Universidade de Brasília
Faculdade de Tecnologia

Desenvolvimento de Instrumentação para Monitoramento de Plantas de Geração Fotovoltaica Off-Grid, Instaladas em Solo e sobre Futuadores em Corpo D'água

Allan Jhonny Lima Maciel

TRABALHO DE GRADUAÇÃO

ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Brasilía

2021

Universidade de Brasília
Faculdade de Tecnologia

Desenvolvimento de Instrumentação para Monitoramento de Plantas de Geração Fotovoltaica Off-Grid, Instaladas em Solo e sobre Futuadores em Corpo D'água

Allan Jhonny Lima Maciel

Trabalho de Graduação submetido como requisito parcial para obtenção do grau de Engenheiro de Controle e Automação.

Orientador: Prof. Dr. Guilherme Caribé de Carvalho
Coorientador: Prof. Dr. Fernando Cardoso Melo

Brasilía
2021

Lima Maciel, Allan Jhonny.
L417d Desenvolvimento de Instrumentação para Monitoramento de Plantas de Geração Fotovoltaica Off-Grid, Instaladas em Solo e sobre Futuadores em Corpo D'água / Allan Jhonny Lima Maciel; orientador Guilherme Caribé de Carvalho; coorientador Fernando Cardoso Melo. -- Brasilía, 2021.
86 p.

Trabalho de Graduação em Engenharia de Controle e Automação -- Universidade de Brasília, 2021.

1. Palavra chave 1. 2. Palavra chave 2. 3. Palavra chave 3. 4. Palavra chave 4. I. Caribé de Carvalho, Guilherme , orient. II. Cardoso Melo, Fernando, coorient. III. Título

Universidade de Brasília
Faculdade de Tecnologia

Desenvolvimento de Instrumentação para Monitoramento de Plantas de Geração Fotovoltaica Off-Grid, Instaladas em Solo e sobre Futuadores em Corpo D'água

Allan Jhonny Lima Maciel

Trabalho de Graduação submetido como requisito parcial para obtenção do grau de Engenheiro de Controle e Automação.

Trabalho aprovado. Brasilía, 10 de dezembro de 2021:

Prof. Dr. Guilherme Caribé de Carvalho, UnB/FT/ENM
Orientador

Prof. Dr. Fernando Cardoso Melo, UnB/FT/ENM
Coorientador

Prof. Dr. Mário Benjamim Baptista de Siqueira, ENM/UnB
Examinador interno

Prof. Dr. Lelio Ribeiro Soares Junior
Examinador externo

Brasilía
2021

Agradecimentos

Agradeço, em primeiro lugar, a Deus, que sempre esteve comigo, ajudando-me e me dando forças para continuar. A Ele, Glória para todo sempre! Amém! Ao meu falecido pai, que sempre me apoiou e acreditou em mim! Ele falava, de modo carinhoso, que eu era seu engenheiro! À minha amada mãe, que sempre me incentivou a sonhar grande, por acreditar em cada um de meus sonhos! À minha Tia Luzileide, que me acolheu e cuidou de mim, oferecendo-se seu apoio e cuidado, tratando-me como filho. Á minha irmã Jhennata Karen, por ser a amiga com quem pude contar em horas de necessidade! À minha esposa, Mirian Alves, por seu amor e compreensão! Aos meus tios, Maria do Socorro, Maria da Paixão e José Sérgio, por seu companheirismo! Aos meus primos André, Rodrigo e Eduardo, Lara Liz, Luara Luiza e Greice, pelas palavras de força e encorajamento! Ao pastor Marcelo, por suas orações, muito bem-vindas e necessárias! Ao meu amigo Davi Rodrigues, pela mão estendida, para me sustentar! Ao professor Dr. Fernando, pela ajuda em todos os momentos difíceis, nas dificuldades, durante o curso e, principalmente, nesta etapa final, quando passei por muitos “perrengues”! Ao professor Dr. Guilherme Caribé, pela orientação, apoio, confiança e dedicação!.

“If you find that you’re spending almost all your time on theory,
start turning some attention to practical things;
it will improve your theories.
If you find that you’re spending almost all your time on practice,
start turning some attention to theoretical things;
it will improve your practice.”
(Donald Knuth)

Resumo

Com o crescimento populacional, aumenta a necessidade por energia, incluindo o fator ambiental torna ainda maior o desafio, assim aumenta a procura por energias renováveis. Dentre as energias renováveis, a solar possui suas vantagens, uma vez que no Brasil, o sol é percebido o ano todo. Essa forma de energia vem Crescendo, e aprimorar a sua eficiência é de fundamental importância, para isso existe dois caminhos: um deles consiste na busca de novos matériais para confecção dos módulos fotovoltáicos mais eficientes e ou outro consiste em reduzi a temperatura dos módulos, este projeto vai ao encontro desse segundo caminho.

Com o intuito de avaliar a eficiência de módulos fótovoltaicos instalados sobre uma lâmina d'água, foram instalados 5 módulos fotovoltaicos, sendo 4 deles instalados sobre uma lâmina d'água, e outro módulo foi instaldo em terra. Dos 4 módulos instalados sobre a lâmina, um foi sumetido a refriamento forçado por meio de um trocador de calor, outro foi sumetido a resfriamento por meio de um sistema de aspersão, outro submetido ao escorrimento de um fina lâmina d'água sobre o módulo e o outro foi sumetido a refrigeração passiva.

Este trabalho vem da necessidade de monitorar a temperatura dos módulos por meio de termopares,a temperatura de entrada e saída da água no caso do módulo com o trocador de calor por meio do sensor Ds18b20 e a irradiação solar por meio do sensor Li-Cor LI-200R, já as grandezas Pressão Atmosferica, humidade relativa e *noth* mánético foram por meio do módulo Rapiberry pi sense hat. Outros sensores também foram descrito no projeto, porém estes não chegaram a tempo para a sua implementação.

A aquisição dos dados do Trabalho foi feita por um Rasberry PI, atrvés de software densenvolvido em python. A comunicação com os sensores foram feitas por meio dos seguintes protocolos one-wire, modbus rtu, e i2c, sendo que todos estes protocolos foram implementado em python através de API's do próprio python.

Palavras-chave: Raspberry Pi. Python. Modbus-Rtu. Energia Solar Palavra chave 4.

Abstract

This is the english abstract.

With population growth, increasing demand for energy, including environmental factor makes the challenge even greater, thus increasing the demand for energy renewable. Among the renewable energies, solar has its advantages, since it that in Brazil, the sun is what happens all year round. This form of energy Crescendo, and improve its methods to improve its efficiency are two of fundamental importance, one of them is the search for new modules, this photovoltaic more efficient and consistent in reducing the temperature of the modules, this photovoltaic project goes along this second path. With the intuitiveness of evaluating the efficiency of photovoltaic modules installed on an installed water depth, 5 photovoltaic modules were installed, 4 of them installed on an installed water depth, and the module was water installed on land. Of the 4 modules installed on the sheet, a heat summator was added through a sprinkler system, another was skimmed from a sprinkler system, another was escorted over a module and the water sheet. another was subjected to automobile liability. This work comes from the need to change the water temperature in the case of temperature of the modules by thermocouples and heat output through the sensor Ds18b20 and the solar irradiation through the sensor Li-Cor LI-200R, already as quantities, Atmospheric, relative and magnetic *th* were via the Rapberry pi sense hat module. Other sensors described in the project, however these also did not reach a time for their implementation.

The acquisition of data from the Work was done by a Rasberry PI, through software developed in python. The communication with the sensors was done through the following protocols one-wire, modbus rtu, and i2c, and all these protocols were implemented in python through API's of python itself. **Keywords**: Raspberry Pi. Python. Modbus-RTU. Solar Energy Keyword

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

A população brasileira projetada para 2021, segundo o IBGE, está na casa dos 212 milhões de pessoas. O censo verifica que serão aproximadamente 71 milhões de endereços, já que o censo de 2010 estimou que a população brasileira seria composta, aproximadamente, por 190,63 milhões de pessoas, notando-se um aumento de cerca de 21,27 milhões de brasileiros (CENSO 2010).

Diante do crescimento da população, aumenta diretamente, a demanda energética, caracterizando um desafio enorme que é o de gerar energias, sem agredir o meio ambiente. Nessa direção, o desenvolvimento de tecnologias de geração de energia renovável tem se tornando mais promissor. Destaca-se, dentre essas, a tecnologia de obtenção de energia solar que possui vantagens, em relação à eólica, à hidroelétrica e à biomassa, principalmente, no Brasil onde o sol é notado durante todas as estações do ano. Para fins de engenharia, a energia solar pode ser dividida em duas vertentes, a energia solar térmica e a energia solar fotovoltaica. A energia solar térmica tem como foco principal, a quantidade de energia que um corpo consegue absorver na forma de calor, a partir da radiação solar sobre ele incidente. Já a energia fotovoltaica é obtida pela conversão direta da luz do sol, em eletricidade, produzindo o efeito fotovoltaico Pinho e Galdino (2014).

A energia solar fotovoltaica é produzida através de painéis fotovoltaicos, composto em sua maioria, por células fotovoltaicas à base silício. Atualmente, a produção de células e módulos fotovoltaicos é classificada em três gerações que variam, de acordo com o material utilizado, a primeira geração é representada pelo silício monocristalino (m-SI) e Silício policristalino (p-Si). Na segunda geração, os materiais utilizados são basicamente silício amorfo (a-Si), disseleneto de cobre e índio (CIS) ou disseleneto de cobre, índio e gálio (CIGS) e telureto de cádmio (CDTe). A terceira geração ainda está em pesquisa e desenvolvimento, ela é dividida em três cadeias produtivas: células fotovoltaicas multijunção e células fotovoltaicas, para concentração (CPV), células sensibilizadas por corante (DSSC), células orgânicas ou poliméricas Pinho e Galdino (2014).

Os painéis fotovoltaicos não transformam, completamente, a energia solar em energia elétrica. A terceira geração apresenta maior eficiência do que as demais, 15 entretanto, o custo ainda não é tão atrativo, quanto a primeira geração. A figura 1 mostra a curva de eficiência das células fotovoltaica com o passar do tempo:

Outra forma de alcançar maior eficiência, seria reduzindo a temperatura em que os módulos fotovoltaicos se encontram. Para isso seria necessário submeter os módulos fotovoltaicos a um tipo de refrigeração forçada ou passiva. Há estudos que demostram que se os módulos fotovoltaicos forem instalados somente sobre uma lâmina de água, eles apontam

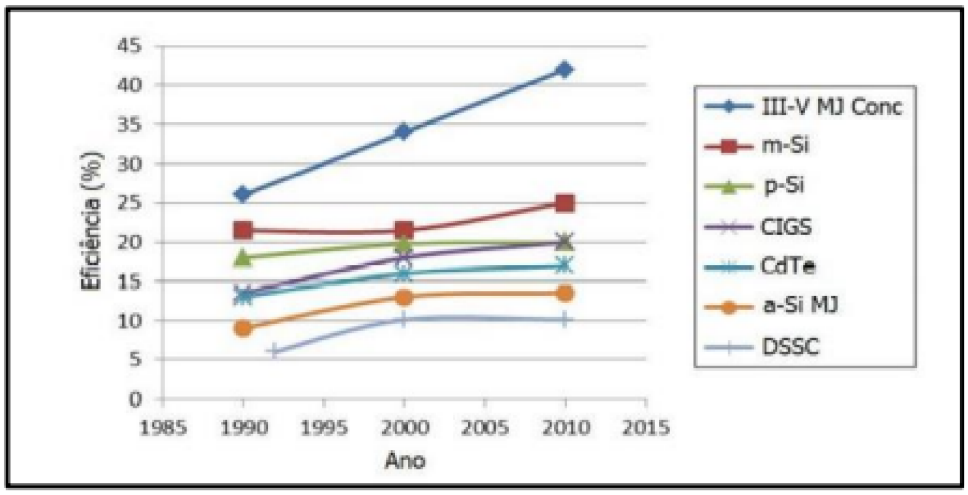


Figura 1 – Evolução das Curvas de Eficiência em Placas Fotovoltaica.

Fonte: Cresceb(2008) (**http://www.cresesb.cepel.br/**).

eficiência de 11% maior se comparados aos módulos instalados em terra Choi (2014).

Um estudo publicado pela Universidade Federal do Ceará constatou que a distância entre o painel fotovoltaico e a lâmina de água não indicam ganhos significativos, em relação à eficiência do painel Alencar Filho, CARVALHO e Dupont (2018). Esse estudo não verificou a influência de outros métodos de refrigeração, como uma refrigeração forçada, por exemplo. Apesar disso, é possível e necessário, evoluir para o caso dos painéis submetido a uma refrigeração forçada.

Assim, a Universidade de Brasília (UnB), com o apoio da Fundação de Apoio à Pesquisa do Distrito Federal, FAP-DF, desenvolvem um projeto que visa comparar a eficiência de módulos fotovoltaicos instalados em terra e sobre uma lâmina de água.

Este projeto de graduação vem da necessidade de desenvolver um aparato experimental que sirva de suporte técnico, para analisar a influência das variáveis: temperatura, pressão, humidade e irradiação solar, sobre a eficiência energética. O resultado servirá de protótipo para o projeto desenvolvido pela UnB e FAP-DF.

1.1 Objetivo do Projeto

Desenvolver aparato experimental, utilizando os protocolos Modbus RTU, One-Wire E I2C, para coleta de dados de refrigeração forçada e passiva de painéis fotovoltaicos, instalados em terra e em lâmina de água.

1.2 Descrição do Projeto

O projeto consiste em aferir dados referentes à temperatura, tensão e corrente de cinco módulos fotovoltaicos, via rede Modbus-RTU, procurando verificar os níveis e aspectos ligados à irradiação solar, umidade, *north* (orientação em relação ao norte magnético e à latitude do local de instalação) e pressão atmosférica.

Dos cinco módulos fotovoltaicos, quatro deles foram instalados sob lâmina de água e o outro, em terra. Daqueles instalados sobre lâmina de água, três foram submetidos à refrigeração forçada, sendo um refrigerado, por um sistema de aspersão; outro, com um trocador de calor e o último, com uma fina lâmina de água, sobre o painel.

Ao longo de todos os módulos fotovoltaicos, foram instalados termopares com o intuito de aferir a temperatura sobre a superfície dos módulos e no backsheet. Os dados referentes a essas temperaturas foram coletados, com o auxílio de um datalogger e um módulo de aquisição de termopares, estes dispositivos enviaram as temperaturas, para o microcomputador (Raspberry Pi), por meio do protocolo Modbus RTU, utilizando a interface RS485.

No módulo que continha um trocador de calor, foram instalados dois sensores DS18B20, para coletar a temperatura de entrada e saída da água. Esses sensores disponibilizaram as informações, por meio do protocolo One-Wire. Foi também instalado o módulo Sense Hat, para aquisição de dados referentes à umidade, north e pressão atmosférica e, para comunicação com o Raspberry PI, por meio do protocolo I2C.

2 Referencial Teórico

Este capítulo apresenta os principais conceitos que estruturam as ideias do projeto, permitindo melhor compreensão de todas as etapas, das avaliações e comparações, assim como do aparato experimental proposto nos objetivos.

2.1 Energia Solar Fotovoltaica

É o tipo de energia adquirida com a transformação da radiação solar em energia elétrica, só sendo possível, por meio de materiais semicondutores. O fenômeno, através do qual, os semicondutores são submetidos para transformarem a energia proveniente do sol, em energia elétrica, é chamado de efeito fotovoltaico Pinho e Galdino (2014).

2.2 Efeito Fotovoltaico

O efeito fotovoltaico foi observado pela primeira vez, em 1839, por Edmund Becquerel, que produziu uma corrente eléctrica ao expor à luz, dois eléctrodos de prata, em um eletrólito 2. Em 1877, W.G. Adams e R.E. Day construíram a primeira célula solar baseada em dois eléctrodos de selénio que produziam uma corrente eléctrica, quando expostos à radiação. No entanto, a eficiência destes sistemas era tão reduzida que o desenvolvimento de células solares, realmente interessantes, teve que esperar, até haver uma compreensão mais completa dos materiais semicondutores Brito e Silva (2006). A Figura 2 verifica os níveis energéticos, em termos de banda ou faixa de energia:

Figura 2 – Faixas de Energia.

Fonte: Braga (2008)

Pela Figura 2 é possível perceber que os átomos possuem faixa de valência, faixa proibida e faixa de condução. A banda proibida, também chamada de faixa proibida, é caracterizada pela quantidade mínima de energia necessária, para passar um elétron da banda de valência, para a banda de condução. Em termos da representação das bandas de energia, a banda de energias proibidas do semicondutor, não é tão grande quanto a do isolante, por isso, alguns elétrons têm a possibilidade de adquirir energia, para passar da banda de valência, para a banda de condução, deixando uma lacuna na banda de valência. Sob a ação de um campo elétrico, os elétrons na banda de condução e, ao mesmo tempo, as lacunas na banda de valência são capazes de ganhar energia cinética, para conduzir a eletricidade Almeida Olivati (2000).

Uma célula fotovoltaica simples consiste, basicamente, em um díodo de grande área, (i.e.), em um substrato de material semicondutor, onde se cria um campo eléctrico interno, permanente (chamado junção pn). Quando a radiação atinge um átomo do semicondutor, ele liberta um elétron que pode ser conduzido pelo campo eléctrico interno, para os contatos, contribuindo com a corrente produzida pela célula fotovoltaica Brito e Silva (2006). A Figura 4 mostra uma célula fotovoltaica:

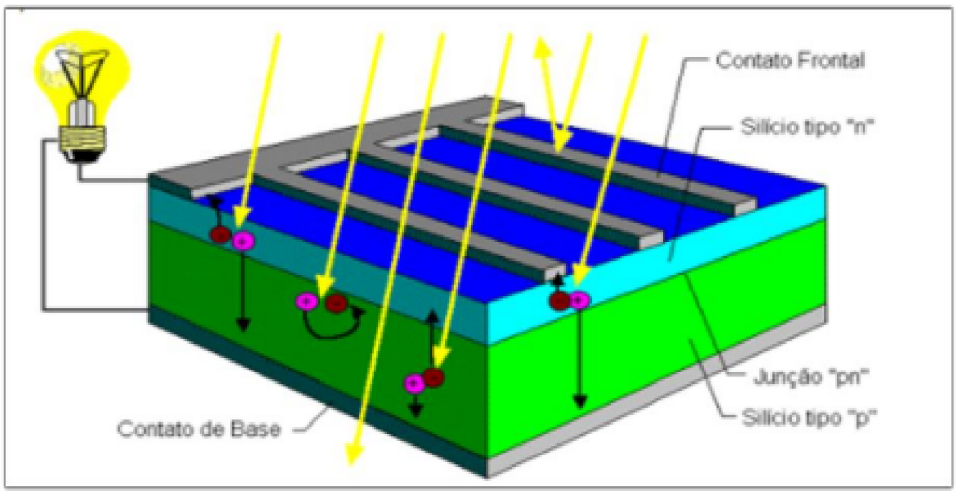


Figura 3 – Corte Transversal de uma Célula Fotovoltaica.

Fonte: CRESEB (2008)

O semicondutor mais utilizado na produção de células fotovoltaicas é o silício. Em sua estrutura molecular, faz 4 ligações com os átomos adjacentes, formando uma rede cristalina. Para que se possa criar um campo elétrico interno permanente, duas estruturas têm que ser criadas, uma é chamada p e a outra, n CRESEB (2008).

A estrutura n pode ser formada, adicionando à rede cristalina do silício, um elemento que tenha 5 elétrons de ligação, como o fósforo, por exemplo. Com isso, fica um elétron sobrando que pode se ligar, fracamente, ao seu átomo de origem. É necessária pouca energia, para que o elétron que está sobrando, ultrapasse a banda proibida e chegue à banda de

condução. O elemento fósforo é chamado de dopante doador, uma vez que ele doará elétrons para a estrutura Pinho e Galdino (2014).

Diferente da estrutura n, na estrutura p é necessário que se introduza na rede cristalina do silício, um elemento onde falte um elétron ligante, como o caso do Boro, que possui 3 elétrons ligantes, faltando um elétron em sua ligação com o silício. A falta de elétrons de ligação é chamada de buraco ou lacuna. Uma lacuna pode se deslocar, caso um elétron, por causa de sua pouca energia, mude de posição. Nesse sentido, o boro é considerado, um aceitador de elétrons CRESEB (2008).

Se, em um silício puro forem introduzidos átomos de boro em uma metade e de fósforo, em outra, será formado o que se denomina de junção pn. O que ocorre nesta junção, é que elétrons livres do lado n passam para o lado p, onde encontram os buracos que os capturam, fazendo com que haja um acúmulo de elétrons, no lado p, tornando-o, negativamente carregado, e uma redução de elétrons do lado n, que o torna, eletricamente positivo. Estas cargas aprisionadas dão origem a um campo elétrico permanente, que dificulta a passagem de mais elétrons do lado n para o lado p. Esse processo alcança um equilíbrio, quando o campo elétrico forma uma barreira capaz de barrar os elétrons livres, remanescentes, no lado n Pinho e Galdino (2014).

