UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Faculdade de Engenharia e Ciências de Guaratinguetá

RODRIGO CARVALHO GUERRA

Sistema de monitoramento de estufa

Guaratinguetá
2023

Rodrigo Carvalho Guerra

Sistema de monitoramento de estufa

Trabalho de Graduação apresentado ao Conselho de Curso de Graduação Engenharia Elétrica da Faculdade de Engenharia e Ciências do Campus de Guaratinguetá, Universidade Estadual Paulista, como parte dos requisitos para obtenção do diploma de Graduação em Engenharia Elétrica.

Orientador: Prof. Dr. Leonardo Mesquita

Guaratinguetá
2023

Guerra, Rodrigo Carvalho
G934s Sistema de monitoramento de estufa / Rodrigo Carvalho Guerra - Guaratinguetá, 2024.
71 f : il.
Bibliografia: f. 54-55

Trabalho de Graduação em Engenharia Elétrica – Universidade Estadual Paulista, Faculdade de Engenharia e Ciências de Guaratinguetá, 2024.

Orientador: Prof. Dr. Leonardo Mesquita

1. Detectores. 2. Inovações tecnológicas. 3. Internet das coisas. 4. Estufas. I. Título.

CDU 681.586

Luciana Máximo
Bibliotecária/CRB-8 3595

unesp **UNIVERSIDADE ESTADUAL PAULISTA**
"JÚLIO DE MESQUITA FILHO"
Faculdade de Engenharia e Ciências de Guaratinguetá

RODRIGO CARVALHO GUERRA

ESTE TRABALHO DE GRADUAÇÃO FOI JULGADO ADEQUADO COMO PARTE DO REQUISITO PARA OBTENÇÃO DO DIPLOMA DE
"GRADUADO(A) EM ENGENHARIA ELÉTRICA"

APROVADO EM SUA FORMA FINAL PELO CONSELHO DE CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Prof. Dr. DANIEL JULIEN BARROS DA SILVA SAMPAIO
Coordenador

BANCA EXAMINADORA:

Prof. Dr. LEONARDO MESQUITA
Orientador(a)/UNESP-FEG

Prof. Dr. GALENO JOSÉ DE SENA
UNESP-FEG

Profa. Dra. RAPHAELA CARVALHO MACHADO
UNESP-FEG

Dezembro 2023

RESUMO

Este trabalho traz um estudo de dois protocolos de comunicação para possibilitar o monitoramento e o controle das variáveis edafoclimáticas, limitadas aqui a temperatura, umidade e iluminância no ambiente de uma estufa, 1-Wire e I²C para a aplicação de sensores de temperatura, umidade, iluminância e distância e o acionamento dos atuadores por meio da comparação dos valores adquiridos com os pré-estabelecidos no firmware. A partir da consulta a folhas de dados fornecidas pelos fabricantes dos sensores, artigos e livros foi possível o entendimento do funcionamento dessas tecnologias e de como as variáveis estudadas interferem no desenvolvimento de uma planta. Por fim, foram feitos dois testes para validar a eficiência dos sensores com diferentes valores alvos estabelecidos para o funcionamento dos atuadores, assim como a plotagem das variáveis e dos estados dos relés ao longo do tempo para a visualização. Concluiu-se que o sistema funcionou para o que se propôs, com uma certa limitação, dado o baixo custo dos atuadores.

PALAVRAS-CHAVE: Sensores; Internet das Coisas; Estufa; 1-Wire; I²C.

ABSTRACT

In order to monitor and control of edaphoclimatic variables, limited to temperature, humidity and luminosity in a greenhouse environment, this work presents a study of two communication protocols, 1-Wire and I²C for the application of temperature, humidity, luminosity and distance sensors and the activation of actuator by comparing the values acquired with those pre-established in the firmware. By consulting the datasheets provided by the sensor manufacturers, articles, and books, it was possible to understand how these technologies work and how the variables studied affect the development of a plant. Finally, two tests were carried out to validate the efficiency of the sensors with different target values establish for the operation of the actuators, as well as plotting the variables and states of the relays over time for visualization. It is concluded that the system worked as intended, with a certain limitation, given the low cost of the actuators.

KEYWORDS: Sensors; Internet of things; Greenhouse; 1-Wire; I²C.

LISTA DE FIGURAS

LISTA DE TABELAS

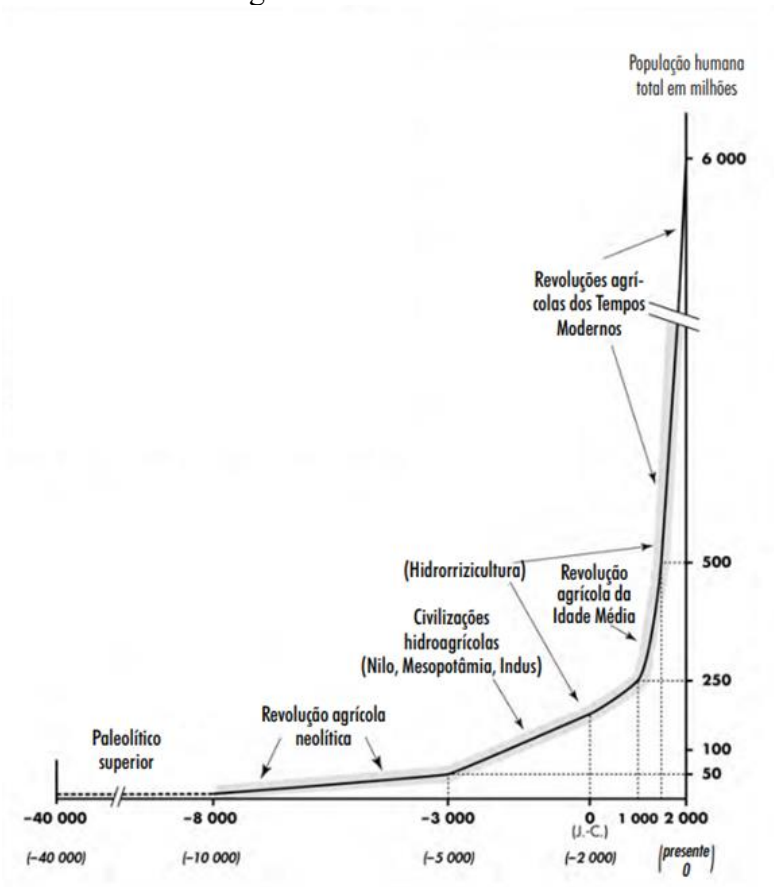
SUMÁRIO

1 INTRODUÇÃO

A Revolução Neolítica, também conhecida como Revolução Agrícola, foi uma das maiores transformações na história, porque permitiu que os seres humanos deixassem suas vidas nômades, ao dependerem da coleta e da caça, para uma vida sedentária, com a produção agrícola. Com a capacidade de produzirem seus próprios alimentos, foi possível o desenvolvimento de sociedades mais complexas, possibilitando, assim, a formação de civilizações.

