Campus Calama Porto Velho
Coordenação do Curso Engenharia de Controle de Automação

RÚBRESSON INOCÊNCIO DE SOUZA

DESENVOLVIMENTO DE SISTEMA DE SUPERVISÃO VISANDO MONITORAMENTO DE IRRADIÂNCIA, TEMPERATURA, TENSÃO E CORRENTE EM PLACAS SOLARES FOTOVOLTAICAS

PORTO VELHO
2022

RÚBRESSON INOCÊNCIO DE SOUZA

DESENVOLVIMENTO DE SISTEMA DE SUPERVISÃO VISANDO MONITORAMENTO DE IRRADIÂNCIA, TEMPERATURA, TENSÃO E CORRENTE EM PLACAS SOLARES FOTOVOLTAICAS

Artigo entregue como Trabalho de Conclusão de Curso ao Instituto Federal de Educação, Ciência e Tecnologia de Rondônia (IFRO), *Campus Calama*, como requisito parcial para obtenção do grau de Bacharel, junto ao Curso de Engenharia de Controle e Automação, sob a orientação do professor Me. Artur Vitório Andrade Santos.

PORTO VELHO
2022

Ficha catalográfica elaborada pelo Sistema Gerador de Ficha Catalográfica do IFRO.

S728d
Souza, Rúbresson Inocêncio de.
Desenvolvimento de sistema de supervisão visando monitoramento de irradiância, temperatura, tensão e corrente em placas solares fotovoltaicas / Rúbresson Inocêncio de Souza, Artur Vitório Andrade Santos. - Porto Velho, 2025.
17 f.

Orientador(a): Prof. Dr. Artur Vitório Andrade Santos.

Trabalho de Conclusão de Curso (Bacharelado em Engenharia de Controle e Automação) – Instituto Federal de Educação, Ciência e Tecnologia de Rondônia - IFRO, Porto Velho, 2025.

1. Eficiência. 2. Placa Fotovoltaica. 3. Supervisório. I. Santos, Artur Vitório Andrade. II. Santos, Artur Vitório Andrade (orient.). III. Instituto Federal de Educação, Ciência e Tecnologia de Rondônia - IFRO. IV. Título.

CDD: 333.792 3

Bibliotecário(a) Responsável: Miria Santana Veiga, CRB-11/898

RÚBRESSON INOCÊNCIO DE SOUZA

DESENVOLVIMENTO DE SISTEMA DE SUPERVISÃO VISANDO MONITORAMENTO DE IRRADIÂNCIA, TEMPERATURA, TENSÃO E CORRENTE EM PLACAS SOLARES FOTOVOLTAICAS

Artigo entregue como Trabalho de Conclusão de Curso ao Instituto Federal de Educação, Ciência e Tecnologia de Rondônia (IFRO), *Campus Calama*, como requisito parcial para obtenção do grau de Bacharel, junto ao Curso de Engenharia de Controle e Automação, sob a orientação do professor Me. Artur Vitório Andrade Santos.

Aprovado em: 18/03/2022 pela banca examinadora.

_____

José Diogo Forte de Oliveira Luna
(Examinador interno)

_____

Ligia Silveria Vieira da Silva
(Examinador interno)

_____

Artur Vitório Andrade Santos
(Orientador)

DESENVOLVIMENTO DE SISTEMA DE SUPERVISÃO VISANDO MONITORAMENTO DE IRRADIÂNCIA, TEMPERATURA, TENSÃO E CORRENTE EM PLACAS SOLARES FOTOVOLTAICAS

Rúbresson Inocêncio de Souza[1] e Artur Vitório Andrade Santos[2]

Resumo: Um sistema fotovoltaico tem a necessidade de manutenção e monitoramento visando sua melhor eficiência na produção de energia. Por isso, o monitoramento de temperatura, tensão, corrente, assim como, a incidência de irradiância em placas fotovoltaicas é de suma importância. Após pesquisas bibliográficas, utilização dos hardwares (Raspberry e Arduino), sensores e módulo controlador, foi desenvolvido um sistema supervisório com base na norma ISA-101, em software *open-source* que dispusesse de linguagem moderna, bem como de mecanismos que pudesse coletar e visualizar de forma estruturada os dados das variáveis características da placa e/ou sistema fotovoltaico. Nesse cenário, tornou-se viável e de baixo custo o desenvolvimento do supervisório, pois os resultados obtidos ficaram claros e de fácil compreensão nas telas do supervisório, inclusive, trazendo a possibilidade de identificação da periodicidade da limpeza da placa fotovoltaica, após análise minuciosa dos dados armazenados.

Palavras-chave: Eficiência, Placa Fotovoltaica, Supervisório.

I. INTRODUÇÃO

A energia elétrica é um elemento necessário para a evolução social, econômica e tecnológica da humanidade. Nesse sentido, é indispensável que os meios de geração de energia elétrica sejam eficientes, com baixo impacto ambiental e acessíveis a toda população. Além disso, alguns meios de geração de energia elétrica podem ser obtidos por fontes naturais virtualmente inesgotáveis e apresentam menor interação negativa com o meio ambiente, são as chamadas fontes de energias renováveis, em específico, a energia solar captável através de um sistema fotovoltaico.

Ainda, no Brasil, a utilização da energia solar é viável no território nacional. A radiação solar é maior nas regiões mais próximas ao Equador. Ademais, a utilização de sistemas fotovoltaicos pode ser considerada uma ótima opção para geradores pontuais de energia elétrica, em residências, áreas isoladas, indústrias, entre outros. Também, destaca que os sistemas fotovoltaicos, atualmente, comercializados no Brasil possuem eficiência variável entre 10 e 16% na conversão de energia solar em energia elétrica, segundo o site Portal Solar [1].

Em relação à eficiência, os sistemas fotovoltaicos raramente operam em condições nominais de funcionamento. O seu desempenho (eficiência) depende diretamente da temperatura dos painéis e da radiação solar incidente no local [2].

Segundo [3], a irradiação solar e a temperatura são os dois principais fatores que influenciam na produção de energia pelas placas fotovoltaicas (painel solar). Além disso, o acúmulo de sujeira pode ocasionar pequenas sombras sobre as placas, podendo reduzir o rendimento do sistema como um todo, pois uma placa quando sombreada pode vir até a atuar como uma carga, levando ao seu aquecimento.

Além disso, [4], ratifica que a sujeira, depois da irradiação e da temperatura ambiente, é o fator que mais impacta o desempenho e eficiência das placas fotovoltaicas. Atualmente, existem trabalhos desenvolvidos que tem como objetivo a monitoração das variáveis elétricas dos painéis solares para otimizar a produção de energia elétrica.

No trabalho de [3], foi realizado o estudo de sujidade mostrando as diferenças entre a produção de energia elétrica em placas fotovoltaicas limpas e as sujas por acúmulo de poeira, lama, folhas, fezes ornitológicas e outros tipos de materiais que impedem o fluxo de raios solares até o interior das placas fotovoltaicas, impactando o seu rendimento na geração de energia elétrica. Para isso, as manutenções de limpeza das placas fotovoltaicas devem ser feitas periodicamente.

Corroborando com os efeitos da sujidade, [5], desenvolveram uma metodologia para quantificar a taxa de sujidade e a relação de sujidade diária em painéis fotovoltaicos, os quais foram determinados a partir de parâmetros elétricos e térmicos dos painéis.

Conforme artigo produzido por [6], confirmaram que a temperatura impacta negativamente na produção de energia elétrica, além do mais, é imprescindível o seu controle para minimizar o efeito negativo e aumentar a eficiência dos painéis.

Já [7], descreve a importância do desenvolvimento de um sistema de aquisição de dados visando uma análise no desempenho de sistema fotovoltaico. Em virtude disso, implantou um sistema de monitoramento na plataforma Arduino com isolação elétrica através de optoacopladores na medição tensão e corrente, possuindo a capacidade de readequação dos módulos permitindo inserção de novos sensores ou *shields*.

Na dissertação apresentada por [8], trouxe como relevância um sistema de medição e caracterização de painéis fotovoltaicos com uso de sensores de baixo custo, de forma a constatar que as faixas de irradiação e temperaturas podem afetar o sistema fotovoltaico.

Por fim, o [9] apresenta as tecnologias fotossensíveis, como os sensores *Ligh Dependent Resistor (LDR)*, como alternativa instrumental de baixo custo para coleta de dados de tensão e medidas de irradiação solar por meio de um *datalogger* para posterior análise estatísticas de regressões entres pontos coletados.

Observa-se que nos trabalhos desenvolvidos, apenas foram armazenados e verificados os dados elétricos do sistema fotovoltaico. Neste contexto, a implementação de um sistema de monitoramento por meio de supervisório pode auxiliar nas tomadas de decisões na manutenção de placas fotovoltaicas, além do mais, torna-se uma alternativa viável a ser concretizada. Com o intuito de verificar esta hipótese, foi desenvolvido um case temporário para implementação e simulação do desenvolvimento de um sistema de supervisão para o monitoramento em placas fotovoltaicas. Através da monitoração dos dados coletados, criar condições de manutenção preditiva e auxiliar a equipe da manutenção na limpeza periódica dos painéis solares.

II. Metodologia

A metodologia utilizada no artigo foram a realização de pesquisas bibliográficas em artigos, trabalhos de conclusão de curso, dissertações, entre outros, assim como a leitura destes, visando buscar conceitos e formas que possibilitem a resolução da hipótese proposta.

Nesse sentido, foi necessário a realização da coleta da luminosidade, utilizando uma matriz de LDR de forma análoga à uma placa fotovoltaica. Para que fosse realizado a coleta e registro dos dados, foram instalados e configurados em um PC os *softwares* e *hardwares* para manipulação dos dispositivos Raspberry PI 3 B+ e Arduino Uno.

Ademais, para capturar as variáveis do sistema, foram colocados sensores de corrente ACS 712 para *corrente e tensão*. Para leitura da temperatura, foi utilizado o sensor Termopar Max 6675. No que tange a comunicação dos *softwares* e *hardwares* empregou-se o protocolo *Modbus TCP/IP*, assim como, a utilização do *hardwares* ethernet ENC28J60 para possibilitar a comunicação entre os dispositivos.

Foram instalados e configurados *softwares* IDE *(Integrated Development Environment)* Arduino para captura dos dados dos sensores, assim como, o Codesys para criação do supervisório e lógica de controle. Para fazer uso dos recursos do Codesys e de sua aplicação com o Raspberry Pi, realizou-se o download do software com a versão V3.5 SP16 Patch 7. Para possibilitar o controle do Raspberry Pi 3 B+, realizou-se o download e instalação do pacote *"Codesys Control for Raspberry Pi"* e, por meio dele, pode ser programado para ser aplica do como um Controlador Lógico Programável (CLP) virtual [10], conforme Fig. 1.

Ainda, no Codesys empregou-se como padrão a linguagem de programação *"Ladder"* para criar as variáveis a serem monitoradas pelo supervisório no *software* Codesys. Para o funcionamento do Raspberry Pi 3 B+ foi necessário o uso do sistema operacional *Raspbian*, que permitiu o controle e acesso aos recursos dos programas e aplicações do hardware [10], conforme Fig. 2.

