Permite modificar o comportamento do sistema supervisório, ou seja, alterar telas, adicionar ou modificar funcionalidades. Consegue reunir todas as funcionalidades de engenharia dos *softwares* dos controladores e das estações de visualização.

2.2.3 Blocos de Um Sistema SCADA

Os SCADAs dividem suas principais tarefas em blocos, permitindo maior ou menor flexibilidade e robustez, de acordo com a solução requerida (Coelho, 2009). As tarefas são assim divididas:

- Núcleo de processamento;
- Comunicação com os Controladores Lógicos Programáveis (CLPs) e Unidades Terminais Remotas (RTUs);
- Gerenciamento de Alarmes;
- Banco de Dados;
- Históricos;
- Lógicas de programação interna;
- Interface gráfica;
- Relatórios;
- Comunicação com outras estações SCADA;

O funcionamento de um sistema supervisório inicia-se no processo de comunicação com os equipamentos de campo, cujas informações são enviadas para o núcleo principal do *software*. O núcleo tem a função de distribuir e coordenar o fluxo das informações para os demais módulos, até chegarem na forma esperada para o operador do sistema.

Tecnologias computacionais utilizadas para o desenvolvimento de um SCADA têm evoluído consideravelmente nos últimos anos, aumentando sua confiabilidade, flexibilidade e conectividade. A inclusão de novas ferramentas permite diminuir cada vez mais o tempo gasto na configuração e adaptação do sistema às necessidades de cada instalação.

Capítulo 3

HARDWARES E SOFTWARE

Este capítulo tem a finalidade de apresentar conceitos sobre os hardwares Raspberry Pi e Arduino, sobre o software CoDeSys e a linguagem de programação Texto Estruturado, que foram aplicados para a automação do protótipo desta monografia, que consiste em um circuito para medição e cálculo de grandezas elétricas.

3.1 Raspberry Pi

3.1.1 Conceitos

O Raspberry Pi é um computador de pequeno porte e de baixo custo, com peso aproximado de 45 gramas. O Raspberry Pi pode ser encontrado em quatro modelos distintos, que se diferenciam basicamente pela capacidade da Memória de Acesso Aleatório (RAM) e pela quantidade de Barramentos Universais de Comunicação (USBs) (Andrade; Soma; Nakamura, 2016). A Figura 5 mostra um exemplo de *hardware* Raspberry Pi modelo 3b.

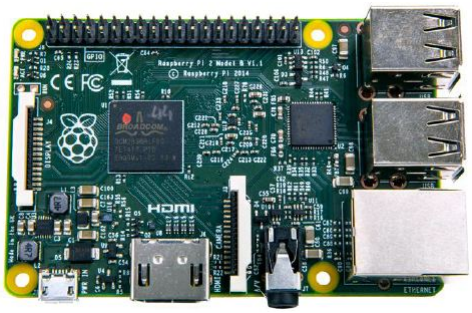

Figura 5 – *Hardware* Raspberry Pi, modelo 3b.

Fonte: Retirado de Andrade, Soma e Nakamura (2016).

Para a automação do sistema elétrico desenvolvido nesta monografia, o Raspberry Pi exerce a mesma função que um Controlador Lógico Programável (CLP) convencional

exerce nas grandes plantas de automação das indústrias.

O CLP surgiu na General Motors em 1968, motivado pelo objetivo de eliminar a necessidade de se alterar a lógica de controle dos painéis de comando a cada mudança na linha de montagem. Resumidamente, o CLP trata-se de um computador industrial capaz de armazenar instruções para aplicações em funções de controle, processar lógicas, cálculos, manipular dados e realizar comunicação em rede com outros equipamentos (Pinto, 2008).

Um CLP pode ser representado basicamente por três partes: os módulos de entradas, a unidade de processamento (CPU) e os módulos de saída, conforme mostra a Figura 6.

Figura 6 – Estrutura de um CLP.

Fonte: Retirado de Pinto (2008).

Quando um CLP é ligado, ele realiza uma autoverificação, processo que é denominado de inicialização. Estando em funcionamento, o controlador realiza uma sequência de operações, denominada ciclo de varredura. A sequência deste ciclo é mostrada na Figura 7.

Figura 7 – Ciclo de varredura de um CLP.

Fonte: Retirado de Pinto (2008).

Caso exista um programa de usuário instalado no CLP, a todo momento, o controlador verifica o estado das entradas e saídas, armazena a leitura na memória e faz a comparação com os parâmetros definidos no programa.

Os CLPs apresentam vantagens em relação aos circuitos de comandos eletromagnéticos convencionais, sendo elas:

- Demandam menor espaço;
- Demandam menor consumo de energia elétrica;
- São reutilizáveis;
- São programáveis;
- Possuem maior confiabilidade;
- Possuem maior flexibilidade;
- Possuem interfaces de comunicação com outros CLPs e computadores.

3.1.2 Vantagens

Em sua aplicação, o Raspberry Pi apresenta algumas vantagens, segundo Andrade, Soma e Nakamura (2016), são elas:

- Consumo de energia: Apresenta baixo nível de consumo de energia ao ser comparado com outros computadores;
- Não possui partes móveis: Todo o sistema está integrado em uma única placa, só há a necessidade de inserir um cartão de memória para armazenamento;
- Tamanho reduzido: Seu tamanho permite ainda que ele seja inserido dentro de outros dispositivos;
- Capacidade de expansão: Há diversos dispositivos disponíveis para o uso com o Raspberry Pi;
- É capaz de alcançar boas resoluções para vídeos e imagens;
- Acessível: Seu baixo custo, em comparação a outros similares, o torna acessível tanto para uso pessoal quanto para uso comercial.

3.1.3 Instalação e Configurações Iniciais

Para o funcionamento do Raspberry Pi é necessário o uso do sistema operacional Raspbian, que permite o controle e acesso aos recursos dos programas e aplicações do *hardware*.

Para instalar o sistema operacional Raspbian do Raspberry Pi, é necessário antes a instalação da interface gráfica New Out Of the Box Software (NOOBS) (Richardson;

Wallace, 2013). Além do Raspbian, a interface NOOBS possui outros sistemas operacionais, como Pidora, RiscOS, Archlinux, OpenELEC e RaspBMC.

Ao conectar um cartão de memória (tipo SD — Seguro Digital) na entrada correspondente do Raspberry Pi, o teclado e o mouse em entradas USB e o monitor na entrada de Interface Multimídia de Alta Definição (HDMI), os dados que foram copiados para o cartão SD são carregados e exibidos na tela.

Em seguida, ao escolher o sistema operacional desejado e instalá-lo, uma tela de configuração denominada Raspi-Config (Figura 8) é carregada. Na janela, o usuário escolhe se o Raspberry Pi deve ser inicializado por uma interface gráfica ou por linha de comando. O usuário pode também fazer configurações de câmera, senhas e outras opções do sistema operacional.

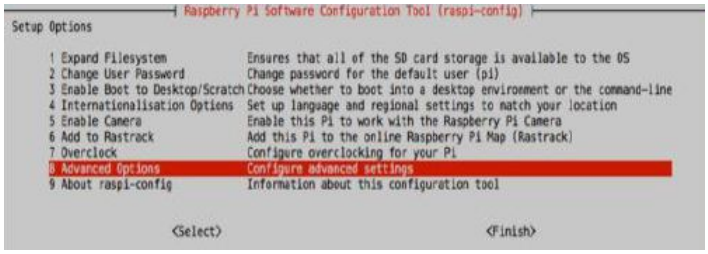

Figura 8 – Tela Raspi-Config.

Fonte: Retirado de Richardson e Wallace (2013).

3.2 Arduino

3.2.1 Conceitos

O Arduino é um *hardware*, composto por um microcontrolador e circuitos de I/O, que foi desenvolvido com o objetivo de ser uma alternativa acessível a estudantes e projetistas amadores. Além disso, para garantir ainda mais acessibilidade aos usuários, o Arduino é um *hardware* livre, o que permite que qualquer pessoa possa montá-lo, modificá-lo e personalizá-lo partindo de uma placa básica.

Para a automação do sistema elétrico desenvolvido nesta monografia, o Arduino tem a função de fazer a leitura dos valores de tensão e corrente, sendo responsável por realizar a comunicação via porta Ethernet com o Raspberry Pi, enviando dados a todo momento. Essa proposta surgiu devido ao fato do computador Raspberry Pi não possuir um conversor analógico-digital integrado, o que é essencial para se realizar as medições.

A Figura 9 apresenta o modelo Uno do Arduino.

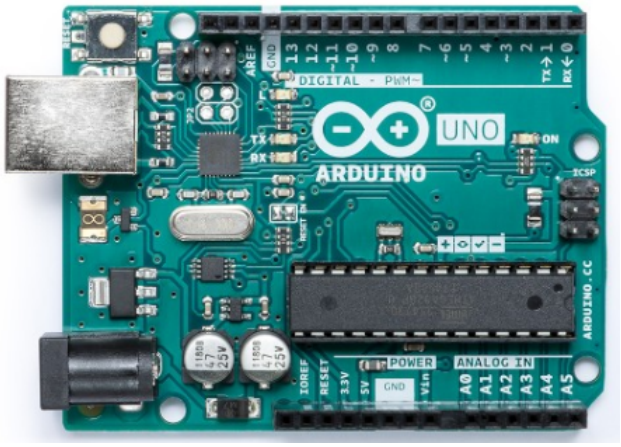

Figura 9 – *Hardware* Arduino, modelo Uno.

Fonte: Do Autor.

3.2.2 Vantagens

Além de ser um *hardware* livre, o Arduino oferece outras vantagens, como exemplos: ótimo custo benefício; plataforma de programação bastante intuitiva e uma placa que permite a integração com diversos sensores.

3.2.3 Instalação e Configurações Iniciais

Para fazer uso dos recursos do Arduino, inicialmente é necessário realizar o *download* do seu ambiente de desenvolvimento, denominado do Arduino *Software* (IDE).

Ao se conectar o Arduino em um computador utilizando um cabo USB, o *hardware* obtém energia a partir dessa conexão e inicia-se o processo de instalação dos *drivers* necessários. Biblioteca básicas já vêm incluídas na instalação do IDE e bibliotecas específicas podem ser obtidas livremente.

As bibliotecas básicas do Arduino são:

- EEPROM: Usada para ler e gravar dados em uma memória EEPROM (*Electrically-Erasable Programmable Read-Only Memory*) no Arduino. No Uno, a EEPROM tem o tamanho de 1024 bytes e no Mega, de 4096 bytes.
- Ethernet: Permite conectar o Arduino à Internet ou à rede local usando um *shield* Ethernet.
- LiquidCrystal: Com essa biblioteca, é possível utilizar *displays* de cristal líquido (LCD).
- Servo: Controle de motores servo.

- SPI: Comunicação com dispositivos usando o barramento *Serial Peripheral Interface* (SPI).