Conforme Mazoyer e Roudart (1997), a agricultura permitiu que as populações crescessem, uma vez que a produção agrícola em larga escala pode sustentar mais pessoas, em relação à dependência da obtenção de alimentos provindos da coleta e da caça, que embora tenha se desenvolvido a fim de se tornar mais eficaz e diversificada, a exploração de algumas espécies encontrava um limite que as faziam regredir ou até mesmo se extinguirem de determinada localidade.

Figura 1 - A progressão da população humana em relação ao desenvolvimento dos sistemas agrários no mundo



Fonte: Adaptado de Mazoyer e Roudart (2010)

Na Figura 1, é possível observar os saltos do crescimento demográfico mundial com as evoluções agrícolas que ocorreram em diferentes partes do mundo ao longo da história.

Com o crescente aumento populacional e o desenvolvimento tecnológico na agricultura, percebeu-se que para o cultivo agrícola mais eficiente requer-se cuidados específicos e manutenção constante para o desenvolvimento sustentável. Cada espécie terá o seu desenvolvimento otimizado se forem colocadas sob algumas variáveis adequadas, tais como: luz solar, umidade, temperatura, nutrientes e espaço.

1.1 CULTIVO A CÉU ABERTO X CULTIVO PROTEGIDO

A agricultura praticada nos moldes tradicionais está sujeita à sazonalidade climática, fazendo com que somente alguns períodos do ano sejam favoráveis ao desenvolvimento das plantas, (BEZERRA, 2003).

Ao ficarem a céu aberto, estão expostas às ocasionalidades climáticas, como ventos e chuvas fortes, geadas e temperaturas extremas, à competição de espaço, água e nutrientes com plantas daninhas, aos ataques de pragas como insetos, ácaros, nematoides, fungos, bactérias, fitoplasmas, vírus, viroides e parasitas.

O cultivo protegido vem para mitigar alguns problemas ocasionados pelo cultivo a céu aberto, uma vez que protege contra condições climáticas adversas, tais como: temperaturas excessivamente baixas e umidade desfavorável, já que o controle dessas duas variáveis é possível em um ambiente controlado. Esse controle pode, inclusive, permitir o cultivo de espécies em épocas que não seriam possíveis se essas fossem cultivadas a céu aberto, podendo assim serem cultivadas ao longo do ano inteiro.

No entanto, o cultivo protegido não isenta a planta de adquirir doenças. A vulnerabilidade desse tipo se dá na maioria dos casos devido ao adensamento realizados nas estufas, criando condições favoráveis para o desenvolvimento de determinadas doenças, devendo então, haver um estudo prévio por profissionais qualificados para o correto dimensionamento do espaço, da terra e dos materiais usados na construção da estufa.

1.2 JUSTIFICATIVA

O clima do Brasil é propício para o cultivo de diferentes tipos de vegetais dado a sua característica tropical, no entanto há diversos gêneros que requerem uma maior estabilidade das variáveis climáticas, tais como a temperatura, a umidade relativa do ar, a radiação solar, a

nebulosidade, as chuvas ou precipitações pluviométricas, a pressão atmosférica e os ventos. Então visando isso é necessário a utilização de estufa para o cultivo de diferentes vegetais para que haja um rigoroso controle de qualidade, podendo assim, ter um crescimento adequado e conservar as propriedades desejadas.

A agricultura moderna enfrenta desafios crescentes, desde a necessidade de aumentar a produtividade até a urgência de práticas sustentáveis. Desse modo, o monitoramento de estufas com sensores surge como uma solução plausível, oferecendo benefícios que justificam sua adoção.

A utilização de sensores possibilita a automação de processos, reduzindo os custos operacionais, uma vez que não há a necessidade de intervenção manual constante, além de trazer práticas mais sustentáveis na agriculta, permitindo, por exemplo, a redução do uso de água e o ajuste ideal da intensidade luminosa.

O monitoramento dos dados coletados a longo prazo, em conjunto com a observação visual e estudos dos resultados, permitem o planejamento de cultivos mais eficientes, podendo ainda se adaptar às mudanças climáticas com a constante melhoria nos processos de produção.

1.3 OBJETIVO GERAL

O objetivo principal desse estudo é fazer a integração entre um microcontrolador, sensores e atuadores a fim de monitorar as variáveis de temperatura, umidade, e iluminância e a partir dos parâmetros configurados, fazer o acionamento dos atuadores para controlar essas variáveis.

1.4 OBJETIVOS ESPECÍFICOS

Os objetivos específicos desse trabalho são fazer os estudos dos protocolos utilizados pelos sensores e disponibilizar os dados coletados em gráficos em um ambiente integrado a Internet das Coisas.

2 REVISÃO BIBLIOGRÁFICA

Nesse capítulo será abordada a revisão da literatura no que diz respeito às características gerais para o desenvolvimento das plantas.

2.1 OLERICULTURA

A olericultura é o termo agronômico que designa o cultivo para consumo alimentar *in natura* das espécies de folhas, frutos, flores, caules ou raízes. (ANDRIOLO, 2002)

Quanto ao aspecto do cultivo, as olerícolas, comumente conhecidas no Brasil como hortaliças, distinguem-se entre si pelo tempo de cultivo, sendo a maioria de ciclo anual, algumas de ciclo bienal e poucas de cultivo perene e pelos cuidados intensivos demandados. Já quanto ao aspecto do consumo, a maioria se assemelha pela consistência macia e por curto período para serem consumidas frescas após a colheita, (MELO;ARAÚJO, 2016).

2.2 CONDIÇÕES PARA O DESENVOLVIMENTO DA PLANTA

Para um melhor desenvolvimento da espécie a ser cultivada, é necessária uma combinação de diversas características edafoclimáticas, no entanto, este estudo irá se limitar a analisar o papel da temperatura, umidade e iluminância para esse fim.

É complexo definir o que cada uma dessas variáveis causa, individualmente, no desenvolvimento das funções vitais das plantas, então será apresentada uma visão geral desses aspectos.

2.2.1 Temperatura

A temperatura é a grandeza física que mede o grau de agitação térmica de um sistema. Quanto maior for a agitação térmica, maior será a temperatura. Em outras palavras, é o grau de calor ou frieza de uma substância.