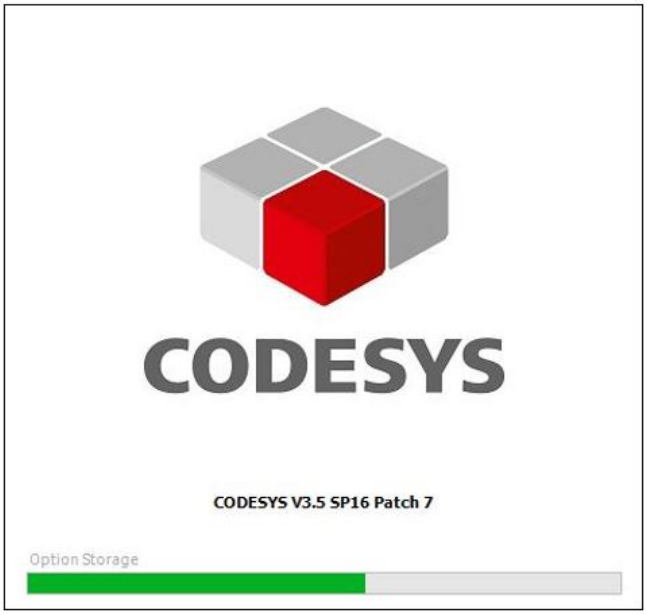

Fig. 1. Processo de instalação e configurações iniciais do Codesys. Fonte: Autor

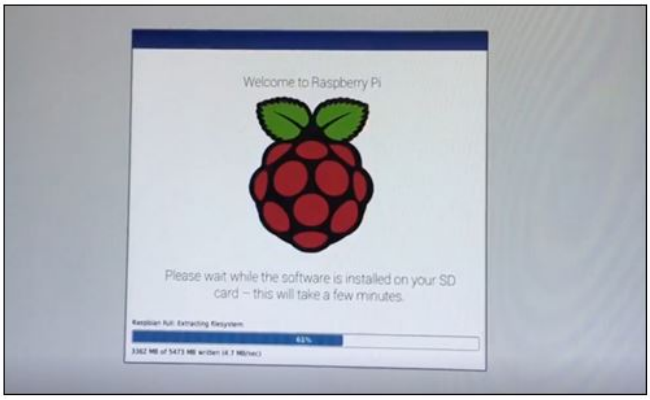

Fig. 2. Processo de instalação e configurações iniciais do Raspberry Pi 3 B+. Fonte: Autor

A. Planta de funcionamento

O funcionamento da planta é descrito na Fig. 3, a placa fotovoltaica gera as variáveis (temperatura, corrente e tensão) a serem medidas, assim como os LDR a luminosidade, os quais serão capturadas pelo *hardware* Arduino e posteriormente visualizadas no sistema de supervisão criado no Codesys. O sistema de supervisão será desenvolvido com base na norma ISA-101, que descreve o desenvolvimento de interfaces homem máquinas (IHM). O sistema de supervisão tem seu funcionamento em um computador pessoal e que através do sistema de comunicação Ethernet, captura e envia as informações para o Raspberry Pi 3 B+ [10].

B. Relação de materiais utilizados

Para possibilitar a implementação do sistema de supervisão foi desenvolvido um case com uma arquitetura de

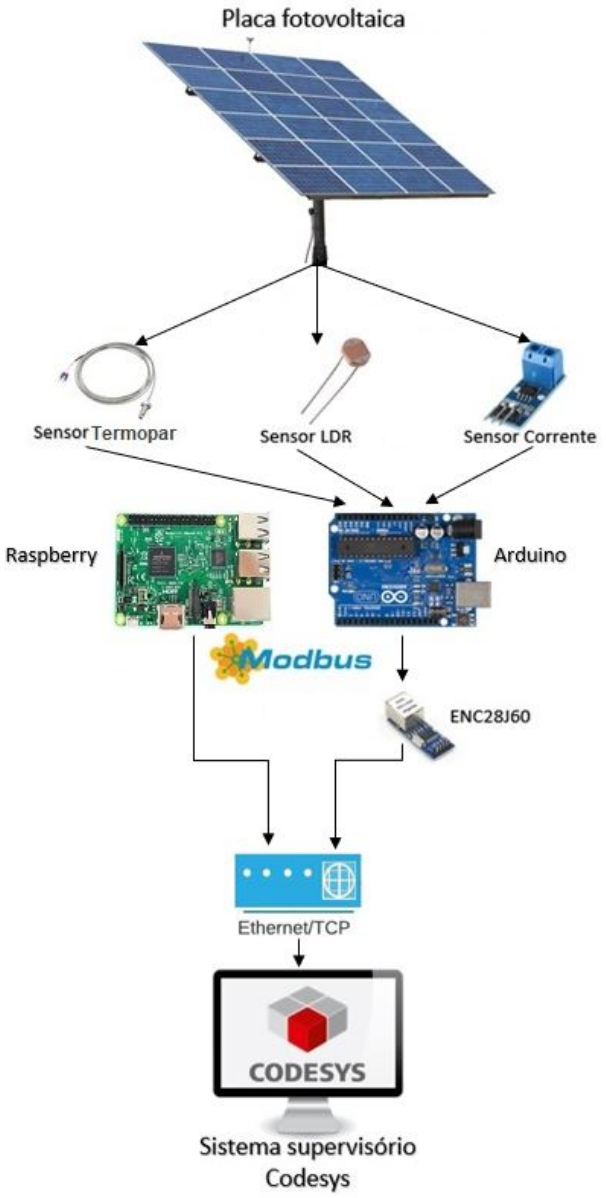

Fig. 3. Planta de funcionamento do projeto de pesquisa. Fonte: Autor.

sensores para monitoração das informações via Arduino e Raspberry Pi. Para construção do case, foram necessários materiais elétricos e eletrônicos de acordo com a Tab. I.

TABLE I
MATERIAIS UTILIZADOS NA PRODUÇÃO DO CASE

ID	Descrição	Quantidade
1	Arduino Uno SMD R3	1
2	Raspberry Pi 3 B+	1
3	Bateria Selada intelbras 12V	1
4	Epever LS0512E 5A 12 V	1
5	Termopar com mód.Max6675	1
6	Sensor Corrente ACS712	1
7	Módulo Ethernet ENC28J60	1
8	LDR 5mm	4
9	Resistor 3.3KOhms	4
10	Kit Jumpers	1
11	Protoboard	1
12	Cabo Ethernet 15m	1
13	Caixa de passagem BRUM	1

III. FUNDAMENTAÇÃO TEÓRICA

A necessidade de energia para o crescimento de um país é indiscutível, todavia a aplicabilidade de tecnologias que procuram o aumento da eficiência energética e a sustentabilidade da produção traz garantia e proporciona um crescimento com redução de impactos ambientais [11].

O Sol através da emissão de radiação solar constitui-se uma inesgotável fonte energética. A radiação solar possibilita um enorme potencial de seu aproveitamento mediante sistemas de captação e conversão, como por exemplo a térmica e a fotovoltaica. No que tange ao efeito fotovoltaico, tem-se que é um fenômeno físico que permite obtenção de energia elétrica, através da conversão direta da energia contida na radiação luminosa, pelo material semicondutor de que é formada a placa fotovoltaica [12].

As placas fotovoltaicas mostram determinados fatores que influenciam na potência gerada, como a irradiância e a temperatura. A irradiância afeta a corrente gerada, de modo que quanto mais irradiância a placa recebe, mais corrente ela é capaz de gerar. Por sua vez, o aumento da temperatura ambiente ocasiona uma queda de tensão gerada pelas placas e um pequeno aumento na corrente, porém não sendo capaz de suprir as perdas causadas pela queda de tensão [2]. Ainda, as elevadas temperaturas podem degradar significativamente as placas fotovoltaicas e, por conseguinte, reduzir a vida útil do módulo [6].

Um outro fator é a sujidade ocasiona pequenas sombras sobre as placas solares, na qual pode influenciar na diminuição da radiação solar sobre a placa fotovoltaica, por consequência, a sua corrente de geração fica limitada reduzindo o rendimento do sistema fotovoltaico como um todo. Uma placa sombreada pode vir atuar como uma carga, por ocasião do aquecimento [3].

Segundo [10], sistemas supervisórios dentro de um processo produtivo ou instalação física permitem que informações sejam monitoradas e rastreadas. As informações são coletadas por meio de equipamentos de aquisição de dados, manipuladas, analisadas, armazenadas e posteriormente apresentadas ao usuário. Sendo assim, o objetivo principal de um sistema supervisório é fornecer recursos ao operador para controlar e monitorar um processo automatizado com maior rapidez, assim como, gerenciar e controlar do processo na tomada de decisões [13].

De acordo com [14], a energia solar fotovoltaica, necessita de monitoramento frequente de seu funcionamento, através da aquisição de dados de irradiância e a utilização de sensores para leitura de tensões e correntes, assim como, da temperatura na superfície da placa fotovoltaica.

A. Software Codesys

O Codesys difere das opções anteriores, que são *open-source*, por ser um *software* proprietário. A utilização do ambiente de desenvolvimento através do sistema operacional

Windows é gratuita e sem limitações, contudo o funcionamento do *runtime* do Raspberry Pi está limitado, na versão gratuita, a duas horas. O preço da licença para remover a limitação é de 50 Euros, havendo a possibilidade da sua aquisição depois da validação do projeto. Ainda, o *Codesys* satisfaz todos os requisitos iniciais relacionados com o protocolo de comunicação *Modbus TCP/IP*. Além do mais, no software é possível a ligação de um sistema SCADA (*Supervisory Control and Data Acquisiton*) e o desenvolvimento de painéis, acessíveis localmente e remotamente por páginas web com suporte HTML5. O servidor web para acesso aos painéis é parte integrante do *Runtime do Codesys* [8].

A plataforma *Codesys*, desenvolvida pela empresa de *software* alemã 3S-Smart, utiliza o padrão de linguagens de programação para *hardwares* de controle. 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, quais sejam: Texto Estruturado; Lista de Instruções; *Ladder*; Diagrama de blocos funcionais; SFC (*Sequential Function Charts*) [10].

O *Codesys* 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 [10].

B. Raspberry

O Raspberry Pi (Fig. 4) 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 (USB) [10].

O que torna o Raspberry Pi um super dispositivo é a harmonização entre a possibilidade de programação científica utilizando linguagens de alto nível e linguagens mais populares e, suas portas *General Purpose Input/Output (GPIO)* que são responsáveis pela entrada e saída de dados [15]. Ademais, o Raspberry Pi apresenta algumas vantagens, são elas: consumo de energia; não possui partes móveis; tamanho reduzido; capacidade de expansão; baixo custo; entre outras [10].

C. Placa Arduino

O Arduino (Fig. 5) 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 neste artigo, o Arduino tem a função de fazer a leitura dos valores de

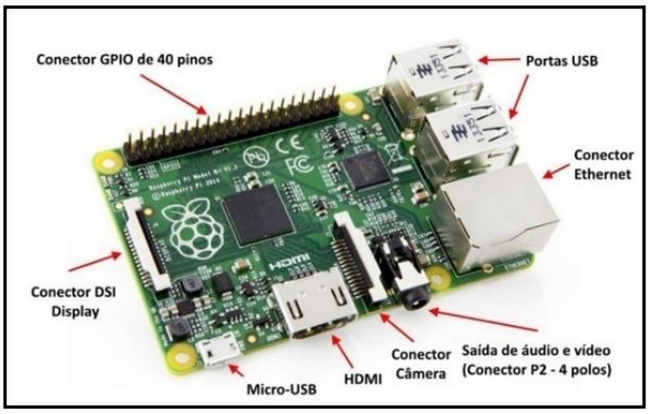

Fig. 4. Placa do Raspberry Pi. Fonte: [15].

temperatura, tensão, corrente e irradiância, sendo responsável por realizar a comunicação via porta *Ethernet* com o Raspberry Pi, enviando dados a todo momento. Essa proposta surgiu devido o microcomputador Raspberry Pi não possuir um conversor analógico-digital integrado, o que é essencial para se realizar as medições. As vantagens para utilização da Placa Arduino são: ótimo custo benefício; plataforma de programação bastante intuitiva; e, a placa que permite a integração com diversos sensores [10].

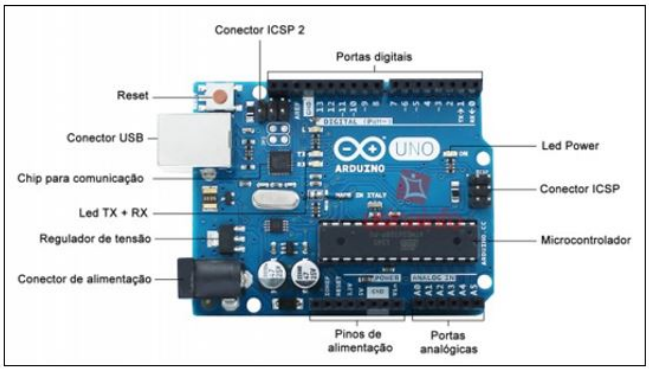

Fig. 5. Placa Arduino com detalhamento. Fonte: [16]

1) Arduino IDE: Para a placa Arduino ser carregada com a programação e consequentemente executada a ação, o Arduino IDE é utilizado para produzir e compilar um *Sketch* [17].

D. Comunicação

1) Protocolo Modbus: É um protocolo aberto que define uma estrutura de mensagens de comunicação usadas para transferir dados discretos e analógicos entre dispositivos microprocessados com detecção e informação de erros de transmissão. O protocolo Modbus faz comunicação usando o método simples de mestre-escravo (cliente-servidor). O dispositivo mestre inicia as operações (chamadas consultas) e os escravos respondem, fornecendo os dados solicitados para o mestre, ou executando a ação solicitada na consulta [18].