Se uma junção pn for exposta a fótons com energia maior que o gap, ocorrerá a geração de pares elétron-lacuna. Se isso acontecer na região onde o campo elétrico é diferente de zero, as cargas serão aceleradas, gerando assim, uma corrente através da junção.

O deslocamento de cargas dá origem a uma diferença de potencial denominada Efeito Fotovoltaico. Se as duas extremidades do "pedaço"de silício forem conectadas por um fio, haverá uma circulação de elétrons. Esta é a base do funcionamento das células fotovoltaicas CRESEB (2008) . A figura 2.3 representa o efeito fotovoltaico na junção pn:

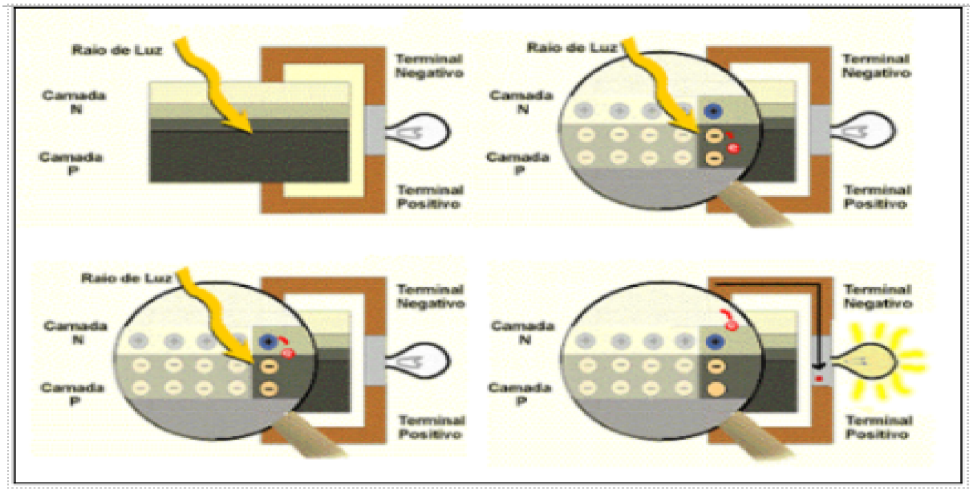


Figura 4 – Efeito fotovoltaico na junção pn.

Fonte: CRESEB (2008)

2.3 Interface Serial

Em uma comunicação serial, os dados são enviados sequencialmente, através de um canal de comunicação ou barramento. Entretanto, é preciso que sejam estabelecidos alguns parâmetros, como os de velocidade de transmissão, de paridade e de controle de fluxo, dentre outros (WEG, 2013).

Segundo Hirakawa, Cugnasca e Cugnasca (2014), a comunicação serial é usada, principalmente, em situações em que os custos de implementação de uma comunicação paralela se tornam proibitivos, como por exemplo, uma implementação que exige grandes distâncias.

A transmissão serial pode acontecer de forma síncrona ou assíncrona. Na forma síncrona, a mensagem transportada não se refere somente a bits de dados, mas também a caracteres de sincronismo, os quais, uma vez detectados pelo receptor, ajustam os seus clocks internos, para receberem os bits de dados, na taxa de comunicação estabelecida.

Na forma assíncrona, cada caractere é transmitido individualmente, e para cada um, existem bits de início (Start Bit) e bits de parada (Stop Bits), conforme verificam André Riyuiti Hirakawa (2014). A figura 5, que vem logo, a seguir, apresenta a estrutura básica de uma transmissão assíncrona:

Figura 5 – Estrutura Básica de Transmissão Assíncrona.

Fonte: MECAWEB (2019)

Na figura 5 é possível verificar a estrutura da mensagem que é composta por um bit de início e um ou dois bits de fim. Nesta direção, observa-se que pode existir um bit de paridade, para verificar erros no processo de transmissão da mensagem. Essa paridade pode receber classificação como ímpar, par ou nenhuma das duas. Nesta figura, a parte de dados é representada por um bit que pode ser menor que a representação de 5 a 7 bits.

2.4 Interface RS485

É importante definir que a interface RS485 não se trata de um protocolo, tratase apenas de um padrão normalizado que especifica detalhes físicos, como os níveis de tensão de operação, número de dispositivos e a distância máxima Freitas (2014).

De acordo com novus (2013), o padrão RS-485 (Recommendad Standart-485) também conhecido como EIA-485 (Electronic Industries Alliance-485) foi aprovado em 1983 pela EIA (Electronics Industries Association), sendo caracterizado pela utilização de um meio de comunicação diferencial (ou balanceado), denominado par trançado. Os circuitos transmissores e receptores adotados nestas interfaces, utilizam como informação, a diferença entre os níveis de tensão em cada condutor do par trançado novus (2013), fazendo com que ocorra uma filtragem eficiente dos possíveis ruídos, para fortalecer o padrão RS485, contra interferências. A figura 6 ilustra esse processo:

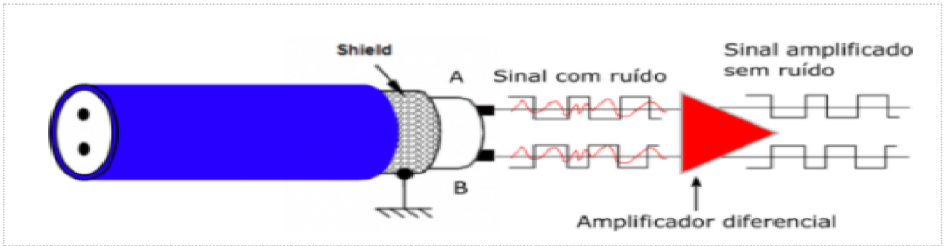


Figura 6 – Sinal Diferencial com o Ruído.

Fonte: Freitas (2014)

A teoria de comunicações descreve a necessidade de terminação de linhas de comunicação, como um valor que correspondente à impedância característica da linha de transmissão novus (2013), com intuito de atenuar as reflexões que distorcem os dados transmitidos. A figura 7 mostra alguns métodos de terminação possíveis:

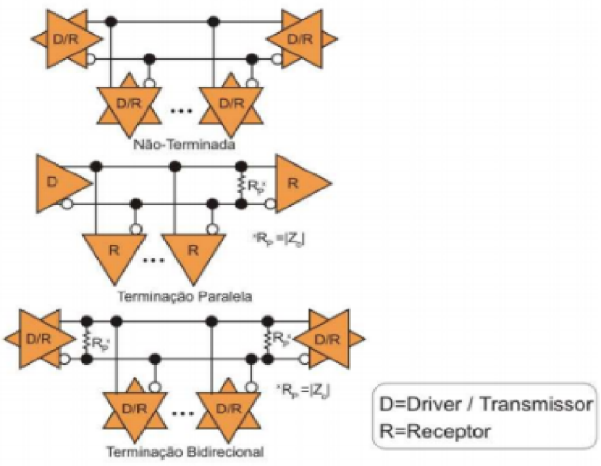


Figura 7 – Métodos de Atenuação.

Fonte: novus (2013)

A implementação da rede não finalizada é considerada simples. Dentre as vantagens que oferece, destaca-se o baixo consumo, e, entre as desvantagens a lentidão de comunicação, por isso, opera com uma taxa de transmissão de até 19.200bps, a uma distância de, no máximo, 100m.

A terminação paralela possui ótimas taxas de transmissão, entretanto, só funciona no modo full duplex, ou seja, um dispositivo fala e os demais da rede, escutam. Já a terminação bidirecional tem como vantagem, uma ótima integridade de sinal e os demais drivers podem ser colocados, em qualquer ponto da rede.

É reconhecido como um dos métodos mais confiáveis, apesar de o consumo da rede aumentar. Para ocorrer melhor rendimento e o casamento de impedâncias, o valor do resistor de terminação, deve ser igual ao da impedância característica de um par trançado (igual, maior ou menor que 120ohms).

A topologia utilizada pelo padrão RS-485 é a de barramento ou sua variante daisy chain que permite a adição e a remoção de dispositivos, sem influenciar outros dispositivos já em operação. Se houver um padrão multiponto, é possível adicionar vários dispositivos na mesma rede, diferente do padrão RS-232, onde apenas é admitida a comunicação entre dois dispositivos Freitas (2014). A figura 8 ilustra essa topologia:

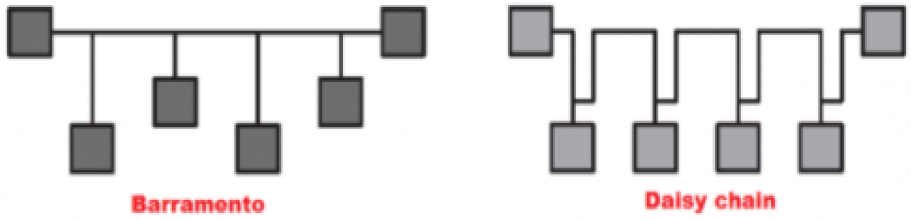


Figura 8 – Efeito fotovoltaico na junção pn.

Fonte: Freitas (2014)

O padrão pode ainda operar no modo half-duplex, com um par de cabos, ou modo full-duplex, com dois pares de cabos. A diferença entre estes dois modos é que no modo half-duplex, só é possível uma operação por vez, no barramento, ou enviar ou receber dados, enquanto no modo full-duplex, é possível fazer as duas operações, ao mesmo tempo. A figura 9 apresenta um layout, para esses modos de operação de cabos:

Quanto à distância de alcance, o padrão RS485 especifica um comprimento de no máximo, 1200 metros. Já a velocidade máxima de comunicação dependerá das características dos dispositivos instalados, da capacitância dos cabos e dos resistores de terminação. Considera-se, para esse cálculo, que a velocidade de comunicação é inversamente proporcional à do comprimento da rede. A figura 10 mostra como acontece essa relação:

Geralmente, são ligados ao protocolo RS485, cerca de 32 dispositivos, em uma mesma

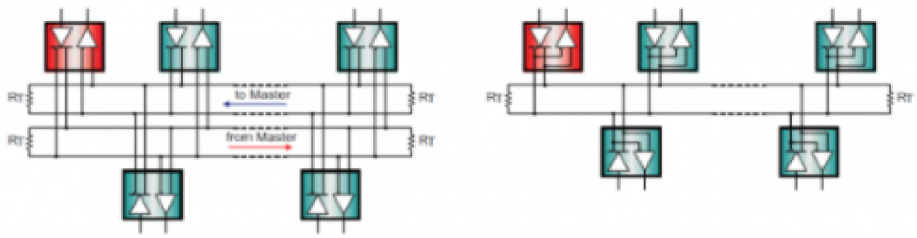


Figura 9 – Estruturas de barramento full-duplex e half-duplex em RS-485.

Fonte: Kugelstadt (2011)

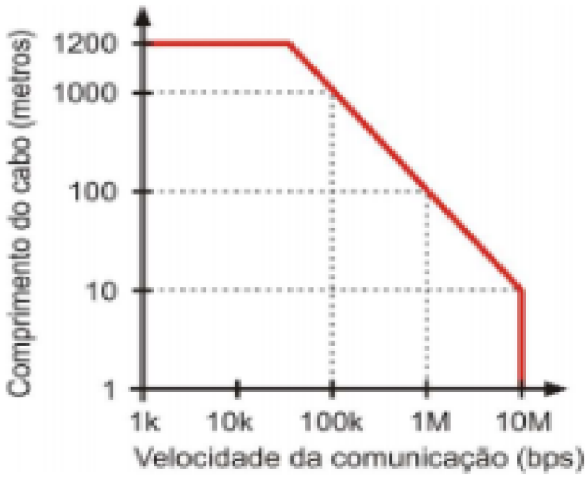


Figura 10 – Comprimento do Cabo Versus Velocidade de Comunicação.

Fonte: novus (2013)

rede, onde é definida uma “unidade de carga” que equivale à 15kohms, para cada dispositivo. Assim, é possível que seja estabelecida uma carga inferior a unitária, 25 de meio, um quarto e um oitavo, que permite a ampliação do número de dispositivos na rede RS485, para até 256 dispositivos novus (2013).

Os circuitos eletrônicos de transmissão e recepção podem ser danificados, se o par trançado apresentar um potencial, excessivamente elevado, em relação ao referencial (comum ou terra). A norma TIA/EIA-485 especifica que a máxima diferença de potencial entre os equipamentos da rede, deve estar entre –7V e +12V novus (2013). Esses circuitos são necessários, para realizar o aterramento na rede.

2.5 Protocolo de Rede de Comunicação

Segundo KUROSE e ROSS (2013), um protocolo de rede é semelhante a um protocolo humano, a única diferença é que as entidades que trocam mensagens e realizam ações, são componentes de hardware ou software de algum dispositivo como computadores, *smartphones, tabletes*, roteadores ou qualquer outro equipamento habilitado, para rede. Assim, o protocolo define o formato e a ordem das mensagens trocadas entre duas ou mais entidades comunicantes, bem como as ações realizadas na transmissão e/ou no recebimento de uma mensagem ou outro evento.

2.5.1 Protocolo Modbus

O protocolo Modbus é um protocolo industrial desenvolvido pela Modicon, em 1979, com o intuito de fazer a comunicação entre diferentes dispositivos, utilizando a estrutura mestre escravo. É um protocolo simples e de fácil implementação, passível de ser utilizado com uma variedade de meios físicos, como o RS232, TCP/IP e RS485, em aplicações com instrumentos e equipamentos de laboratório e, até mesmo com a automação de navios. A velocidade de transmissão dependerá do padrão escolhido do comprimento da rede Modbus-IDA (2006).

Por ser um protocolo simples, criado em cima da comunicação serial, não poderia ser fracionado em camadas, como na estrutura TCP/IP (Transmission Control Protocol/Internet Protocol). Porém, com a criação de novas unidades de dados, o uso das estruturas TCP/IP e UDP (User Datagrama Protocol) foram permitidas, gerando uma separação entre o protocolo básico, que estabelece uma unidade de dados de protocolo (PDU) e a camada de rede, que define a unidade de dados de aplicação (ADU) INSTRUMENTS (2019).

A comunicação ocorre entre os pares, utilizando a forma mestre/escravo, operando pelo seguinte esquema: um dispositivo faz uma requisição e aguarda por uma resposta, dentro de um tempo indeterminado. A figura 11 ilustra o relacionamento denominado de mestre-escravo, como se observa:

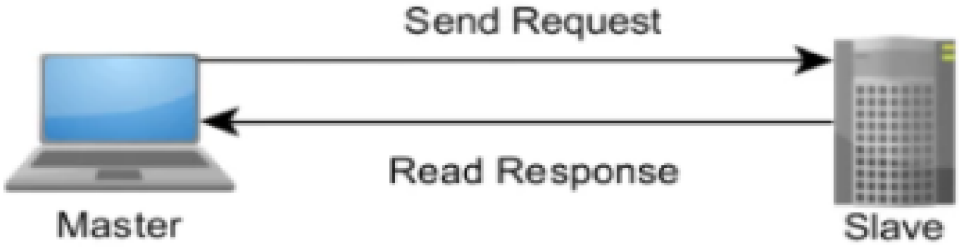


Figura 11 – Relacionamentos de Rede do Tipo Mestre Escravo.

Fonte: INSTRUMENTS (2019)

A estação mestre inicia a comunicação, solicitando que os escravos enviem seus

dados. Por sua vez, os escravos recebem a requisição do mestre e retornam, indicando os dados solicitados. Os dados transmitidos podem ser discretos ou numéricos, sendo possível enviar valores numéricos indicadores de temperatura ou pressão, por exemplo, e bits para ligar ou desligar um motor Freitas (2014). A figura 12 dedica-se à ilustração do processo de troca de mensagens, usando o protocolo Modbus, como se observa:

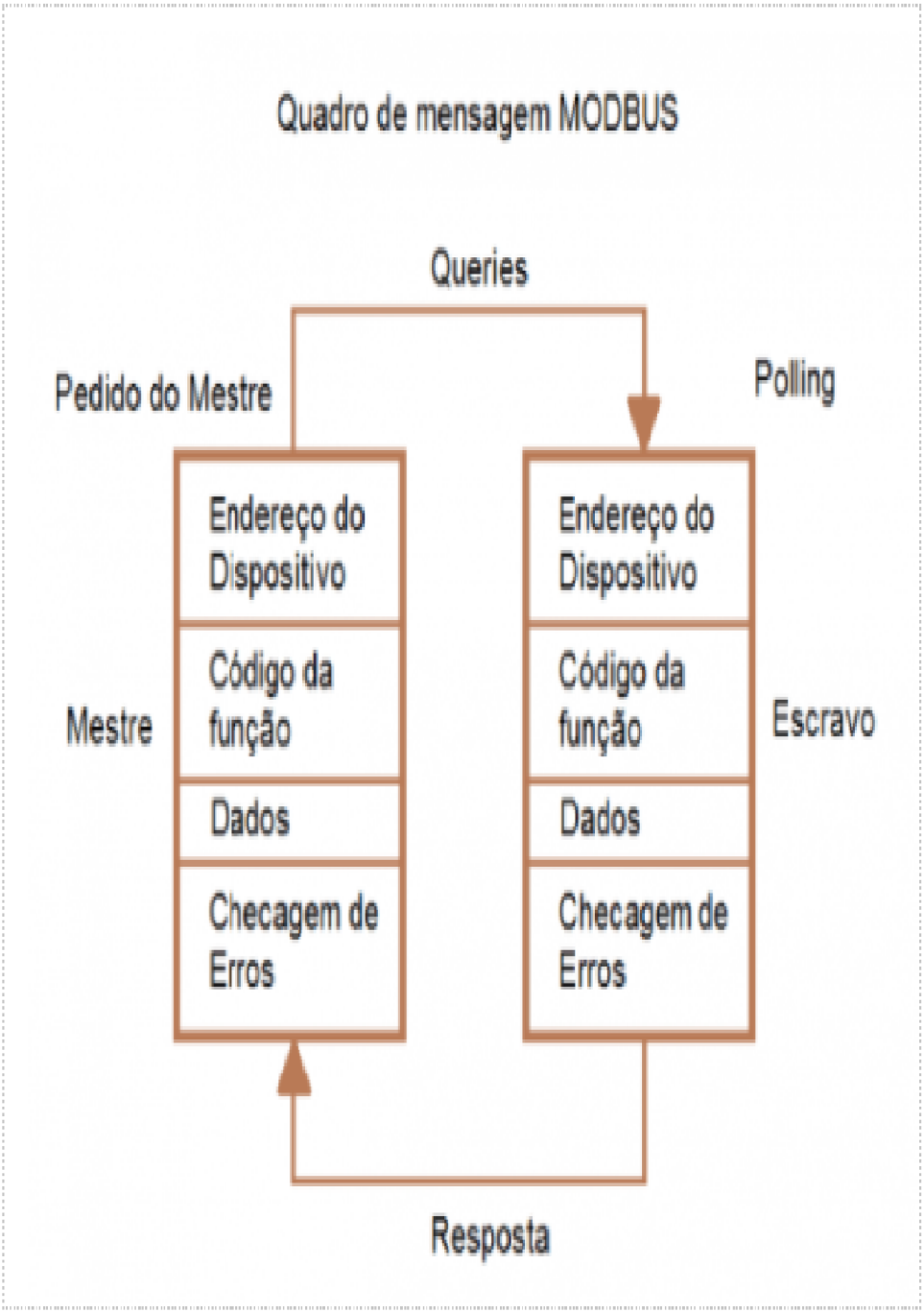


Figura 12 – Comprimento do Cabo Versus Velocidade de Comunicação.

Fonte: Freitas (2014)

As estruturas das mensagens trocadas são compostas por uma unidade de dados do protocolo (PDU) e por uma unidade da aplicação (ADU). Na figura 12, a PDU é representada

na estrutura, como um código de função de um bit, seguido por até 252 bits de dados específicos da função.

O endereço do dispositivo e a checagem dos erros fazem parte da ADU, variando, em relação ao tamanho e formato, uma vez que seguem um padrão de checagem. As ADUs possuem três formatos padrão: o TCP, o RTU (Unidade de Terminal Remoto) e o ASCII. As ADUs dos tipos RTU e ASCII são, tradicionalmente, utilizadas por uma linha serial, enquanto o TCP é usado em redes mais modernaas de TCP/IP ou UDP/IP INSTRUMENTS (2019) .

Em caso de erro relacionado à função Modbus solicitada, o campo contém um código de exceção que permite ao aplicativo do servidor, usar essa função, para determinar uma ação consequente, que deverá ser executada, ao final Modbus-IDA (2006). As duas figuras que se seguem, identificam situações com erro presente ou ausente. A figura 13 indica uma transação livre de erro, como se observa:

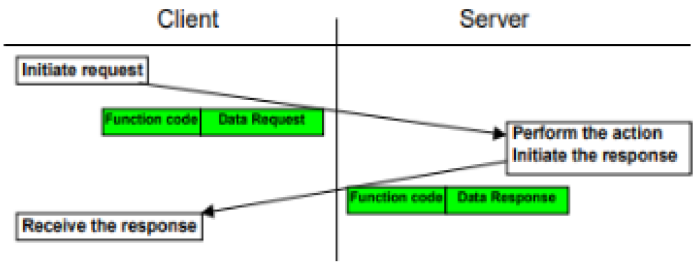


Figura 13 – Transação Modbus sem Erro.