3.2.3.1 Comunicação com o Arduino Ethernet Shield

O envio de informações remotamente é um dos grandes objetivos de muitos projetos envolvendo o Arduino. O dispositivo Ethernet *Shield* W5100 pertence a família do Arduino e possibilita o acesso às informações na sua rede local, podendo ser monitorado de qualquer lugar do mundo.

A Figura 10 apresenta o modelo da placa Ethernet Shield do Arduino.

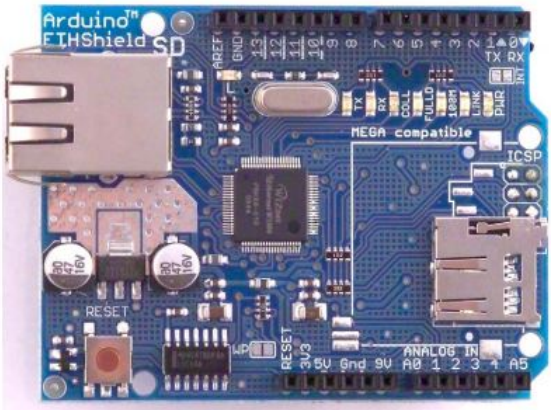

Figura 10 – *Hardware* Arduino, placa Ethernet Shield.

Fonte: Do Autor.

Para realizar o envio de informações em uma determinada rede é necessário encaixar a placa Ethernet Shield no Arduino, como mostra a Figura **??** e, em seguida, conectar-se a um roteador, por meio de um cabo de rede.

Primeiro, deve-se realizar a configuração do *Internet Protocol* (IP) que será atribuído ao Arduino. Para isso, o usuário deve entrar com o comando *CMD* no menu iniciar de seu computador e uma nova janela, denominada *prompt* de comando, do Windows é carregada. Na nova tela, deve-se digitar o comando *ipconfig/all* e pressionar a tecla *enter*.

Em seguida, várias informações são apresentadas na tela do *prompt*, como mostra a Figura 11. As informações são relacionadas à configuração da placa de rede do computador e de sua rede local. O usuário deve anotar as informações da sua conexão local, do endereço IP, da máscara de sub-rede e o *gateway* padrão, que são necessárias para a configuração do Arduino Ethernet Shield.

As informações da conexão da rede local devem ser diretamente passadas para o ambiente de programação do Arduino, para que assim a comunicação seja efetivada. A forma padrão de declaração das informações deve estar de acordo com a Figura 12.

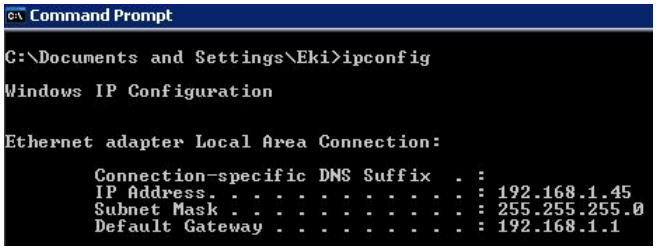

Figura 11 – Janela do *prompt* de comando do Windows.

Fonte: Do Autor.

```
#include <SPI.h>
#include <Ethernet.h>
byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED }; //MAC address
IPAddress ip(192,168,1,177);     // IP da lan
IPAddress gateway(192,168,1,255);// Gateway Padrão
IPAddress subnet(255,255,255,0); // Máscara de sub-rede
```

Figura 12 – Declaração das informações de rede local na IDE do Arduino.

Fonte: Do Autor.

3.3 A Linguagem de Programação Texto Estruturado

Existem no mercado diferentes linguagens e sistemas de programação voltados para *hadwares* de controle. No entanto, antes de 1992, a variedade de linguagens começou a representar aumento de custos nas indústrias, fazendo com que a troca de um controlador por outro, de um fabricante diferente, fosse inviável.

A inviabilidade se dava porque o programador não conseguia trocar unidades de programas e nem reutilizar funções já programadas. Para solucionar tal problema, em 1992, foi publicado um conjunto de normas que estabelecem padrões para a programação de *hadwares* de controle e, desde então, os usuários não são mais dependentes de seus fabricantes.

Para a automação do sistema elétrico desenvolvido nesta monografia, foi utilizada a linguagem de programação Texto Estruturado.

O Texto Estruturado é uma linguagem de alto nível com raízes em Pascal e C. Contém todos os elementos essenciais de uma linguagem de programação moderna, incluindo condicionais e iterações. A Figura 13 apresenta um exemplo de programação utilizando o texto estruturado.

3.4 CoDeSys

3.4.1 Conceitos

A plataforma CoDeSys, desenvolvida pela empresa de *software* alemã 3S-Smart, utiliza o padrão de linguagens de programação para *hardwares* de controle que segue a

```
I:=25;
WHILE J<5 DO
      Z:= F(I+J);
END_WHILE

IF B_1 THEN
      %QW100:= INT_TO_BCD(Display)
ENDIF

CASE TW OF
      1,5:   TEMP := TEMP_1;
      2:     TEMP := 40;
      4:     TEMP := FTMP(TEMP_2);
ELSE
      TEMP := 0;
      B_ERROR :=1;
END_CASE
```

Figura 13 – Exemplo de programação em texto estruturado.

Fonte: Retirado de Nakaygawa (2009).

norma IEC 61131, Parte 3 (Linguagens de Programação) (Souza; Pereira, 2015). Desse modo, é permitido ao programador o uso de qualquer uma das cinco linguagens que fazem parte da norma para desenvolver sua aplicação ou mesmo utilizar mais de uma linguagem no mesmo programa. As cinco linguagens são:

- Texto Estruturado;
- Lista de Instruções;
- Ladder;
- Diagrama de blocos funcionais;
- SFC (Sequential Function Charts).

O *software* CoDeSys basicamente pode ser dividido em 3 camadas, sendo elas:

- Camada de Desenvolvimento: Na qual se encontra o ambiente de desenvolvimento do programa
- Camada de Comunicação: Serve de interface de comunicação entre a camada de desenvolvimento e a cama do dispositivo, sendo responsável pela troca de variáveis entre camadas.
- Camada do Dispositivo: Para que um sistema ou dispositivo possa ser programado com o CoDeSys, é necessário que tenha o Runtime System do CoDeSys implementado e adaptado para o mesmo *hardware*.

Atualmente a plataforma CoDeSys é utilizada por mais de 250 fabricantes de CLPs, como Mitsubishi, BECKHOFF, ABB, Schneider, SEW e Bosch.

O *software* possui um simulador integrado que permite representar o programa e a aplicação, conforme visualizado na Figura 14.

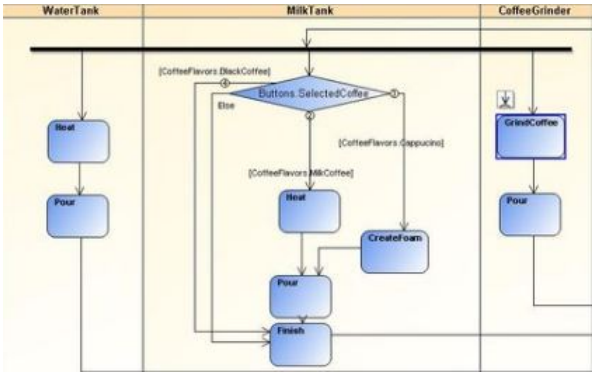

Figura 14 – Exemplo de programação no simulador CoDeSys.

Fonte: Retirado de Souza e Pereira (2015).

Há também um editor de Interface Homem Máquina (IHM) em que é possível programar supervisórios. Este recurso foi o utilizado para a automação do sistema elétrico desenvolvido nesta monografia.

3.4.2 Vantagens

O uso do recurso de edição de interface do CoDeSys oferece a vantagem de possuir licença gratuita, podendo ser instalado em qualquer estação de trabalho. Além disso, seu *software* abrange diversas áreas da indústria em uma única plataforma, sendo uma das melhores e mais completas do mercado, que está em constante atualização.

O *software* permite que o usuário desenhe gráficos em que possa visualizar os dados e avaliar o desempenho do controlador facilmente. Todas as ferramentas necessárias para diversas aplicações utilizando o CodeSys fazem parte de sua instalação descartando a necessidade de itens adicionais.

3.4.3 Instalação e Configurações Iniciais

Para fazer uso dos recursos do CoDeSys e de sua aplicação com o Raspberry Pi, realiza-se o *download* do *software* com a versão V3.5 SP11 Patch 4. O usuário necessita realizar um registro na plataforma do *software* para que o *download* seja liberado.

Em seguida, deve-se realizar o *download* e instalação do pacote “CoDeSys Control for Raspberry Pi"e, por meio dele, o Raspberry Pi pode ser programado para ser aplicado como um CLP.

Capítulo 4

O SISTEMA ELÉTRICO DESENVOLVIDO

Este capítulo apresenta o sistema elétrico controlado pelo sistema supervisório utilizando o Raspberry Pi.

4.1 Apresentação

Com o objetivo de desenvolver o sistema supervisório de baixo custo proposto, foi projetado e montado um sistema elétrico de corrente alternada capaz de medir as grandezas elétricas tensão e corrente em uma carga. Para o desenvolvimento dessa monografia de se implementar um sistema supervisório de baixo custo, o projeto não leva em conta os parâmetros necessários propostos pela norma PRODIST módulo 8. Dessa forma, em sistemas comerciais disponíveis no mercado se enquadram nesse padrão imposto pela ANEEL.

O objetivo principal da escolha de uma planta que trabalhe em corrente alternada, foi o fato de que nas indústrias e residências esse tipo de carga é a que está presente em maior número, levando a uma maior aplicabilidade do projeto. Um organograma que representa todo o projeto desenvolvido é apresentado na Figura 15.

Na etapa 1, os circuitos de medição de tensão e corrente são conectados ao Arduino para medição das duas grandezas na carga. Na etapa 2, o Arduino, quando solicitado, fornece ao Raspberry Pi (etapa 4), por meio de uma rede Ethernet (etapa 3), as informações de corrente e potência ativa na carga. Na etapa 5, o *software* CoDeSys faz o cálculo das grandezas: potência aparente, potência reativa, fator de potência e energia a partir dos dados de corrente e potência ativa que lhes são enviados. Em seguida, o CoDeSys apresenta as informações ao usuário em um sistema supervisório.

Os detalhes dos métodos de medição e cálculo das grandezas, assim como a sequência em que eles ocorrem, são apresentados nas seções a seguir.