2) Controlador Ethernet ENC28J60: É um controlador *Ethernet* autônomo com uma Interface Periférica Serial padrão da indústria (SPI). Projetado para servir como uma rede *Ethernet interface* para qualquer controlador equipado com SPI. O ENC28J60 (Fig. 6), atende a todas as especificações IEEE 802.3. Ainda, no *Datasheet* consta a forma de ligação do módulo [19].

Fig. 6. Módulo do Controlador ENC28J60. Fonte: [19].

E. Sensores

1) Sensor Resistor Dependente de Luz (LDR): O Sensor LDR (*Light Dependent Resistor* – Resistor Dependente de Luz) é um sensor resistivo analógico cuja resistência varia de acordo com a luminosidade. Quando submetido a uma luz cada vez mais intensa, pode-se verificar que sua resistência 1 diminuirá gradativamente. Utilizando um circuito divisor de tensão, podemos fazer com que através dessa variação da resistência, haja uma variação na tensão [20], conforme observa-se na Fig. 7.

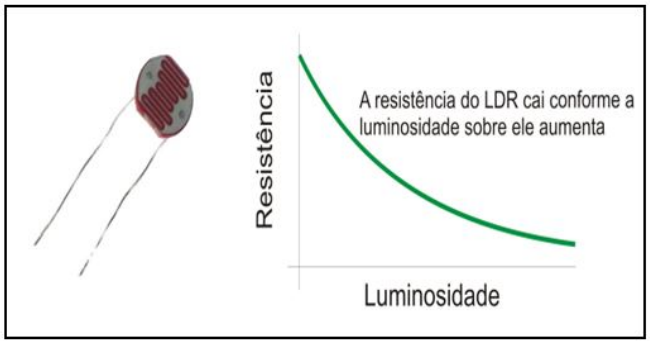

Fig. 7. LDR e gráfico Resistência x Luminosidade. Fonte: [20].

2) Sensor Termopar Tipo K com módulo MAX6675: Este tipo de sensor de temperatura já vem com o módulo que trata os sinais recebidos, facilitando bastante o uso do termopar. O Max6675 (Fig. 8), realiza compensação de junção fria e digitaliza o sinal de um termopar tipo K. Os dados são emitidos em uma resolução de 12 bits, compatível com SPI. O conversor resolve temperaturas até 0,25°C e permite leituras tão altas quanto 1024°C [21].

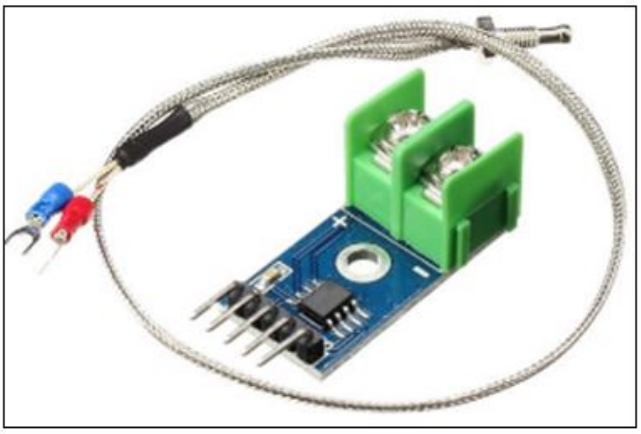

Fig. 8. Sensor de temperatura Max6675. Fonte: [22]

,

3) Sensor de corrente ACS712: Denominado Allegro ACS712 (Fig. 9), oferece um desempenho econômico e preciso para detecção de corrente AC ou DC em áreas industriais, comerciais, e sistemas de comunicações. Aplicações típicas incluem controle de motor, detecção e gerenciamento de carga, fontes de alimentação chaveadas e proteção contra falhas de sobrecorrente [23].

Fig. 9. Sensor de corrente ACS712. Fonte: [24].

,

F. Norma ISA-101

A ISA-101 é uma norma, que visa direcionar o design, implementação e manutenção de interfaces homem-máquina

(IHM) para processos de sistemas automatizados, resultando em IHMs mais seguras, eficazes e que proporcionam um controle mais eficiente de processos. Além de aprimorar a habilidade dos usuários de detectar, diagnosticar e agir de forma apropriada a situações incomuns [25].

IV. Apresentação e Análise dos Dados

A. Desenvolvimento elétrico e eletrônico.

Utilizando quatro sensores resistivos dependente de luminosidade, conhecido com LDR, conectados em série com os resistores *pull-down*, tendo a qualidade de um divisor de tensão, onde nos terminais foi possível coletar o sinal analógico da luminosidade através das saídas analógicas A1, A2, A3 e A4, através do microcontrolador Arduino Uno. A conexão da matriz LDR no Arduino, pode ser observada conforme Fig. 10 e o esquemático produzido no *TinkerCad* através da Fig. 11.

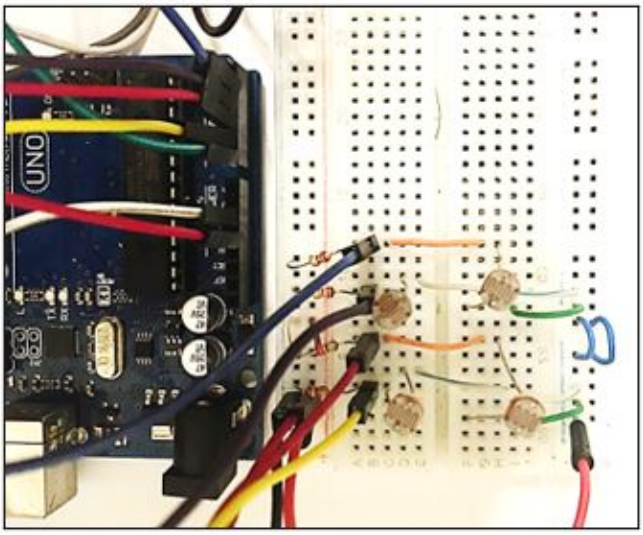

Fig. 10. Conexão da Matriz LDR a placa Arduino. Fonte: Autor.

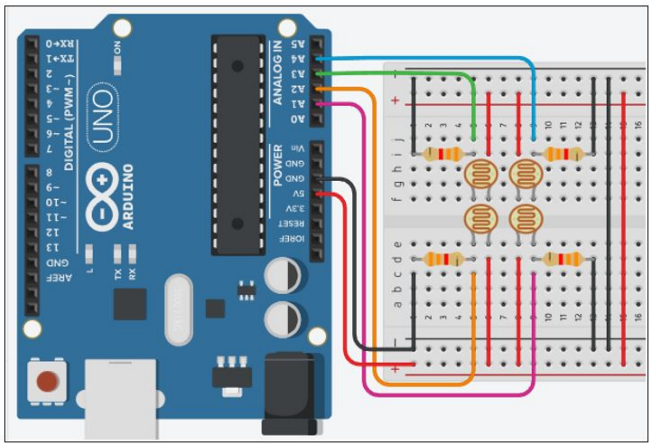

Fig. 11. Esquemático no *TinkerCad* da Matriz LDR conectada a placa Arduino. Fonte: Autor.

Posteriormente, o sensor de corrente ACS712 foi conectado na saída analógica A0 da placa Arduino (Fig. 12), visando a coleta da corrente e, utilizando o *firmware* apresentado na seção seguinte, os cálculos da mensuração da tensão do sistema de alimentação da placa fotovoltaica. O fio positivo que alimenta a placa foi conectado respectivamente nos bornes verdes. O *led* vermelho demonstra que o sensor está ligado em tensão 5V, através da Fig. 13 pode ser observado o esquemático produzido no *TinkerCad*.

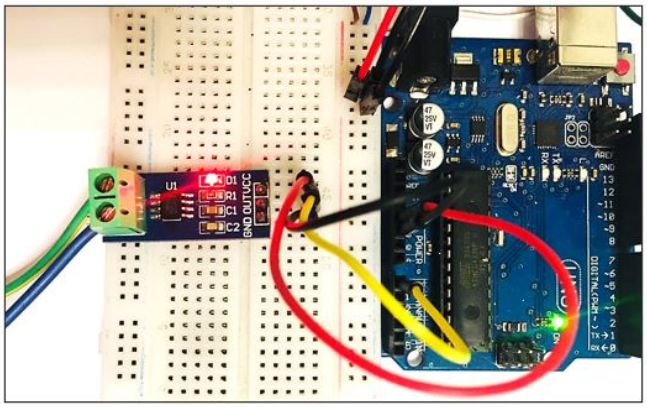

Fig. 12. Conexão do sensor ACS712 na placa Arduino. Fonte: Autor.

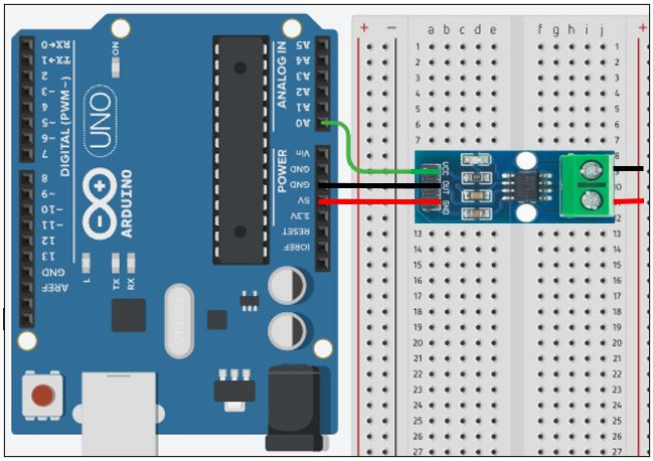

Fig. 13. Esquemático no *TinkerCad* da conexão do sensor ACS712 na placa Arduino. Fonte: Autor

Já o sensor termopar tipo K com módulo Max6675, foi utilizado em conexão com o Arduino nas portas de saídas digitais 3, 4 e 5, para obtenção dos dados de temperatura ao ser encapsulado na placa fotovoltaica, de acordo com a conexão física mostrada na Fig. 14 e esquemático produzido no *TinkerCad* na Fig. 15.

Realizadas as conexões elétricas e eletrônicas dos componentes e sensores com a placa Arduino, bem como, a conexão elétrica do microcomputador Raspberry Pi 3 B+ e os seus periféricos teclado, mouse e monitor (Fig. 16), montou-se as partes citadas anteriormente em um *case* temporário de monitoramento das variáveis. Ainda, como partes do *case*,

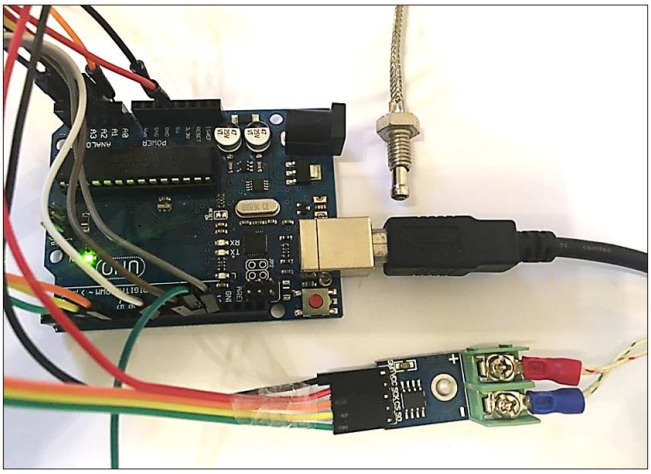

Fig. 14. Conexão do sensor Termopar na placa Arduino. Fonte: Autor.

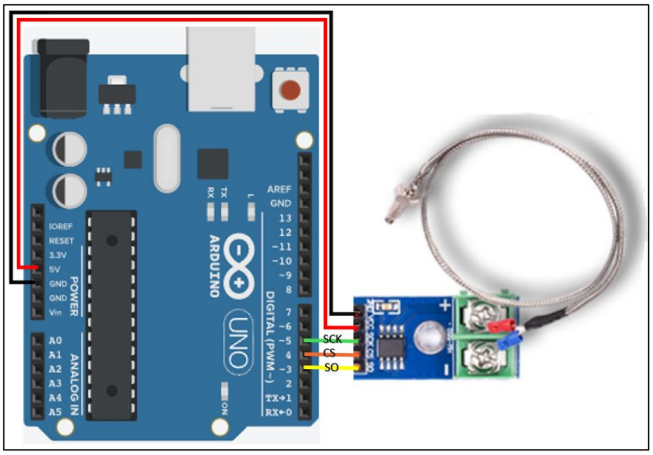

Fig. 15. Esquemático no *TinkerCad* da Conexão do sensor Termopar na placa Arduino. Fonte: Autor.

utilizou-se de uma bateria estacionária 12V e um controlador de carregamento solar Epever 12V, os quais juntamente com a energia fornecida pela placa solar, forneciam a alimentação elétrica contínua aos componentes do *case*, conforme pode ser observado através da montagem da Fig. 17.