Fonte: Modbus-IDA (2006)

A figura 14 apresenta uma situação onde ocorre um erro, relacionado à função Modbus. Nesta condição, o servidor envia um código de função de exceção, para que o erro evidenciado, seja tratado e/ou corrigido, pelo cliente, de forma adequada:

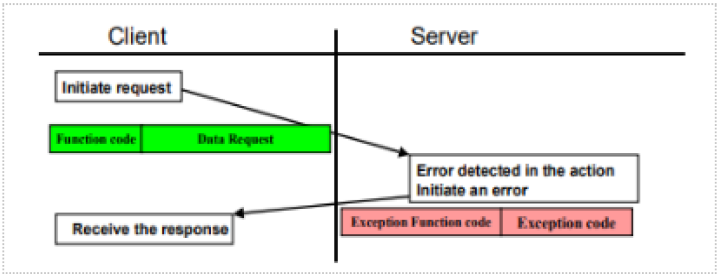


Figura 14 – Transação Modbus com Erro.

Fonte: Modbus-IDA (2006)

2.5.2 Modbus-RTU

Diferente da ADU ASCII, a ADU RTU não tem um caractere especifico, para delimitar o início ou o fim de uma mensagem que é definida pela ausência de transmissão, durante

um tempo mínimo, correspondente a três vezes e meia do tempo de transmissão de um bit de dados. Se esse tempo passar e não houver transmissão de nenhuma mensagem, o sistema entenderá que a transmissão acabou. Assim, se o sistema precisar transmitir uma nova mensagem, terá que realizar a transmissão de um novo telegrama, mas se durante a transmissão, o tempo entre os bits for maior que o tempo mínimo, o telegrama será considerado inválido, pois o controlador descartará os bits recebidos e aguardará o tempo mínimo, para montar um novo telegrama, com os bits transmitidos Freitas (2014).

O tempo de transmissão de uma palavra do quadro pode variar e depender da taxa de comunicação do sistema. Com isso, quanto maior a taxa de comunicação, menor será o tempo de transmissão. O tempo de silêncio mínimo pode ser menor que dois minutos, mas não depende da taxa de comunicação INSTRUMENTS (2019).

No modo RTU, cada mensagem é transmitida como um fluxo contínuo de caracteres e tem um byte de tamanho, ou seja, dois caracteres hexadecimais de 4 bits. Dentre as vantagens do modo RTU está o fato de que nele ocorre uma maior densidade de caracteres, que permite um melhor processamento de dados, se comparado ao modo ASCII. Entre as desvantagens, destaca-se o fato de que o dispositivo tem que esperar o término do tempo mínimo, antes de poder processar o pacote, pondo em risco o desempenho do sistema de comunicação. A figura 15 mostra como é a estrutura da ADU da RTU:

Figura 15 – Estrutura da ADU

Fonte: INSTRUMENTS (2019)

Considera-se que a estrutura do Modbus PDU seja bem simples, uma vez que é composta, apenas por duas partes, além da PDU. Dentro dessas partes encontramse o endereço do escravo, propriamente dito, e o CRC (Cycling Redundancy Check), o qual servirá para identificar possíveis erros na transmissão do telegrama. A figura 16 mostra a composição de bits do endereço do escravo, da função da PDU, da parte de dados da PDU e dos bits reservados, para a verificação de erros:

2.6 Protocolo I2C

Diferente do protocolo Modbus RTU, O protocolo I2C é síncrono, exigindo um sinal de relógio, para que seja feia uma transmissão de mensagem. A comunicação é feita, por meio de um barramento bidirecional que possui duas linhas de barramento, uma, de dados serial

Address	Function	Data	CRC Check
8 bits	8 bits	N x 8 bits	16 bits

Figura 16 – Composição da PDU.

Fonte: Freitas (2014)

(SDA) e outra, de relógio serial (SCL). Com isso, a mensagem é transmitida no formato de um bit bidirecional, podendo alcançar diferentes velocidades: de 100 kbit/s no modo padrão, de 400 kbit/s e 1 Mbit/s, em modo rápido, de até de 3,4Mbit/s, em alta velocidade, sendo possível aumentar ainda mais a velocidade para 5Mbit/s, desde que feita uma transmissão no modo unidirecional Semiconductors e Eindhoven (2021).

Outra diferença entre os protocolos Modbus e o I2C é que o segundo admite mais de um controlador em rede. Essa montagem é descrita na figura 17:

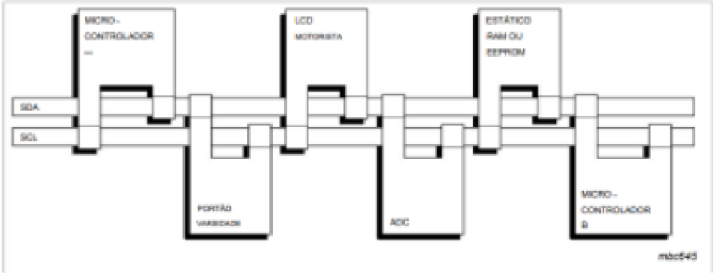


Figura 17 – Exemplo de Montagem uma Rede I2C.

Fonte: Semiconductors e Eindhoven (2021)

2.7 Protocolo One-Wire

O protocolo One-Wire é usado para fazer a interface de sensores e dispositivo simples, por meio de um único fio, possui dois modos de velocidade de operação, o modo de velocidade padrão e o modo de Overdrive. No primeiro, a velocidade é de 16,3kbit/s, enquanto no segundo, a velocidade é 10 vezes maior que o modo padrão Instruments (2018).

A transmissão acontece, por meio de uma única linha de dados, de um dispositivo para outro. Salienta-se que a comunicação no barramento é Halfduplex, quer seja, nenhum dispositivo pode transmitir e receber as mensagens, ao mesmo tempo. O mestre é responsável pelo início de toda a transferência, na linha de dados Instruments (2018).

O barramento One-Wire pode conter vários escravos submetidos a um único mestre, responsável pelo controle da transferência de informações na rede. A comunicação acontece entre mestre e escravo, considerando o fato de que os escravos não se comunicam entre si Instruments (2018).

A figura 18 traz a topologia de um barramento One-Wire, ilustrando como todos os dispositivos se ligam à mesma linha de terra, destacando a necessidade da existência de um resistor entre o VDD e a linha de dados Instruments (2018).

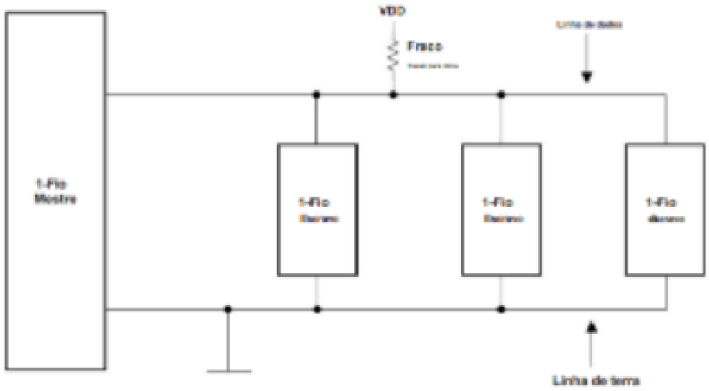


Figura 18 – Topologia de Barramento One-Wire

Fonte: Instruments (2018)

2.8 Pyserial

Este módulo encapsula o acesso à porta serial, fornecendo backends, para Python, sendo compatível com Windows, OSX, Linux, BSD, IronPython e, possivelmente, com qualquer sistema compatível com Posix. O módulo denominado “serial” seleciona, automaticamente, o backend apropriado, sendo lançado, sob uma licença de software livre Liechti (2021). A figura 2.18 apresenta um exemplo de escrita pela porta serial, da seguinte forma:

```
>>> import serial
>>> ser = serial.Serial('/dev/ttyUSB0')  # open serial port
>>> print(ser.name)         # check which port was really used
>>> ser.write(b'hello')     # write a string
>>> ser.close()             # close port
```

Figura 19 – Exemplo de Escrita pela Porta Serial.

Fonte: Liechti (2021)

O exemplo apresentado na figura acima, não especifica a taxa de transmissão. Quando essa taxa não é especificada, considera-se uma taxa de 9600. Nesse módulo, a comunicação é simples, ela é iniciada com a abertura de uma porta serial, podendo ser feita em diferentes velocidades e formas, como mostra a figura 20:

A Figura 20, além de exemplificar a leitura feita por porta serial, ilustra três casos/tipos de leituras, quer seja, a leitura de um único bit, a leitura de 10 bits e a leitura da uma linha completa que delimita a linha, com o caractere de fim de linha, o “/n”.

```
>>> with serial.Serial('/dev/ttyS1', 19200, timeout=1) as ser:
...     x = ser.read()          # read one byte
...     s = ser.read(10)        # read up to ten bytes (timeout)
...     line = ser.readline()   # read a '\n' terminated line
```

Figura 20 – Exemplo de Leitura de Mensagem pela Porta Serial.

Fonte: Liechti (2021)

2.9 Bibliotecas Modbus Python

Para fazer a comunicação Modbus utilizando a linguagem de programação Python, existem, basicamente, três bibliotecas e cada uma, com sua particularidade, sendo elas: modbustk, pymodbus e minimalmodbus.

2.9.1 Modbus TK

A biblioteca Modbus tk, é uma api de fácil de utilização, na medida em que usa uma estrutura de pilha Modbus e oferece suporte mestre/escravo, tanto para Modbus TCP IP, quanto para Modbus RTU. No caso do Modbus RTU, há demanda pela utilização da api Pyserial. Essa biblioteca possui limitada documentação, agrupada em um único grupo criado no Google Drive e na página do desenvolvedor no GitHub.

2.9.2 Pymodbus

A biblioteca Pymodbus é a mais completa, uma vez que fornece suporte para MODBUS mestre e escravo, funcionando com uma estrutura de pilha MODBUS que usa o twisted, considerado um mecanismo de rede, orientado para eventos escritos em Python e licenciado sobre código aberto Collins (2013).

Além disso, a api Pymodbus fornece suporte à modbus TCP, UDP, serial ascii, serial RTU e serial binário, isto tanto para mestre, quanto para escravo. Nos dois modos, mestre (cliente) e escravo (servidor) pode ser assíncrono (alimentado por twisted/tornado/asyncio) ou síncrono.

2.9.3 Minimalmodbus

O MinimalModbus é um módulo Python de fácil utilização que facilita a comunicação entre os instrumentos (escravos) de um computador (mestre). Usa o protocolo MODBUS e é destinado a ser executado no mestre. A única dependência é o módulo PySerial (também python puro) Berg (2021). Apresenta como limitação, o fato de que apenas os modos ascii e

RTU sejam habilitados para ele. A figura 21 mostra o processo de estabelecer a comunicação, mediante o uso do módulo Python MinimalModbus:

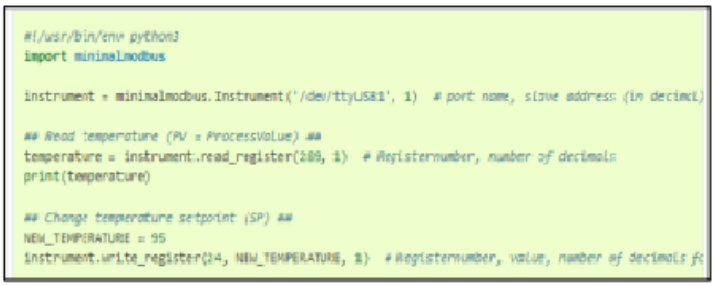


Figura 21 – Exemplo do Estabelecimento de uma Comunicação MinimalModbus.

Fonte: Berg (2021)

No código representado na figura acima, primeiro ocorre a importação do módulo, em seguida, o estabelecimento do nome da porta de comunicação e do endereço do escravo, onde se deseja receber ou enviar as mensagens. A figura ilustra o processo de envio de uma mensagem, para escrita no registrador, do escravo.

2.10 PyDrive

Pydrive é uma api do Python, em que é possível realizar a comunicação entre o dispositivo e o Google Drive. Ao transmitir dados e obter uma autenticação, o Pydrive realiza o protocolo de autenticação OAuth2.0, pela execução dos seguintes passos disponivel na documentação:

1. Vá para o console de APIs e faça seu próprio projeto.
2. Pesquise ’API do Google Drive’, selecione a entrada e clique em ’Ativar’.
3. Selecione ’Credenciais’ no menu à esquerda, clique em ’Criar credenciais’ e selecione ’ID do cliente OAuth’.
4. Agora, o nome do produto e a tela de consentimento precisam ser definidos -> clique em ’Configurar tela de consentimento’ e siga as instruções. Depois de terminar:
    a. Selecione ’Tipo de aplicativo’ para ser um aplicativo da web.
    b. Insira um nome apropriado.
    c. Insira http: // localhost: 8080 para ’Origens JavaScript autorizadas’.
    d. Insira http: // localhost: 8080 / para ’URIs de redirecionamento autorizados’.
    e. Clique em ’Salvar’.

5. Clique em ’Baixar JSON’ no lado direito do ID do cliente para baixar.

O arquivo baixado deve estar no mesmo diretório do trabalho que será realizado. Assim que todos os passos (1 a 5) forem executados, o sistema estará apto, para realizar a autenticação no Google Drive, usando o Pydrive. Um exemplo de autenticação simples, usando essa api, é descrita na Figura 22:

```
from pydrive.auth import GoogleAuth

gauth = GoogleAuth()
gauth.LocalWebserverAuth() # Creates local webserver and auto handles authentication.
```

Figura 22 – Código Exemplo para Autenticação Pydrive.

Fonte: Nabel (2016)

3 Descrição de Materiais

Este capítulo fala sobre os materiais selecionados e utilizados, para a produção e realização do projeto que aferiu dados relacionados à temperatura, tensão e corrente de cinco módulos fotovoltaicos, usando os protocolos Modbus-RTU, OneWire E I2C, mediante a averiguação de irradiação solar, umidade, north e pressão atmosférica. Destaca-se que esses materiais foram usados na coleta de dados de refrigeração forçada e passiva de painéis fotovoltaicos, instalados em terra e em lâmina de água.

3.1 FieldLogger

De acordo com novus (2021) o FieldLogger é um equipamento de aquisição e registro de dados analógicos e digitais, de alta resolução e velocidade. Resultante de um avançado desenvolvimento tecnológico, o produto se destaca em diversos aspectos, como alto desempenho, alta conectividade e facilidade na configuração e operação. Esta tecnologia é considerada a solução ideal, para aplicações que requerem flexibilidade e funcionalidade, em relação a diversos padrões de redes industriais. A Tabela 1 apresenta um FieldLogger, apontando suas dimensões e algumas de suas principais características:

Tabela 1 – Especificações do Fieldlogger.

Nome	especificações
Entradas analógicas	8 entradas
Entradas/saídas digitais	8 entradas/saídas
Saídas a relé	1 uma saída
Memória interna	2 MB
Interface para Cartão SD	Até 16 GB - não disponível em alguns modelos
Interface RS485 o Principal	Modbus RTU mestre e escravo
Serviços Ethernet (não disponível em alguns modelos)	O DHCP o HTTP, o FTP ,o SMTP, o SNMP e o Modbus TCP

Fonte: adaptado de novus (2021)

A figura 23 e a figura 24 mostram como é um FieldLogger, indicando suas respectivas dimensões. Na primeira, é possível perceber duas linhas de terminais,uma em cima e a outra embaixo, as duas linhas possuem 50 terminais, os quais são usados, para realizar conexões diversas, que são, basicamente, ethernet, conexão de entrada, alimentação, relés de saída, saídas para alimentação auxiliar, entradas digitais e comunicação serial. Essas informações estão representadas em seu layout:

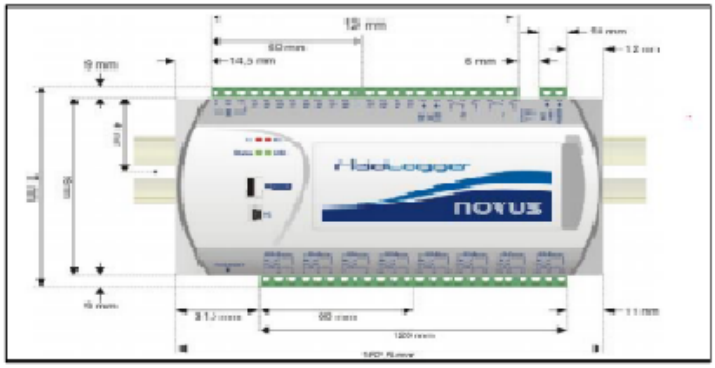


Figura 23 – Código Exemplo para Autenticação Pydrive.

Fonte: novus (2021)

A figura 24 apresenta uma dimensão da lateral do FieldLogger, indicando também, suas dimensões:

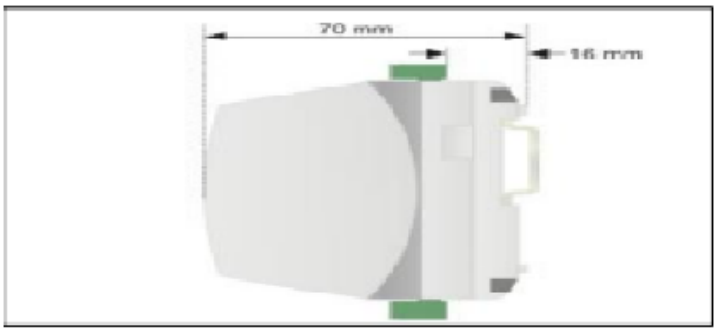


Figura 24 – Código Exemplo para Autenticação Pydrive.

Fonte: novus (2021)

A figura 25 apresenta a parte superior dos terminais com numeração de 25 até 50. Nesse desenho, a alimentação é de 24v DC/AC, sendo que, para tensão AC, as frequências permitidas são de 50hz ou de 60hz, além disso, os sistemas possuem um fusível de proteção de 0.5A. Os terminais de 27 a 32 são destinados aos relés de saída, os quais podem ser usados como um alarme, para caso de falha no sistema. Os terminais de 25 a 47 são destinados a entradas e saídas digitais que têm uma tensão de 5v. Já os terminais de 48 a 50 destinam-se à camada física da rede RS485, como se observa:

Destaca-se que a figura 25 representa o lado superior do equipamento de FieldLogger, com especificações para versão standard e 24 V, sem citar outra versão ou modelo. Já a figura 26 ilustra a conexão inferior que se destina às entradas analógicas:

O FiedLogger possui um software configurador que permite verificar as condições de trabalho do hardware. Neste sentido, o projeto de placas flutuantes discriminado neste estudo, utilizará a opção escravo.

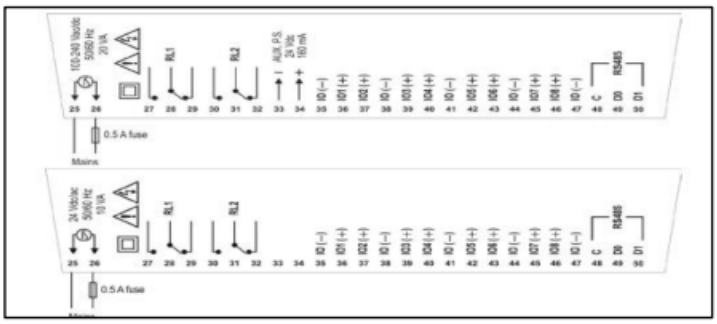


Figura 25 – Código Exemplo para Autenticação Pydrive.

Fonte: novus (2021)

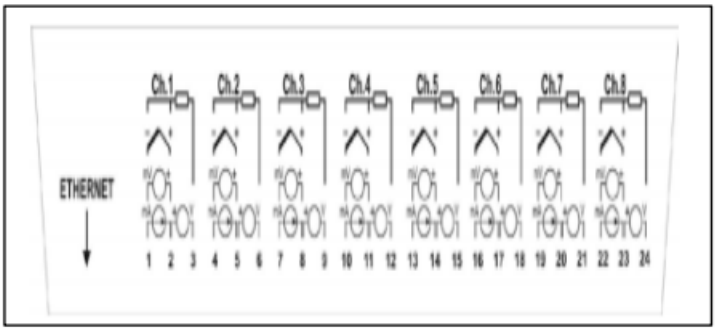


Figura 26 – Código Exemplo para Autenticação Pydrive.

Fonte: novus (2021)

3.2 RK300-02

É um sensor de concentração de poeira que usa o princípio de espalhamento a laser, para detectar a partícula de poeira de até um micrometro de tamanho. Existem dois modelos de uso: o tipo interno, chamado de indoor e o tipo externo, chamado outdoor, ambos possuem alta sensibilidade, excelente estabilidade, baixo consumo de energia e vida útil, longa. Os dois tipos são representados, na figura 27:

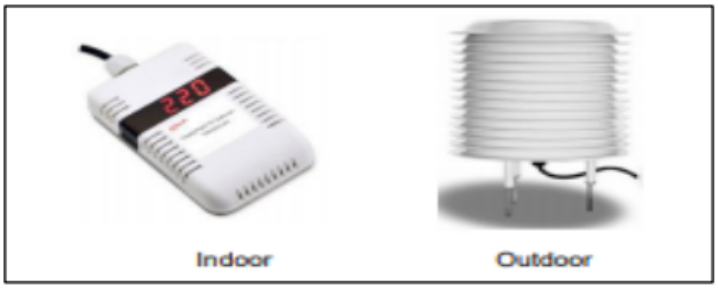


Figura 27 – Layout RK300-02.