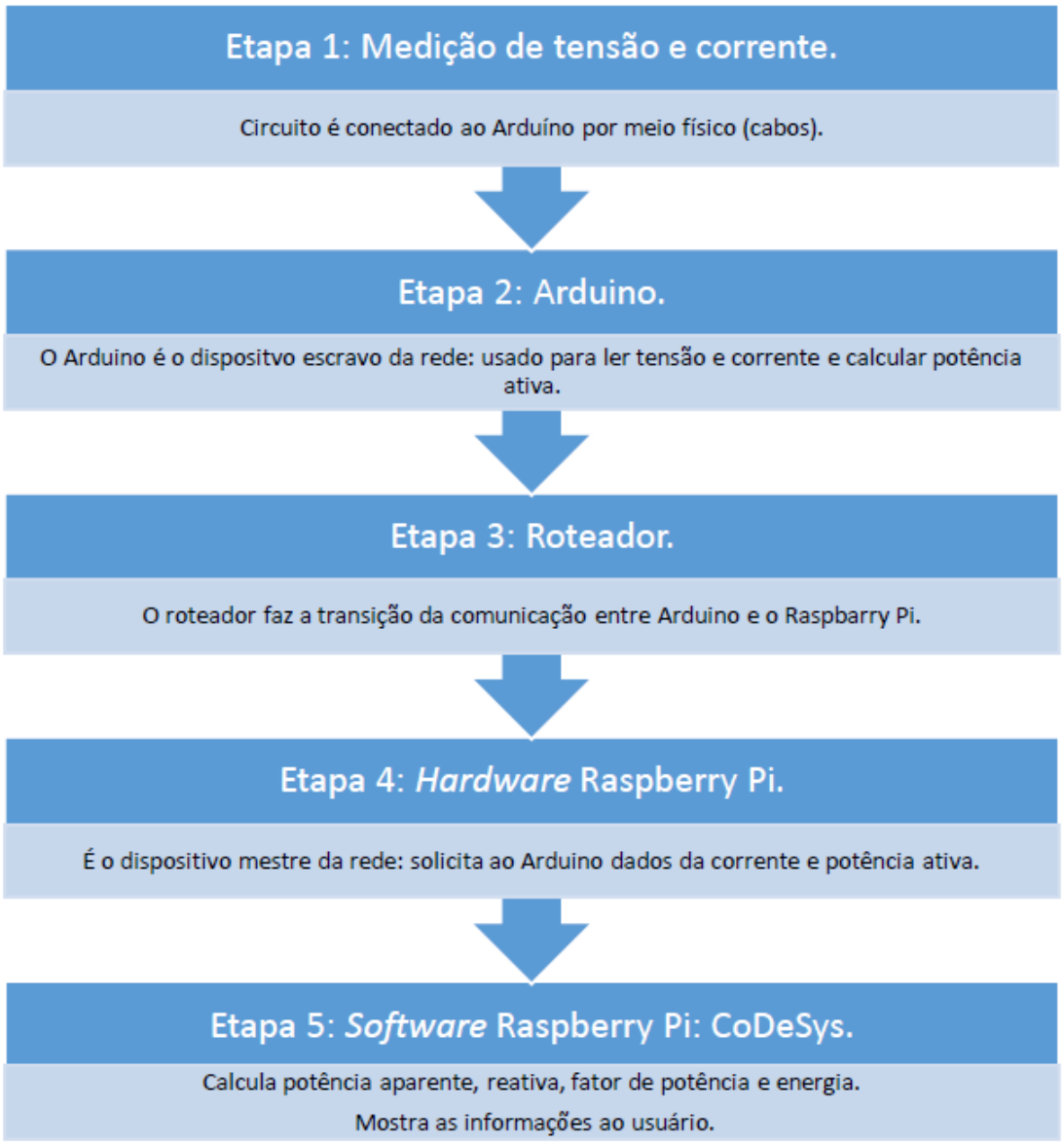

Figura 15 – Organograma do projeto.

Fonte: Do Autor.

4.2 Classificações do Sistema

Considerando a hierarquia dos sistemas de automação e as estações e blocos de um sistema SCADA, o projeto desenvolvido tem as características:

- Hierarquia:
  - No nível 1 estão os circuitos de medição de corrente e tensão;
  - No nível 2 está o sistema supervisório;
  - Os níveis 3 e 4 não foram implementados no projeto.
- Estações:

- O computador em que está instalado o CoDeSys funciona como uma estação de monitoramento do supervisório;
- O Arduino funciona como estação servidora de base de dados.

- Blocos:

  Quanto aos blocos de um sistema SCADA, no projeto implementado têm-se o bloco de processamento: CoDeSys; o bloco de comunicação: rede Ethernet; o bloco de lógica de programação: Arduino e CoDeSys e o bloco de interface gráfica: sistema supervisório.

4.3 Monitoramento de Corrente

4.3.1 Sensor de Corrente

Para a leitura da corrente alternada optou-se pelo sensor SCT-013, apresentado na Figura 16. O modelo consegue realizar leituras de até 100A, tendo como principal vantagem não precisar de contato elétrico com o circuito para realizar a medição. Dessa forma, não é necessário que se realize modificações nas instalações do cliente para realizar as leituras.

Figura 16 – Sensor de corrente SCT-013 100A.

Fonte: Retirado de Ramos e Andrade (2015).

A sigla SCT significa *Split-Core Current Transformer*, ou seja, transformador de corrente de núcleo dividido.

4.3.1.1 Princípio de Funcionamento

Para o SCT-013 conseguir realizar a leitura da corrente elétrica sem a necessidade de contato elétrico com o circuito, ele utiliza as propriedades magnéticas da corrente elétrica.

O sensor utiliza o efeito Hall para medir a corrente, ou seja, o campo elétrico gerado pela ação de um campo magnético no fluxo de corrente de um condutor (Ramos;

Andrade, 2015). O SCT-013 possui uma bobina (conjunto de 2000 espiras) interna em sua estrutura, conforme mostra a Figura 17, que são distribuídas ao redor de um condutor no qual deseja-se medir a corrente.

Figura 17 – Bobina interna do sensor SCT-013.

Fonte: Retirado de Ramos e Andrade (2015).

4.3.2 Leitura de Corrente

Para realizar a leitura da corrente utilizando o sensor SCT-013 é necessário o *download* da biblioteca *Emonlib* da plataforma Arduino, por meio da qual é possível calcular a corrente elétrica consumida na carga.

Utilizando informações do *datasheet*, percebe-se que o SCT-013 apresenta em sua saída uma variação de corrente. Assim, torna-se necessário a utilização de um resistor em paralelo com a saída do sensor para gerar a variação de tensão que é lida pelo Arduino. Para o cálculo do resistor descrito anteriormente, foram adotadas as seguintes etapas:

- Conversão da corrente máxima do sensor para a corrente de pico máxima no primário do transformador de corrente

$$I_{pico1} = I_{rms}\sqrt{2} = 100A\sqrt{2} = 141.4A \tag{4.1}$$

  em que I_{pico1} é a corrente máxima no primário do transformador de corrente (TC).

- Divisão da corrente de pico máxima pelo número de espiras do transformador de corrente do sensor

$$I_{pico2} = \frac{I_{pico1}}{2000} = 0.0707A \tag{4.2}$$

  em que I_{pico2} é a corrente máxima no secundário do TC.

- Cálculo final do resistor a ser inserido em paralelo com a saída do sensor.

  A tensão aplicada ao resistor no momento de pico da corrente é igual à metade da tensão de referência do Arduino (5V), dessa forma calcula-se o valor do resistor

$$R = \frac{5}{2} I_{pico2} = \frac{5}{2} 0.0707 = 35.4\Omega. \tag{4.3}$$

Para verificar a precisão do sensor, realizou-se a medida da corrente de uma lâmpada incandescente de 60 *Watts*. As figuras 18 e 19 mostram como o experimento de leitura da corrente foi montado.

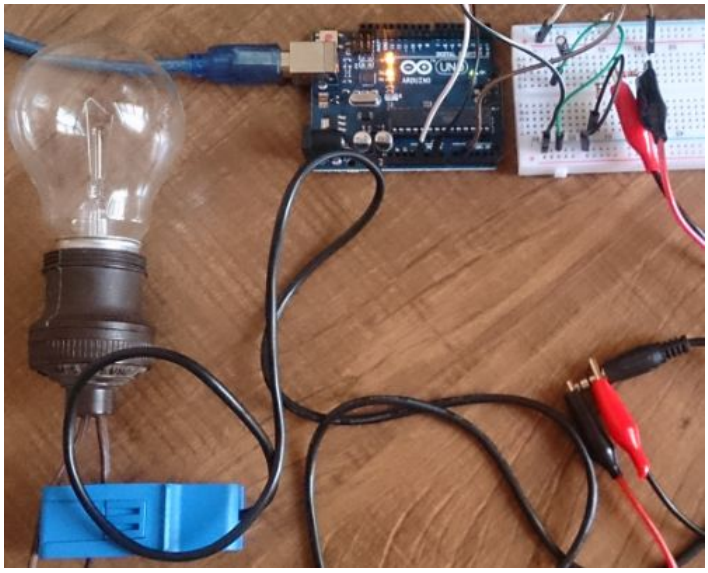

Figura 18 – Realização de leitura da corrente de uma lâmpada incandescente.

Fonte: Do Autor.

O SCT-013 apresentou boa precisão na leitura da corrente, informando na porta serial do Arduino o valor de 0.47A. Este valor está condizente com a teoria, uma vez que para a lâmpada de 60 *watts* / 127V a corrente nominal é igual a

$$I_{nominal} = \frac{P}{V} = \frac{60}{127} = 0.472A. \tag{4.4}$$

4.4 Monitoramento de Tensão

4.4.1 Descrição do Circuito

Para realizar a leitura da tensão alternada em um determinado equipamento, inicialmente precisa-se de um transformador monofásico para abaixar a tensão da rede para aplicação em um circuito divisor de tensão. O divisor torna-se necessário porque a

Figura 19 – Modo correto de leitura da corrente.

Fonte: Do Autor.

tensão no secundário do transformador é maior que a tensão de entrada máxima suportada pelo Arduino.

O transformador utilizado nessa monografia é um abaixador de 127/12 Volts com uma corrente nominal de 0.5A no lado de alta tensão.

Na Figura 20 pode-se visualizar o circuito responsável pela leitura da tensão.

O trecho do circuito composto pelo resistor R1 e o potenciômetro RV1 garante que o máximo valor de tensão que possa se encontrar na porta analógica A1 do Arduino seja de 4,8 Volts. Esse módulo de tensão encontra-se dentro do aceitável nas entradas analógicas de um Arduino.

O circuito divisor de tensão, composto pelos resistores R2 e R3, é responsável por criar um *offset* na onda senoidal, isso devido ao fato de que a entrada do Arduino não suporta tensões negativas.

4.4.2 Implementação do Circuito

Para o cálculo dos componentes do circuito visualizado na Figura 20 que realiza a leitura da tensão, considerou-se uma tensão de pico máxima de $12\sqrt{2} = 16{,}97$ Volts na saída do transformador monofásico. A malha do circuito analisado é apresentada na Figura 21.