Por conseguinte, realizou-se a instalação do *case* temporário em campo (Fig. 18) em conjunto com a placa fotovoltaica YINGLI YL022-17B-1/7–22WP, objetivando a realização da coleta dos dados das variáveis e também a posterior análise e tratamento dos dados pelo *software Excel*.

Em seguida, foram efetuadas as configurações do Raspberry PI 3 B+ em conexão com a rede *Ethernet* via *Wifi* e identificação do IP do Raspberry constante no VNC Server. Além disso, as configurações de senhas exigidas na comunicação com o *Codesys* e habilitação de interfaces em *Applications Menu>Preferences>Raspberry Pi Configuration>System>Password e Interfaces*, conforme observado na Fig. 19.

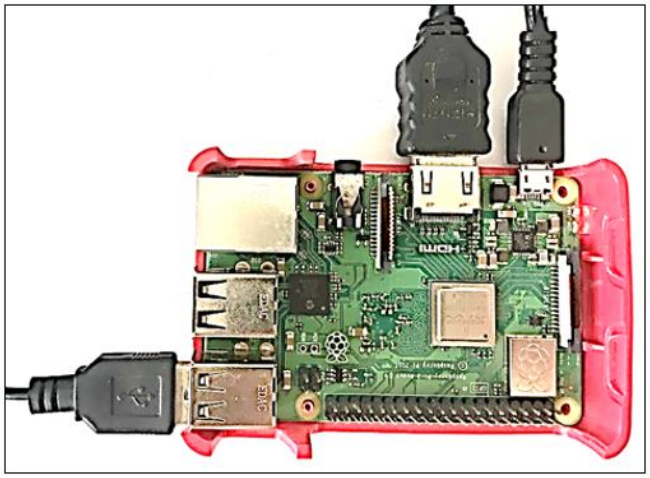

Fig. 16. Conexão Elétrica Raspberry PI 3 B+ e periféricos. Fonte: Autor.

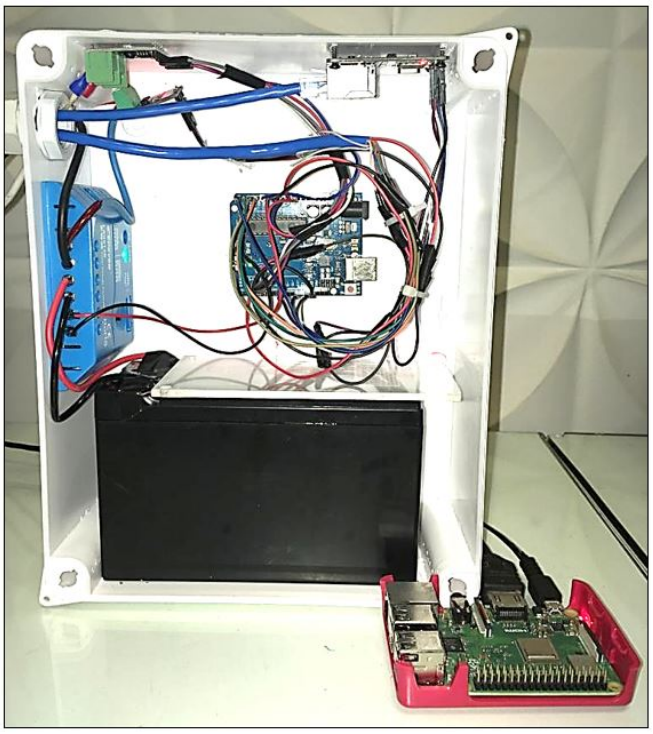

Fig. 17. Case temporário para o desenvolvimento do sistema de supervisão. Fonte: Autor.

B. Firmware *no Arduino IDE*

Para possibilitar a interpretação dos dados elétricos pelo Arduino a ser transmitido para Raspberry Pi, o *firmware* desenvolvido para aplicação, consiste no *Sketch* onde primeiramente, foram adicionadas as bibliotecas Max6675.h para o sensor Max6675; EtherneCard.h com o *driver* IPv4 para módulo ENC28J60; Modbus.h e ModbusIP; ENC28J60.h para o módulo ENC28J60 e o protocolo Modbus TCP, através do comando *include* <biblioteca>, visando o desenvolvimento do algoritmo para comunicação, conexão à internet e reconhecimento do Termopar, conforme pode ser observado

Fig. 18. Case temporário instalado em campo para o desenvolvimento do sistema de supervisão. Fonte: Autor.

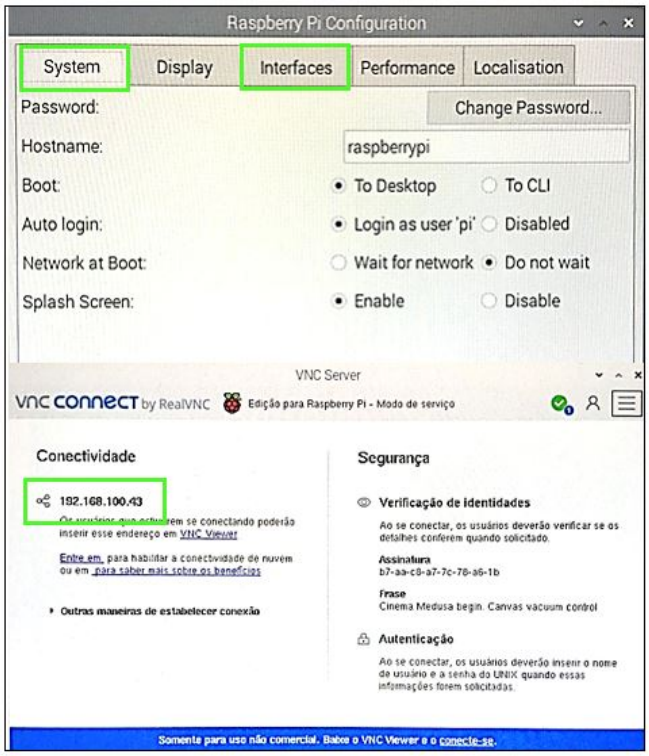

Fig. 19. Configurações Raspberry PI 3 B+. Fonte: Autor.

na Fig. 20.

Em seguida, foram definidos os endereços IP da interface *Ethernet* do Arduino, assim como o *Gateway* da rede. O IP utilizado foi o residencial 192.168.100.10.

Posteriormente, no ambiente de programação, foram inseridos os comandos de configuração do registro *Modbus* para representar os valores das variáveis. Este valor é o deslocamento (baseado em 0) a ser colocado no *software* de supervisão. O software utiliza deslocamentos baseados em 1, o valor definido foi a partir de 100. A declaração dos

```
ModbusTCP_V1003_corrente_e_tens_oV3 §
// inclusão da biblioteca do sensor/transdutor de sinais
#include <max6675.h>
// inclusão da biblioteca para comunicação Ethernet
#include <EtherCard.h>
// inclusão da biblioteca IP Modbus apra chip ENC28J60
#include <ModbusIP_ENC28J60.h>
// inclusão da biblioteca base do protocolo Modbus
#include <Modbus.h>
```

Fig. 20. Bibliotecas inclusas no algoritmo desenvolvido. Fonte: Autor.

registros *Modbus* para leitura pode ser observado na Fig. 21.

```
ModbusTCP_V1003_corrente_e_tens_oV3 §
const int SENSOR_IREG_CORRENTE = 100;
const int SENSOR_IREG_LUMINO1 = 101;
const int SENSOR_IREG_LUMINO2 = 102;
const int SENSOR_IREG_LUMINO3 = 103;
const int SENSOR_IREG_LUMINO4 = 104;
const int SENSOR_IREG_VOLTAGE = 105;
const int SENSOR_IREG_TEMP = 106;
```

Fig. 21. Declaração dos registros Modbus para leitura das variáveis. Fonte: Autor.

Continuando a programação, no ambiente do Arduino IDE, no *void setup*, foram adicionados os comandos de registrador do tipo de entrada analógica, as quais são responsáveis pelo acionamento das variáveis, assim como, a verificação de seus estados (Fig. 22).

```
ModbusTCP_V1003_corrente_e_tens_oV3 §
void setup()
{
  mb.config(mymac, myip); // configura IP Modbus
 // Adiciona registrador tipo registro de entrada analógicas
  mb.addIreg(SENSOR_IREG_CORRENTE);
  mb.addIreg(SENSOR_IREG_VOLTAGE);
  mb.addIreg(SENSOR_IREG_LUMINO1);
  mb.addIreg(SENSOR_IREG_LUMINO2);
  mb.addIreg(SENSOR_IREG_LUMINO3);
  mb.addIreg(SENSOR_IREG_LUMINO4);
  mb.addIreg(SENSOR_IREG_TEMP);
```

Fig. 22. Declaração dos registro de entrada analógica. Fonte: Autor.

No *void loop* (Fig. 23), foram inseridos os comandos para realização das leituras/escrita das variáveis analógicas definidas em mb.addIreg. Cada tipo de variável deve ser declarada conforme sua especificidade, se for analógica/digital, leitura/escrita, conforme os parâmetros *Modbus*. Também foi adicionado os comandos para cálculo e conversão da corrente e voltagem através do sensor ACS712 (Fig. 24).

Todo o código desenvolvido está disponível via *GitHub* para ser feito o download e testado,

```
ModbusTCP_V1003_corrente_e_tens_oV3 §

    //Realiza o valor do registro da variáveis analógicas
    //definidas em mb.addIreg para acioná-la.
    mb.Ireg(SENSOR_IREG_CORRENTE, analogRead(Amps));
    mb.Ireg(SENSOR_IREG_VOLTAGE, analogRead(Voltage));
    mb.Ireg(SENSOR_IREG_LUMINO1, analogRead(sensorPin1));
    mb.Ireg(SENSOR_IREG_LUMINO2, analogRead(sensorPin2));
    mb.Ireg(SENSOR_IREG_LUMINO3, analogRead(sensorPin3));
    mb.Ireg(SENSOR_IREG_LUMINO4, analogRead(sensorPin4));
    mb.Ireg(SENSOR_IREG_TEMP, thermocouple.readCelsius());
```

Fig. 23. Comando de acionamento das entradas analógicas. Fonte: Autor.

```
ModbusTCP_V1003_corrente_e_tens_oV3 §

}
void Calcula_corrente_voltage()
{ //calcula a corrente e voltagem
  RawValue = analogRead(sensorPin);
  Voltage = (RawValue / 1024.0) * 5000;
  Amps = ((Voltage - ACSoffset) / mVperAmp);
}
```

Fig. 24. Comandos para cálculos e conversão de tensão e corrente. Fonte: Autor.

assim como, aprimorado, através do link: *https://github.com/rubresson/Supervisorio2.git*.

C. Desenvolvimento do Supervisório

Sistemas supervisórios permitem o monitoramento de informações de processos produtivos ou instalações físicas, através de coletas por meio de equipamentos de aquisição de dados, visando a manipulação, análise e armazenamento para facilitar a interação com seu usuário. Um sistema supervisório é conhecido como *Supervisory Control And Data Acquisition (SCADA)*, ou seja, Sistema de Supervisão e Aquisição de Dados [10].

Inicialmente, para desenvolvimento do supervisório foi realizado os testes de conexão à rede internet, do Raspberry PI 3 B+ via *Wifi* e do Arduino pelo controlador ENC28J60. Após carregar o *firmware* na placa do Arduino, a verificação de conexão é efetuada no *Prompt* de Comando do *Windows* mediante comando *ping* e o IP do Raspberry (192.168.100.43) e o IP inserido no *firmware* (192.168.100.10) para comunicação do Arduino. A referida conexão é estabelecida quando o IP dispara e as respostas são enviadas em tempo e as respostas são confirmadas via parâmetro TTL, conforme observado na Fig. 25.