Fonte: RK300-02 (indefinida)

Contribui para a interpretação desses dados e compreensão de funcionamento dos modelos de indoor e outdoor, a tabela 2 que aponta as especificações técnicas do sensor RK

300-02:

Tabela 2 – RK300-03 Piranômetro

Nome	especificações
Objeto De Amostragem	PM1.0, PM2.5, PM10 de concentração
Alcance	0-1000ug/m
Precisão	± 3% FS @ 25
Alimentação	5VDC, 12-24VDC
Saída	4-20mA, 0-5V, RS485
Consumo	<50mA@24V(4-20mA)
STempo De Aquecimento	3min
Tempo De Resposta	<90s
Desvio De Temperatura	0.2%FS/
Estabilidade	<±2%FS
Repetibilidade	<±1%FS
Temperatura De Operação	-20-+50@15-80% RH
Temperatura De Armazenamento	-40-60@20%-90%RH

Fonte: adaptado de RK300-02 (indefinida)

3.3 RK200-03 Piranômetro

O RK200-03 Piranômetro tem como princípio de funcionamento, o termoelétrico, uma vez que a superfície do sensor possui um revestimento preto, com alta taxa de absorção, fazendo com que os contatos da junção quente dos sensores fiquem sobre sua superfície, enquanto a junção fria fica localizada dentro do corpo. O efeito do termoelétrico é proporcional à radiação solar. A figura 28 apresenta o layout do piranômetro:

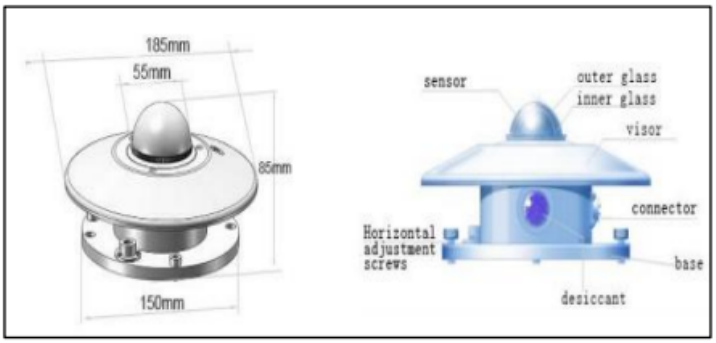


Figura 28 – Layout RK200-03.

Fonte: RK200-03 (indefinida)

A tabela 3 foi construída, para apontar as especificações de cada item que compõe o sensor RK 200-03 – Piranômetro. Os dados obtidos foram condensados em referenciais, discriminados nas categorias de item e especificação, conforme a apresentação que se segue:

Tabela 3 – RK200-03 Piranômetro

Nome	especificações
Faixa Espectral	300-3200nm
Alimentação	5V,12-24VDC
Alcance	2-2000W/m^2
Saídas 0-20mV,0-5V,4-20mA,RS485	
Saída	4-20mA, 0-5V, RS485
Sensibilidade	7-14uV*W-1*m^2
Resistência Interna	350ohm
Não Linearidade	<±2%
Angulo De Medição	Ângulo sólido 2
Desvio Zero (Desvio de Temperatura: 5k / H)	±5W/m^2
Estabilidade	±2%/ano
Correção de Cosseno	±7%(angulo de elevação solar=10°)
Efeito da Temperatura	±2%(-10-+40)
Temperatura de Operação	-40-+80
Intervalo de Recalibração	2 anos
Dessecante	Dessecante de sílica gel
Peso (Descompactado)	2,5kg
Pacote	Caixa de instrumento de liga de alumínio
Dimensão ø185*120mm	
Suporte De Instalação(opcional)	Suporte De Instalação(opcional)
Proteção De Entrada	IP65
Condição De Armazenamento	10-60@20%-90%RH

Fonte: adaptado de RK200-03 (indefinida)

3.4 RK900-09 - Estação Meteorológica Ultrassônica, em Miniatura

A estação meteorológica ultrassônica, em miniatura, RK900-09, é composta por sensores de velocidade do vento ultrassônico, pelo sensor de direção do vento, de temperatura digital, de umidade e de pressão atmosférica, permitindo a detecção precisa e ágil da velocidade e da direção do vento, de sua temperatura, pressão e umidade atmosférica.

O RK900-09 tem uma unidade de processamento de sinal, embutida, que pode integrar sinais, conforme a demanda do usuário. Sua estrutura possui alta resistência, conferindo-lhe alta confiabilidade, para ser utilizado em ambientes mais agressivos. O sensor também possui uma saída RS485 que permite a leitura de dados, via Modbus RTU. A Tabela 7 apresenta as especificações deste modelo, levando em conta, a análise de cinco aspectos que somados, atribuem caráter único ao sensor, quer seja: item, princípio, faixa de operação, resolução e acurácia.

Tabela 4 – Especificações do RK900-09.

Item	**Principio**	**Faixa de operação**	**Resolu ção**	**Resolu ção**
Velocidade do vento	Ultrassônico	0-40m/s	0.01m/s	±0.5+2%FS
Velocidade do vento	Ultrassônico	0-359°	1°C	±3°C
Temperatura atmosférica MÊS	MEMS	-40-+100°C	0.1°C	±0.5
Temperatura atmosférica MÊS	MEMS	0-100%RH	0.1%RH	±3%
Pressão atmosférica	MEMS	10-1100hPa	0.1hpa	±1.5hPa
Alimentação	24V			
Saída	RS485			
Consumo de energia	0.6W			
Temperatura de operação	-40-+80°C			
Avaliação ip	IP65			
Material principal	ABS +Aluminium alloy			

Fonte:adaptado de RK900-09 (indefinida)

O modo de transmissão do Modbus RTU já vem pré-configurado, apresentando uma taxa de transmissão 9600, com 8 bits de dados por mensagem, 1 bit de parada, e nenhum bit de verificação. O endereção de escravo também é pré-definido, tendo como padrão de fábrica, o endereço 01H, podendo ser definida a faixa de 00H a FFH, dependendo da necessidade do usuário. A Figura 29 apresenta o layout deste sensor, contendo algumas dimensões, para aferição de medidas de temperatura, dentre outras habilitações:

3.5 Módulo de Termopares AM8t

É um módulo de aquisição de dados de termopares. Com ele é possível ler até 8 termopares, sendo estes do tipo J, K ou T. Quanto à forma de comunicação ela ocorre por meio de uma interface de comunicação serial RS485 isolada, se valendo de dois protocolos Modbus rtu e o protocolo LG inverter.

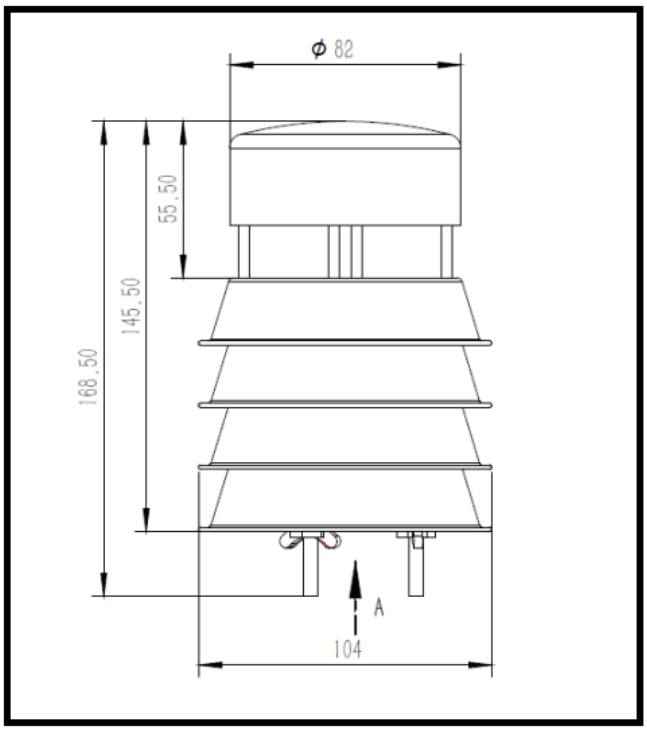


Figura 29 – Layout RK200-03.

Fonte: RK900-09 (indefinida)

Quanto à forma de ligação, ela é feita mediante a ligação direta dos termopares, no módulo AM8T, através de um conjunto de terminais que fica localizado, na parte superior do dispositivo.

É alimentado com uma tensão de 24Vcc, sendo recomendado pelo fabricante, o uso de uma fonte individual para o AM8T, para evitar eventuais falhas no isolamento dos termopares, impedindo que outros equipamentos ligados à rede, sejam danificados também.

Para sanar dúvidas referentes ao processo de alimentação do módulo de termopares AM8t, construir conhecimentos sobre as demais especificações e características do módulo, a Tecnolog (2020), disponibiliza um manual para o usuário, que foi muito útil para o desenvolvimento e produção deste projeto. A figura a seguir traz uma imagem do módulo:

A configuração do módulo é feita através de chaves localizadas do lado esquerdo do equipamento como mostra a figura 3.8. A posição da chave para cima significa que está ligada, posicionada para baixo, desligada. As chaves são enumeradas da esquerda para direita, na base binária com uma representação litter edian (o bit mais à direita é o menos significativo). As duas primeiras chaves representam o tipo de termopar utilizado. Na posição desligada (00) representam o termopar do tipo J. Se a chave 1 estiver desligada e a chave 2, ligada, representa o termopar K. Com a chave 1 ligada e a chave 2 desligada, há um termopar do tipo T. Se duas primeiras chaves estiverem ligadas, caracterizará uma posição reservada. A

Figura 30 – Layout RK200-03.

Fonte: TECNOLOG (2020)

chave 3 representa o tipo de protocolo utilizado. Na posição ligada significa que o protocolo utilizado foi o Modbus RTU, caso contrário, o LG inverter. A chave 4 representa a velocidade de comunicação, sendo a posição ligada, para uma velocidade 19200bps e a posição desligada, para uma velocidade de 9600bps. Já as chaves de 5 a 8 representam o endereço do módulo na rede, sendo que este pode tomar como endereço a faixa entre 00001a 1111. Normalmente, o endereço 0000 é reservado ao mestre da rede.

Uma vez definida a configuração, para o protocolo Modbus RTU, o AM8T possui as funções 03 e 04, que representam, frespecivamene, a leituira os registradores de retenção e ler os registradores de entrada respectivamente. Os registradores são disponibilizados, conforme o objetivo do usuário, sendo o registrador 0 destinado à junta fria e os registradores de 1 a 8, determinantes da temperatura dos canais 1 a 8, com resolução normal. Já os registradores de 20 a 27 destinam-se à temperatura nos canais de 1 a 8, com resolução total, ou seja, de 10 vezes. As demais especificações são mostradas na tabela 5:

Tabela 5 – Especificações do AM8T

Nome	especificações
Tensão de alimentação:	24VDC ± 20%
Precisão:	0.25% leitura + 1°C
Deriva Térmica:	0.3% FE (0 a 60°C).
Taxa amostragem:	1Hz.
Consumo máximo:	1W.
Dimensões:	98x71x36mm
Temperatura operação:	0 a 60 oC.
Grau de proteção	IP30

Fonte: adaptado de TECNOLOG (2020)

3.6 Controlador de Carga (TRIRON-N)

Os controladores de cargas TRIRON-N têm como característica principal, o fato de ser em um projeto de caráter modular que pode ser integrado a diferentes módulos de display e interface diferentes, para atender uma ampla gama de requisitos funcionais. Além disso, possui capacidade para identificar e carregar os drives de vários módulos conectados a ele.

Com o moderno algoritmo de controle MPPT, o controlador é capaz de obter o máximo de energia solar, sobre as mais diferentes condições, garantindo eficiência superior de 20% a 30%, se comparado aos controladores de cargas PWM. O equipamento possui um modo adaptativo de três estágio, que se baseia no controle de circuito digital, sendo possível aumentar o ciclo de vida das baterias, melhorar, significativamente, o desempenho do sistema e atuar como suporte, para todas as funções de proteção eletrônica, incluindo proteção de sobrecarga e descarga efetiva das baterias e minimizando os danos causados aos componentes que são provocados por falhas na instalação.

Além das propriedades supracitadas, o dispositivo consegue se comunicar, através de uma interface RS485, permitindo a configuração via software pc. Com isso, é possível acompanhar a geração em tempo real, ajustar parâmetros e até mesmo, ligar e desligar os sistemas (EPEVER, 2020). A figura 31 apresenta o controlador de carga, descrito neste item:

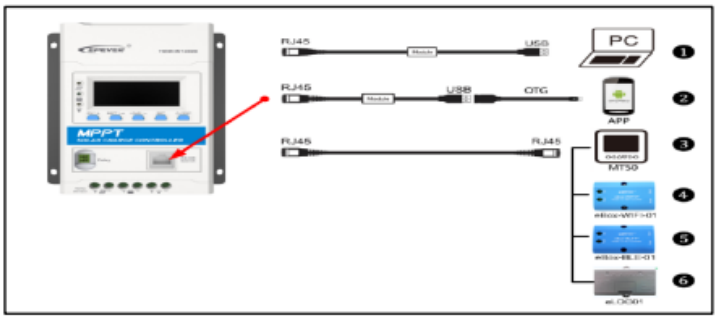


Figura 31 – Layout RK200-03.

Fonte: EPEVER (indefinida)

Nota-se, pela figura 3.9, que o controlador precisa de um cabo com entrada RJ45, um módulo de conversão de RJ45 para USB, a fim de que possa controlar e monitorar, via protocolo Modbus RTU, o dispositivo, quando estiver no modo escravo EPEVER (indefinida). A tabela 6 apresenta características do controlador, relacionadas ao seu comportamento elétrico:

3.7 Raspberry PI 4

É um microcomputador composto por um processador Quard-core de 64 bits, de alto desempenho, com duas entradas HDMI, com capacidade para suportar dois monitores, com

Tabela 6 – Especificações do RK900-09.

Item	**TRIRON 4210N**
Tensão nominal	12/24VDC AUTO
Corrente de carga nominal (A)	40
Corrente de descarga nominal (A)	40
Faixa de tensão na bateria	8 32v
Tensão máxima de circuito aberto	100V Tamb e 92V à 25°C
Faixa de tensão MPP	(tensão na bateria +2V) 72V
Máxima potência de entrada fotovoltaica	520W/12V 1040W/24V
Auto consumo	
Queda de tensão no circuito de descarga	14mA(12V)15mA(24V)
Coeficiente de compensação de temperatura	-3mV//2V (padrão)
Aterramento	negativo comum
Interface RS485	5VDC/100mA
Interface relé	30VDC/1A
Tempo de retroalimentação	Padrão:60S, faixa: 0 999S(0S: a luz de fundo está LIGADA o tempo todo)

Fonte: adaptado de EPEVER (indefinida)

uma resolução de até 4k. O hardware vem com até 8gb de memória ram, conferindo-lhe boa velocidade de funcionamento.

Em relação à conectividade, pode ocorrer, por meio de Wi-fi de 2.4GHz e 5.0 GHZ no padrão IEEE 802.11b/g/n/ac; por meio de Bluetooth 5.5, ou através de uma porta BLE Gigabit ethernet. O microcomputador possui 4 portas USB, sendo duas 2.0 e as outras 3.0 . A Figura 32 apresenta o layout do Raspberry PI 4::

Figura 32 – Layout Raspberry PI 4.

Fonte: raspberry (2019)

A figura 32 mostra o microcomputador possui entradas e saídas de uso geral, chamadas de GPIO, localizadas na parte superior do Raspberry PI4, são compostas por 40 pinos,

através dos quais torna-se possível controlar atuadores, receber e enviar informações. A alimentação é de 5VDC e é feita, via conector USB-C, com no mínimo de 3A raspberry (2019).

O Raspberry PI 4 possui uma entrada para cartão micro SD que é usado para armazenamento e hospedagem do sistema operacional, chamado de Raspberry Pi OS (antigo Raspbian), é um sistema que tem como base, o Linux e a distribuição Debian raspberry (2019).

3.7.1 Raspberry PI e Sense Hat

O Sense Hat é um complemento ao Raspberry PI, lançado em 2015, na estação espacial internacional, tendo sido construído, para a missão Astro PI. O Sense Hat é composto por uma matriz de LED RGB 8x8; um joystick de cinco botões que inclui sensores de giroscópio, de acelerômetro, de magnetômetro, de temperatura, de pressão barométrica e de umidade.

Os recursos do Raspberry Pi Sense Hat são utilizados, através da programação do Raspberry PI. Para esta finalidade, existe uma biblioteca do Python, onde é possível acessar, facilmente, esses sensores, utilizando o protocolo I2C. A Figura 33 apresenta o layout do Raspberry Pi Sense Hat:

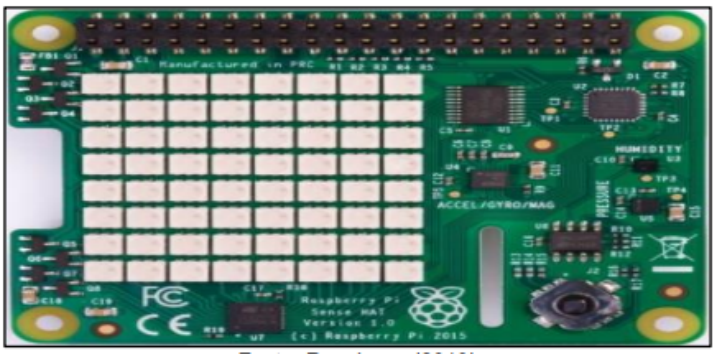


Figura 33 – Layout Raspberry Pi Sense Hat

Fonte: raspberry (2019)

3.7.2 Sensor de Temperatura DBS18B20

O sensor de temperatura DS18B20 é bastante versátil. Sua faixa de medição varia de -55°C à +125°C, o que lhe permite trabalhar nos mais diversos ambientes. Ele possui uma boa resolução, pois pode ter de 9 a 12 bits, para representar a temperatura Dallas (indefinida).

O DBS18B20 possui uma interface de comunicação simples, ligando vários sensores de temperatura em um fio único, em uma única saída digital, como o que acontece com um Arduino, por exemplo. Pela interface 1-Wire, cada sensor tem um número de série único, de 64 bits, permitindo o controle de uma infinidade de sensores. Além desta característica,

o sensor possui precisão de mais ou menos 0.5% e boa confiabilidade, em suas medições Dallas (indefinida). Quanto à forma de alimentação, está é de 3.0 a 5.5VDC. A figura 34 apresenta um modelo do sensor de fio único:

Figura 34 – Layout Raspberry PI 4.

Fonte: Robocore Dallas (indefinida)

3.8 Termopar

Um termopar é um sensor utilizado para a medição da temperatura. Ele é constituído de dois metais distintos, unidos por suas extremidades e ligados a um termopar, ou a outro dispositivo, com capacidade termopar, na outra extremidade. Quando configurado, corretamente, termopares podem fornecer medições de temperatura, em uma ampla faixa de temperatura ÔMEGA. (indefida).

Quanto ao modo de funcionamento, o termopar consiste na junção de dois metais diferentes nas duas extremidades. Em caso de aquecimento em uma das extremidades, o sensor fará com que circule uma corrente contínua, ao longo do circuito termoelétrico. Agora, se esse circuito termoelétrico for interrompido no centro, constituirá um circuito aberto. A tensão de circuito aberto é a função da temperatura, e o efeito da variação da tensão com a temperatura, é chamado de efeito de “Seebeck’ ÔMEGA. (indefida).

Quanto ao tipo do termopar, existe uma variedade de diferentes combinações de metais ou calibrações, sendo mais comuns, as do tipo J, K, T, E. Para o caso especifico deste projeto, foi utilizada a calibração do tipo T que possui um intervalo de temperatura entre –250ºC a 350ºC, conforme descrito na figura 35, demonstrando que a escolha do termopar do tipo T é compatível com este projeto ÔMEGA. (indefida).

3.9 Módulo Fotovoltaico

Com 36 células, o painel fotovoltaico RSM-100P é robusto e habilitado, tanto para sistemas on-grid, quanto para sistemas off-grid, por possuir um revestimento hidrofóbico,

Intervalos de Temperatura Mais Comuns para Termopares			
Calibração	Temperatura Intervalo	Limites Padrão de Erros	Limites Especiais de Erros
J	0° a 750°C (32° a 1382°F)	Superior a 2.2°C ou 0.75%	Superior a 1.1°C ou 0.4%
K	-200° a 1250°C (-328° a 2282°F)	Superior a 2.2°C ou 0.75%	Superior a 1.1°C ou 0.4%
E	-200° a 900°C (-328° a 1652°F)	Superior a 1.7°C ou 0.5%	Superior a 1.0°C ou 0.4%
T	-250° a 350°C (-328° a 662°F)	Superior a 1.0°C ou 0.75%	Superior a 0.5°C ou 0.4%

Figura 35 – Calibração do Termopar

Fonte: ÔMEGA. (indefida)

com antirreflexo, fazendo com que o painel possa obter uma eficiência de conversão de até 15,44%.