Segundo Nemali (2021), a faixa de temperatura ótima para a maioria das espécies de oleráceas está entre 20 e 25 °C. Espécies sensíveis a baixas temperaturas podem apresentar um crescimento lento e uma iniciação floral atrasada em estufas quando a temperatura ambiente está abaixo da ideal. Muitas espécies param completamente o seu crescimento se estiverem em temperaturas baixas, em torno de 2 a 12 °C, outras, no entanto, quanto expostas a temperaturas

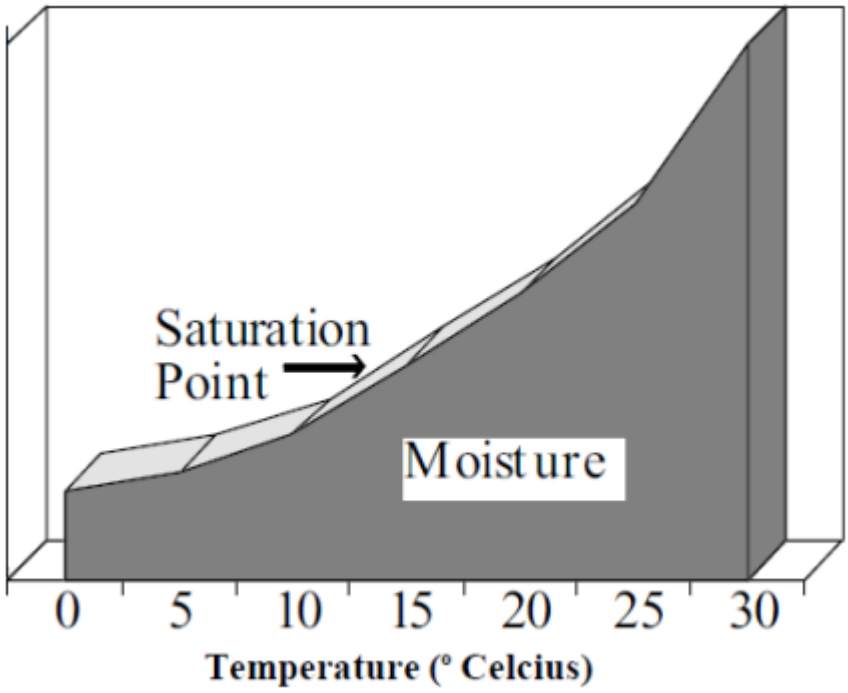
mais elevadas, entre 25 e 30 °C, podem apresentar crescimento exagerado. Danos por alta temperatura, como senescência foliar e queda de flores podem ocorrer se forem expostas a temperaturas superiores a 32 °C.

Para controlar a temperatura em cultivo protegido há diferentes métodos, tanto para abaixar, quanto para elevar a temperatura. Para o primeiro caso, há o uso de telas para o sombreamento, a construção da estufa de modo que permita uma boa ventilação natural e a ventilação forçada com o uso de exaustores. Já para elevar a temperatura de um cultivo protegido as alternativas são o uso de luzes de aquecimento, aquecedores com ventiladores acoplados, aquecimento das bancadas direcionado para as raízes e o aquecimento geotérmico, que é a utilização da temperatura do solo para trocar calor com o ambiente, usado principalmente no inverno em regiões mais frias, visto que a temperatura de 2 a 3 metros de profundidades se aproxima da temperatura média anual da localidade.

2.2.2 Umidade

Segundo o Ministério da Agricultura, Pesca e Alimentação da Colúmbia Britânica, Canadá (1994), a umidade é uma expressão da quantidade de vapor d'água presente no ar e quando se quer falar sobre a umidade do ar, geralmente essa é dada em termos de umidade relativa, porque a quantidade absoluta de água presente no ar varia conforme a temperatura, como pode ser visto na figura 2.

Figura 2 - Relação entre a temperatura e a máxima umidade presente no ar



Fonte: Ministério da Agricultura, pesca e alimentação da Colúmbia Britânica, Canadá. (1994)

A umidade relativa do ar é a razão entre o vapor d´água e a quantidade total de água em saturação no ar, observando o gráfico da Figura 2, pode-se perceber que quanto maior a temperatura do ar, maior a quantidade de vapor d'água necessária para saturá-lo.

O principal mecanismo de uma planta para lidar com a umidade é o ajuste dos estômatos foliares, que podem se abrir ou fechar, conforme a necessidade. Quanto à fotossíntese, os níveis de umidade a afetam indiretamente, uma vez que o CO_2 é absorvido pela abertura dos estômatos.

Não há um nível de umidade relativa do ar que seja ótimo para todos os tipos de plantações, mas geralmente, em cultivo protegido, as plantações costumam ter um melhor crescimento em níveis mais altos de umidade relativa do ar, mas deve-se evitar um elevado nível de umidade relativa do ar, perto do ponto de condensação da água, pois pode favorecer o desenvolvimento de pragas. Quando está no ponto de saturação, as plantas não conseguem evaporar a água que está presente nas suas folhas, limitando a absorção de nutrientes.

Para abaixar a umidade relativa do ar, as duas estratégias mais comuns são o aquecimento do ambiente e a realização de uma ventilação forçada para trocar umidade com o ar exterior.

Já para elevar a umidade relativa do ar, as estratégias mais utilizadas são o uso de máquinas umidificadoras de ar, como nebulizadores e aspersores, e o uso de telas.

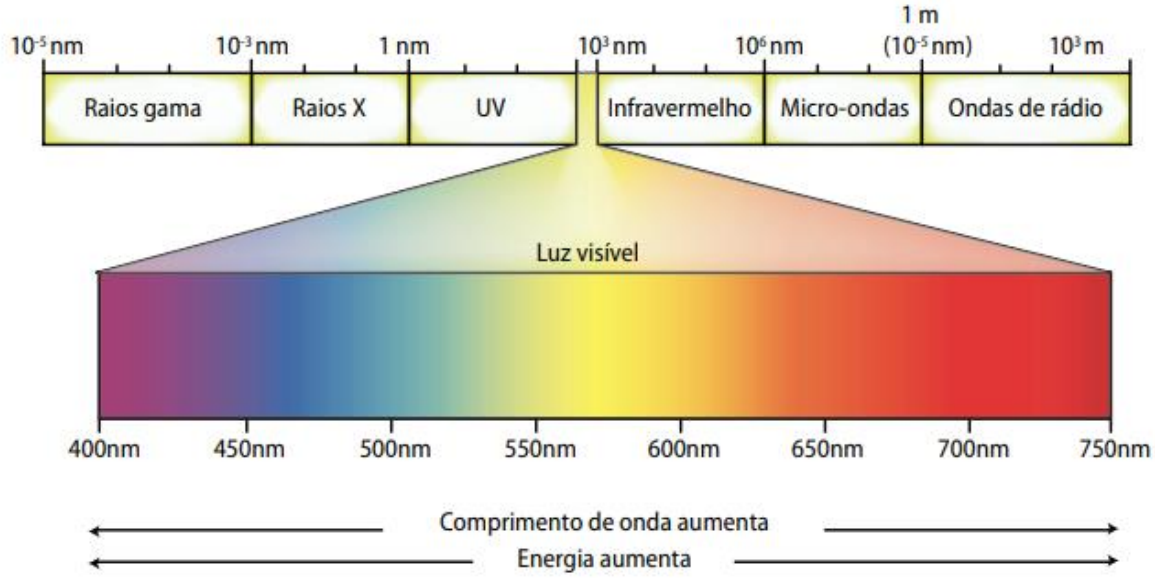
2.2.3 Iluminância

A iluminância é uma medida da quantidade de luz que incide sobre uma superfície. É definida como a razão entre o fluxo luminoso e a área da superfície iluminada, dado em lux, que é equivalente a lúmen/m².