Seguidamente, no Codesys foram realizadas as configurações de credenciais de *login*, o *Scan* para seleção do IP do Raspberry Pi e a biblioteca *Runtime Package Raspberry 4.0.0.0*, para efetivação das comunicações entre *Hardwares* e *Softwares* (Fig. 26). Uma vez sendo realizadas as configurações anteriores, criou-se um projeto padrão denominado "Supervisório2", com um dispositivo programável *Codesys Control for Raspberry PI SL(3S – Smart Software Solutins*

```
Prompt de Comando

C:\Users\FAMILIA>ping 192.168.100.43

Disparando 192.168.100.43 com 32 bytes de dados:
Resposta de 192.168.100.43: bytes=32 tempo=2ms TTL=64
Resposta de 192.168.100.43: bytes=32 tempo=11ms TTL=64
Resposta de 192.168.100.43: bytes=32 tempo=4ms TTL=64
Resposta de 192.168.100.43: bytes=32 tempo=3ms TTL=64

Estatísticas do Ping para 192.168.100.43:
    Pacotes: Enviados = 4, Recebidos = 4, Perdidos = 0 (0% de
             perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 2ms, Máximo = 11ms, Média = 5ms

C:\Users\FAMILIA>ping 192.168.100.10

Disparando 192.168.100.10 com 32 bytes de dados:
Resposta de 192.168.100.10: bytes=32 tempo=1ms TTL=64
Resposta de 192.168.100.10: bytes=32 tempo=27ms TTL=64
Resposta de 192.168.100.10: bytes=32 tempo=1ms TTL=64
Resposta de 192.168.100.10: bytes=32 tempo=1ms TTL=64

Estatísticas do Ping para 192.168.100.10:
    Pacotes: Enviados = 4, Recebidos = 4, Perdidos = 0 (0% de
             perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 1ms, Máximo = 27ms, Média = 7ms

C:\Users\FAMILIA>
```

Fig. 25. Resposta da conexão com Raspberry Pi e Arduino. Fonte: Autor.

GmbH) com tarefa cíclica que chama PLC-PRG em *Ladder Logic Diagram*(LD), ambas configuradas na criação do projeto (Fig. 27).

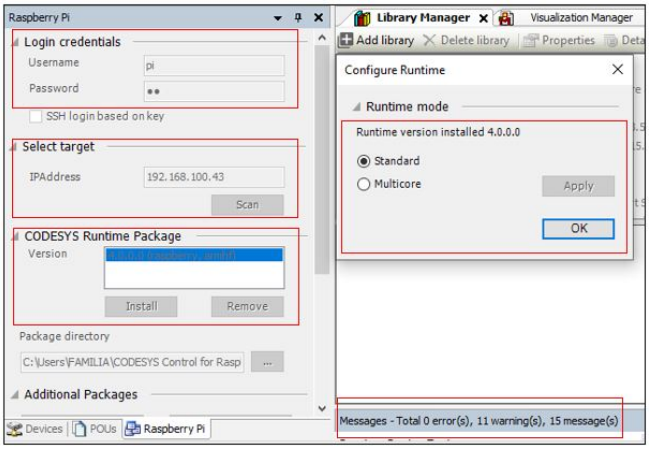

Fig. 26. Configurações de credenciais, IP e biblioteca. Fonte: Autor.

Posteriormente, para configurar a comunicação com o Arduino em maneira *Master* (mestre) e *Slave* (escravo) no Codesys, utilizou-se em *Device (Codesys Control for Raspberry Pi SL) > Add Device > Ethernet Adapter>3S - Smart Software Solutions GmbH* 3.5.1.0 (Fig. 28). Após gerado o *Ethernet Adapter* para criação do *Master* e *Slave*, adicionou-se ambos em *Ethernet Adapter > Add Device > Fieldbuses > ModBus > Modbus TCP Master* e o *Slave Device* (Fig. 29).

Ainda, pelo *Scan Network*, foi efetuado o teste de comunicação entre o Raspberry Pi e Codesys. Por conseguinte, em *Ethernet > General*, inseriu-se a *Wlan0, IP Address* 192.168.100.43, entre outras configurações, visando comunicar via *modbus*, o *codesys* com o raspberry (Fig. 30).

Após, foi implementada a configuração do *Slave* (escravo),

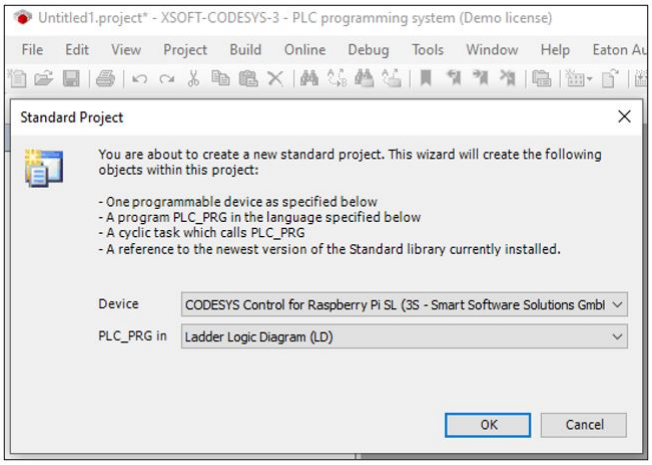

Fig. 27. Configuração de criação de projeto padrão no Codesys. Fonte: Autor.

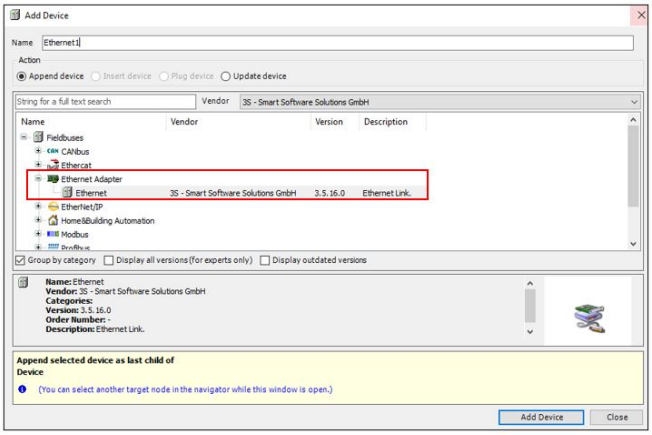

Fig. 28. Inserção do Ethernet Adapter no Codesys. Fonte: Autor.

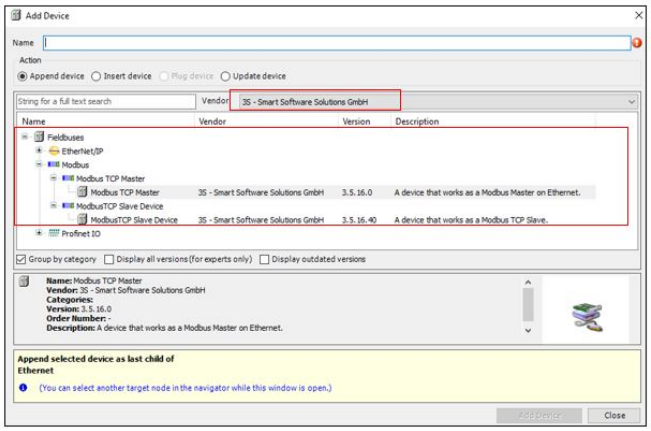

Fig. 29. Inserção da comunicação Modbus Master/Slave no Codesys. Fonte: Autor.

em *Ethernet* > *Modbus TCP Slave Device*> *General*, inserindo o IP utilizado no *firmware* compilado para a placa do Arduino, assim, como outras configurações (Fig. 31).

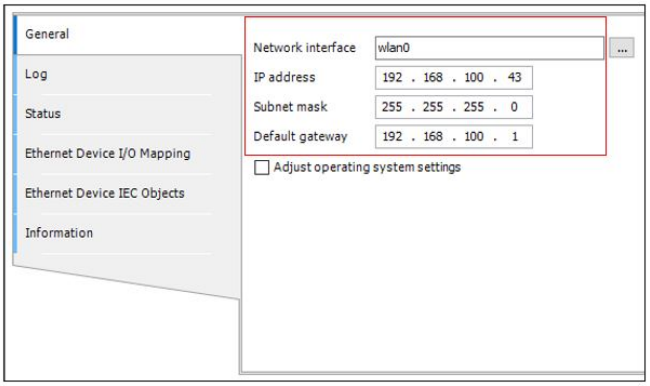

Fig. 30. Configuração do Ethernet General no Codesys. Fonte: Autor.

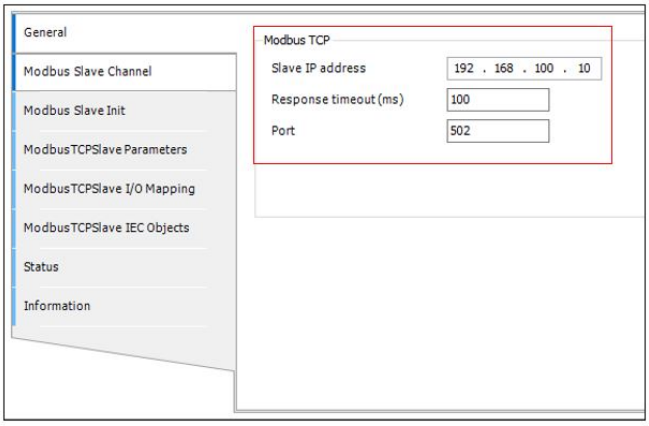

Fig. 31. Configuração do *Slave* (escravo) no Codesys. Fonte: Autor.

Em razão das ações de configurações realizadas, foi efetuado o acionamento para verificação da comunicação entre o Raspberry, Arduino e o *Codesys*, mediante o *Login* e *Start*, demonstrada pela indicação nas marcações em verde em destaque na Fig.32.

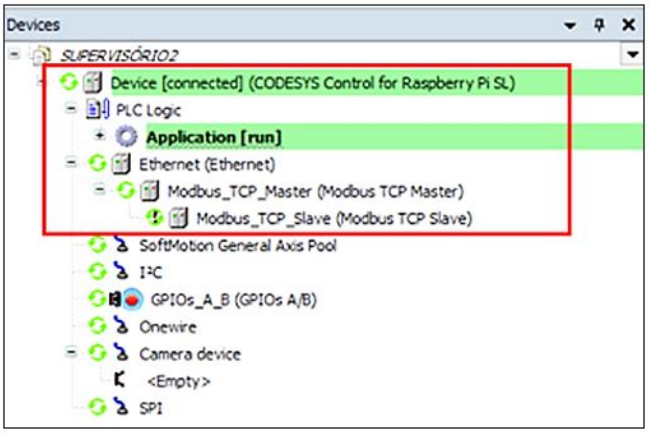

Fig. 32. Confirmação das comunicações dos *Hardwares* e o *Codesys*. Fonte: Autor.

Em continuação ao desenvolvimento do supervisório, foi

criado o ambiente PLC(PRG) para geração das variáveis, em *Application* > *Add Object* > *POU* > selecionado *Program* e na *Implementations Language*, selecionado *Ladder Logic Diagram* (LD), conforme Fig. 33. Por conseguinte, o ambiente para as conversões de variáveis, em *Application*>*Add Object*> *Unit Conversion*.

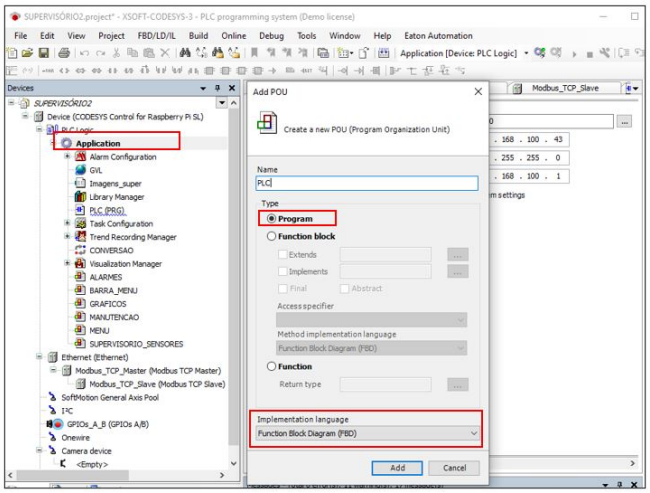

Fig. 33. Ambiente de Programação no Codesys. Fonte: Autor.

Em *Modbus TCP Slave*, na aba *Modbus Slave Chaneel* > *Add Chaneel*, foram criadas os canais das principais variáveis de corrente, tensão, temperatura, luminosidade 1, 2, 3, e 4 atribuindo a estas o nome (*name*), tipo de acesso (*Acess Type*), acionamento (*trigger*), leitura de deslocamento (*Read Offset*), entre outros, visando a coleta dos dados dos dispositivos eletrônicos instalados (Fig. 34). Além disso, na aba *Modbus TCPSlave I/O Mapping*, foi feito o direcionamento das variáveis da programação *Ladder* do supervisório para possível visualização dos dados coletados durante o funcionamento do sistema (Fig. 35).