A potência que este painel pode disponibilizar, na saída, é de até 100w, dependendo da irradiação e da temperatura a que for submetido. As duas figuras a seguir, ilustram o comportamento do painel. A figura 36 mostra, especificamente o comportamento da corrente enralação a tensão, em virtude da irradiação solar:

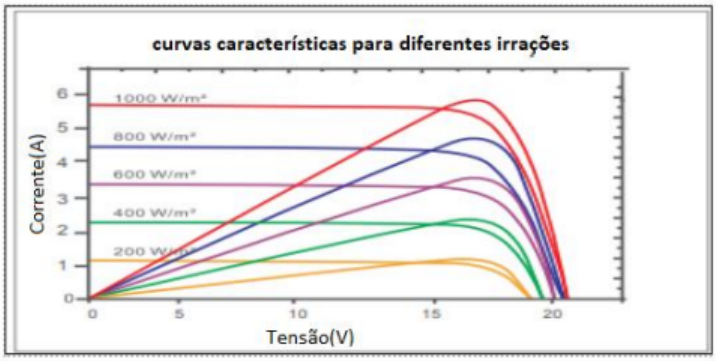


Figura 36 – Comportamento do Módulo, Corrente(I) versus Tensão(T), em Relação à Irradiação.

Fonte: datasheet do Módulo.

Pelo gráfico contido na figura 36, é possível observar o comportamento da corrente e da tensão, com a irradiação de $1000W/m^2$ até $200w/m^2$. Tendo, para irradiação de $1000w/m^2$, a tensão de máxima, a máxima potência é de 17,4V e a corrente de máxima potência de 5.75A. Na mesma direção, com a finalidade de analisar o painel frente à temperatura, a figura 37 ilustra seu comportamento:

Com a figura 37 é possível perceber que a eficiência diminui, com o aumento da temperatura, fazendo com que a tensão de máxima potência diminua e a corrente também, mostrando que quanto menor a temperatura, maior a geração.

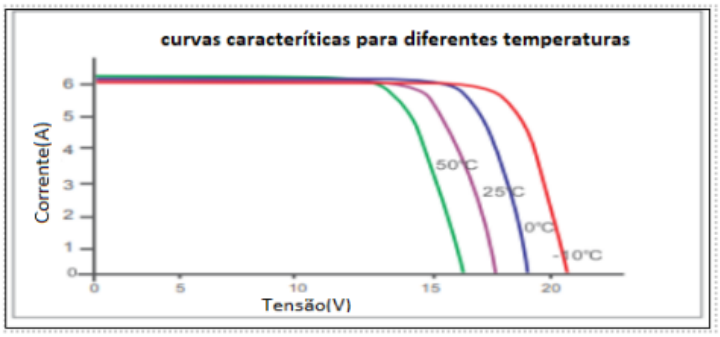


Figura 37 – Comportamento do módulo Relacionado à Temperatura.

Fonte: datasheet do Módulo.

4 Comunicação de Sistema

A comunicação do sistema foi feita através dos seguintes protocolos, Modbus RTU, I2C e o protocolo *One-Wire*. A comunicação dos dispositivos com mestre da rede Raspberry PI foi desenvolvida, via software escrito em *Python*, com o auxílio de API's do próprio *Python*. Os dispositivos utilizados na rede foram o Sense Hat, o AM8T, o FieldLogger, além dos sensores de temperatura DS18B20.

4.1 Mapa da Rede

Pela ocorrência de atraso na entrega dos materiais solicitados pela universidade, para a construção do projeto em questão, somente alguns sensores foram utilizados e integrados à rede, como demonstra a figura 38 que contém o diagrama da rede deste estudo:

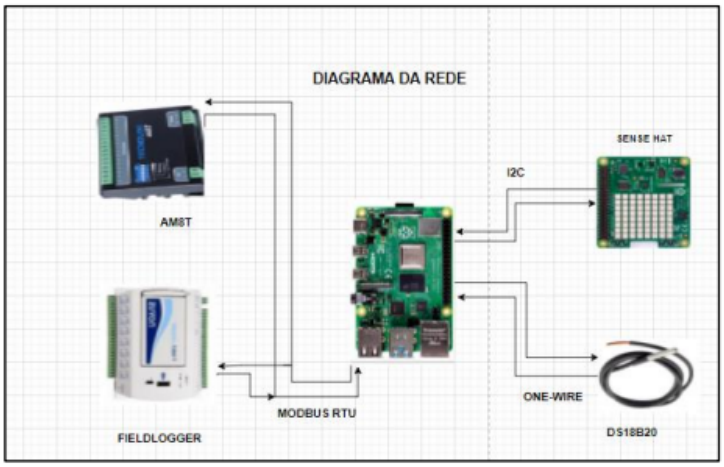


Figura 38 – Diagrama da Rede.

Fonte: Produzida pelo autor

A figura 38 mostra todos os dispositivos conectados ao Raspberry PI, sendo o FieldLogger e o AM8T, escravos Modbus. O *Sense Hat* é escravo do I2C. Os sensores DS18B20 se comunicam com o Raspberry PI, através do protocolo *One-Wire*.

O FieldLogger e o AM8T foram usados para aferição de temperatura dos termopares, instalados nos módulos fotovoltaicos. O Sense Hat foi usado, para verificar os dados referentes à umidade, *north* magnético e pressão atmosférica, enquanto o DS18B20 foi utilizado, para medir a temperatura de entrada e saída do trocador de calor.

O Raspberry PI 4 ficou encarregado de ler os módulos e os sensores. Cada leitura ficou salva em um arquivo no formato CSV, tendo em vista que, após 1300 leituras, o Raspberry

PI envia os dados, para uma pasta no Google Drive. Se houver algum erro na leitura, o Raspberry PI envia um e-mail para o desenvolvedor, informando qual módulo ou sensor apresentou problemas.

4.2 Software de Configuração e Coleta do FieldLogger

Para estabelecer uma operação do FieldLogger, é preciso configurar o dispositivo, via *software* Configurador, que é disponibilizado no site do fabricante. Uma vez instalado, é necessário o uso de um cabo USB, para fazer a comunicação entre o PC e o dispositivo, conforme representação da figura 39:

Figura 39 – Tela Principal

Fonte: novus (2021)

Na tela principal do software configurador, destacam-se quatro botões, usados para realizar a configuração, o diagnóstico, a coleta e selecionar as preferências. No botão de configuração, é possível alterar a configuração do FieldLogger. No de diagnóstico, pode-se ler os canais habilitados, a situação de alarmes configurados, bem como as informações gerais e de status do dispositivo. O botão de coleta permite que se efetue a coleta dos dados de registro do FieldLogger, bem como visualizá-los, em diversos formatos. E, no quinto botão, é possível alterar algumas preferências de *software*.

4.2.1 Configuração

A configuração do software é bem simples, bastar clicar no botão de configuração, indicado na figura 39 que, automaticamente, o software será direcionado, para a segunda tela do configurador, como exemplo a seguir:

Na tela de configuração, permite que seja feita a verificação das configurações mais atualizadas de um FieldLogger, assim também, fazer uma nova configuração ou abrir outra já existente.

Figura 40 – Tela de Configuração

Fonte: novus (2021)

Para realizar uma nova configuração, é preciso clicar no botão: nova configuração, descrito na figura 40, o que levará o software a direcionar o usuário, para um novo arquivo de configuração. Essa janela pode ser visualizada na figura 41:

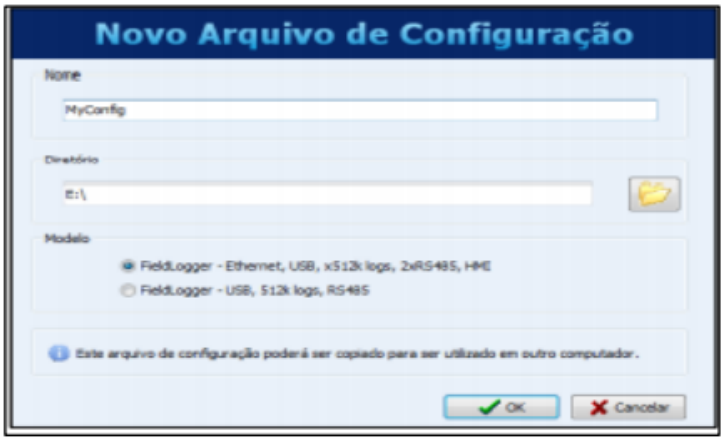


Figura 41 – Novo Arquivo de Configuração.

Fonte: novus (2021)

A figura 41 mostra que se inicia um novo arquivo de configuração. Para que uma nova configuração seja criada, é preciso definir um nome para a configuração, bem como o caminho do diretório onde ela se encontra. Depois disto, deve-se selecionar o modelo de FieldLogger utilizado, para apertar o botão de “ok”. A figura 42 apresenta o painel de configurações gerais:

Por meio das configurações gerais do dispositivo, é possível configurar o nome do equipamento, quer seja, a “Tag do equipamento”; configurarção uma IHM (interfacehomem máquina); testar o equipamento, diariamente, em um horário determinado, sendo necessária a seleção da opção: “habilitar o reset diário”; determinar o horário em que essa habilitação ocorrerá.

Figura 42 – Configurações Gerais.

Fonte: novus (2021)

Com isso, automaticamente, o PC informará o horário, ao FieldLogger. Para salvar os dados no Pen drive é só habilitá-lo, para coleta.

Logo após as configurações gerais, o sistema configurador é direcionado para a interface RS485 que pode ser configurada, no modo desabilitado, no modo mestre e no modo escravo.

Também existe a possibilidade de definir o endereço no dispositivo da rede Modbus RTU, dos bits de parada, da taxa de transmissão e de paridade. A interface RS485 é apresentada pela Figura 43:

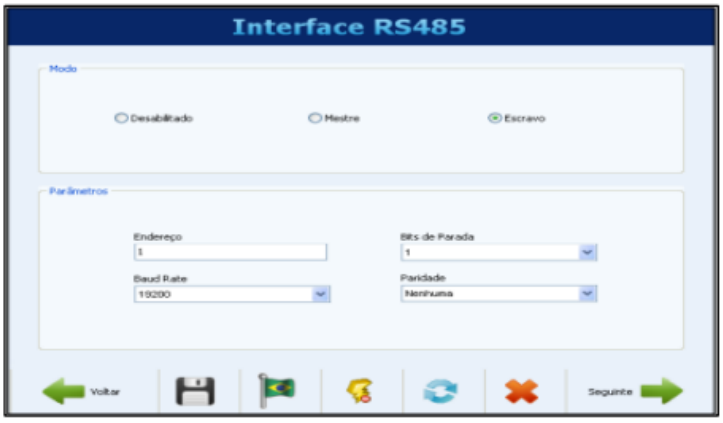


Figura 43 – Interface RS485

Fonte: novus (2021)

Em seguida, o sistema é conduzido para a interface ethernet, como mostra a figura 44 , com ela, é possível configurar os protocolos TCP/IP, FTP, SMTP, SNMP, HTTP e Modbus TCP. A figura 44 ilustra a tela que apresenta a interface de ethernet:

Para enviar um e-mail, é preciso estabelecer uma relação entre o e-mail do destinatário

Figura 44 – Interface Ethernet - Configuração SMTP.

Fonte: novus (2021)

e uma lista de destinatários. Para isso, isto é necessário que o usuário clique na indicação da tela de editar lista. Após essa ação, aparecerá uma janela, onde será possível adicionar ou remover e-mails dos destinatários. Todo esse processo é descrito pela figura 45, a seguir:

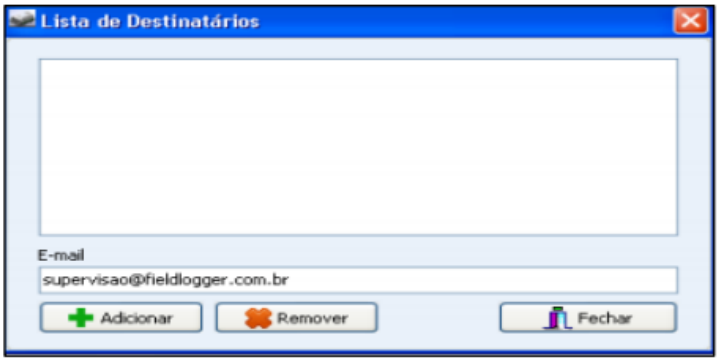


Figura 45 – Interface Ethernet - Configuração dos Destinatários de E-mail.

Fonte: novus (2021)

Havendo necessidade de alimentar uma página na Web, com os dados do dispositivo, é possível configurar a interface ethernet-HTTP, uma vez que essa página possui um parâmetro de auto atualização, que informará ao software navegador, quando a próxima página tiver que ser recarregada, com as devidas e necessárias atualizações. A figura 46 demonstra como esse processo ocorre:

Realizadas as configurações de rede, clicando no ícone “seguinte”, pode-se seguir, para as configurações de entradas analógicas e digitais. Na configuração para uma entrada analógica, primeiro é definido o nome do sensor, no campo chamado “Tag”, onde se estipula um erro, para, em seguida, estabelecer o tipo de entrada que o equipamento lerá, verificando se é um termopar ou outro sensor.

No caso de um termopar, é preciso definir a unidade da temperatura (Celsius ou Fahrenheit), sendo permitida a colocação de um filtro digital, estabelecendo-se as casas

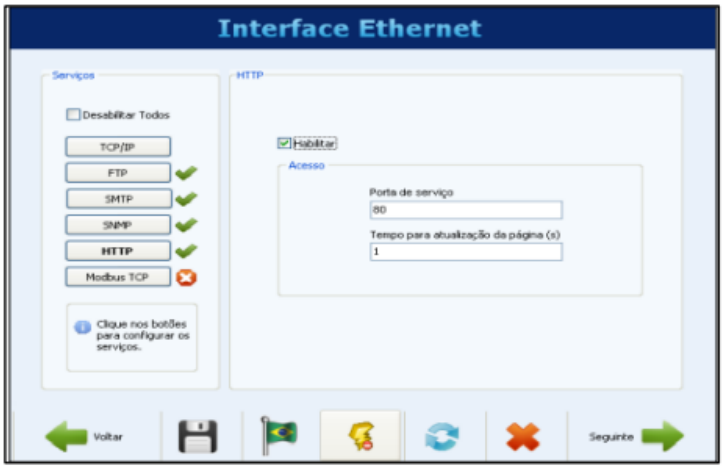


Figura 46 – Interface Ethernet - Configuração HTTP.

Fonte: novus (2021)

decimais da leitura. Feita essa configuração, é preciso definir qual canal será habilitado para leitura, assim como o intervalo de varredura. O processo de configuração de canais analógicos é apresentado pela Figura 47, da seguinte forma:

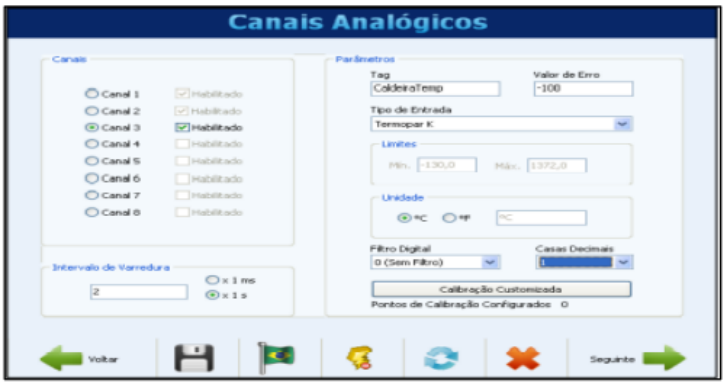


Figura 47 – Canais Analógicos e Configuração de Canal Linear.

Fonte: novus (2021)

Terminada a configuração dos canais analógicos, se o sistema demandar entradas ou saídas digitais, é possível fazer a configuração de entrada ou saída.

A figura 48 mostra como é feita a entrada digital: primeiro, seleciona-se o canal que se pretende trabalhar, depois, marca-se o modo de entrada e habilita-se a leitura, no campo de parâmetros, para, em seguida, definir um nome, para a entrada e os níveis lógicos que serão lidos, conforme demonstração:

A configuração de uma saída na escolha do canal, percorre o mesmo caminho da entrada. Para isso, seleciona-se o modo “saída”, em seguida, é feita a escolha de uma saída controlada por alarmes ou por comandos do Modbus. A figura 49 se refere à última página de configuração:

Figura 48 – Canais Digitais.

Fonte: novus (2021)

Figura 49 – Canais Analógicos e Configuração de Canal Linear.

Fonte: novus (2021)

Para configurar os registros, em caso de trabalho com rede Modbus, é necessário que sejam configurados os modos de início e de término, via comando Modbus, assim como precisa ser feita a seleção de canais para registro, o levantamento de memória, para armazená-los e o cálculo do intervalo entre esses registros.

A definição do intervalo entre os registros é de suma importância, uma vez que não adianta ter uma taxa de registro mais alta que a taxa de leitura dos canais analógicos, ou que a taxa de leitura dos canais remotos, isso só fará com que os registros tenham dados repetidos. Feitas as configurações, as mensagens podem ser enviadas ou salvas no equipamento.

4.3 Comunicação Raspberry PI

A comunicação via Raspberry é realizada com o software escrito, na linguagem de programação Python, com o auxílio de API's do próprio Python, dentre as API's que foram utilizadas, destacam-se a minimalmodbus e a sense. O algoritmo do software é descrito na

figura 50:

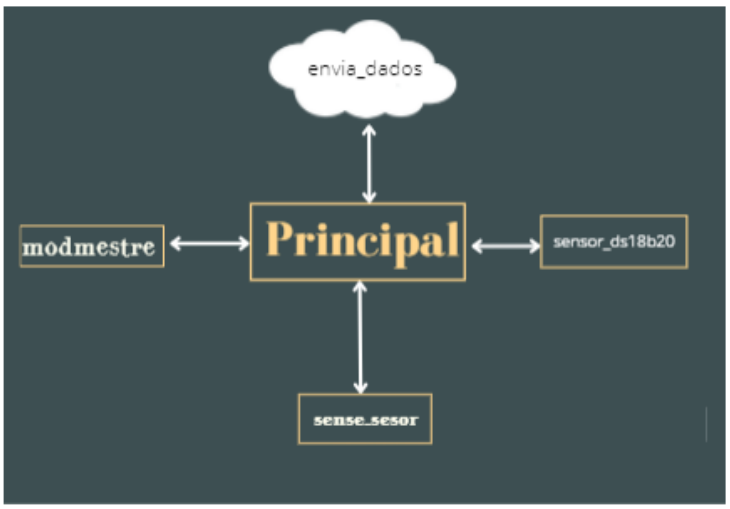


Figura 50 – Mapa do Software

Fonte: Produzida pelo autor (2021)

O algoritmo é centrado no módulo principal, fazendo a coleta dos dados dos sensores que utilizam o protocolo Modbus, I2C e One-Wire. Para cada um desses protocolos foram implementadas interfaces feitas, através de módulos do Python.

O módulo Modmestre é uma interface com os dispositivos que utilizam o protocolo Modbus, para sua comunicação. O Sense_Sesor é uma interface que coleta os dados do Sense Hat. O módulo sensor_DS18B20 é uma interface que faz a comunicação com os sensores ds18B20.

Os dados coletados por essas interfaces são salvos pelo módulo principal, em um arquivo com formato CSV, no intervalo de 1 minuto. Ao completar 1300 medições, uma cópia deste arquivo é enviada para o Google Drive, através do módulo envia_dados. O código que implementa o módulo principal é apresentado no Apêndice A.4 deste estudo.

4.3.1 Módulo Modmestre

O módulo Modmestre possibilita a comunicação Modbus RTU, fazendo com que os dispositivos Modbus RTU conectem-se com a interface RS485. A figura 51 ilustra o algoritmo que implementa e realiza a função da leitura do módulo, sendo possível analisa-lade forma bem objetiva:

O algoritmo descrito na figura acima, funciona quando o módulo principal faz a chamada da função dos dados_modbus. Esta função tem como parâmetro, o endereço do dispositivo na rede Modbus e o nome dos registradores de retenção que já foram e ainda serão lidos.

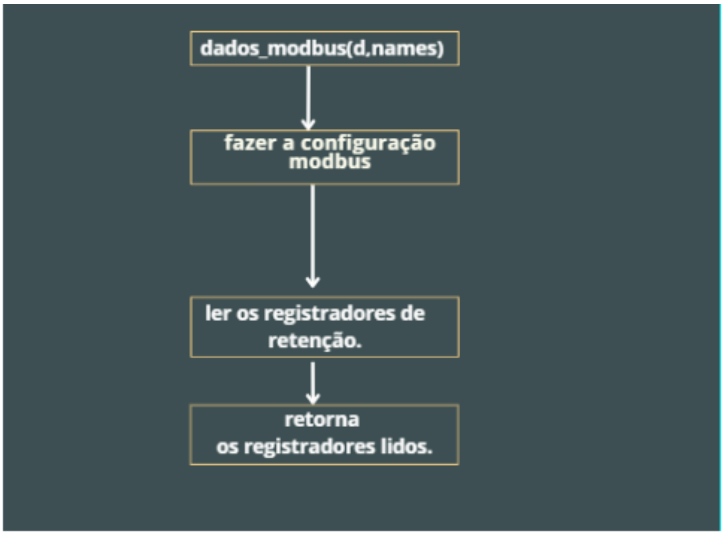


Figura 51 – Algoritmo de Implementação da Função de Leitura Modbus.