Segundo Paulilo *et al*. (2015), a fotossíntese pode ser considerada como um dos processos biológicos mais importantes na Terra, pois através dele é consumido o dióxido de carbono e liberado o oxigênio. A energia trazida pelas luzes permite a ocorrência do fenômeno da fotossíntese nas plantas, no qual o dióxido de carbono em conjunto com a água se transforma em oxigênio e glicose.

A iluminância ainda é essencial para o desenvolvimento da planta por agir no seu crescimento, na sua estrutura, orientação e reprodução. Entretanto para isso ocorrer, deve haver um controle da intensidade luminosa direcionada à planta. Caso seja muito baixa, poderão ser usadas fontes de iluminação auxiliares, além do sol. No caso de ter uma intensidade luminosa elevada, além do necessário para o desenvolvimento saudável da planta, é necessário reduzi-la através do uso de técnicas de sombreamento.

Figura 3 - Espectro de radiação eletromagnética emitida pelo Sol.



Fonte: Paulilo (2015)

Como pode ser vista na Figura 3, a radiação solar pode ser dividida em três grupos, a ultravioleta, que corresponde a comprimentos de onda menores que 400nm e que podem causar queimaduras de pele devido a sua alta energia, o segundo grupo consiste na luz visível, que compreende a faixa de 380 a 770 nm, onde se encontra a faixa de radiação principal que vai

fornecer energia para a fotossíntese e o terceiro grupo é o da faixa do infravermelho, que corresponde a comprimentos de onda maiores que 770nm e possui um efeito de aquecimento.

3 METODOLOGIA

Esse capítulo descreverá os componentes físicos (hardwares), utilizados na montagem do sistema de controle da estufa, bem como a lógica para o controle e monitoramento desse sistema (firmware).

3.1 HARDWARE

A proposta inicial desse trabalho era de utilizar um FPGA (Field Programmable Gate Arrays) como controlador central do projeto. No entanto, ao serem analisados o custo e a complexidade na utilização do que estava sendo proposto, foi encontrada uma alternativa mais viável para o projeto através da utilização de um microcontrolador com Wi-Fi integrado, o ESP32, da empresa chinesa Espressif Systems.

Tabela 1 - Comparação entre um FPGA e o ESP32

Recurso	EP3C25E144C8N	ESPWROOM32
Clock do processador	Até 315MHz	Até 240MHz
Memória	75 kB	520 kB SRAM
Consumo de energia	Alto	Baixo
Programação	Complexa	Simples
Aplicação	Aplicações de alta performance	Aplicações de baixo consumo de energia
Preço (Sem periféricos)	$88,86	$3,00

Fonte: Mouser

Na Tabela 1, é efetuada uma comparação entre o FPGA modelo EP3C25E144C8N da série Cyclone III da Intel com o ESP-WROOM-32. Pode-se perceber que a utilização do microcontrolador será mais adequada ao projeto proposto, por ser mais simples de ser programado e ter um custo de aquisição inferior.

3.1.1 Esp32

A fabricante do ESP-WROOM-32 o descreve como um módulo Wi-Fi + Bluetooth que visa atender uma vasta gama de aplicações, desde redes de sensores de baixo consumo, até tarefas mais exigentes como codificação de voz e decodificação de áudio no formato MP3.

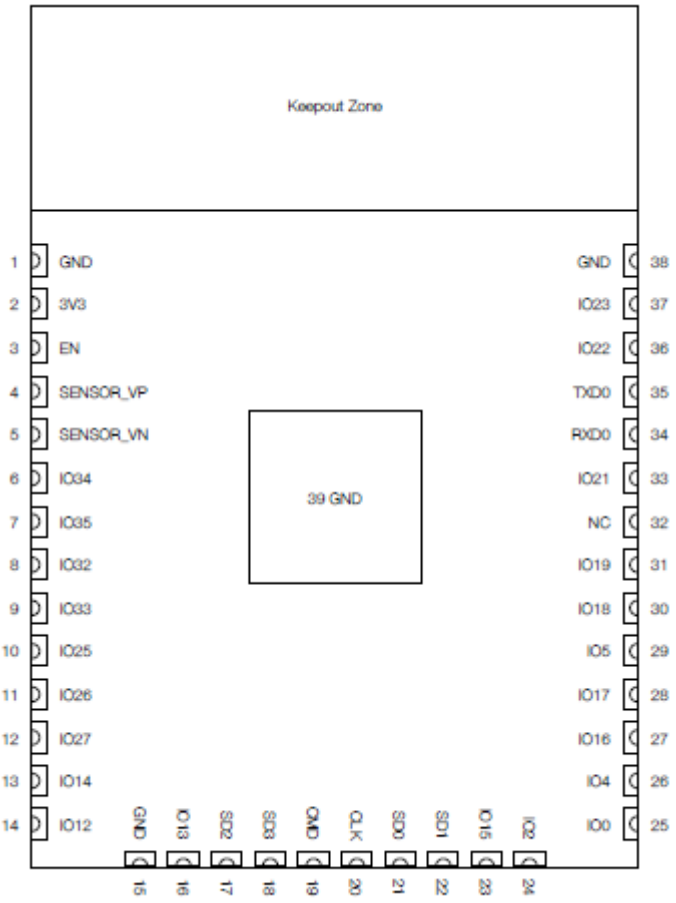
Tabela 2 - Especificações do módulo ESP32-WROOM-32

Categoria	Item	Especificação
Certificação	RF	Conforme segundo a Diretriz de Equipamentos de Rádio da União Europeia
	Wi-Fi	Wi-Fi Alliance
	Bluetooh	BQB
	Green	RoHS/REACH
Teste	Confiabilidade	HTOL/HTSL/uHAST/TCT/ESD
Wifi	Protocolos	802.11 b/g/n (802.11n até 150 Mbps) A-MPDU e A-MSDU
	Frequência de operação	2412 ~ 2484 MHz
Bluetooth	Protocolos	Bluetooth v4.2 BR/EDR e Bluetooth LE
	Rádio	Receptor NZIF com sensibilidade de -97 dBM Classe-1, Classe-2 e transmissor classe-3 AFH
	Áudio	CVSD e SBC
Hardware	Módulos de interface	Cartão SD, UART, SPI, SDIO, I2C, LED PWM, Motor PWM, I2S, IR, contador de pulso, GPIO, sensor de toque capacitivo, ADC, DAC, TWAI® e CAN 2.0
	Cristal Integrado	Frequência de 40 MHz
	Flash SPI integrado	4 MB
	Tensão de operação	3,0 V ~3,6 V
	Corrente de operação	Média de 80 mA
	Corrente mínima da fonte a ele conectado	500mA
	Temperatura recomendada de operação	-40°C ~ 85°C
	Tamanho	18mm x 25,5 mm x 3,1 mm
	Nível de sensibilidade à umidade (MSL)	Nível 3