Adotando uma queda de tensão de pico máxima de 2,3 Volts sobre o potenciômetro R_{V1}, obtém-se a máxima queda de tensão admissível em cima do resistor R_1

$$V_{\mathrm{R1}} = (12\sqrt{2}) - 2,3 = 14,67V. \tag{4.5}$$

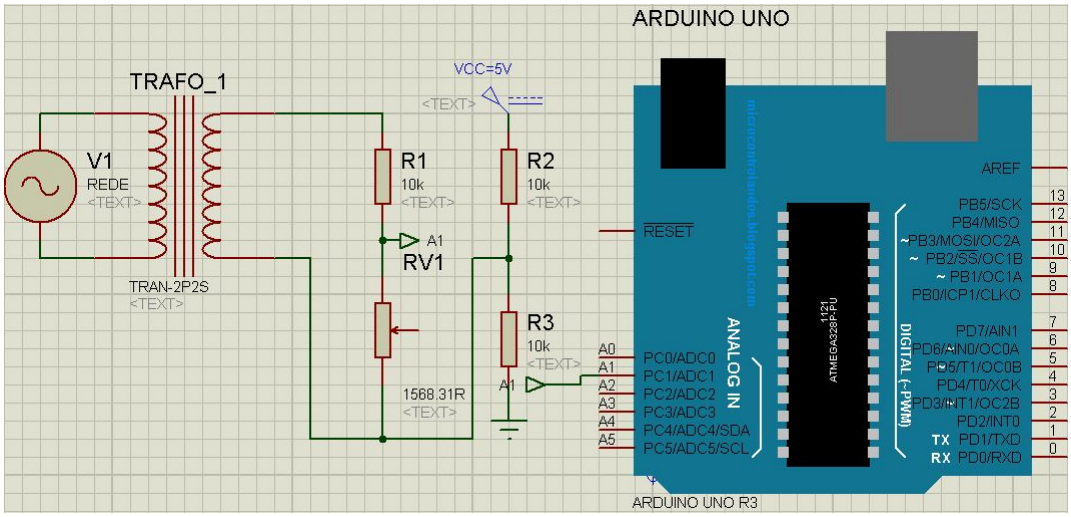

Figura 20 – Circuito de medição de tensão.

Fonte: Do Autor.

Por definições de projeto, adota-se o resistor $R_1 = 10$ kΩ. Agora é calculada a corrente de pico do circuito

$$I_{MAX} = \frac{V_{R1}}{R_1} = \frac{14,67}{10000} = 1,467mA. \tag{4.6}$$

Em seguida, calcula-se o valor de resistência que deverá ser ajustada no potenciômetro R_{V1} para uma leitura precisa da tensão alternada da rede

$$R_{V1} = \frac{V_{RV1}}{I_{MAX}} = \frac{2,3}{1,467 \times 10^{-3}} = 1568,31\Omega. \tag{4.7}$$

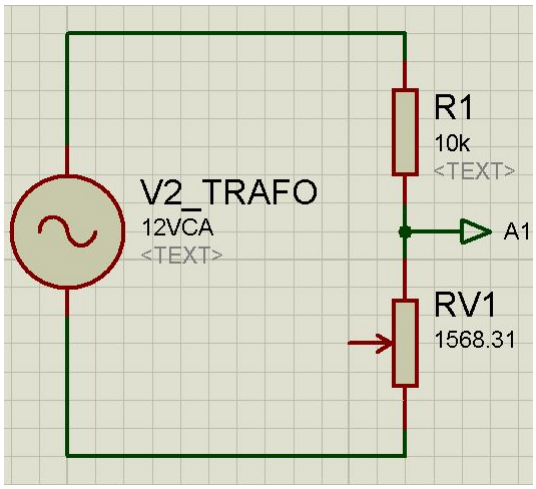

Figura 21 – Representação em malha do circuito composto pelo resistor R_1 e o potenciômetro R_{V1}.

Fonte: Do Autor.

4.5 Calculando as Grandezas

Conforme relatado, o Arduino é responsável por fazer a leitura da medição de tensão e corrente na carga. As medidas da tensão e da corrente instantânea são tomadas, e um intervalo de 6000 medições são consideradas para os cálculos das demais grandezas.

Inicialmente é calculada a corrente RMS

$$I_{rms} = \sqrt{\sum_{n=1}^{6000} I_n^2}. \tag{4.8}$$

De modo igual,

$$V_{rms} = \sqrt{\sum_{n=1}^{6000} V_n^2}. \tag{4.9}$$

Ainda no programa implementado no Arduino, a cada medição de tensão e corrente instantânea, é determinado o valor da potência ativa instantânea, e então a potência ativa média pode ser determinada por

$$P_{cc} = \frac{\sum_{n=1}^{6000} V_n I_n}{6000}. \tag{4.10}$$

Em seguida, os resultados de corrente I_{rms} e potência ativa P_{cc} são enviados ao CoDeSys, que calcula as demais grandezas desejadas: potência aparente, potência reativa, fator de potência e energia. Para calcular essas quatro grandezas, seria necessário, além dos resultados de corrente e potência ativa, a tensão V_{rms}, porém, o projeto foi implementado de tal forma que é possível o envio de duas variáveis do Arduino para o Raspberry. Então, optou-se por fazer o envio dos resultados da corrente e da potência ativa ao invés da corrente e da tensão. Com isso, foi determinado que a carga conectada aos circuitos de medição implementados, para fins de cálculo, sempre tem valor igual a 127 Volts (RMS).

Então, a potência aparente pode ser determinada

$$S = V_{rms} I_{rms} = 127 \times I_{rms}. \tag{4.11}$$

Nas cargas indutivas há uma defasem angular entre a tensão e a corrente e, para estabelecer uma relação da defasagem, usa-se o fator de potência (FP). O FP determina a relação que existe entre a potência ativa e a potência aparente, dada por 4.12.

$$FP = \frac{P_{cc}}{S}. \tag{4.12}$$

Quando o FP tem um valor alto, há a indicação de bom desempenho do sistema, pois a maior parte da energia está sendo convertida em trabalho. De forma inversa, quando o FP tem um baixo valor, há a indicação de desempenho ruim do sistema.

Além da tensão, medida em Volts (V), da corrente, medida em Ampere (A), da potência ativa, medida em Watts (W), da potência aparente, medida em Volt-Ampere

(VA) e do fator de potência, o sistema supervisório fornece ao usuário o valor da potência reativa (Q), medida em Volt-Ampere reativo (VAr), que é determinada por

$$Q = \sqrt{S^2 - P_{cc}^2}. \tag{4.13}$$

Por último, a energia, medida em Joules, ou trabalho realizado pela carga, é determinada através da potência ativa na carga (P_{cc}) em função do tempo (t) de leitura em que o equipamento permaneceu ligado.

$$E = \sum_{n=1}^{6000} P_{cc} \times t. \tag{4.14}$$

Capítulo 5

RESULTADOS

Este capítulo apresenta os resultados esperados e obtidos com o desenvolvimento do projeto.

5.1 Testes

Para verificar a veracidade das grandezas obtidas pelos circuitos de medição e pelos algoritmos de cálculo, implementados tanto no Arduino quanto no CoDeSys, foram testadas três cargas diferentes. As características das cargas são mostradas na Tabela 1. É importante destacar que, para mostrar a energia (E) da carga ao usuário, o gráfico é atualizado a cada 1 segundo, logo, a energia de cada carga mostrada na Tabela 1 foi calculada considerando o tempo de 1 segundo da carga ligada.

Tabela 1 – Cargas de testes.

Carga	S (VA)	P (W)	Q (VAr)	FP	E (J)
Ferro de passar	1100	1100	0	1	1100
Lâmpada fluorescente	45,45	25	37,95	0.55	25
Lâmpada incandescente	60	60	0	1	60

Fonte: Do Autor.

As figuras 22 a 34 mostram os resultados da potência aparente, potência ativa, potência reativa, fator de potência e energia na carga para os três cenários de teste utilizados. Observe que para o ferro de passar roupas e para a lâmpada incandescente não há resultado de potência reativa, uma vez que o FP das cargas é unitário.

O eixo horizontal dos gráficos que apresentam a energia consumida pelas cargas não corresponde ao tempo real em que foram efetuadas as medições em determinada carga, ou seja, apenas uma escala de tempo aleatória gerada pelo *software* CoDeSys.

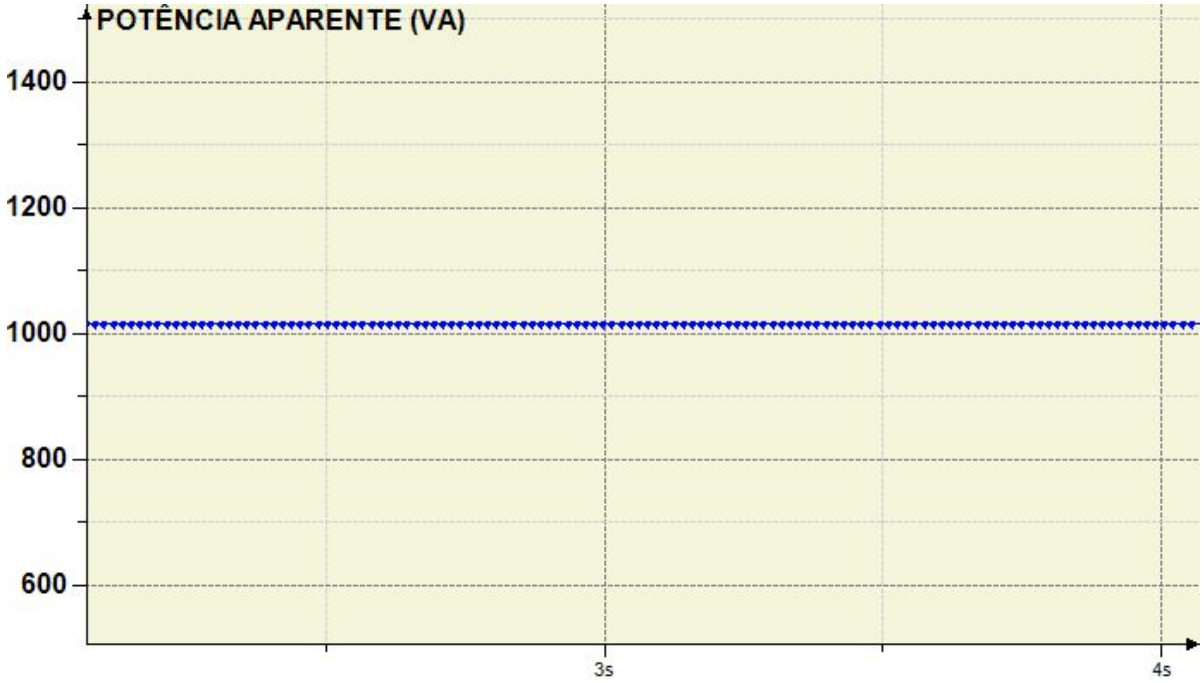

Figura 22 – Potência aparente do ferro de passar roupas.