Fonte: Produzida pelo autor (2021)

Feita a chamada da função, pelo módulo principal, é preciso que seja feita uma configuração, para o estabelecimento de uma comunicação Modbus que define a taxa de transmissão dos dados, o bit de parada, o tamanho de cada mensagem e o tempo de espera.

Depois de estabelecidos os parâmetros, é necessário ler os registradores de retenção dos dispositivos. Os dados lidos serão unidos aos seus respectivos nomes de registradores. Percorrido esse caminho, a função dados_modbus (d,names) retorna para um dicionário contendo os nomes dos registradores de retenção e seus valores. O código com a implementação do algoritmo se encontra no Apêndice A.2.

4.3.2 Módulo Sense_Hat

O módulo Sense_Hat é chamado pelo módulo principal, uma vez que implementa a função ler_sensors, através da qual, realiza a leitura dos sensores espaciais e ambientais do Sense Hat. Feita a leitura, a função retorna os valores e os nomes dos sensores lidos. A figura 52 apresenta o algoritmo que realiza essa função.

Ressalta-se, que o módulo Sense Hat utiliza o protocolo I2C, para estabelecer comunicação com o Raspberry, sendo implementado pela própria api do Sense Hat. O código que implementa esse algoritmo, encontra-se no Apêndice A.5 deste estudo.

4.3.3 Módulo Sensor_DS18B20

O Módulo Sensor_DS18B20 implementa a comunicação com o sensor de temperatura DS18B20, mas essa operação exige que sejam feitas algumas configurações do protocolo One-Wire no Raspberry PI. Para fazer essas configurações, é preciso executar os seguintes

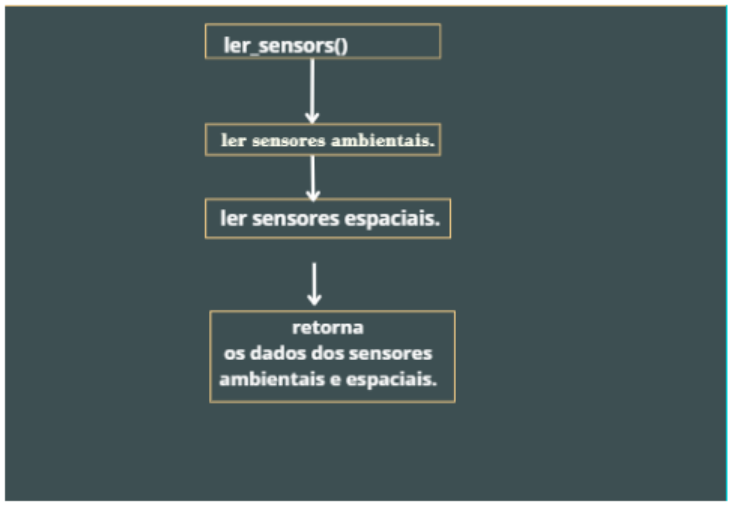


Figura 52 – Algoritmo que Implementa a Leitura dos Sensores do Sense Hat.

Fonte: Produzida pelo autor (2021)

comandos, no terminal do Raspberry PI:

1. Sudo modprobe w1-gpio.
2. Sudo modprobe w1-therm.
3. Sudo nano /boot/config.txt.
4. Sudo nano /boot/config.txt.

Uma vez configurado o Raspberry PI, o módulo principal estará habilitado para chamar o módulo Sensor_DS18B20 que executará o algoritmo descrito na figura 53, o qual, por sua vez, desempenhará uma função chamada de Tempagua (names), onde receberá os nomes dos sensores, fará a leitura desses sensores e retornará com um dicionário contendo os nomes e os valores dos sensores. O algoritmo que implementa o módulo Sensor_DS18B20, em Python está registrado no Apêndice A.3. A figura 53 ilustra esse algoritmo, da seguinte forma:

4.3.4 Módulo Envia_Dados

O módulo envia_dados é chamado pelo módulo principal, ao final de 1300 leituras. Os dados ficam na forma de um arquivo CSV, que é enviado para uma pasta no Google Drive. Para tanto, o módulo envia_dados implementa a função up_dados, que realiza a comunicação com o Google Drive. Estabelecida a comunicação, faz-se a autenticação do dispositivo, com a api do Google Drive, por meio do protocolo de autenticação OAuth2.0. Concluída a autenticação, o algoritmo é formulado e habilitado, para enviar o arquivo CSV, com os dados diários dos sensores. Esse processo é descrito na figura 54:

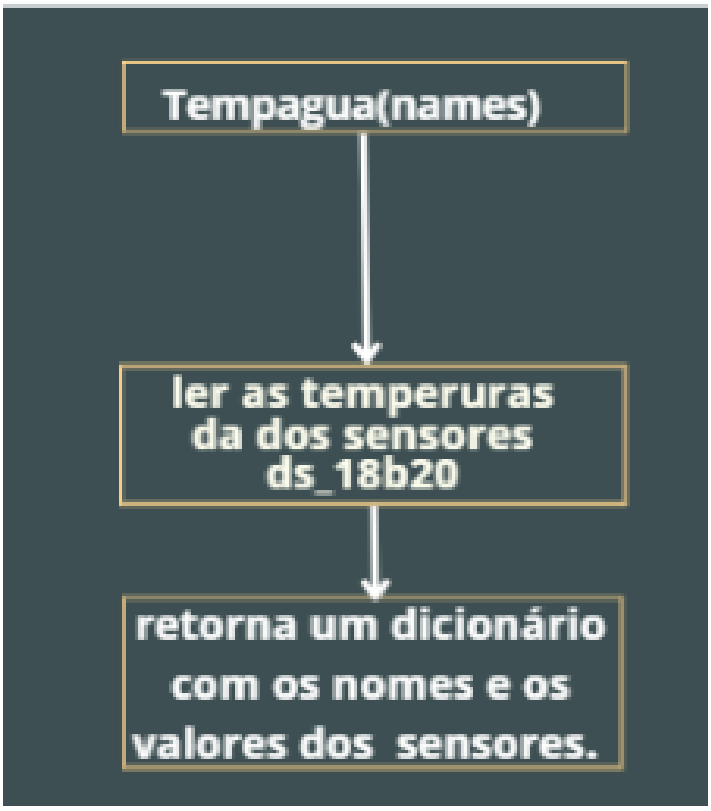


Figura 53 – Algoritmo que Implementa a Leitura dos Sensores DS18B20.

Fonte: Produzida pelo autor (2021)

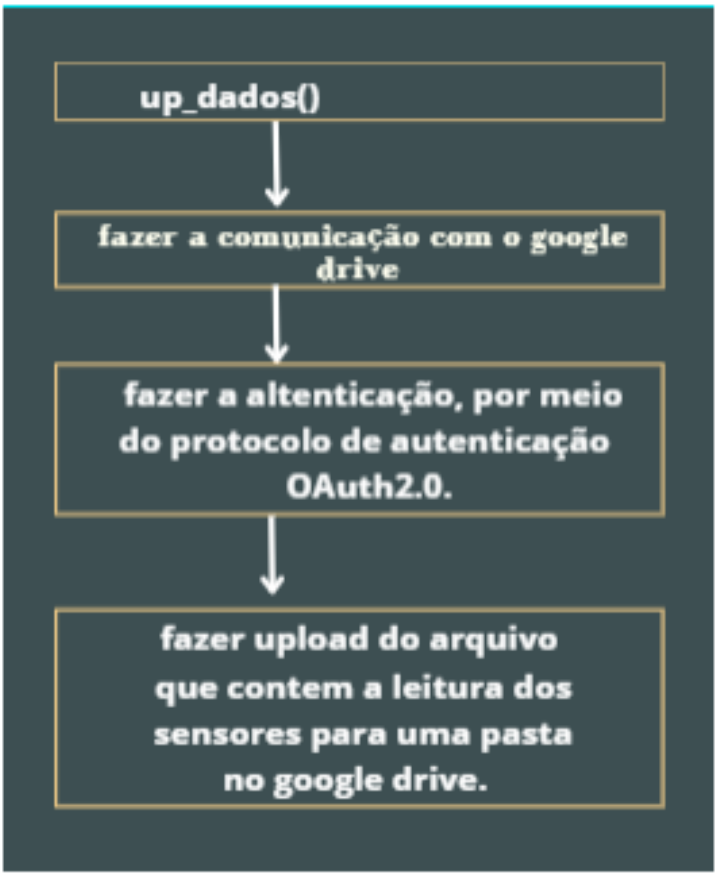


Figura 54 – Algoritmo que Implementa o Upload dos Dados.

Fonte: Produzida pelo autor (2021)

5 Experimento

Com o intuito de validar o sistema de medição, foi montado um experimento com 5 módulos que funcionará como protótipo, para dois sistemas fotovoltaicos, um desses sistemas será organizado com módulos instalados em terra e o outro, com módulos instalados sobre uma lâmina de água. Esse Capítulo se dedica à descrição desse experimento

5.1 Descrição

O experimento foi feito com dois arranjos de painéis fotovoltaicos idênticos, sendo um arranjo com dois painéis e outro com três painéis. Foi feito um arranjo com dois painéis amparados sobre uma estrutura de alumínio sustentada por eps, ficando em estado de flutuação sobre uma lâmina d'água. Já o arranjo com três painéis foi configurado, mantendo-se dois deles amparados em uma estrutura de alumínio com eps, sobre uma lâmina d'água e um terceiro, fixado ao solo. A disposição e organização desses painéis é representada na figura 55:

Figura 55 – Arranjos de Módulos Fotovoltaicos.

Fonte: Produzida pelo autor (2021)

Na figura 55, os painéis foram numerados para facilitar a compreensão do experimento e de seus resultados. Os módulos representados pelos módulos 4(FV4) e 5 (FV5) aparecem em série e os painéis representados pelos módulos 1, 2 e 3 foram instalados também em série em outro controlador de carga.

O módulo 1 (FV1) é o painel que possui um trocador de calor, como demonstrado na figura 56. Destaca-se que para ocorrer a troca de calor, tem que haver uma circulação forçada de água, na serpentina do trocador de calor. No módulo representado na Figura 56, foram instalados dois sensores DS18B20, com a finalidade de verificar a entrada e a saída da água e enviar calor, para Raspberry, através de um protocolo *One-Wire*.

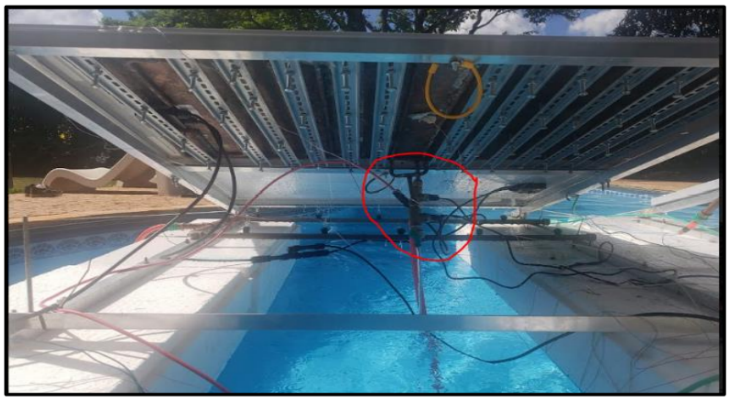


Figura 56 – Instalação dos sensores ds18b20 no módulo do trocador de calor.

Fonte: Produzida pelo autor (2021)

O módulo 2 (FV2), apresentado na figura 57 possui um sistema de aspersão que realiza a refrigeração forçada do backsheet do módulo. Já módulo 3 (FV3) não possui nenhum sistema de resfriamento, é simplesmente um painel instalado sobre o solo.

Figura 57 – representações do sistema de aspersão.

Fonte: Produzida pelo autor (2021)

O módulo 4 (FV4) não possui nenhum sistema de refrigeração forçada, entretanto, foi instalado sobre uma lâmina d'água, ficando responsável pela refrigeração passiva. O módulo 5 (FV5), representado pela figura 58 possui uma fina lâmina de água que passa por sua superfície, servindo para evitar a acumulação de poeira sobre o painel e para promover o resfriamento da superfície.

5.2 Mapas de Termopares

Com o intuito de monitorar a temperatura dos painéis foram dispostos 15 termopares, devidamente numerados, de 1 a 15 e distribuídos entre os 5 painéis. A figura 59 ilustra essas distribuições, da seguinte forma:

Figura 58 – Representações do Módulo 5(FV5).

Fonte: Produzida pelo autor (2021)

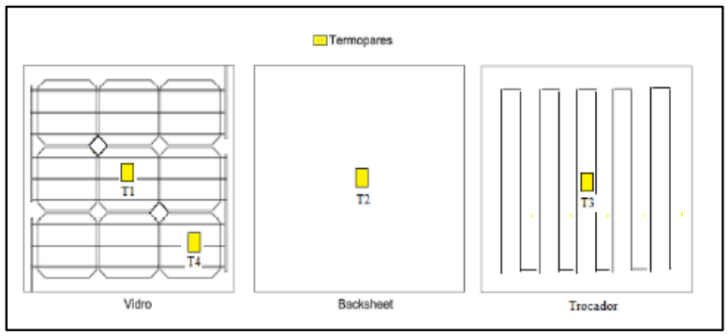


Figura 59 – Distribuições Dos Termopares No Painel Com Um Trocador De Calor FV1.

Fonte: FERREIRA (2021)

O painel com trocador de calor FV1 é representado pela figura 59. A figura mostra que quatro termopares numerados foram separados e posicionados, dois sobre uma superfície de vidro, denominados T1 e T4 e dois, no trocador de calor, o T2 no backsheet e o T3 no trocador de calor. A figura 60 mostra a localização e a posição dos termopares dos outros quatro módulos Fotovoltaicos.

Na figura 5.6 o módulo FV2 é representado pelo primeiro painel localizado a canto superior esquerdo, ele possui 4 termopares numerados de T5 a T8, estando os termopares T5 e T7 localizados na superfície superior do painel e outros dois localizados no *backsheet.*

O módulo FV3, O qual foi instalado em solo, é representado pelo módulo que contém os termopares enumerado T12 e T13 na figura 5.6, O termopar T12 está localizado no backsheet do painel, enquanto o termopar T13 está na superfície do painel.

O módulo FV4, foram instalados 3 termopares de T9 a T11, sendo os termopares T9 e T10 localizado na parte inferior do módulo, já o termopar T11 está localizado na superficie do painel superior.

Por fim o módulo FV5, é o módulo da figura 5.6 que foram instalados dois termopares T14 e T15, sendo ambos instalados na superfície superior do painel, sendo o T14 localizado

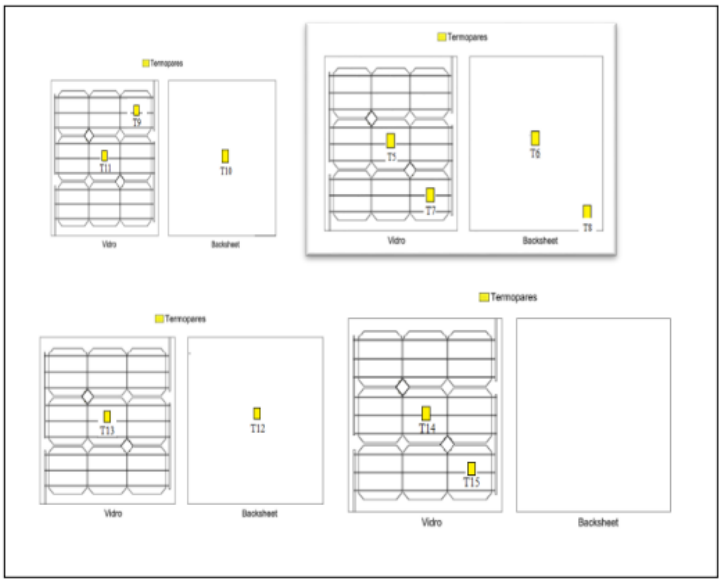


Figura 60 – Representações da Localização dos Termopares nos Módulos Fotovoltaicos FV2, FV3, FV4 e FV5.

Fonte: FERREIRA (2021)

no centro da superfície e T15 no canto inferior direito da superfície superior.

5.3 Montagem Elétrica

Ambos arranjos de painéis alimentam dois quadros de distribuição, os quais estão conectados a um banco de baterias, composto por 3 baterias de 50Ah, duas delas da marca Heliar e a outra da Bosch.

Foi feita uma montagem padrão para o quadro de distribuição,composta por 2 dispositivos de proteção de surto, que foram devidamente aterrados, 2 disjuntores bipolares e 1 disjuntor unipolar, um controlador de carga. No projeto foram utilizados dois quadros, um com um fieldLogger e um módulo de aquisição de termopares, o outro continha Raspberry pi e o sense Hat. O diagrama unifilar dos quadros se encontra em anexo. Nota-se pela figura 61 que os componentes do quadro elétrico estão protegidos por caixa hermética de 50x60 com proteção contra raios.

Figura 61 – layouts padrão do quadro de força.

Fonte: Produzida pelo autor (2021)

6 ANÁLISE DE DADOS E RESULTADOS

Este capítulo traz uma síntese dos resultados alcançados, com a experimentação usando os protocolos Modbus RTU, One-Wire e I2C, na coleta de dados de refrigeração forçada e passiva, com cinco painéis fotovoltaicos, instalados em terra e em lâmina de água que, com ajuda de termopares, tiveram a função de verificar aspectos de irradiação solar, umidade, north, pressão atmosférica e aferir a temperatura sobre a superfície dos módulos e no *backsheet*.

No decorrer do projeto, houve atraso na entrega de alguns materiais que não foram integrados à rede, impedindo que os objetivos fossem totalmente alcançados. Diante disso, não foi possível obter dados relacionados à velocidade do vento, dos sensores ambientais, do sensor de concentração de poeira e dos piranômetros. Não obstante, foram observados dados referentes à temperatura e irradiação solar, destacando-se a diferença entre as temperaturas dos sistemas refrigerados.

Observou-se, em relação às medidas de temperatura e irradiação solar, que os módulos de aquisição usados neste estudo, foram capazes de aferir medidas, com precisão, consideravelmente alta e de mostrar que as temperaturas dos sistemas refrigerados apresentaram níveis menores que as dos sistemas não refrigerados.

O FieldLogger e o módulo de aquisição de Termopares AM8T possuem, cada um, oito canais de aquisição, sendo responsáveis pela leitura das temperaturas. O FieldLogger também é usado, para leitura da irradiação solar. Em razão dessas características, neste estudo, foram usados sete canais do FieldLogger, para leitura de temperatura e um canal, para leitura do sensor de irradiação solar Li-Cor LI-200R. Após a coleta, os dados foram enviados, através do modbus RTU, para o mestre da rede (Raspberry PI) que comporta até 1300 medições, para ser condensado em um arquivo CSV e armazenado, na nuvem.

O Raspberry PI recebeu os dados do FieldLogger, do AM8T, do Sense Hat e do sensor de temperatura DS18B20. Os dados do Sense Hat foram enviados pelo protocolo I2C. Já os dados do DS18B20, responsável por medir a temperatura de entrada e saída da água do trocador, foram enviados pelo protocolo One-Wire. A forma de instalação do sensor de temperatura foi ilustrada pela Figura 62:

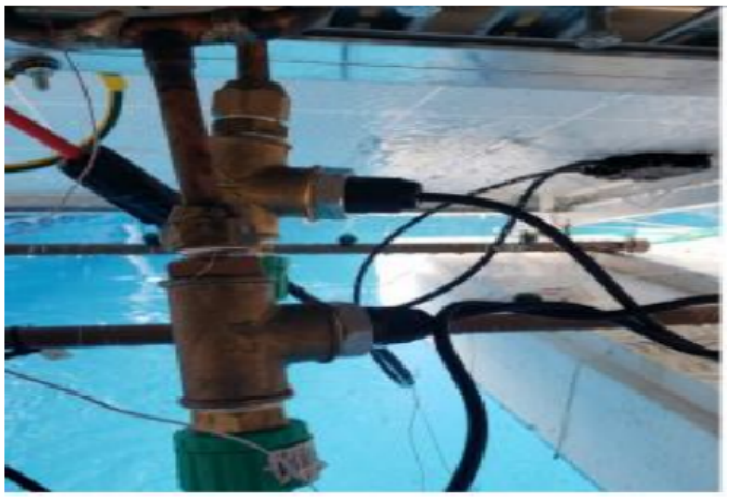


Figura 62 – Sensor de Temperatura da Água - Trocador de Calor.