Fonte: ESPRESSIF Systems

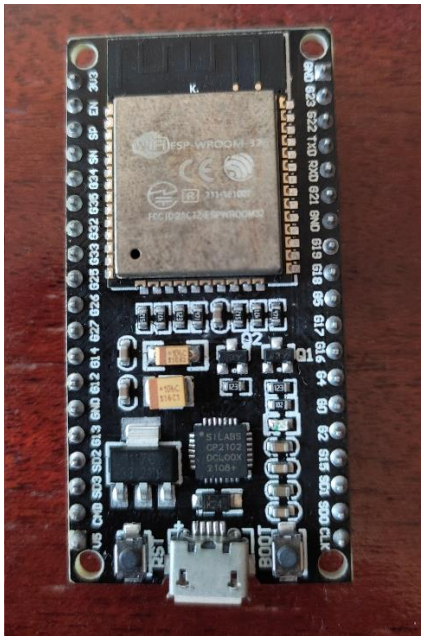
Na tabela 2 estão descritas as especificações gerais do módulo presente na placa de desenvolvimento utilizada no projeto. Os pinos correspondes às funções do ESP-WROOM-32 estão numerados e descritos na Figura 4, abaixo.

Figura 4 - Posicionamento dos pinos no chip do ESP-WROOM-32



Fonte: ESPRESSIF Systems

Figura 5 - Placa de desenvolvimento ESP-WROOM-32



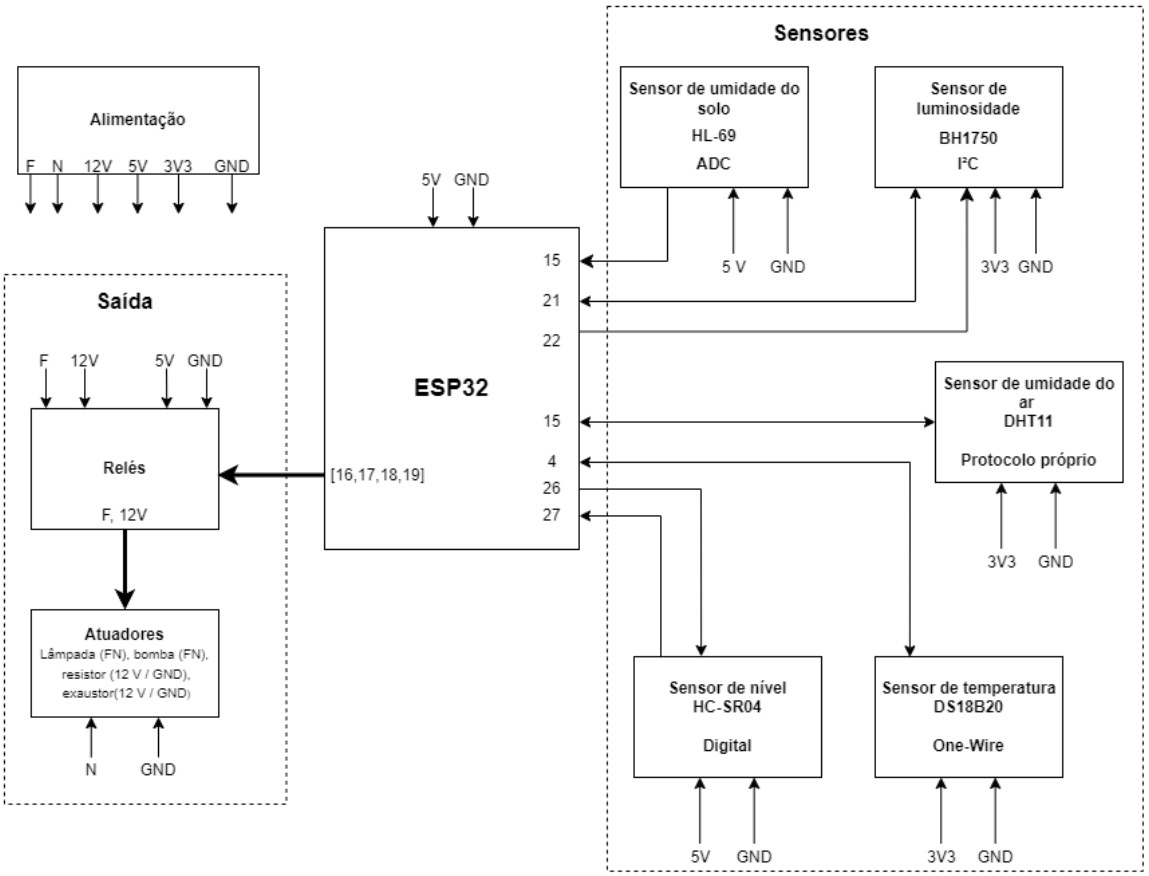
Fonte: Autoria própria

A placa de desenvolvimento da Figura 5 é utilizada para facilitar a construção de protótipos, uma vez que nela está presente um circuito integrado responsável pela interface de comunicação entre a porta USB do computador e a UART do microcontrolador e um circuito integrado regulador de tensão de 5 V para 3,3V, podendo assim ser alimentada diretamente pela porta USB de um computador, ou por um carregador genérico de celular de 5 V através de um cabo de terminais USB-A e Micro-USB.

3.1.2 Diagrama em blocos do projeto

No diagrama em blocos da Figura 6 é mostrada uma visão geral do projeto para uma melhor visualização de como foram feitas as conexões realizadas, protocolos utilizados para cada sensor e as suas alimentações.

Figura 6 - Diagrama de blocos do projeto



Fonte: Autoria Própria

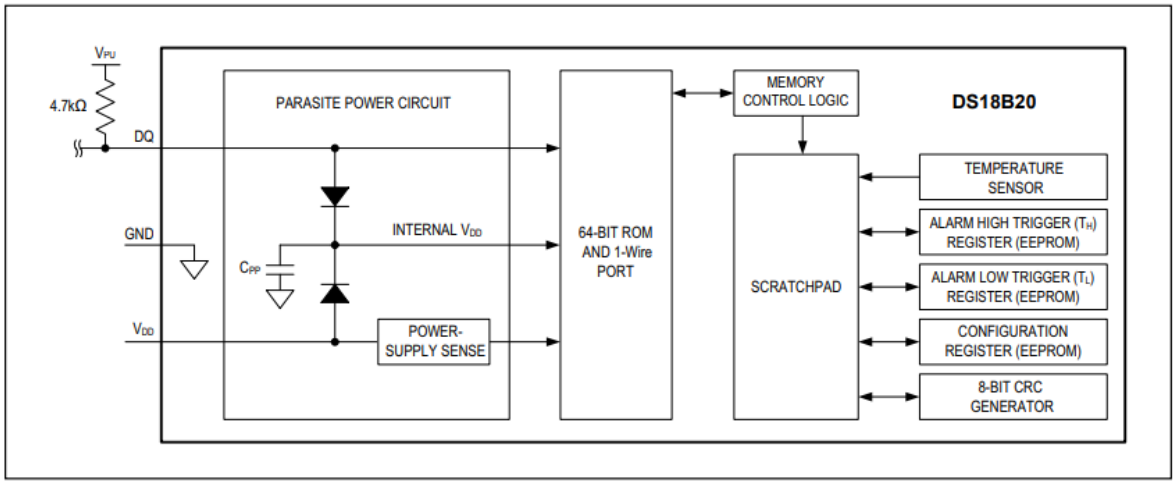
3.1.3 Sensor de temperatura

O sensor de temperatura que será usado para medir a temperatura ambiente será o DS18B20, fabricado pela Maxim Integrated, que o descreve em sua documentação técnica como sendo um termômetro digital que fornece uma medida de temperatura em graus Celsius com resolução de 9 a 12 bits, possuindo um alarme com função não-volátil de disparo por pontos de alta e baixa temperatura, programados pelo usuário.