Fonte: Do Autor.

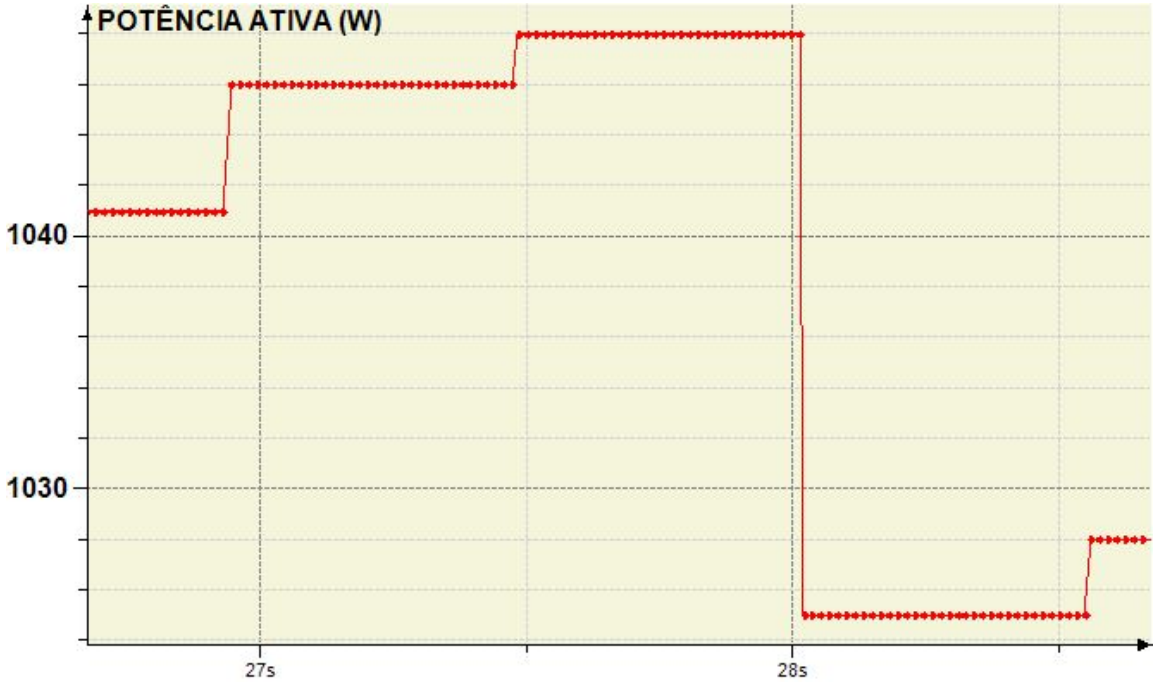

Figura 23 – Potência ativa do ferro de passar roupas.

Fonte: Do Autor.

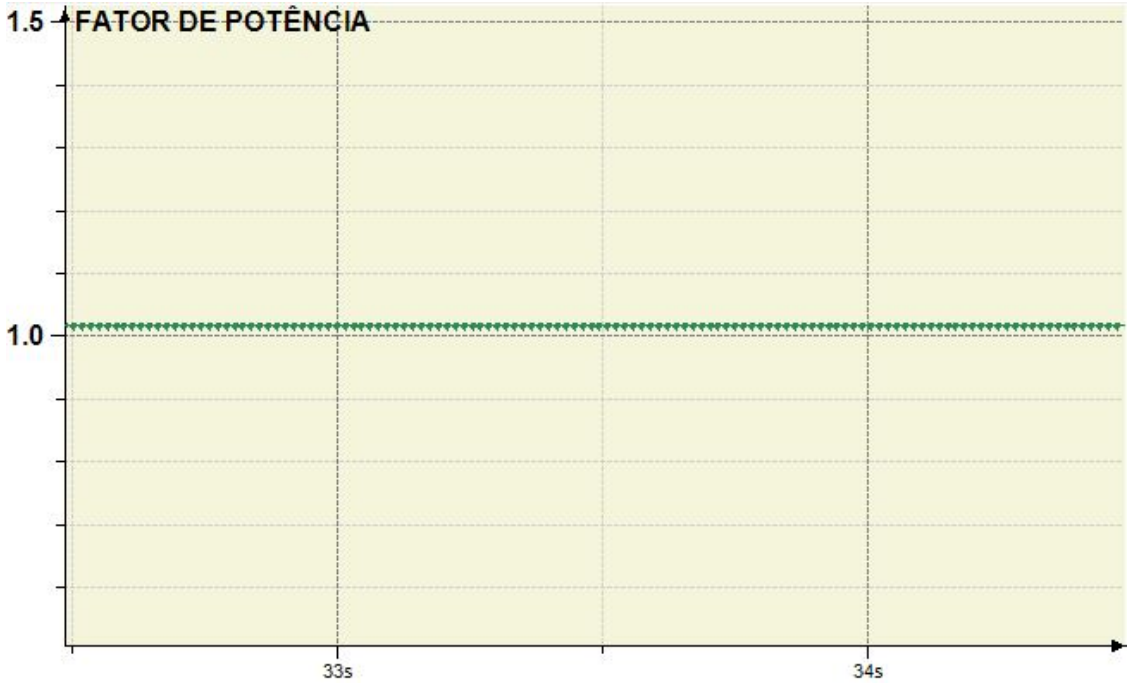

Figura 24 – Fator de potência do ferro de passar roupas.

Fonte: Do Autor.

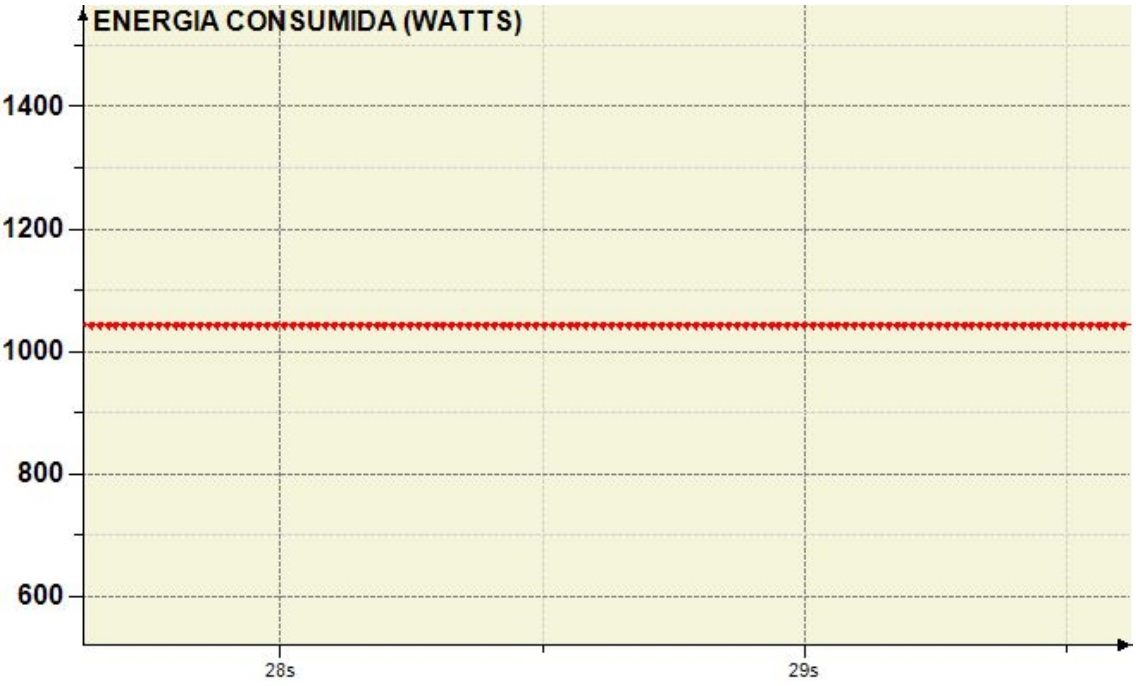

Figura 25 – Energia do ferro de passar roupas.

Fonte: Do Autor.

5.2 Análise Financeira

Os custos para implementação do presente trabalho são apresentados na Tabela 2. Observe que os circuitos de medição e o roteador, usado para transmissão das informações,

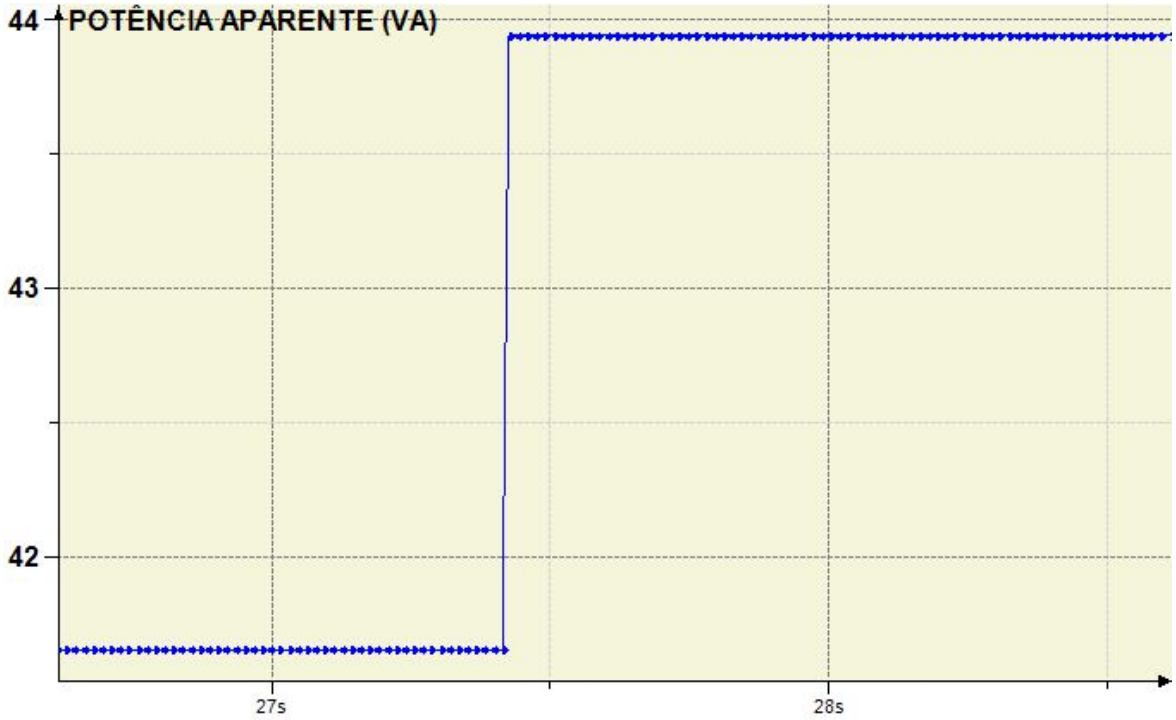

Figura 26 – Potência aparente da lâmpada fluorescente.