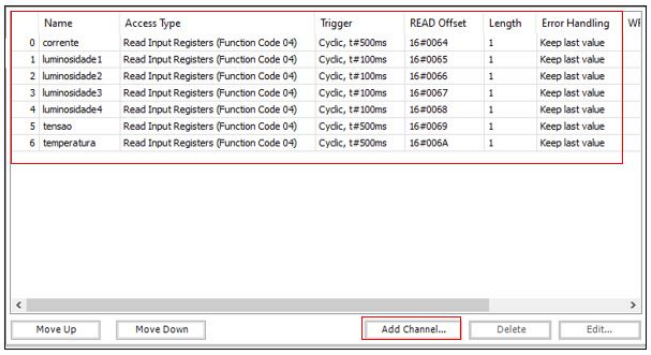

Fig. 34. Inserção das variáveis físicas no *Slave* (escravo) através do Codesys. Fonte: Autor.

Com a criação do ambiente de programação *Ladder* os blocos e variáveis foram estabelecidas de acordo com as necessidades no supervisório, dentre as quais pode-se citar: temperatura, corrente, tensão, MediaGeral, luminosidade1, luminosidade2, luminosidade3, luminosidade4, Contaluxsaida, entre outras. As mencionadas variáveis passaram pelo processo de conversão para serem identificadas no supervisório, ademais, alguns cálculos foram realizados nos blocos matemáticos da programação *Ladder*. Para representação dos *Leds* utilizados no supervisório foram criadas no *Codesys* variáveis virtuais. A seguir especificados alguns blocos de programação *Ladder* utilizados.

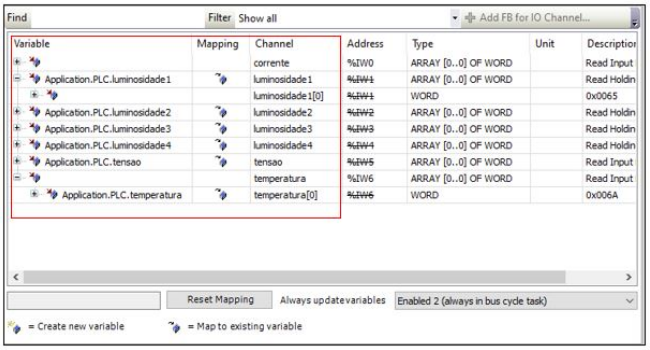

Fig. 35. Direcionamento de variáveis para visualização dos dados da programação *Ladder* do supervisório. Fonte: Autor.

Primeiramente, temos a programação para leitura e conversão das variáveis físicas coletadas nos sensores Termopar, ACS712 e de cada LDR da matriz para serem lidas no supervisório. A programação está em funcionamento coletando os respectivos valores de acordo com a Fig. 36.

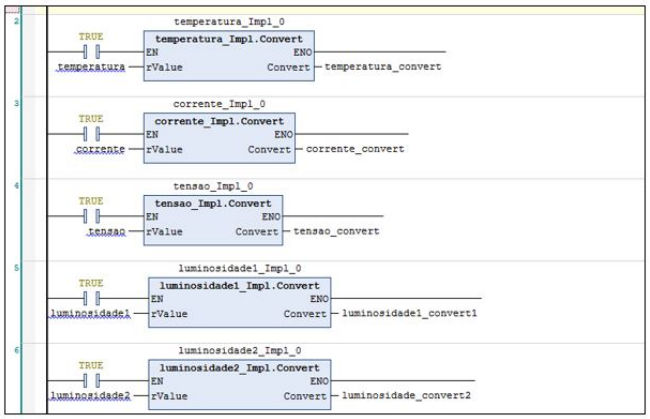

Fig. 36. Programação *Ladder* em blocos para coleta de dados. Fonte: Autor.

Ainda, na programação foi adicionado blocos matemáticos para comparação da leitura com um valor especificado, com objetivo de parametrizar o momento de acionamento do *Led* virtual para sinal de alerta (Fig. 37).

Também, foi calculada a média aritmética e ponderada da luminosidade, assim como, a média aritmética de ambas para leitura no supervisório, utilizando os blocos matemáticos, com intuito de nortear a coleta de dados para uma futura análise (Fig. 38).

Foram criados outros blocos de programação

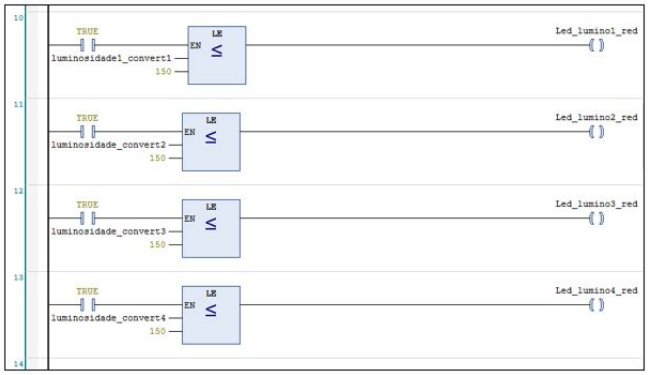

Fig. 37. Programação *Ladder* para comparação de valores. Fonte: Autor.

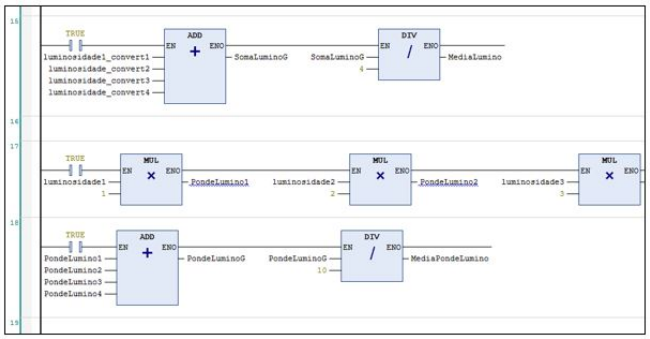

Fig. 38. Blocos matemáticos utilizados para obtenção da média aritmética e ponderada da luminosidade. Fonte: Autor.

Ladder os quais estão especificados no arquivo SUPERVISORIO2.project.Pdf e disponibilizado via *GitHub* para ser feito o download através do link: *https://github.com/rubresson/Supervisorio2.git*, extraído do *Codesys*, com as respectivas identificação e tipo de variáveis. Ainda, para o supervisório foram inseridas imagens em *Application* > *Add Object* > *Imagem Pool*, as quais foram utilizadas na *interface* do menu no supervisório.

Ainda, adicionou-se variáveis globais em *Application* > *Add Object* > *Global Variable List* para utilização na implementação dos Alarmes e alternância nos botões da barra menu do supervisório (Fig. 39).

No supervisório foram implementadas telas para operação do sistema (Menu, Operação, Gráficos, Manutenção e Alarmes), a ação foi realizada em *Application* > *Add Object* > *Visualization* e, posteriormente inserido acesso controlado mediante senha. Cada tela pode contar com os instrumentos imprescindíveis para a visualização dos dados das variáveis de tensão, corrente, temperatura e luminosidade. Em primeiro lugar, foi criada uma tela padrão com uma barra menu contendo botões para acesso as telas supracitadas, *Login* e *Logout*, que foi utilizada nas demais telas em conformidade com a Fig. 40.

Também, foi configurado os botões de acesso de cada tela, de forma a ficar em destaque verde quando acessada. Além do mais, em *Properties* > *Vizualization*, da Tela padrão

```
1  {attribute 'qualified_only'}
2  VAR_GLOBAL
3
4  iScreen : INT:=0;
5
6  penTemperatura: BOOL;
7  penTensao: BOOL;
8  penCorrente: BOOL;
9  penMediaGeral: BOOL;
10
11
12 END_VAR
```

Fig. 39. Descrição das variáveis globais. Fonte: Autor.

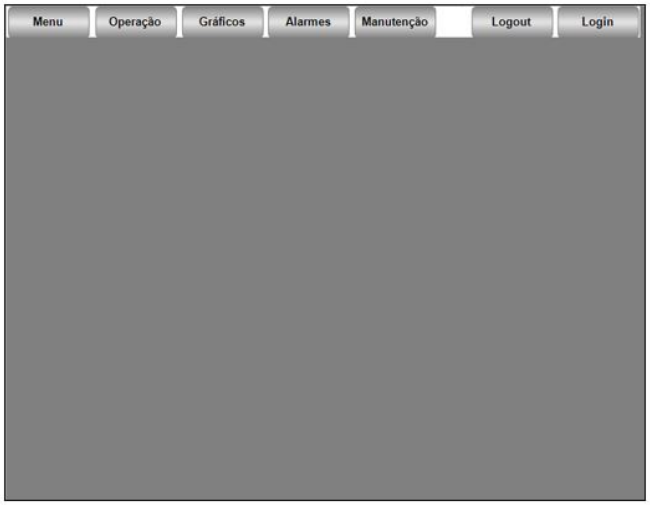

Fig. 40. Tela padrão. Fonte: Autor.

(BARRAMENU), o dimensionamento de visualização da tela foi estabelecido no padrão 1920 x 50 (Fig. 41). As demais telas foram padronizadas com a dimensão de 1920 x 1080.

Seguidamente, as telas citadas anteriormente foram adicionadas e configuradas, visando a visualização das variáveis criadas pelo usuário. Na tela inicial foi inserida uma interface de indicação da denominação do supervisório, bem como, o local indicado para realizar o login com acesso controlado e limitado para cada usuário (Fig. 42).

No supervisório foram definidos três usuários (Admin, Operator e Service) distintos com controle e limite de acesso. O usuário Admin tem acesso a todas as telas, por outro lado o usuário operador tem acesso as telas de Menu, Operação e Gráficos. Já o usuário de serviço a tela Menu e a de manutenção.

Na tela de operação foram inseridos a visualização dos dados coletados de temperatura, tensão, corrente, luminosidade e alarme de necessidade de limpeza. A inserção contou com a utilização de ferramentas do *Visualization Toolbox*, quais sejam: Controles Comuns, Gerenciador de Alarmes, Controle de Medição, Lâmpadas/Interruptores/Bitmaps, Con-

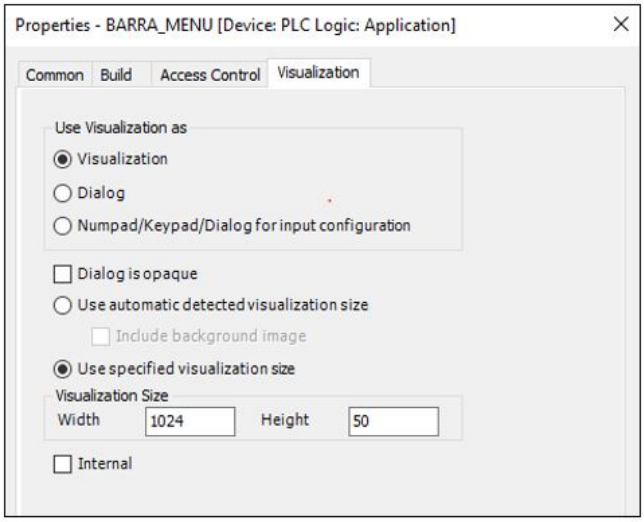

Fig. 41. Configuração da dimensão da Tela padrão. Fonte: Autor.

Fig. 42. Tela Menu Inicial. Fonte: Autor.

troles especiais, entre outros. Na tela, o *Led* verde indica que o sistema está em funcionamento e o *Led* vermelho indica alerta, além disso, nos campos de textos (*Texfield*) são mostrados os valores (Fig. 43).

Na tela de gráficos, com acesso aos usuários Admin e Operator. A visualização da forma gráfica de temperatura, tensão, corrente e luminosidade poderá ser realizada mediante seleção. Também, foram inseridos *Leds* verdes e vermelhos para indicação do funcionamento e de alerta. A visualização dos gráficos permite verificar possíveis aumentos ou diminuições dos valores das variáveis citadas no Codesys (Fig. 44).