Ainda segundo sua documentação técnica, ele é capaz de medir temperaturas que vão de -55°C a +125°C, com ± 2°C de precisão e tem sua precisão melhorada para ± 0,5°C a temperaturas entre -10°C e +85°C. Sendo assim, adequado para o que está sendo proposto nesse trabalho.

O sensor pode ser alimentado tanto por uma fonte de alimentação externa de 3,0 V a 5,5 V, quanto através de um modo de alimentação que utiliza apenas um único fio para alimentação e para transmissão de dados. Isso pode ocorrer porque ele possui um circuito interno específico para armazenar energia em um capacitor, assim que o barramento fica em nível alto, e esse mesmo capacitor consegue fornecer a energia necessária para o funcionamento do sensor quando o barramento está em nível baixo. O diagrama de blocos da Figura 7 ilustra o seu funcionamento interno.

Figura 7 - Diagrama bloco do sensor de temperatura DS18B20



Fonte: Maxim Integrated

Por questão de confiabilidade e por ter acesso a uma fonte de alimentação externa, essa será utilizada para a alimentação do sensor.

O DS18B20 se comunica pelo protocolo 1-Wire®, e assim como os outros dispositivos de mesmo protocolo, contém um endereço serial único de 64 bits que lhe é dado no momento

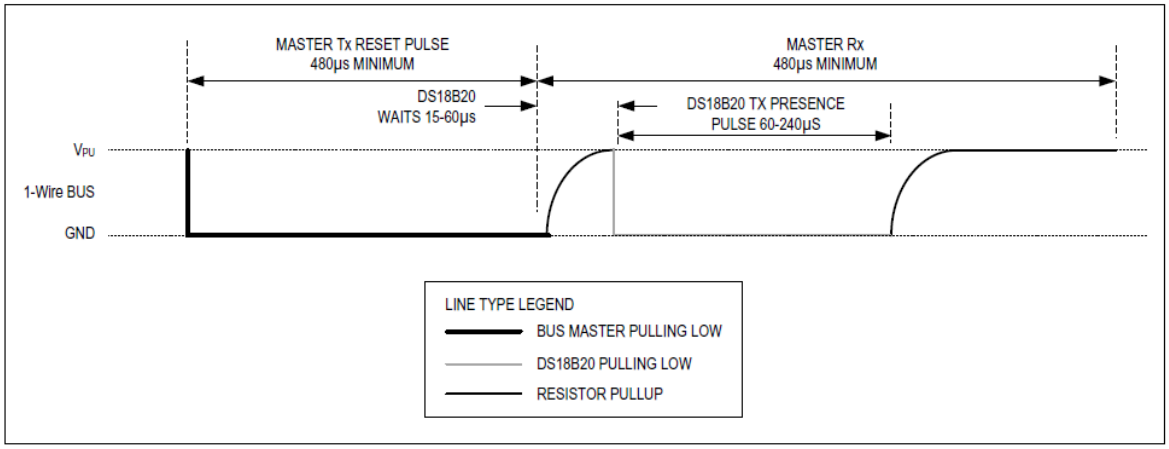
em que é manufaturado, sendo um byte para indicar o tipo de dispositivo e outro byte para detecção de erros. Horowitz e Hill (2015) descrevem esse protocolo como sendo capaz de conectar múltiplos dispositivos a um microcontrolador e de transmitir, de maneira serial, dados, endereços e alimentação através de um único fio.

Ainda segundo Horowitz e Hill (2015), o protocolo 1-Wire® permite que múltiplos dispositivos escravos se conectem entre um mesmo barramento de dados e um terminal de terra ao dispositivo mestre e a transmissão de dados acontece da seguinte forma: a tensão do barramento é levada para 5 V, o que permite que os dispositivos pendurados ao barramento se energizem e respondam com um breve estado baixo, então o mestre iniciará a comunicação, confirmando os endereço dos dispositivos e depois transmitindo ou recebendo os dados. Os dados são codificados em formato binário, sendo “1” um pulso de duração menor que 15 µs e “0” um pulso de 60 µs. Para transmitir os dados, o microcontrolador apenas envia os dados codificados da maneira que foi explicada anteriormente, já para receber os dados, ele envia o pulso correspondente a “1” e o endereço do dispositivo escravo, que por sua vez já deixa o nível lógico do barramento alto, sinalizando “1”, ou mantém o nível lógico do barramento baixo por 60 µs, sinalizando “0”.

Na folha de dados fornecida pela fabricante do sensor são dados os passos para que ocorra a sequência de transição, que têm que ser seguidos para que haja uma correta transmissão dos dados.

O primeiro passo é sobre a sequência de inicialização, que consiste num pulso de reset transmitido do microcontrolador para o sensor, seguido de um pulso de presença transmitido pelo sensor para o microcontrolador, avisando-o que está pronto para a operação. Os tempos de cada ação podem ser observados na Figura 8. O microcontrolador mantém o nível lógico baixo por um período de pelo menos 480 µs, o microcontrolador então libera o barramento e o sensor detecta que houve uma borda de subida, aguardando de 15 µs a 60 µs para então manter o nível lógico do barramento baixo por um período de 60 µs a 240 µs e em sequência libera o barramento, para então voltar para o nível lógico alto.

Figura 8 - Tempos de inicialização do sensor



Fonte: Maxim Integrated

A Figura 9 mostra o comportamento de inicialização do sensor utilizando o analisador lógico USB e o seu *software* “Logic 2”, ambos fornecidos pela empresa “Saleae”.