Fonte: Produzida pelo autor (2021)

6.1 Armazenamento dos dados

Os dados referentes a este projeto estão salvos, em pastas no Google Drive, sendo armazenados, de acordo com as datas de coleta, que começaram a ser registradas em 28 de outubro e concluídas em 01 de novembro de 2021. Obtidas a leitura de 1300 medições, o Raspberry enviou os dados e os armazenou, na nuvem, conforme demonstrado na Figura 63.

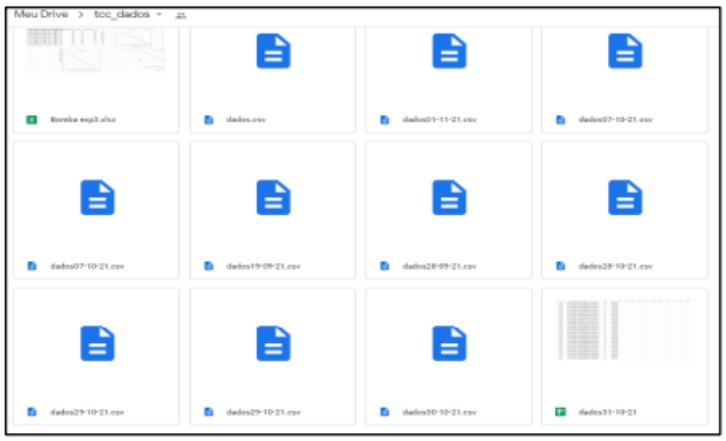


Figura 63 – Pastas de Dados do Raspberry.

Fonte: novus (2021)

6.2 Dados

Os dados obtidos, através do experimento, foram condensados em gráficos comparativos, agrupados de acordo com o assunto e com os instrumentos de medição. A figura 64 ilustra a aferição da temperatura do ar, pelo Raspberry PI Sense Hat e sua aplicação em 5 dias consecutivos, considerando o tempo de realização e as mudanças ocorridas.

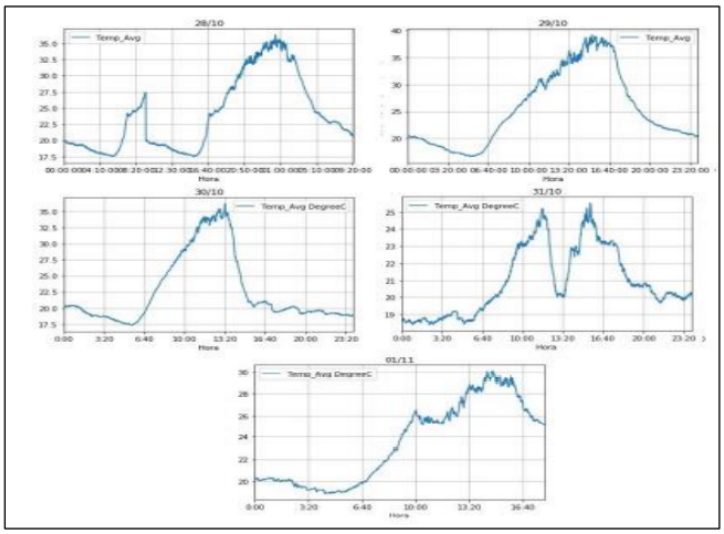


Figura 64 – Temperatura do Ar ao Longo do Experimento, Sense Hat.

Fonte: FERREIRA (2021)

A Figura 65 apresenta a leitura da irradiação solar, observada pelo piranômetro li200r, durante o período em que o experimento se realizava, verificando-se alterações importantes, durante essa leitura, registrando-se níveis altos e baixos, em um mesmo dia, demonstrando o potencial de precisão do instrumento de medição:

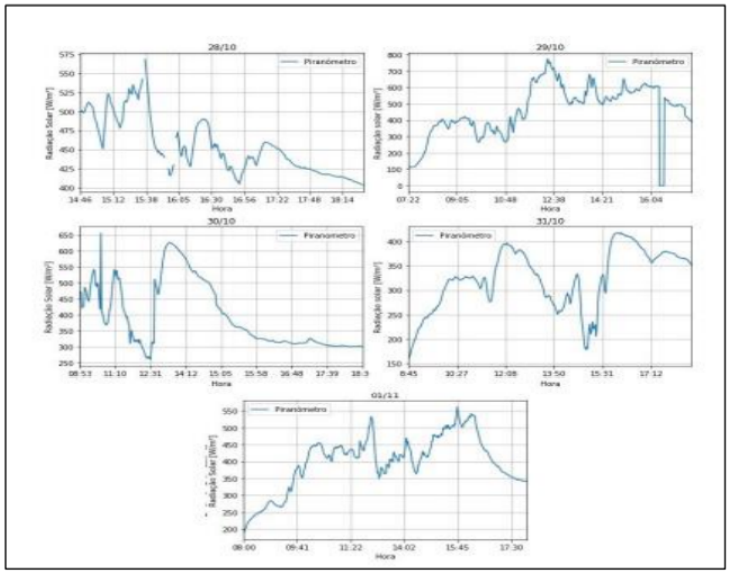


Figura 65 – Radiação solar medida durante os experimentos..

Fonte: FERREIRA (2021)

A variação de temperatura aferida pelo FieldLogger e pelos termopares dos quatro módulos postos sobre lâmina de água, é apresentada em uma sequência, tomando como base, as representações de Boas e Ferreira (2021), através das quais, torna-se possível evidenciar as diferenças ocorridas, em período curto de observação, conferindo aos instrumentos

de aferição de medidas, sensibilidade e confiabilidade. A Figura 66 mostra a variação de temperatura nos módulos 1, 3 e 4, ao longo do tempo:

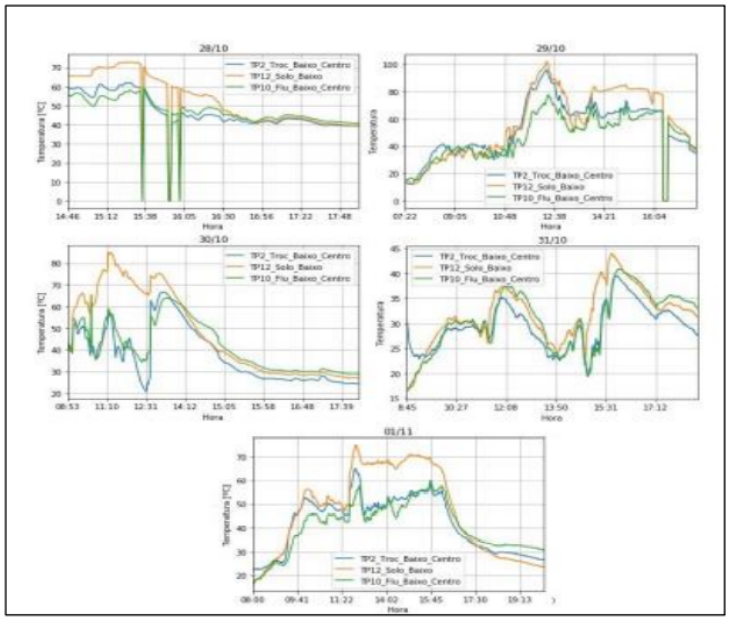


Figura 66 – Variação da temperatura nos termopares da superfície superior dos módulos FV 1, 3 e 4.

Fonte: FERREIRA (2021)

Seguindo a mesma dinâmica, a Figura 67 apresenta, de forma comparativa, a variação da temperatura dos termopares do módulo FV 2, ao longo dos cinco dias que durou o período de observação:

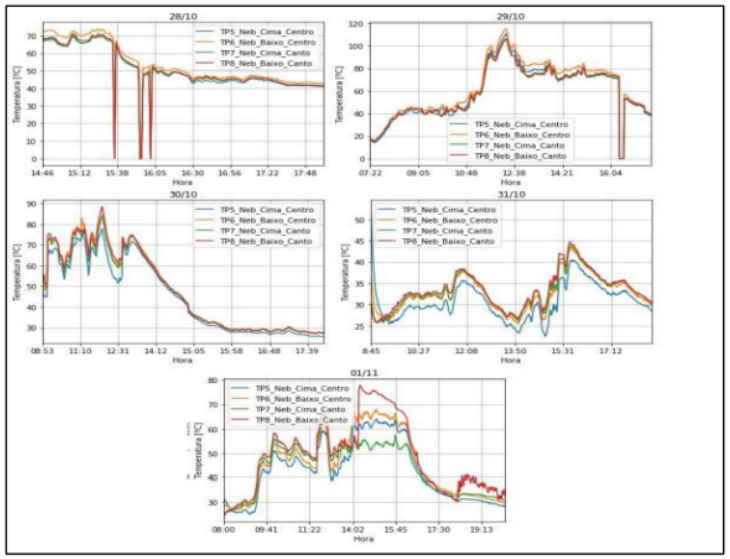


Figura 67 – Variação da temperatura dos termopares do módulo FV 2.

Fonte: FERREIRA (2021)

A variação da temperatura do módulo FV4, no percurso da observação, foi descrita pela Figura 68, como visto, logo a seguir:

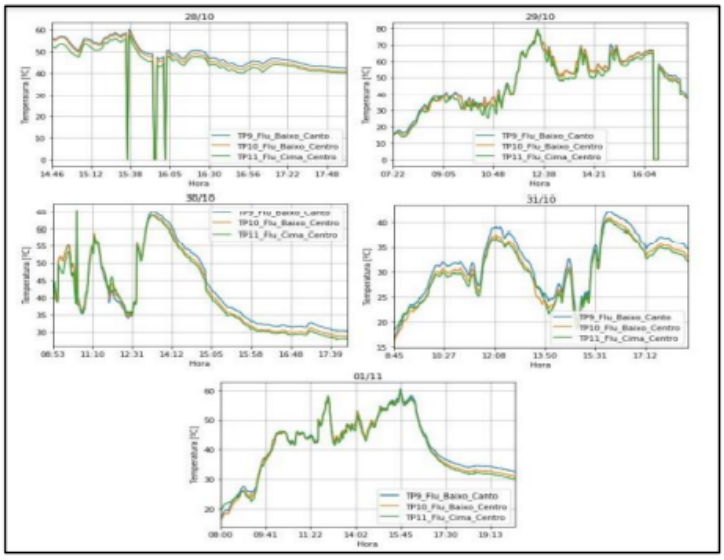


Figura 68 – Variação da temperatura dos termopares do módulo FV 4.

Fonte: FERREIRA (2021)

Por último, a Figura 69 ilustra a variação da temperatura verificada nos termopares do módulo FV3, durante o período de observação e de duração do experimento:

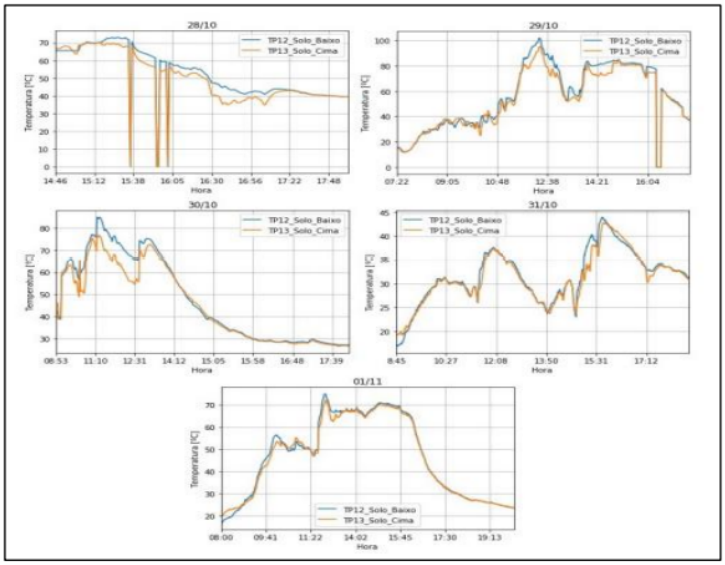


Figura 69 – Variação da temperatura dos termopares do módulo FV 3.

Fonte: FERREIRA (2021)

A falta de materiais adequados impossibilitou a coleta de dados automática, em relação às medidas de tensão, com isso, foi preciso realizar uma aferição de valores de 10 em

10 minutos, a contar do dia 28 de outubro ao dia 01 de novembro. Para acompanhar essa leitura, foi organizada a Tabela 9.12, onde são apresentadas médias de cada módulo:

Tabela 7 – Especificações do RK900-09.

Dia	**Tensão Solo (FV3)**	**Tensão Flutuante (FV4)**	**Tensão Aspesor (FV2)**	**Tensão Lâmina (FV5)**	**Tensão Trocador (FV1)**
28-10	7.15	16.36	9.73	9.42	15.71
29-10	16.26	17.82	17.83	17.21	18.27
30-10	16.94	17.37	18.89	18.70	18.80
01-11	17.09	16.37	18.5	14.80	18.45

Fonte: Produzida pelo autor.

A comparação das medidas médias de tensão, aferidas manualmente, indica que houve uma diferença considerável entre os painéis dispostos no solo e os demais. A geração de tensão desses painéis foi menor, em relação aos outros painéis. Dentre os painéis refrigerados, o painel com os aspersores se mostrou o mais promissor, ainda que ele apresentasse valores abaixo do painel com o trocador e com a lâmina de água. Esses valores podem ser explicados ou podem ter sido provocados pelo sombreamento da vegetação local.

7 CONCLUSÃO

Este projeto objetivou o desenvolvimento de experiência, usando protocolos Modbus RTU, One-Wire E I2C, para coletar dados de refrigeração forçada e passiva, analisando e comparando as temperaturas, a tensão e a corrente de cinco painéis fotovoltaicos, sendo quatro instalados sob lâmina de água e um, em terra, além de aferir a irradiação solar, medir a umidade, north e a pressão atmosférica.

A tarefa de leitura das variáveis foi possibilitada graças aos módulos de aquisição fieldlogger, Am8t e sense hat, e aos sensores termopar tipo T, ds18b20 e piranometro Li-Cor LI-200R. Não somente esses, mas também o Rasbperry pi com os módulos python minimalmodbus e pyserial possibilitaram a comunicação com a rede Modbus.

Todas as experiências resultaram na produção de imagens, tabelas e gráficos, para permitir melhor visualização do trabalho que, mesmo tendo sofrido interferências do tempo e da falta de material adequado, não foi interrompido, mas redirecionado, para ser concluído, uma vez que foram realizadas as leituras das temperaturas, aferindo medidas e variáveis, através das quais foi possível concluir que o sistema de maior geração foi o de painéis, com aspersores, no *backsheet*.

As representações e conclusões deste projeto ensejam uma continuidade, a fim de que o primeiro objetivo seja alcançado, mediante a aplicação de todos os recursos e materiais requeridos para a pesquisa. Com isso, seria possível propor, futuramente, a construção de um sistema em Python, para tratamento dos dados observados, bem como de um gráfico, em tempo real, para cada uma das variáveis. Contribuem para esse projeto futuro, o fornecimento dos códigos gerados para este projeto que possibilitou e facilitou a comunicação de rede Modbus com outros sensores que conversam Modbus RTU.

Referências

ALENCAR FILHO, A. A. R. de; CARVALHO, P. C.; DUPONT, I. M. INFLUÊNCIA DA DISTÂNCIA DE PAINÉIS FOTOVOLTAICOS EM RELAÇÃO À ÁGUA SOBRE A EFICIÊNCIA ENERGÉTICA. In: VII Congresso Brasileiro de Energia Solar-CBENS 2018. 2018. Citado na p. 15.

ALMEIDA OLIVATI, C. de. **Efeito fotovoltaico e fotocondutividade em dispositivos poliméricos**. 2000. Dissertação de Mestrado – Universidade de São paulo. Citado na p. 18.

ANDRÉ RIYUITI HIRAKAWA, P. S. C. **Experiência 4: implementação de um cronômetro**. Nov. 2014. disponivel em <http://sites.poli.usp.br/d/pcs2529/index$_arquivos$/2529e062014.p $acessoemAcesso$: 8mai.2021. Citado na p. 20.

BERG, J. **MinimalModbus Documentation Release 2.0.1**. Ago. 2021. Disponível em: <https://minimalmodbus.readthedocs.io/>. Citado nas pp. 30, 31.

BRAGA, R. P. **ENERGIA SOLAR FOTOVOLTAICA: FUNDAMENTOS E APLICAÇÕES**. Trabalho de conclusão do de curso. http://www.repositorio.poli.ufrj.br/monografias/monopoli100 nov. 2008. acessado em 15 mai. 2021. Citado na p. 17.

BRITO, M. C.; SILVA, J. A. Energia fotovoltaica: conversão de energia solar em electricidade. **O instalador**, v. 25, n. 676, p. 07, 2006. Citado nas pp. 17, 18.

CHOI, Y.-K. A study on power generation analysis of floating PV system considering environmental impact. **International journal of software engineering and its applications**, Science e Engineering Research Support Society, v. 8, n. 1, p. 75–84, 2014. Citado na p. 15.

COLLINS, G. **Pymodbus Documentation**. June, 2013. Disponível em: <https://pyserial.readthedocs.io/en/latest/pyserial.html>. Acesso em: 5 mai. 2021. Citado na p. 30.

CRESEB. **Tutorial de Energia Solar Fotovoltaica**. Set. 2008. Centro de Referência das Energias Solar e Eólica. Disponível em: <http://www.cresesb.cepel.br/index.php?section=com_content%5C&lang=pt%5C&catid=4>. Acesso em: 3 jul. 2011. Citado nas pp. 18, 19.

DALLAS. **DS18B20 Programmable Resolution 1-Wire® Digital Thermometer**. Jun. indefinida. Disponível em: <https://www.robocore.net/sensor-ambiente/sensor-de-temperatura-ds18b20-a-prova-de-agua?gclid=Cj0KCQjw4PKTBhD8ARIsAHChzRLN1ZbNMtYc_Ug-R4z7UIh73oOFJms6NC29KxBJ7vC5q3HQn_dMOY8aAiuiEALw_wcB>. Citado nas pp. 43, 44.

EPEVER. **Manual do Usuário. Série TRIRON N. Controlador Modular de Carga Solar MPPT.** indefinida. Disponível em: <https://www.epever.com/>. Citado nas pp. 41, 42.

FERREIRA, B. S. V. B. I. A. **ESTUDO EXPERIMENTAL DE MÓDULOS FOTOVOLTAICOS REFRIGERADOS EM AMBIENTE AQUÁTICO**. Nov. 2021. Citado nas pp. 61, 62, 66–68.

FREITAS, C. M. **Redes de comunicação em RS-485**. https://www.embarcados.com.br/redes-de-comunicacao-em-rs-485/, nov. 2014. acessado em: mai. 2021. Citado nas pp. 20–22, 25, 27, 28.

INSTRUMENTS, N. **The Modbus Protocol In-Depth**. https://www.ni.com/, mai. 2019. Disponível em: <https://www.ni.com/>. Acesso em: 2021. Citado nas pp. 24, 26, 27.

INSTRUMENTS, T. **1-Wire Enumeration**. https://www.ti.com/, jan. 2018. Disponível em: <https://www.ti.com/>. Citado nas pp. 28, 29.

KUGELSTADT, T. **The RS-485 Design Guide**. Mai. 2011. acessado em: mai. 2021. Disponível em: <https://www.ti.com/>. Acesso em: 3 mai. 2021. Citado na p. 23.

KUROSE, J. F.; ROSS, K. W. **Redes de Computadores, Uma Abordagem Top-Down**. Pearson, 2013. Citado na p. 24.

LIECHTI, C. PySerial documentation. **Versión: 2.6, MAI 2021**, 2021. Disponível em: <https://pyserial.readthedocs.io/>. Acesso em: 5 mai. 2021. Citado nas pp. 29, 30.

MECAWEB. **Porta Serial**. http://www.mecaweb.com.br/eletronica/content/e$_s$*erial*, set. 2019. acessado em 21 jun. 2021 . Citado nas pp. Citado na p. 20.

MODBUS-IDA, M. A. P. S. V. 1.1 b. **Hopkinton, Massachusetts (www. modbus. org/-docs/Modbus Application Proto col V1 1b. pdf)**, 2006. Citado nas pp. 24, 26.

NABEL, R. **Documentação da Biblioteca Wrapper do Google-api-phyton-client. Documentação oficial nas páginas do GitHub.** https://pythonhosted.org/PyDrive/, out. 2016. Disponível em: <https://pythonhosted.org/PyDrive/>. Citado na p. 32.

NOVUS. **Conceitos Básicos de Conceitos Básicos de RS485 e RS422**. Jan. 2013. acessado em: mai. 2021. Disponível em: <https://www.novus.com.br/downloads/Arquivos/conceitos>. Acesso em: 3 jul. 2011. Citado nas pp. 21, 23.

NOVUS. **Manual de Operação FieldLogger V1.7x A**. https://www.novus.com.br/, ago. 2021. Disponível em: <https://www.novus.com.br/>. Citado nas pp. 33–35, 48–53, 65.

ÔMEGA. **Saiba o que é um Termopar.** indefida. Disponível em: <https://br.omega.com/>. Citado nas pp. 44, 45.

PINHO, J.; GALDINO, M. Engineering manual for photovoltaic systems retrieved from Rio de Janeiro: CEPEL—CRESESB. **Manual de engenharia para sistemas fotovoltaicos**, 2014. Citado nas pp. 14, 17, 19.