Na tela de Alarmes, em *Application* > *Alarm Configuration* > *Add Object* > *Alarm Class e Alarm Group*, foram gerados e configurados os Alarmes de temperatura, luminosidade, tensão e corrente. Por conseguinte, em *Visualization Toobox* > *Alarm Manager* (gerenciador de alarmes), foi inserido o local de observação dos alarmes quando forem acionados. Foram também, configurados *Leds* para indicação

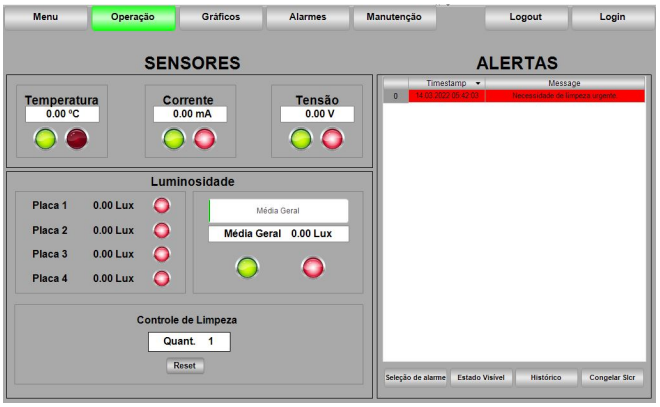

Fig. 43. Tela de Operação em funcionamento. Fonte: Autor.

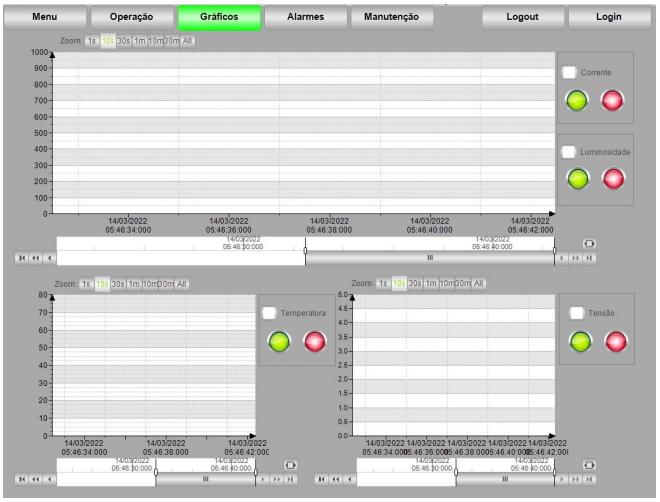

Fig. 44. Tela de Gráficos em funcionamento. Fonte: Autor.

e alerta (Fig. 45).

Fig. 45. Tela de Alarmes. Fonte: Autor.

Na tela de manutenção, com acesso limitado aos usuários Admin, Operator e Service, possibilitou a visualização grá-

fica das variáveis do sistema, assim como, o armazenamento dos dados coletados para posterior análise criteriosa quanto as possíveis manutenções de limpezas. A tela também, possui *Leds* vermelhos indicadores de funcionamento de alertas, assim como os botões *Reset*, Iniciar, Parar e Armazenar, de acordo com a Fig. 46.

Fig. 46. Tela de manutenção em funcionamento. Fonte: Autor.

Para armazenamento dos dados, após a inicialização da simulação, a ação tomada inicia-se em "parar"a visualização > nomear "exemplo.trace"o arquivo de dados no campo de texto > acionar o botão "armazenar". Com relação aos dados armazenados na tela de manutenção, em arquivo com extensão ".TRACE", é possível, utilizando-se do *software* Excel, transformá-lo em uma base de dados coletados para conversão em gráficos com parâmetros de tempo para futuras análises e decisões. Os dados na figura são da coleta de luminosidade e corrente realizados em testes com a ação humana para verificação do funcionamento do case e sensores. (Fig. 47).

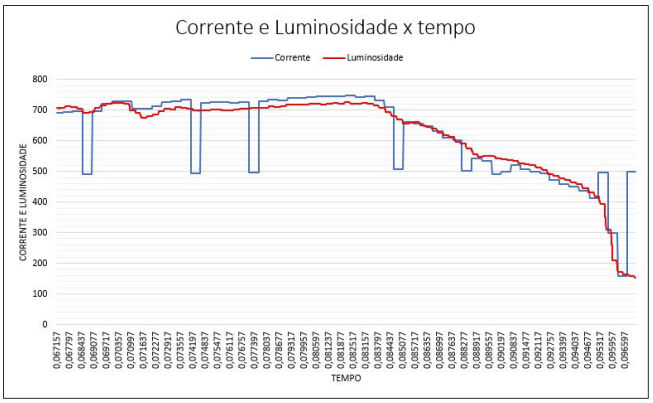

Fig. 47. Gráfico gerado no *software* Excel a partir do dados coletados. Fonte: Autor.

Nas (Fig. 48 e 49), tem-se as simulações realizadas em campo sob as intempéries naturais, no horário das 8h00 até as 9h30min. Nas simulações, também, foi realizado teste com ação humana, pretendendo reproduzir as condições de sobreamento/sujidade, para que no supervisório fosse observado as possíveis variações captáveis pelos sensores.

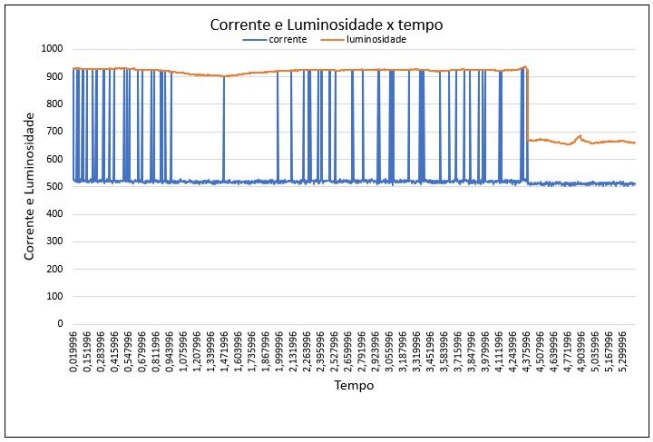

Fig. 48. Gráfico da simulação das 8h até 9h30 Luminosidade e Corrente. Fonte: O autor.

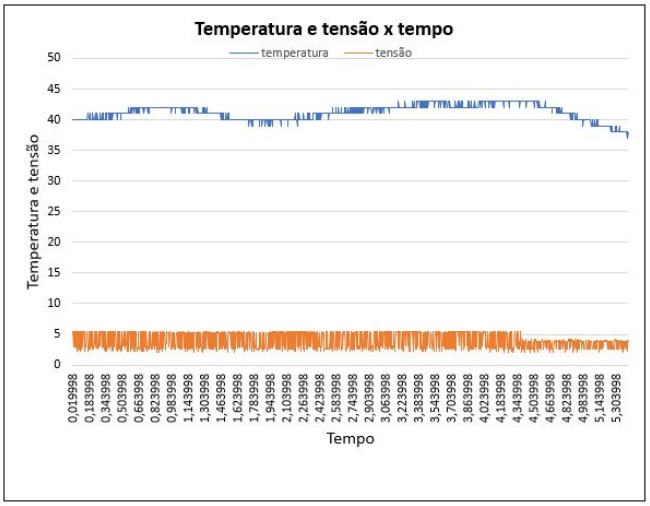

Fig. 49. Gráfico da simulação das 8h até 9h30 Temperatura e tensão. Fonte: O autor.

A limitação do tempo, se deu em virtude da falta de licença para o monitoramento 24 horas pelo *Codesys*.

Ainda, nas simulações realizadas ao longo do dia, conforme (Fig. 50 e 51, no horário das 10h40min até as 12h30min), observou-se no supervisório que as variáveis foram monitoradas e tiveram seus valores coletados para possíveis análises. Além do mais, foi aplicado teste com ação humana, objetivando reproduzir as condições de sobreamento/sujidade para vericações das variações.

Posteriormente, no período da tarde (Fig. 52 e 53), verificou-se uma aumento acima da temperatura de operação da placa fotovoltaica.

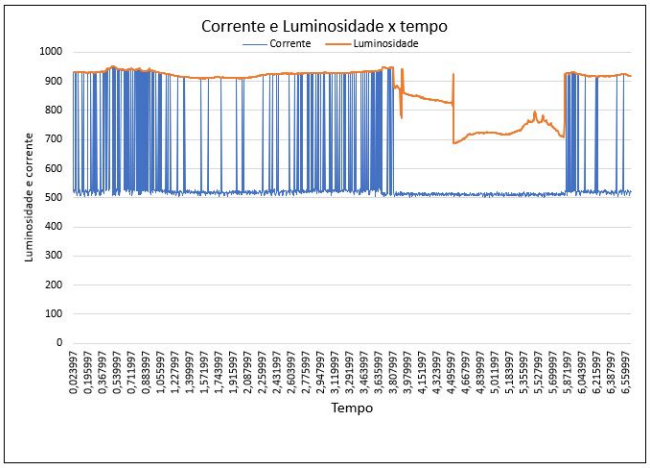

Fig. 50. Gráfico da simulação das 10h40min até 12h30min Luminosidade e Corrente. Fonte: O autor.

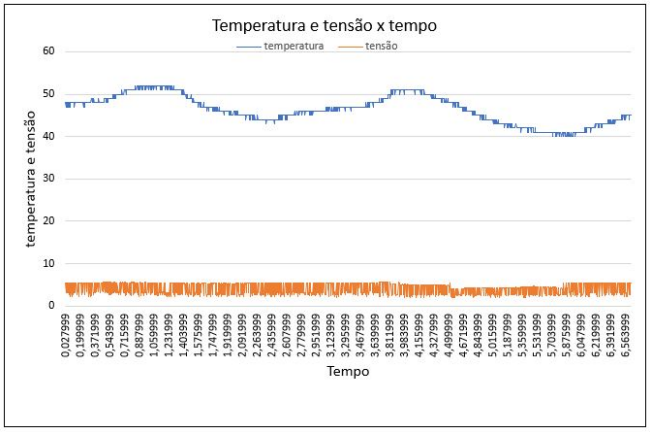

Fig. 51. Gráfico da simulação das 10h40min até 12h30min Temperatura e tensão. Fonte: O autor.

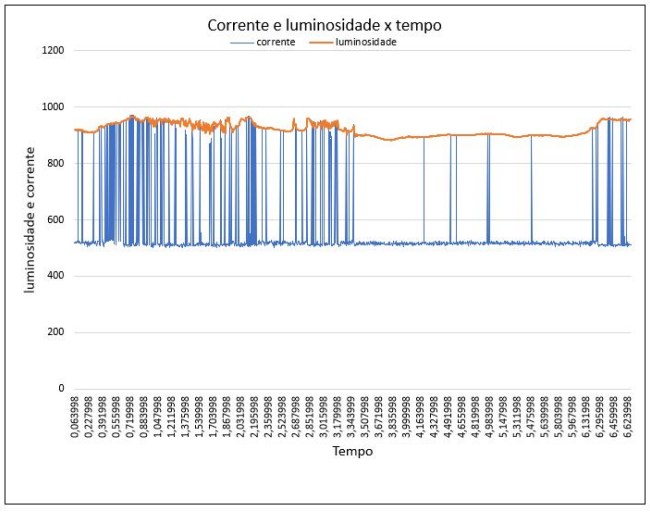

Fig. 52. Gráfico da simulação no período da tarde Luminosidade e Corrente. Fonte: O autor.

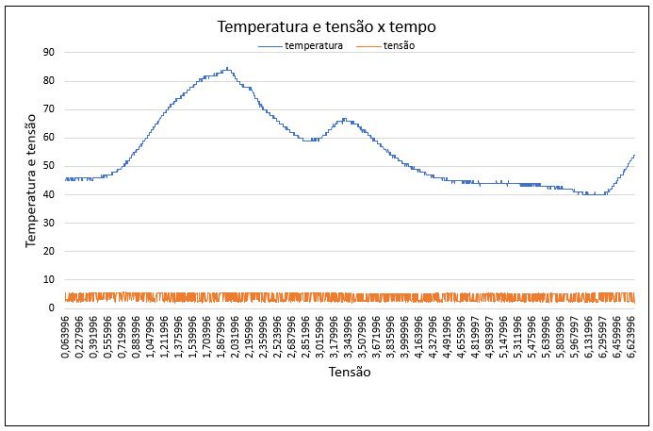

Fig. 53. Gráfico da simulação no período da tarde Temperatura e tensão. Fonte: O autor.