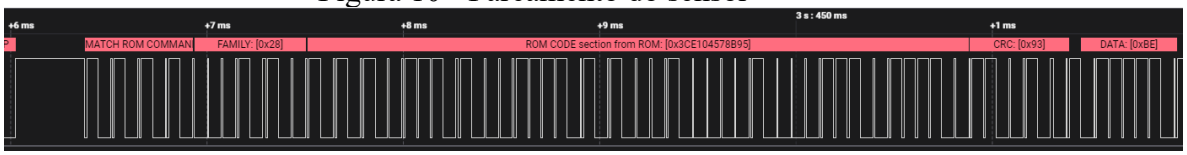
Figura 9 - Inicialização do sensor DS18B20 utilizando um analisador lógico

Fonte: Autoria própria

O segundo passo é executar os comandos de ROM, o “Search ROM” sendo usado para identificar quais e quantos são os sensores e os endereços de 64 bits de cada um que estão pendurados no barramento. Caso só haja um sensor no barramento é usado um outro comando, “Read ROM”, para obter a sequência de 64 bits do sensor. Também há um comando, “Match ROM”, que é seguido pelo endereço de 64 bits para que haja a comunicação apenas com o sensor endereçado. Os 64 bits do endereço de um sensor são alocados da seguinte forma: os 8 bits menos significativos são referentes à família do sensor, sendo 28h para o DS18B20, os próximos 48 bits contém um número serial único e os 8 bits mais significativos são referentes ao CRC (*Cyclic Redundancy Check*), que consiste no resultado de uma rotina de verificação da validade dos 56 outros bits transmitidos.

A Figura 10 mostra o comando “Match ROM [0x55]” sendo executado seguido do endereço de 64 bits do sensor.

Figura 10 - Pareamento do sensor

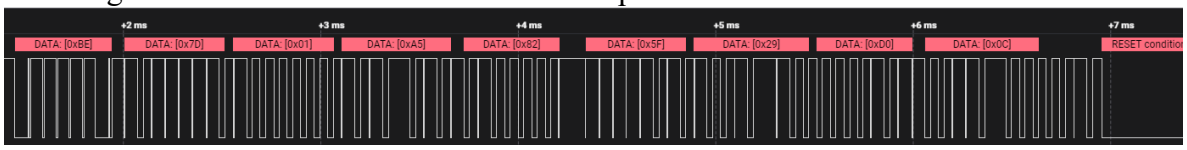


Fonte: Autoria própria

O último passo consiste na execução dos comandos de função, que permitem o microcontrolador ler e escrever dados na SPM (Scratchpad memory) do sensor. O principal e único comando que será utilizado nessa aplicação será o de leitura da SPM, “Read Scratchpad”, que consiste em 8 bytes de dados e um byte de CRC. Nos 2 primeiros bytes são armazenados no formato de complemento de 2 os dados referentes à temperatura obtida pelo sensor.

Na Figura 11, é exibida a execução do comando “Read Scratchpad” [0xBE] e logo em sequência a leitura dos dois bytes de dados do sensor. O byte “0” é o menos significativo (LSB) e contém o dado 0x7D, enquanto o byte “1” é o mais significativo (MSB), contendo o dado 0x01.

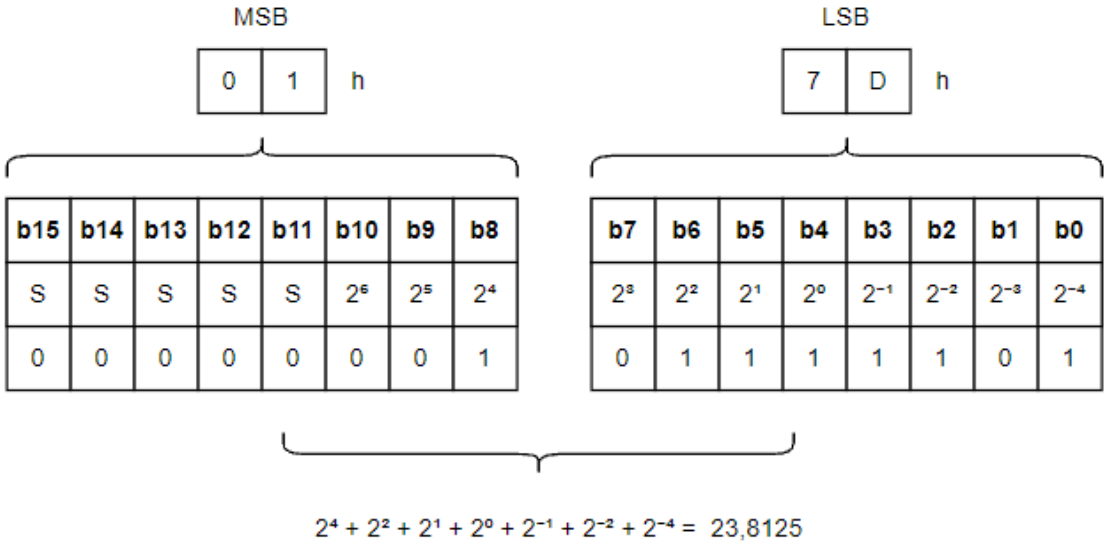
Figura 11 - Transmissão dos dados de temperatura do sensor DS18B20



Fonte: Autoria própria

Na Figura 12, é demonstrado como é feita a conversão dos dois primeiros bytes de dados obtidos pelo comando “Read Scratchpad” em um valor de temperatura em graus Celsius.

Figura 12 - Interpretação dos dados de temperatura obtidos na Figura 11

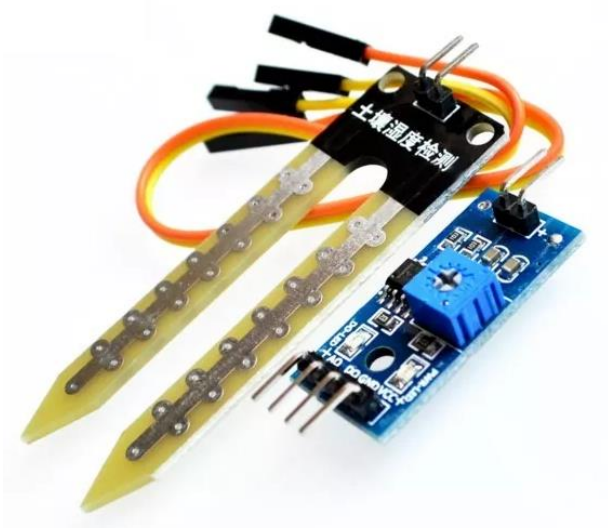


Fonte: Autoria própria

Os bits de “b11” a “b15” correspondem ao sinal do valor da temperatura, se os bits nas posições “S” forem iguais a “0” significa que o valor da temperatura é positivo, caso sejam iguais a “1”, o valor é negativo. Os bits de “b0” a “b10” possuem um peso na base 2 associado à sua posição no vetor, começando por 2^{-4} na posição “b0” até o peso 2^{6} na posição “b10”. Ao ser realizada a soma dos dados obtidos nesse caso, tem-se a temperatura de 23,8125 °C.

3.1.4 Sensor de umidade do solo

Figura 13 - Sensor de Umidade HL-69



Fonte: Vamuino

O sensor de umidade utilizado no projeto para aferir a umidade do solo é o HL-69, conforme ilustrado na Figura 13. Ele é formado por duas placas, uma contendo dois eletrodos que são inseridos no solo, sendo que a resistência entre os eletrodos é alta quando o solo está seco e baixa quando úmido. Já a outra placa contém um circuito comparador com tensão limiar controlada por um potenciômetro, que irá comparar a tensão ajustada pelo usuário com a tensão sob o sensor, devolvendo um sinal de nível lógico alto ou baixo na saída digital (D0) e um sinal analógico indicando o nível de umidade do solo, sendo que quando o solo estiver seco, a tensão de saída será alta e quando estiver úmido, será baixa.