Fonte: Do Autor.

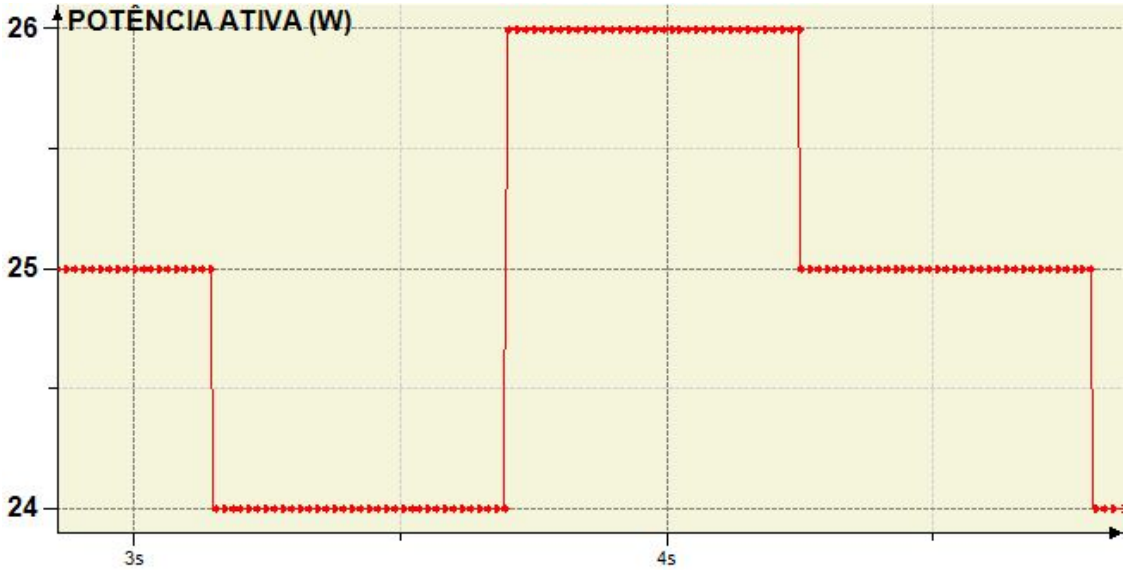

Figura 27 – Potência ativa da lâmpada fluorescente.

Fonte: Do Autor.

não são considerados, uma vez que eles teriam que ser adquiridos mesmo que fosse usado um dos sistemas supervisórios disponíveis no mercado.

A Tabela 3 apresenta o custo de alguns sistemas disponíveis no mercado.

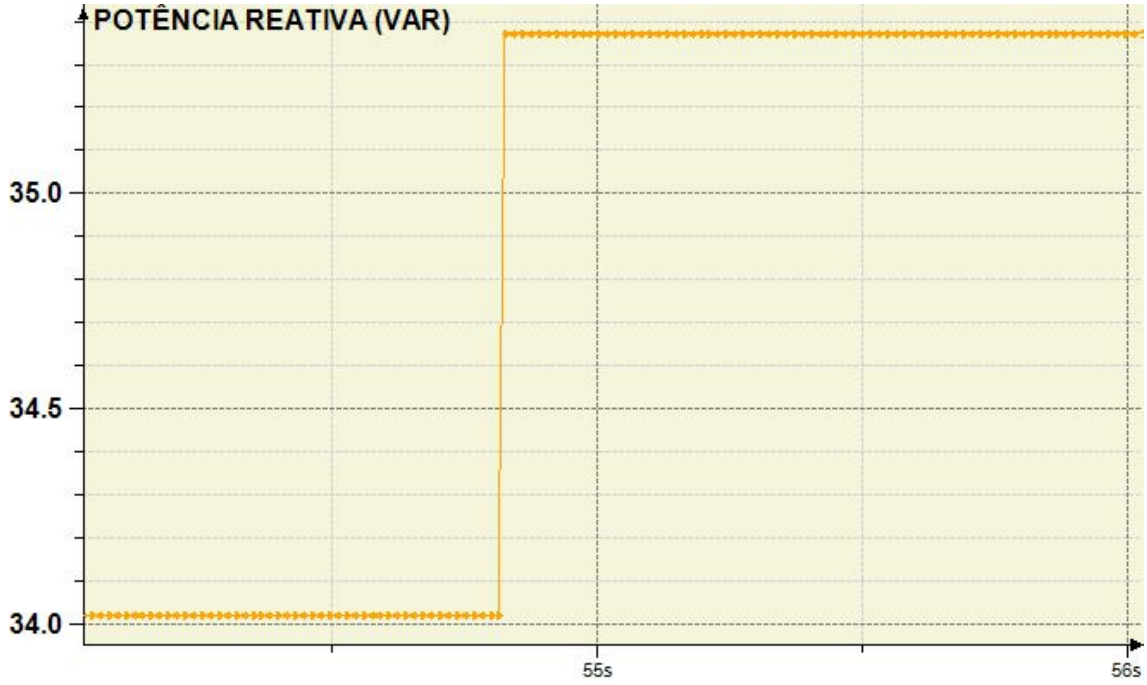

Figura 28 – Potência reativa da lâmpada fluorescente.

Fonte: Do Autor.

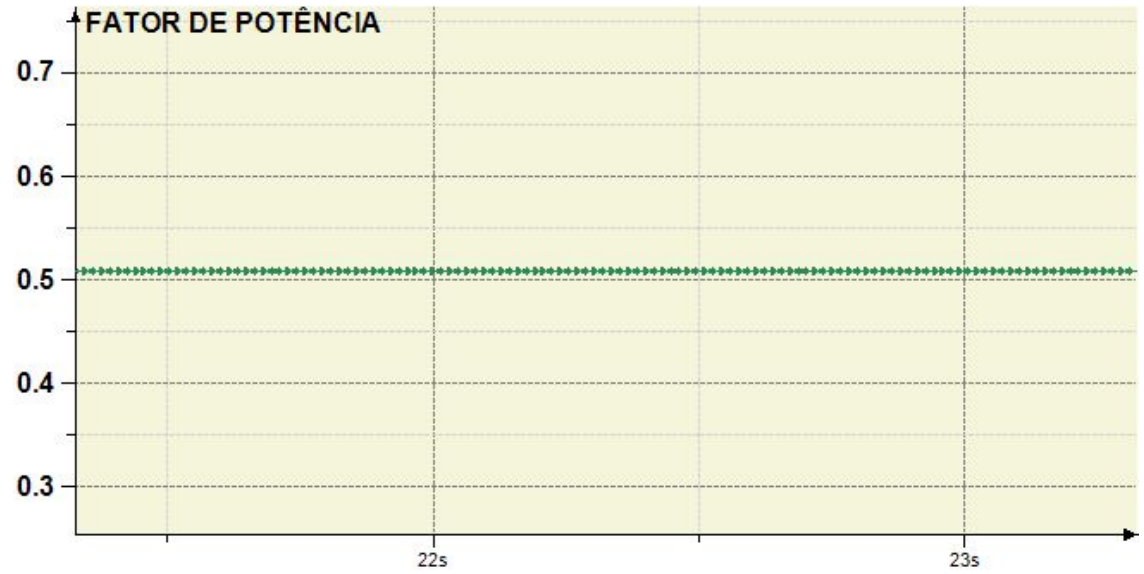

Figura 29 – Fator de potência da lâmpada fluorescente.

Fonte: Do Autor.

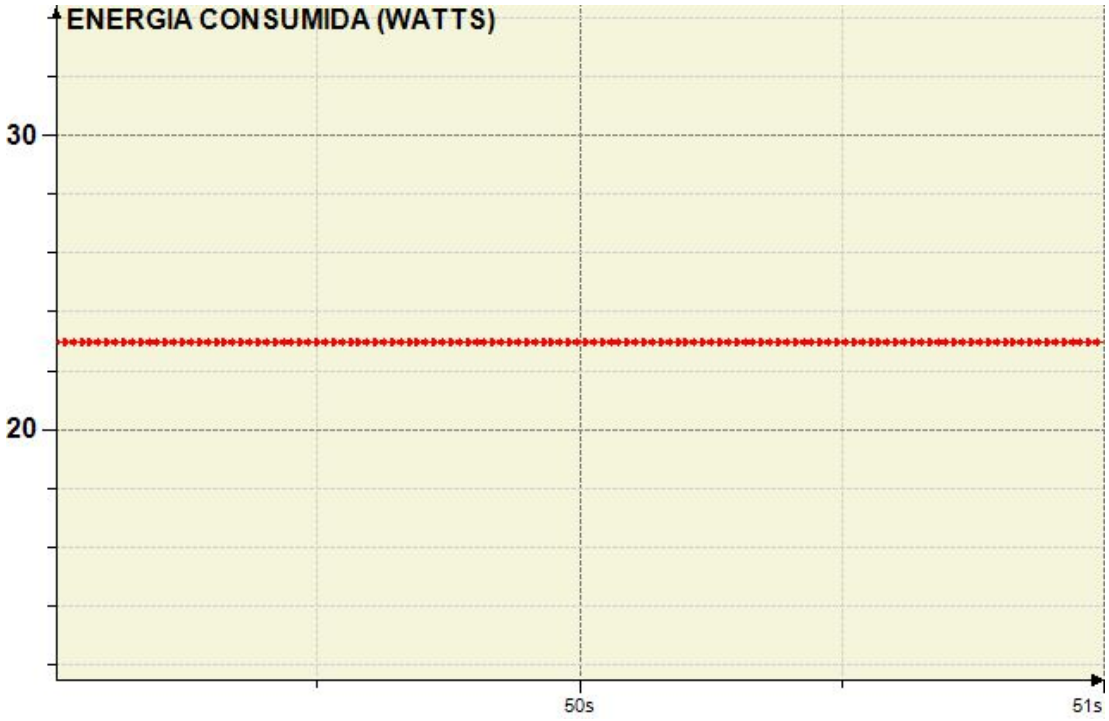

Figura 30 – Energia da lâmpada fluorescente.

Fonte: Do Autor.