Por fim, ao entardecer, a luminosidade foi diminuindo, bem como, os valores, ademais, de temperatura foi baixando de acordo com as (Fig. 54 e 55)

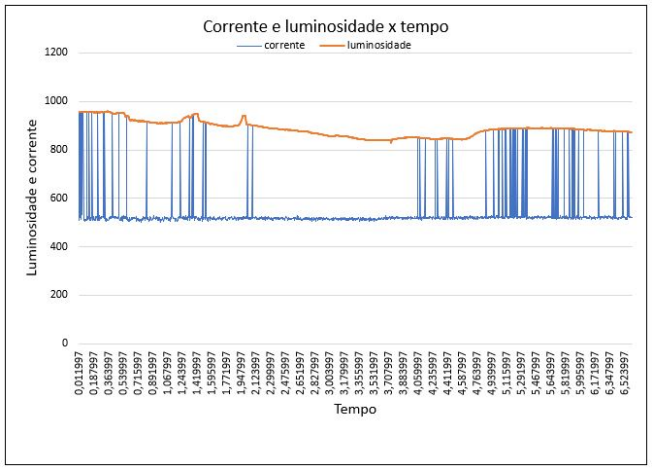

Fig. 54. Gráfico da simulação ao entardecer Luminosidade e Corrente. Fonte: O autor.

Finalizando, foi configurado no Codesys em *Visualizaton* > *WebVisu*, a visualização via internet para os usuários da mesma rede a qual encontra-se o sistema de comunicação *Ethernet e Wifi*. O endereço utilizado foi http://192.168.100.43:8080/webvisu.htm pode ser observado na Fig. 56.

V. Conclusão

Ao analisar o funcionamento do supervisório e a coleta dos dados verificou-se que os objetivos foram satisfatórios e que é viável a sua implementação com a utilização do Raspberry, Arduino, Codesys e os demais componentes citados neste artigo. Os testes realizados garantiram a funcionalidade do supervisório permitindo de forma clara a visualização dos dados das variáveis de temperatura, tensão, corrente e luminosidade coletadas pelos sensores. Observa-se o acionamento de alarmes devido instabilidades na placa fotovoltaica. O

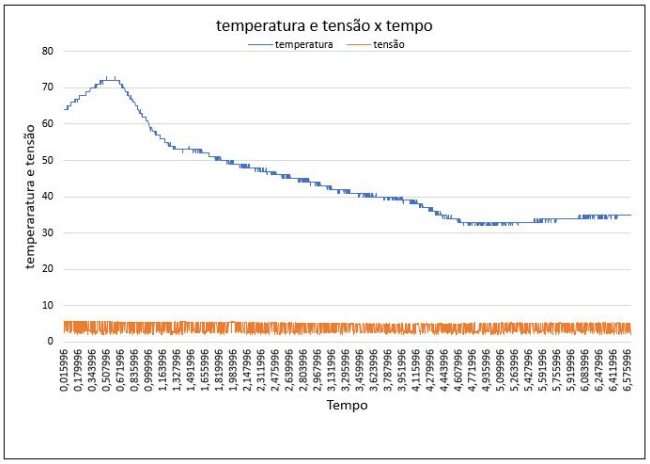

Fig. 55. Gráfico da simulação ao entardecer Temperatura e tensão. Fonte: O autor.

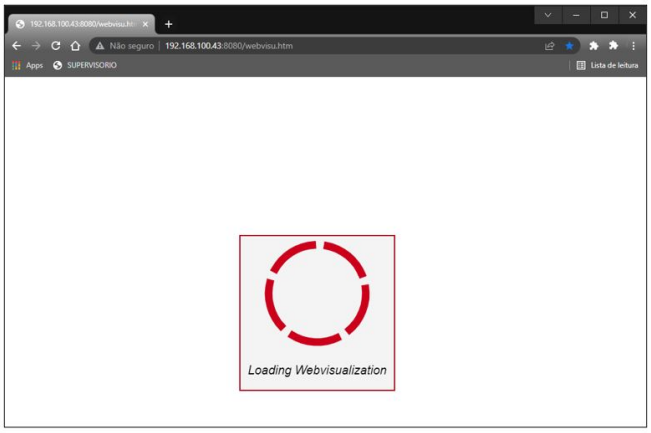

Fig. 56. Acessando o supervisório pela Internet.. Fonte: Autor.

supervisório também permitiu a coleta de dados e sua conversão para análise de possíveis instabilidades e pela redução de sua eficiência, a deliberação quanto a limpeza.

Ademais, a eficácia do *Firmware* inserido na placa do Arduino foi satisfatória para comunicação *Ethernet* e aplicação do protocolo *Modbus*, sendo de suma importância este fator para obtenção dos dados coletados das variáveis e posteriormente visualização e armazenamento no supervisório.

Para trabalhos futuros, existe as seguintes possibilidades:

a) Melhoria do case temporário com base no desenvolvimento elétrico e eletrônico dos *hadwares* e *softwares* utilizados neste artigo;

b) Inserção ou utilização de outros sensores que possam maximizar o monitoramento e/ou realizar outras formas de diagnósticos;

c) Inclusão de módulo para comunicação IOT;

d) Aquisição de pacotes para o Codesys com a finalidade de monitoramento 24 horas;

d) Melhoria quanto o desenvolvimento e proteção da matriz LDR;e,

e) Implementação de um sistema de arrefecimento de temperatura e/ou limpeza automatizado das placas fotovoltaicas.

REFERENCES

[1] V. BLASZCZAK. (2017) Análise de eficiência de painel fotovoltaico com sistema sracker seguidor solar. tcc apresentado ao curso de engenharia ambiental na uffs. Acesso em: 23 maio 2021. [Online]. Available: https://rd.uffs.edu.br/bitstream/prefix/1695/1/BLASZCZAK.pdf

[2] D. O. PAULO JÚNIOR. (2018) Sistema de monitoramento para detecção de falhas em placas fotovoltaicas. trabalho de conclusão de curso de engenharia eletrônica e de telecomunicações. Acesso em: 18 jun. 2021. [Online]. Available: https://repositorio.ufu.br/bitstream/123456789/23504/1/SistemaMonitoramentoDeteccao.pdf

[3] E. BARBOSA *et al.* (2018) Influência da sujeira na geração fotovoltaica. vii congresso brasileiro de energia solar - gramado. Acesso em: 22 maio 2021. [Online]. Available: https://anaiscbens.emnuvens.com.br/cbens/article/view/655

[4] D. N. ARAÚJO *et al.* (2019) Efeitos da acumulação de sujeira sobre o desempenho de módulos fotovoltaicos. revista tecnologia. [Online]. Available: https://periodicos.unifor.br/tec/article/view/9414#:~:text=Os%20resultados%20mostram%20que%2C%20devido,20%25%20da%20efici%C3%AAncia%20da%20planta

[5] S. C. S. COSTA *et al.* (2020) Determinação das taxas de sujidade para módulos fotovoltaicos de filme fino e silício cristalino instalados em diferentes zonas climáticas brasileiras. Acesso em: 11 Jan 2022. [Online]. Available: https://anaiscbens.emnuvens.com.br/cbens/article/view/821

[6] L. J. PIOTROWSKI *et al.* (2020) Otimização da geração de energia elétrica fotovoltaica pelo controle da temperatura. artigo. universidade federal de santa maria – ufsm, santa maria, brasil. Acesso em: 20 jun. 2021. [Online]. Available: https://www.ufsm.br/app/uploads/sites/553/2020/07/91657-field_submission_abstract_file2.pdf

[7] J. P. REGES. (2017) Desenvolvimento de um sistema de aquisição de dados para sistemas fotovoltaicos. Acesso em: 11 Jan 2022. [Online]. Available: http://ppger.ifce.edu.br/wp-content/uploads/2017/12/REGES-J.-P.pdf

[8] P. R. SILVA. (2018) Estudo do efeito da composição solar no desempenho dos módulos fv caracterizados em condições reais de operação. dissertação de pós-graduação em engenharia elétrica. Acesso em: 18 jun. 2021. [Online]. Available: https://ufsj.edu.br/portal2-repositorio/File/ppgel/152-2017-12-20-DissertacaoPedroRodriguesSilva.pdf

[9] R. C. BARROS *et al.* (2019) Low-cost solar irradiance meter using ldr sensors. Acesso em: 12 Jan 2022. [Online]. Available: https://ieeexplore.ieee.org/document/8627176

[10] A. F. NUNES. (2018) Desenvolvimento de um sistema supervisório de baixo custo para sistemas elétricos. trabalho de conclusão do curso de engenharia elétrica, universidade federal de ouro preto, minas gerais. [Online]. Available: https://www.monografias.ufop.br/bitstream/35400000/1626/6/monografia_desenvolvimentosistemasupervis%c3%b3rio.pdf

[11] P. E. B *et al.* (2018) Atlas brasileiro de energia solar. são josé dos campos – brasil. 2ª edição. Acesso em: 12 jun. 2021. [Online]. Available: http://repositorio.unifesp.br/handle/11600/58353

[12] L. F. S. ROZZA, B; PEREIRA. (2019) Processo automático para limpeza de sistema fotovoltaico. trabalho de conclusão de curso engenharia elétrica da unisul/sc. Acesso em: 12 jun. 2021. [Online]. Available: https://www.riuni.unisul.br/bitstream/handle/12345/7397/TCC_Bruno_Segala_Final_Entrega.pdf?sequence=1&isAllowed=y

[13] T. O. JESUS, J. N; NASCIMENTO. (2020) Sistema supervisório aplicável ao ensino de sistema fotovoltaico autônomo. trabalho de conclusão do curso de bacharel em engenharia elétrica, universidade federal do amapá. Acesso em: 20 jun. 2021. [Online]. Available: http://repositorio.unifap.br/handle/123456789/575

[14] B. BASTOS. (2019) Desenvolvimento de um dispositivo para monitoramento de painéis fotovoltaicos. [Online]. Available: https://ppgee.ufersa.edu.br/wp-content/uploads/sites/61/2020/03/Disserta%C3%A7%C3%A3o-Matheus-Fonseca-Bastos.pdf

[15] F. A. A *et al.* (2018) Wm water monitoring. tcc apresentado no curso de tecnólogo em análise e desenvolvimento de sistemas, na ufpr. [Online]. Available: https://acervodigital.ufpr.br/bitstream/handle/1884/59211/TCC%20SW%20Smart%20Waterpdf.pdf?sequence=1&isAllowed=y

[16] F. SOUZA. (2018) Arduino uno. Acesso em 25 Set 2021. [Online]. Available: https://www.embarcados.com.br/arduino-uno/

[17] G. BUKMA. (2018) Desenvolvimento de um protótipo de baixo custo para segurança residencial. monografia de graduação em engenharia do controle e automação, na universidade federal de ouro preto/mg. Acesso em 20 Out 2021. [Online]. Available: https://www.researchgate.net/publication/338053235_desenvolvimento_de_prototipo_de_sistema_de_alarme_para_ambiente_laboratorial

[18] A. A. GUEDES *et al.* (2021) Desenvolvimento de uma i/o remota com comunicação modbus de baixo custo. Acesso em: 20 Nov 2021. [Online]. Available: https://semanaacademica.org.br/system/files/artigos/final_-_desenvolvimento_de_uma_io_remota_com_comunicacao_modbus.pdf

[19] MICROCHIP. (2004) Datasheet enc28j60. Acesso em 10 Jan 2022. [Online]. Available: http://ww1.microchip.com/downloads/en/devicedoc/39662a.pdf

[20] L. F. PATSKO. (2006) Tutorial de aplicações, funcionamento e utilização de sensores. Acesso em: 10 Jan 2022. [Online]. Available: https://www.maxwellbohr.com.br/downloads/robotica/mec1000_kdr5000/tutorial_eletronica_-_aplicacoes_e_funcionamento_de_sensores.pdf

[21] M. INTEGRATED. Datasheet max6675s. Acesso em 10 Jan 2022. [Online]. Available: https://datasheets.maximintegrated.com/en/ds/MAX6675.pdf

[22] FILIPEFLOP. Imagem sensor max6675. Acesso em 10 Jan 2022. [Online]. Available: https://www.filipeflop.com/produto/termopar-tipo-k-com-modulo-max6675/

[23] M. ALLEGRO. Datasheet do sensor corrente acs712. Acesso em: 10 Jan 2022. [Online]. Available: https://www.allegromicro.com/en/products/sense/current-sensor-ics/zero-to-fifty-amp-integrated-conductor-sensor-ics/acs712

[24] C. CIRCUITO. Imagem senhor acs712. Acesso em 10 Jan 2022. [Online]. Available: https://www.curtocircuito.com.br/sensor-de-corrente-30adc-acs712.html

[25] L. L. VIEIRA *et al.*, "Desenvolvimento de interfaces homem-máquina de alta performance." 2018, acesso em: 14 jun. 2021. [Online]. Available: https://monografias.ufop.br/handle/35400000/1542