RASPBERRY. **DATASHEET: Raspberry Pi 4 Model B. Reliase 1.** Jun. 2019. Disponível em: <https://www.raspberrypi.org/>. Acesso em: 5 mai. 2021. Citado nas pp. 42, 43.

RK200-03. **RK200-03 Pyranometer**. https://www.rikasensor.com/rk200-03-pyranometer-radiation-monitor.html, indefinida. Disponível em: <https://www.rikasensor.com/rk200-03-pyranometer-radiation-monitor.html>. Citado nas pp. 36, 37.

RK300-02. **RK300-02 Dust Concentration Sensor**. https://www.rikasensor.com/rk300-02-dust-sensor.html, indefinida. Disponível em: <https://www.rikasensor.com/rk300-02-dust-sensor.html>. Citado nas pp. 35, 36.

RK900-09. **RK900-09 Miniature Ultrasonic Weather Station**. https://www.rikasensor.com/rk900-09-miniature-ultrasonic-weather-station.html, indefinida. Disponível em: <https://www.rikasensor.com/rk900-09-miniature-ultrasonic-weather-station.html>. Citado nas pp. 38, 39.

SEMICONDUCTORS, N.; EINDHOVEN, N. I2C-bus specification and user manual, 2021. **URL<https://www.nxp.com/docs/en/user-guide/UM10204.pdf**, 2021. Citado na p. 28.

TECNOLOG. **AM8T Módulo de termopares Manual do Usuário. 2020.** 2020. Disponível em: <https://www.rikasensor.com/rk900-09-miniature-ultrasonic-weather-station.html>. Citado na p. 40.

Apêndices

APÊNDICE A – Códigos de Programação

A.1 Código que Faz a Comunicação do Raspberry Pi com o Google Drive

Código A.1 – Código de Python

```
#import ler_csv1
#import csv
import time
from datetime import datetime as date
from  pydrive.auth  import  GoogleAuth
from pydrive.drive import GoogleDrive
import csv
#1OBo-HBRI4LyHK16VOaB7sY4woIwOf4ys
def up_dados():
  hora=date.now()
  c=hora.strftime('%d/%m/%y')
# c='01/11/21'
  c= c.replace('/','-')
  name="dados"+c+".csv"
  gauth=GoogleAuth()
  gauth.LocalWebserverAuth ()  # Cria um servidor da web local e
     gerencia a autenticação automática
  drive = GoogleDrive(gauth)
  #file1 = drive.CreateFile({'title': 'dados.csv'})  # Create
     GoogleDriveFile instance with title 'Hello.txt'.
  file1 = drive.CreateFile({'parents': [{'id':
     '1OBo-HBRI4LyHK16VOaB7sY4woIwOf4ys'}],'title':
     name})#'dados17-09-21.csv'
  file1.SetContentFile(name)#'dados17-09-21.csv'
  #file1.SetContentString('olá sou allan') # Set content of the
     file from given string.'''
  file1.Upload()
  return True
```

A.2 Código que faz a comuicação com a Rede Modbus

Código A.2 – Código de Python

```
import serial
import minimalmodbus
```

```
#import matplotlibpyplot as plt
import csv
# try:

#   #primeiro tem que anotar o id de cada componente.
#'/dev/ttyUSB1'
def dados_modbus(d,names):
  instrument = minimalmodbus.Instrument('/dev/ttyUSB0',d, debug
     =True)
  instrument.serial.port                      # this is the serial
     port name
  instrument.serial.baudrate = 9600        # Baud
  instrument.serial.bytesize = 8
  instrument.serial.parity   = serial.PARITY_NONE
  instrument.serial.stopbits = 1
  instrument.serial.timeout  = 0.5          # seconds

  instrument.address                         # this is the slave
     address number
  instrument.mode = minimalmodbus.MODE_RTU   # rtu or ascii mode
  instrument.clear_buffers_before_each_transaction = True
  i=[1,2,3]
  a=instrument.read_registers(1,8,3)
  instrument.serial.close()
  c={}
  for i in range(0,len(a)):
    k={str(names[i]):float(a[i])}
    c.update(k)


  return c
```

A.3 Código que Faz a Comunicação com o Sensor DS18B20

Código A.3 – Código de Python

```
#!/usr/bin/python
#adaptado de
#https://www.raspberrypi-spy.co.uk/2013/03/raspberry-pi-1-wire-digital-therm
def gettemp(id):
  try:
    mytemp = ''
    filename = 'w1_slave'
    f = open('/sys/bus/w1/devices/' + id + '/' + filename, 'r')
    line = f.readline() # read 1st line
    crc = line.rsplit(' ',1)
    crc = crc[1].replace('\n', '')
    if crc=='YES':
```

```
13        line = f.readline() # read 2nd line
14        mytemp = line.rsplit('t=',1)
15      else:
16        mytemp = 99999
17      f.close()
18
19      return int(mytemp[1])
20
21    except:
22      return 99999
23
24 #if __name__ == '__main__':
25 def Tempagua(names):
26   # Script has been called directly
27   #28-00000b939f52, 28-0302977936c6
28   #28-00000b56fa73  28-011927687b1a  28-00000b939f52
        28-0302977936c6
29   T=[]
30   #id =
        ['28-00000b56fa73','28-011927687b1a','28-00000b939f52','28-0302977936c6
31   id = ['28-00000b939f52','28-00000b56fa73']
32   #28-00000b939f52
33   #28-100000000001  28-500000000001  28-900000000001
34   for i in id:
35     a=gettemp(i)/float(1000)#temperatura de entrada
36     T.append(a)
37   #a=gettemp(id[1])/float(1000)#temperatura de saída
38   #T.append(a)
39   #print ("Temp : " + '{:.3f}'.format(gettemp(id[0])/float(1000)))
40   #print ("Temp : " + '{:.3f}'.format(gettemp(id[1])/float(1000)))
41
42   k={}
43   b={names[0]:T[0]}
44   k.update(b)
45   b={names[1]:T[1]}
46   k.update(b)
47
48
49
50
51   return k
52 names=['TA1','TA2']
53 a=Tempagua(names)
54 print(a)
55 #Temp=Tempagua()
56 #print(Temp)
```

A.4 Código que Controla o Fluxo de Execução das Ações do Sistema

Código A.4 – Código de Python

```
#importação da bibliotecas
import time
import modmestre
from datetime import datetime as date
import sense_sesor as S
import sensor_ds18b20 as T
import envia_email as EM
import envia_dados as D
import modmestretwo as M

minutes=0
seconds=0
b=False


hoje='2021/10/29'

def execucao():
  #sense hdat

  flagemail=False
  flagemailA=False
  fieldNames,dados=S.ler_sensors()
  #a=datetime.today().strftime('%Y/%m/%d')


  #leitura de dados da rede modbus
  #=================================================================================
  try:

    Fnames_sesor=['FPAR15','FPAR14','FPAR13','FPAR12','FPAR11','FPAR10','FPA
    c=modmestre.dados_modbus(2,Fnames_sesor)#leitura de dados do
       fieldlogger
    dados.update(c)
    for i in Fnames_sesor:
      fieldNames.append(i)
  except:
    if flagemail==False:
      print("não foi possivel fazer a comunicação modbus")
      a=EM.envia_lista(1)
      flagemail=True

  try:

    Anames_sesor=['APAR1','APAR2','APAR3','APAR4','APAR5','APAR6','APAR7','A
       dos sensores do am8t
```

```
45      d=modmestre.dados_modbus(3,Anames_sesor)#leitura de dados do
            am8t
46      dados.update(d)
47      for i in Anames_sesor:
48        fieldNames.append(i)
49    except:
50      if flagemailA==False:
51        print("não foi possivel fazer a comunicação modbus")
52        a=EM.envia_lista(2)
53        flagemail1=True
54  #leitura dos medidores de tensão
55    try:
56
57      T_sesor=['TenA','tenTr','Tsolo','lamina','flutuante','cc3','cc2','APAR8'
            dos sensores do am8t
58      d=M.dados_modbus(4,T_sesor)#leitura de dados do am8t
59      dados.update(d)
60      for i in T_sesor:
61        fieldNames.append(i)
62    except:
63      if flagemailA==False:
64        print("não foi possivel fazer a comunicação modbus")
65        a=EM.envia_lista(2)
66        flagemail1=True
67    #leitura da temperatura da agua via sensores ds18b20
68    #=============================================================================
69    #Tagua=[]
70    try:
71      Tnames=['TA1','TA2']
72      Tagua=T.Tempagua(Tnames)
73      print("sensores de temperatura")#Tagua[0] temperatura de
            entrada da agua, Tagua[1] é a temperatrua de sída
74      for i in Tnames:
75        fieldNames.append(i)
76      dados.update(Tagua)
77    except:
78      a=EM.envia_lista(3)
79      print("algo deu errado com os sensores de temperatura")
80    print(dados)
81    print(fieldNames)
82    b=S.grava_dados(fieldNames,dados)
83    print(b)
84    return True
85
86  def executar_tcc():
87    #contar de 20 em vinte minuto para fazer uma coleta
88    b=execucao()
89    return True
90  flag_envio=False
91  print("primeira flag de envio\n")
92  print(flag_envio)
93  q=0#quantidades de leitura
```

```
94
95
96
97
98  while True:
99    executar_tcc()
100   hora=date.now()
101   H=hora.strftime('%H')
102   M=hora.strftime('%M')
103   print(H+":"+M)
104   print("printando flag de envio\n")
105
106   q+=1
107   if q >= 1300 and flag_envio==False:
108     try:
109       flag_envio=D.up_dados()
110     except:
111       print("erro de envio");
112   print("/n")
113   print(flag_envio)
114   print("quantidade de leituras")
115   print(q)
116   D=hora.strftime('%Y/%m/%d')
117   if D!=hoje:
118     hoje=D
119     q=0
120     flag_envio=False
121   time.sleep(60)
122 '''
123   print(a)
124   print("dados do sense hat\n")
125   print()
126   print("\n")
127   try:
128     print("dados da rede modbus do fieldlogger\n")
129     print(c)
130     print("dados da rede modbus do am8t\n")
131     print(d)
132   except:
133     a=EM.envia_lista()
134     print("não é possivel mostrar a comunicação modbus")
135     print("erro na rede modbus")
136
137   '''
```

A.5 Código que faz a Comunicação com o Sense Hat

Código A.5 – Código de Python

```
1 from sense_hat import SenseHat
```

```
import time
import csv
import os
from datetime import datetime as date
sense = SenseHat()
#temp,humidade,precao,north

def grava_dados(fieldNames,dl):
  hora=date.now()
  c=hora.strftime('%d/%m/%y')
  c= c.replace('/','-')
  name="dados"+c+".csv"
  fileName=r"/home/pi/Desktop/projeto"+"/"+name
  print(name)
  print(fileName)

  #"/home/pi/Documents/dados.csv"
  if os.path.exists(fileName)==False:
    with open(name,'w',newline='') as csvfile:
      #fieldNames=["temperatura","humidade","pressao"]
      writer = csv.DictWriter(csvfile, fieldnames=fieldNames)
      writer.writeheader()
      writer.writerow(dl)
      csvfile.close()
  else:
    with open(name,'a',newline='') as csvfile:

      writer = csv.DictWriter(csvfile, fieldnames=fieldNames)
      #writer.writeheader()
      #writer.writerow({"temperatura":float(a[0]),"humidade":float(a[1]),"pr
      writer.writerow(dl)
      csvfile.close()
  return True
def amb_sensors():
  fieldNames=[]
  l={}
  d={}


  print('cabeçarios\n')
#print(fieldNames)
#==========ordem de leitura
  fieldNames.append('Temperatura')
  fieldNames.append('Humidade')
  fieldNames.append('precao')
  fieldNames.append('north')

# lendo a temperatura
  temp=sense.get_temperature()
  l={fieldNames[0]:float(temp)}
  d.update(l)

```

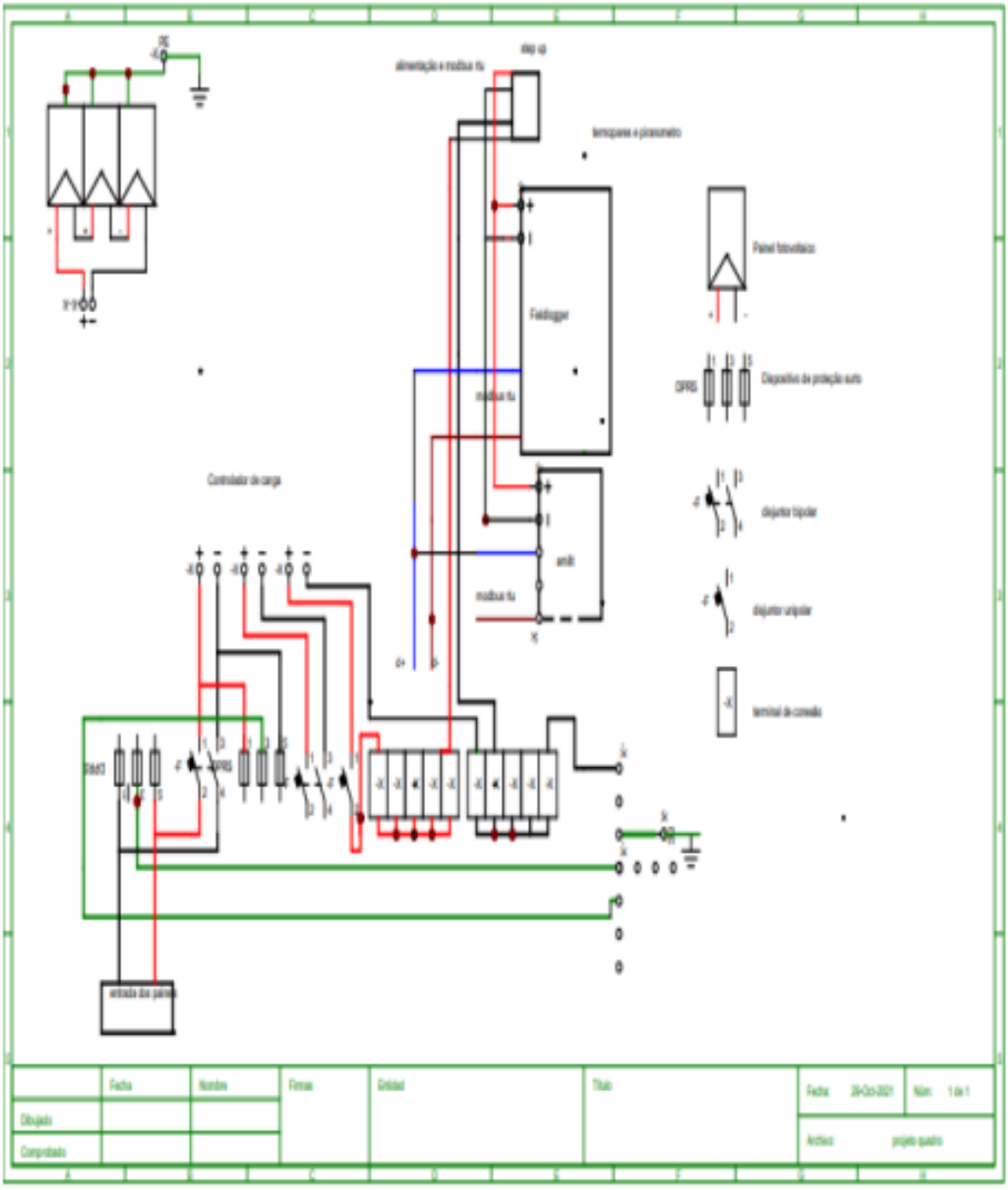
```
54
55 #   lendo humidade
56   humi=sense.get_humidity()
57   l={fieldNames[1]:float(humi)}
58   d.update(l)
59
60 # lendo pressão
61   precao= sense.get_pressure()
62   l={fieldNames[2]:float(precao)}
63   d.update(l)
64
65 # lendo o north magnético
66   north = sense.get_compass()
67   l={fieldNames[3]:float(north)}
68   d.update(l)
69
70
71   #a=[temp,humi,precao,north]
72   #fieldNames=["temperatura","humidade","pressao","north"]
73   #print(fieldNames)
74   #print(d)
75   return fieldNames,d
76 #lendo os agulos de sensores espaciais
77 def space_sensors():
78   ori=sense.get_orientation()
79   fieldNames=[]
80   fieldNames.append('roll')
81   fieldNames.append('pitch')
82   fieldNames.append('yaw')
83
84   return fieldNames,ori
85
86 def data_hora():
87   d={}
88   fieldNames=[]
89   hora=date.now()
90   b=hora.strftime('%H:%M')
91   c=hora.strftime('%d/%m/%y')
92   fieldNames.append('Data')
93   fieldNames.append('Hora')
94
95   l={fieldNames[0]:str(c),fieldNames[1]:str(b)}
96   d.update(l)
97   return fieldNames,d
98
99 def ler_sensors():
100   fieldNames=[]
101   d={}
102   #variaveis auxiliares
103
104   auxic,auxid=data_hora()
105   for i in auxic:
```

```
106       fieldNames.append(i)
107     d.update(auxid)
108 
109     auxic,auxid=amb_sensors()#lendo sensores ambientais
110     for i in auxic:
111       fieldNames.append(i)
112     d.update(auxid)
113 
114     auxic,auxid=space_sensors()# lendo sensores espaciais
115     for i in auxic:
116       fieldNames.append(i)
117     #fieldNames.append(auxic)
118     d.update(auxid)
119     return fieldNames,d
120 """
121 fieldNames,dl=ler_sensors()
122 #b=grava_dados(a,fieldNames,dl)
123 print(dl)
124 print(fieldNames)
125 print('dados')
126 #print(dl)
127 a=grava_dados(fieldNames,dl)
128 print(a)
129 #for i in range(0,10):
130 """
```

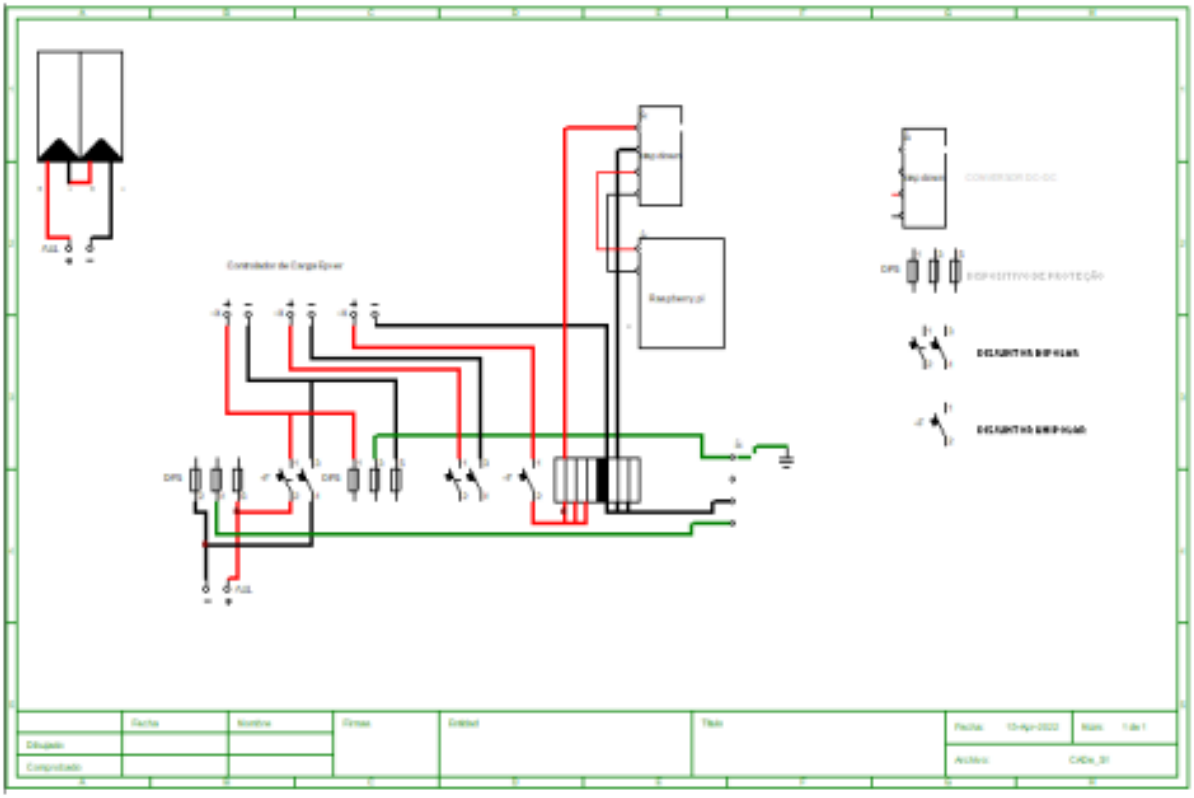
Anexos

ANEXO A – Diagramas Unifilares

A.1 Diagrama unifilar com o fieldlogger



A.2 Diagrama unifilar com o Raspberry pi


Raspberry pi
DISJUNTOR MINIATURA
DISJUNTOR MINIATURA