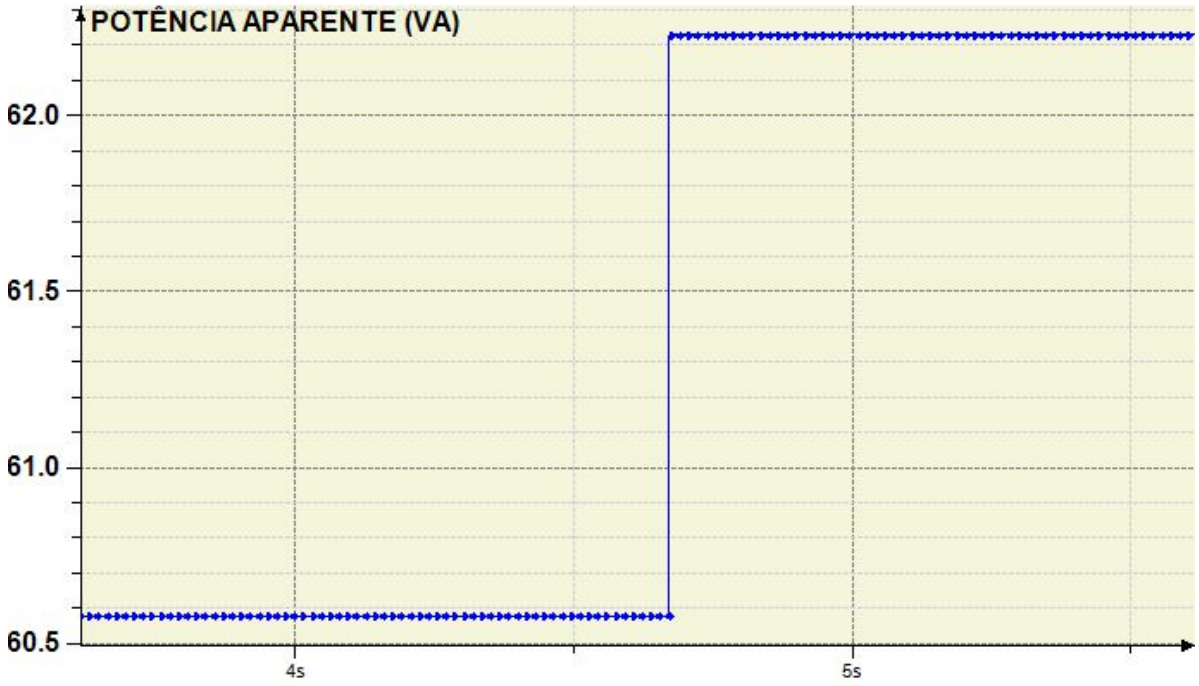

Figura 31 – Potência aparente da lâmpada incandescente.

Fonte: Do Autor.

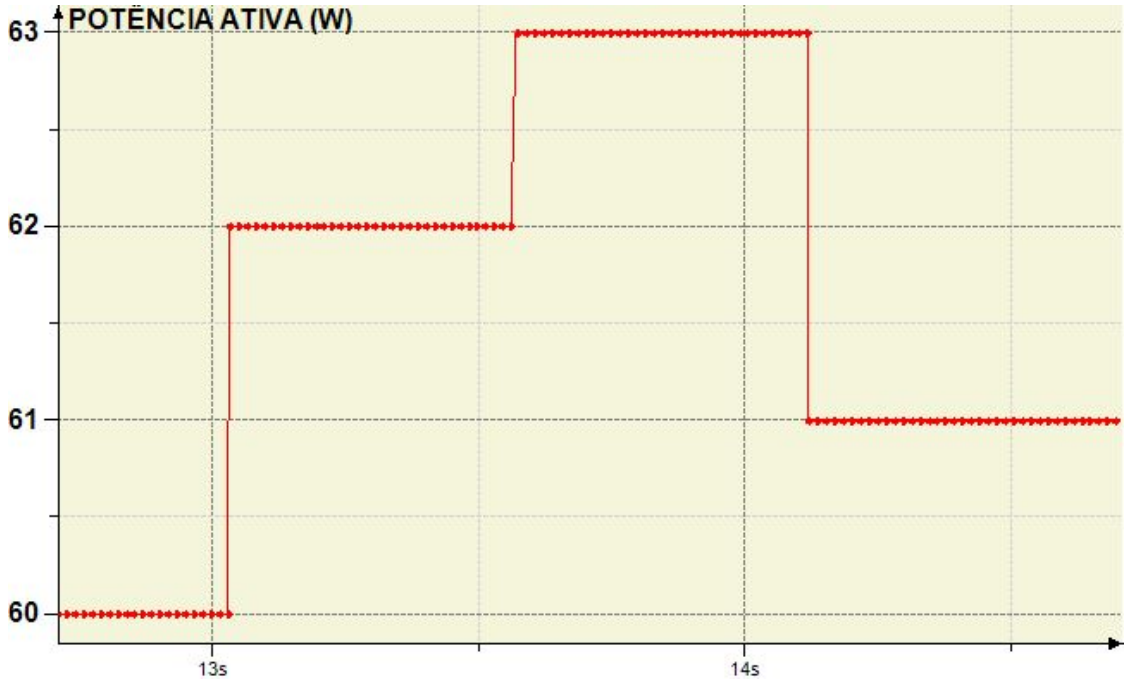

Figura 32 – Potência ativa da lâmpada incandescente.

Fonte: Do Autor.

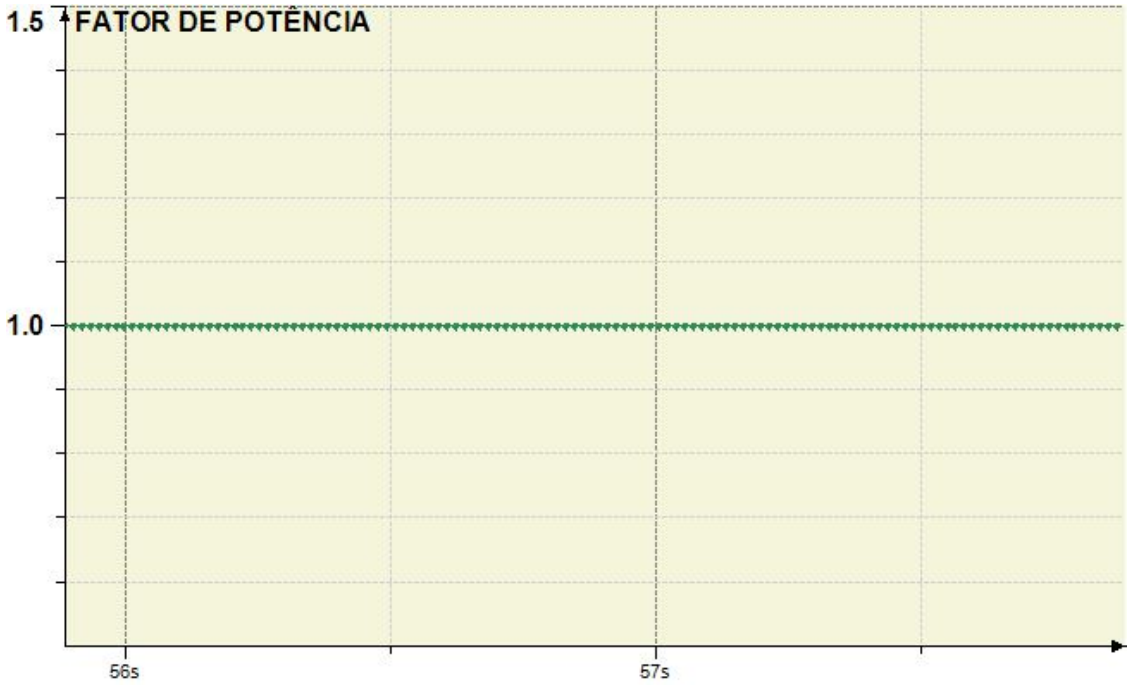

Figura 33 – Fator de potência da lâmpada incandescente.

Fonte: Do Autor.

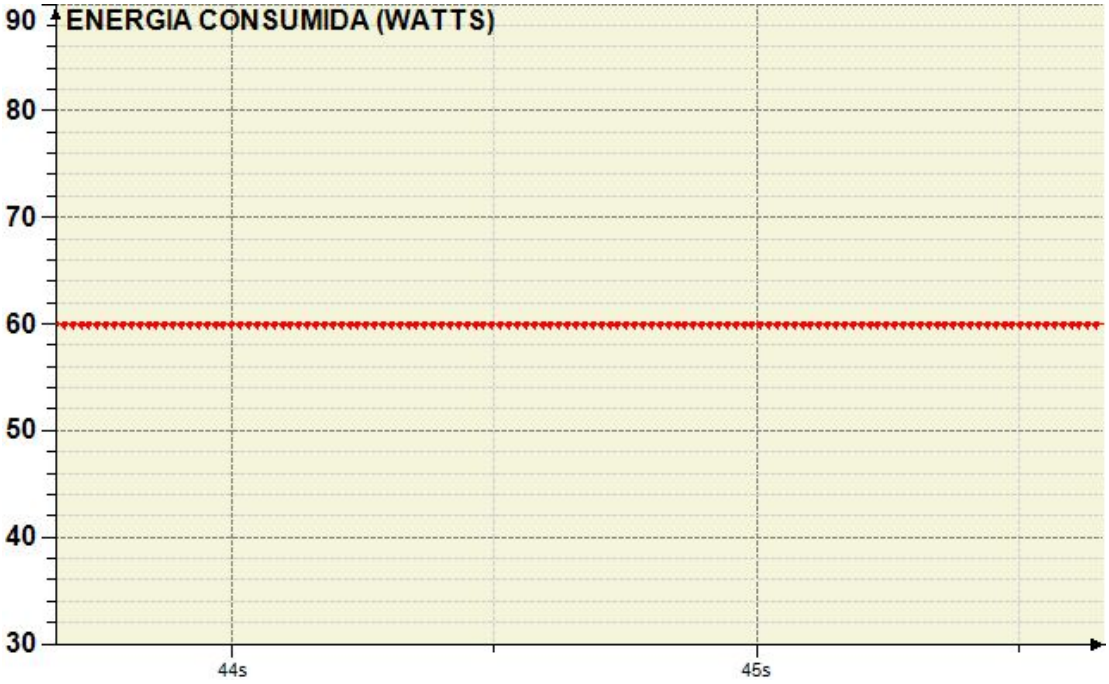

Figura 34 – Energia da lâmpada incandescente.

Fonte: Do Autor.

Tabela 2 – Custos do projeto.

Item	Custo
Arduino	R$39,59
Módulo Ethernet	R$55,82
Raspberry	R$209,90

Fonte: Do Autor.

Tabela 3 – Custo de alguns Sistemas Supervisórios de Monitoramento de Energia.

Sistema	Fabricante	Custo
Medidor de Energia lem 3350	Schneider Eletric	R$1080,00
Medidor de Consumo de Energia	Landis	R$498,44
Medidor de Energia Monofásico	Kienzle	R$530,00

Fonte: Do Autor.

Capítulo 6

CONCLUSÃO E SUGESTÕES PARA TRABALHOS FUTUROS

Este capítulo apresenta a conclusão a que se chegou com a realização do projeto, além de sugerir melhorias para o seu aperfeiçoamento.

Sobre o trabalho elaborado, os resultados levam a conclusão de que é viável a implementação de um sistema supervisório com o uso do Rasberry Pi. A conclusão da viabilidade se deve ao fato de que os resultados das grandezas: potência aparente, potência ativa, potência reativa, fator de potência e energia apresentadas ao usuário são coerentes com os resultados esperados, conforme características nominais das cargas de teste utilizadas.

Entretanto, o fato mais importante para se garantir a viabilidade do projeto, considerando a proposta principal da realização do presente trabalho, é o resultado obtido com a análise financeira, que confirma que o Rasberry Pi é uma alternativa econômica para implementação de sistemas supervisórios.

Para elaboração de trabalhos futuros, que tenham o objetivo de implementar sistemas supervisórios com alternativas econômicas aos modelos de mercado, eis algumas sugestões:

- Utilização do medidor de energia elétrica proposto nessa monografia interligado com sistemas de automação residencial, de tal forma a ter atuadores automatizados para ocasionar intervenções em caso de consumo excessivo de energia elétrica.
- Desenvolver o sistema proposto a fim de monitorar o consumo de energia em equipamentos ligados em redes trifásicas obtendo uma grande aplicabilidade em indústrias em geral.
- Realizar um maior desenvolvimento no *software* para gerenciar os dados do consumo de energia elétrica obtidos, proporcionando aos clientes um controle de consumo eficiente de energia.

Referências

Andrade, A. A.; Soma, A. T.; Nakamura, C. E. Automação de baixo custo baseada no Raspberry Pi. Trabalho em desenvolvimento, São Bernardo do Campo, 2016. 1, 10, 12

Coelho, M. S. *Sistemas Supervisórios*. [S.l.]: -, 2009. 3, 4, 5, 8

Jurizato, L. A.; Pereira, P. S. R. et al. Sistemas supervisórios. *Nova Odessa, Network Technologies*, v. 1, p. 2, 2003. 5, 7, 8

Nakaygawa, H. Controle de vazão de líquido utilizando *software* de programação de CLP. *Monografia (Trabalho de Final de Curso em Engenharia de Controle e Automação). Universidade Federal de Ouro Preto, Minas Gerais*, 2009. 17

Pinto, P. Funcionamento de um controlador lógico programável (clp). *Revista Controle de Contaminaçao. Disponivel em: http://www. pharmaster. com. br/artigos*, 2008. 11

Queiroz, M. H. d.; Cury, J. E. Controle supervisório modular de sistemas de manufatura. *Sba: Controle & Automação Sociedade Brasileira de Automatica*, SciELO Brasil, v. 13, n. 2, p. 123–133, 2002. 1, 5, 6

Ramos, M.; Andrade, V. S. Desenvolvimento, construção e calibração de uma central de monitoramento de consumo de energia elétrica e de água utilizando o microcontrolador arduino. *Anais do ENEDS*, 2015. 21, 22

Richardson, M.; Wallace, S. Primeiros passos com o Raspberry Pi. *Primeira Edição. Novatec Editora Ltda*, p. 20, 2013. 13

Souza, L. C.; Pereira, A. L. S. Estudo e aplicação de linguagens de programação utilizando o software codesys. 2015. 17, 18

ANEXOS

A1

O algoritmo utilizado para realizar a leitura da corrente e tensão de uma carga alternada e o envio dos dados através da comunicação modbus tcp/ip implementado na plataforma IDE do Arduino é apresentado a seguir.

```
#include<SPI.h> //BIBLIOTECA PARA COMUNICAÇÃO COM PERIFÉRICOS
#include<Ethernet.h> //BIBLIOTECA PARA COMUNICAÇÃO VIA ETHERNET
#include "MgsModbus.h"//BIBLIOTECA QUE REALIZA A COMUNICAÇÃO
MODBUS TCP/IP DO ARDUINO
#include "EmonLib.h"//BIBLIOTECA UTILIZADA PARA O SENSOR DE
CORRENTE SCT013

MgsModbus Mb;
EnergyMonitor emon1;

//DEFINIÇÕES DOS PARÂMETROS DE TENSÃO:
float tensao=0, vrms=0, soma=0;
long int i=0, periodo=1000;
#define pino_tensao A1

//DEFINIÇÕES DOS PARÂMETROS DE CORRENTE:
#define pino_sct A0
#define TxEnablePin 2
int inByte=0;

byte mac[]= 0x90, 0xA2, 0x0A, 0x00, 0x51, 0x06 ;
IPAddress ip(192, 168, 1, 100); //ENDEREÇAMENTO DE IP FIXO AO ARDUINO
IPAddress gateway(192, 168, 1, 1); //DEFINE O GATEWAY PADRÃO
```

```
IPAddress subnet(255, 255, 255, 0); //DEFINE A MÁSCARA DE REDE

void setup() {
pinMode(pino_sct,INPUT); //DEFINE A ENTRADA PARA O SENSOR DE
CORRENTE
pinMode(pino_tensao,INPUT); //DEFINE A ENTRADA PARA O SENSOR DE
TENSÃO
Serial.begin(9600);
analogReference(DEFAULT);
Ethernet.begin(mac,ip,gateway,subnet); //INICIA A COMUNICAÇÃO ETHERNET DO
ARDUINO
emon1.voltage(pino_tensao,67.8, 1.7); // REALIZA A CALIBRAÇÃO DA TENSÃO DA
REDE
//Pino, calibracao - Cur Const= Ratio/BurdenR. 2000/33 = 60 emon1.current(pino_sct,
60); //REALIZA A CALIBRAÇÃO DO SENSOR DE CORRENTE
}

void loop() {
Mb.MbsRun(); //REALIZA A ATUALIZAÇÃO DOS REGISTRADORES PARA
COMUNICAÇÃO MODBUS TCP/IP
emon1.calcVI(20,2000);
float realPower = emon1.realPower;
float supplyVoltage = emon1.Vrms;
Serial.println("TENSÃO : ");
Serial.println(supplyVoltage);
Serial.println("POTENCIA MEDIA : ");
Serial.println(realPower); // POTÊNCIA ATIVA
//FUNÇÃO QUE REALIZA A LEITURA DA CORRENTE DE UMA FONTE
ALTERNADA:
double Irms = emon1.calcIrms(1480);
Serial.println("Corrente : "); //EXIBE O VALOR DA CORRENTE NO MONITOR
SERIAL
Serial.println(Irms);
Mb.MbData[0]=Irms; //REALIZA O ENVIO DO VALOR DA CORRENTE VIA REDE
Mb.MbData[1]=realPower; //REALIZA O ENVIO DO VALOR DA POTÊNCIA ATIVA
VIA REDE
delay(100); //REALIZA O DELAY DE 100MS
}
```

```
//FUNÇÃO QUE REALIZA A LEITURA DA TENSÃO DE UMA FONTE
ALTERNADA:
void calcula_tensao(){
for (i=1;i<periodo;i++)
{
tensao = (analogRead(1) - 512);// REMOVE O OFFSET DA ONDA SENOIDAL
tensao = tensao*0.3277831; // CONSTANTE DE CALIBRAÇÃO DA TENSÃO
soma=soma+(tensao*tensao);
}
soma = soma/periodo;
vrms = sqrt(soma);
soma = 0;
}
```

A2

O algoritmo utilizado para implementação do sistema supervisório no *software* CoDeSys é apresentado a seguir.

```
PROGRAM PLC_PRG

//DECLARAÇÃO DE VARIÁVEIS:

VAR
CORRENTE: WORD:=0;
CORRENTEAMP: REAL:=0;
AUX: REAL:=0;
TENSAO: WORD:=0;
ENERGIA: REAL:=0;
ENERGIA2: REAL:=0;
POTENCIA: REAL:=0;
APARENTE: REAL:=0;
REATIVA: REAL:=0;
SOMAV: REAL:=0;
SOMAI: REAL:=0;
CONT: INT:=0;
PMED: WORD:=0;
ATIVA: REAL:=0;
FP: REAL:=0;
```

```
N: INT:=1000;
K: INT:=1;
VRMS: REAL:=0;
REDE: INT:=127;
IRMS: REAL:=0;
J: INT:=0;
END_VAR

//DESENVOLVIMENTO DO PROGRAMA:

CORRENTE:=%IW0; //Realiza a leitura da corrente através do registrador IW0
PMED:=%IW1; //Realiza a leitura da potência ativa através do registrador IW1
ATIVA:=PMED; //Atribui o valor da variável PMED a variável PATIVA
CORRENTEAMP:=CORRENTE; //Atribui o valor da variável CORRENTE a
variável CORRENTE AMP

IF K=N THEN

IRMS:=CORRENTEAMP;
APARENTE:=REDE*IRMS; //Realiza o cálculo da potência aparente
FP:=ATIVA/APARENTE; //Realiza o cálculo do fator de potência da carga
REATIVA:=SQRT((APARENTE*APARENTE)-(ATIVA*ATIVA)); //Realiza o
cálculo da potência reativa
ENERGIA:= ENERGIA+(ATIVA*N*(1/6000)); //Realiza o cálculo da energia
K:=0;
SOMAV:=0;
SOMAI:=0;
POTENCIA:=0;
END_IF

IF J=6000 THEN

ENERGIA2:=ENERGIA2+ATIVA; //Atualiza a exibição de energia no gráfico a
cada 1 segundo
J:=0;
END_IF

K:=K+1; //Incrementa o valor da variável auxiliar K
J:=J+1; //Incrementa o valor da variável auxiliar J
```

MINISTÉRIO DA EDUCAÇÃO
Universidade Federal de Ouro Preto – UFOP
Instituto de Ciências Exatas e Aplicadas
Colegiado do Curso de Engenharia de Elétrica

UFOP

TERMO DE RESPONSABILIDADE

O texto do trabalho de conclusão de curso intitulado "Desenvolvimento de um sistema supervisório de baixo custo para sistemas elétricos" é de minha inteira responsabilidade. Declaro que não há utilização indevida de texto, material fotográfico ou qualquer outro material pertencente a terceiros sem a devida citação ou consentimento dos referidos autores.

João Monlevade, 18 de dezembro de 2018.

Alisson Fernandes Nunes

Alisson Fernandes Nunes

MINISTÉRIO DA EDUCAÇÃO
Universidade Federal de Ouro Preto – UFOP
Instituto de Ciências Exatas e Aplicadas
Colegiado do Curso de Engenharia de Elétrica

ANEXO XI - DECLARAÇÃO DE CONFERÊNCIA DA VERSÃO FINAL

Declaro que conferi a versão final a ser entregue pelo aluno Alisson Fernandes Nunes, autor do trabalho de conclusão de curso intitulado Desenvolvimento de um sistema supervisório de baixo custo para sistemas elétricos quanto à conformidade nos seguintes itens:

1. A monografia corresponde a versão final, estando de acordo com as sugestões e correções sugeridas pela banca e seguindo as normas ABNT;
2. A versão final da monografia inclui a ata de defesa (ANEXO IV - apenas verso), a ficha catalográfica e o termo de responsabilidade (ANEXO X -) devidamente assinados.

João Monlevade, 18 de dezembro de 2018.

Vitor Costa de Silva Campos
Nome do(a) Professor(a)