A leitura do sinal analógico é feita através de um canal do módulo ADC (Analog to Digital Converter), presente no ESP 32. Segundo Valvano (2012), o ADC o responsável por fazer a conversão de um sinal analógico para o formato digital, sendo ele quem faz a aquisição do sinal de entrada, normalmente uma tensão analógica, e retorna na saída um valor binário.

A precisão de um ADC é o número de entradas distintas que ele pode reconhecer, sendo calculada segundo a equação (1). O intervalo é a tensão de leitura de uma amostra e resolução é a menor parte distinguível de um sinal. A equação (2) descreve a relação entre esses parâmetros.

$$\text{Precisão}_{\text{ADC}} = 2^{\text{N}} - 1 \tag{1}$$

Sendo N o número de bits do ADC.

$$\text{Intervalo [V]} = \text{Precisão} \times \text{Resolução [V]} \tag{2}$$

Segundo a folha de dados do microcontrolador ESP-32-WROOM32, o ADC possui 12 bits e uma tensão máxima na entrada do canal de 3,3 V, utilizando a equação (2) tem-se a resolução, em Volts, do ADC.

$$\text{Resolução [V]} = \frac{\text{Intervalo [V]}}{\text{Precisão}} = \frac{3{,}3}{2^{12} - 1} = 0{,}805808 \text{ mV}$$

O sensor de umidade HL-69 irá entregar um maior valor de tensão quanto mais seco o solo estiver 3V e ao ser submergido em água entrega um valor de tensão de 1,2 V. Algumas alterações tendo que ser feitas via firmware para adequar esse intervalo entregue pelo sensor em um nível percentual de umidade do solo. Pela equação (2), pode-se retirar o valor da precisão correspondente ao valor de 1,2V e 3V.

$$\text{Precisão}_{\text{úmido}} = \frac{\text{Intervalo[V]}}{\text{Resolução [V]}} = \frac{1{,}2 \text{ V}}{0{,}805808 \text{ mV}} = 1489{,}08$$

$$\text{Precisão}_{\text{seco}} = \frac{\text{Intervalo[V]}}{\text{Resolução [V]}} = \frac{3 \text{ V}}{0{,}805808 \text{ mV}} = 3722{,}97$$

Como o ADC retorna apenas valores inteiros, serão adotados no firmware os valores de 1489 e 3723 como sendo os limites inferior e superior, respectivamente, da faixa de operação.

3.1.5 Sensor de iluminância

O sensor de iluminância a ser utilizado no projeto é o BH1750FVI, da fabricante japonesa ROHM Semiconductor. Segundo as notas técnicas fornecidas pela fabricante, o circuito integrado desse sensor pode ser alimentado com tensões entre 2,4 V e 3,6 V, comunica-se pelo

protocolo I²C, possui uma resolução de 16 bits, tendo assim uma faixa de alta precisão de leitura, de 1 a 65536 lux, e possui uma baixa sensibilidade a luzes infravermelhas.

Embora esse sensor tenha sido desenvolvido para ser utilizado no interior de celulares para o controle da intensidade luminosa de telas LCD e luzes de fundo de teclados, ele será usado para medir a intensidade luminosa no interior da estufa, uma vez que ela não estará exposta diretamente ao sol e, assim, a iluminância não ultrapassará a capacidade de leitura do sensor.

A tabela 3 exibe a intensidade luminosa medida em diferentes situações parar trazer uma noção de grandeza dessa unidade.

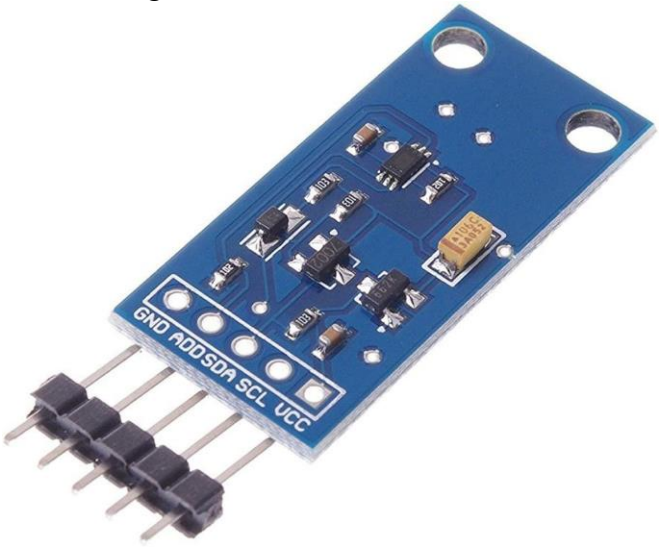
Tabela 3- Ordem de grandeza da intensidade luminosa em algumas situações

Situação	Intensidade luminosa [lux]
Luz das estrelas	~ 0,002
Luar	~ 0,2
Iluminação das ruas	6 ~12
Luz do dia em interiores	500 - 2000
Luz do dia em exteriores	100 ~ 20000
Luz do Sol direta	50000 ~ 100000

Fonte: Universidade de São Paulo

Para facilitar a aplicação no projeto, será utilizado o módulo GY-30, ilustrado na Figura 14, que contém o sensor BH1750FVI, permitindo sua alimentação com 5 VDC.

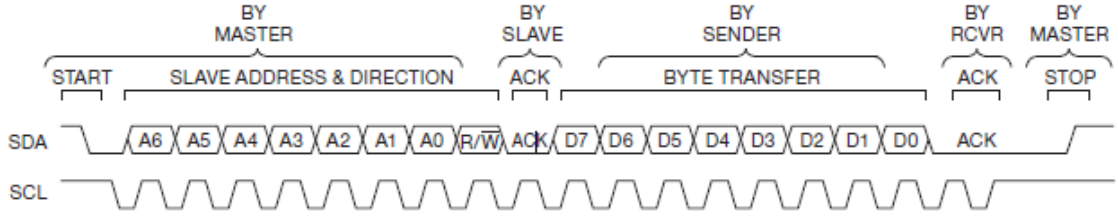
Figura 14 - Módulo GY-30



Fonte: Amazon

O protocolo I²C (Inter-Integrated-Circuit), utilizado por esse sensor, funciona do seguinte modo segundo Horowitz e Hill (2015): ele possui dois barramentos de comunicação que são comuns para todos os dispositivos escravos conectados ao mestre, sendo uma linha de dados seriais (SDA) e uma linha de clock (SCL) para sincronizar a transmissão. Os bits de dados são transmitidos em sincronismo com a borda de subida do sinal do barramento SCL. A linha de transmissão de dados seriais é "half-duplex", ou seja, os dados são enviados e recebidos por ela, mas apenas em uma direção por vez.

Figura 15 - Transmissão de dados usando o protocolo I²C



Fonte: HOROWITZ e HILL (2015)

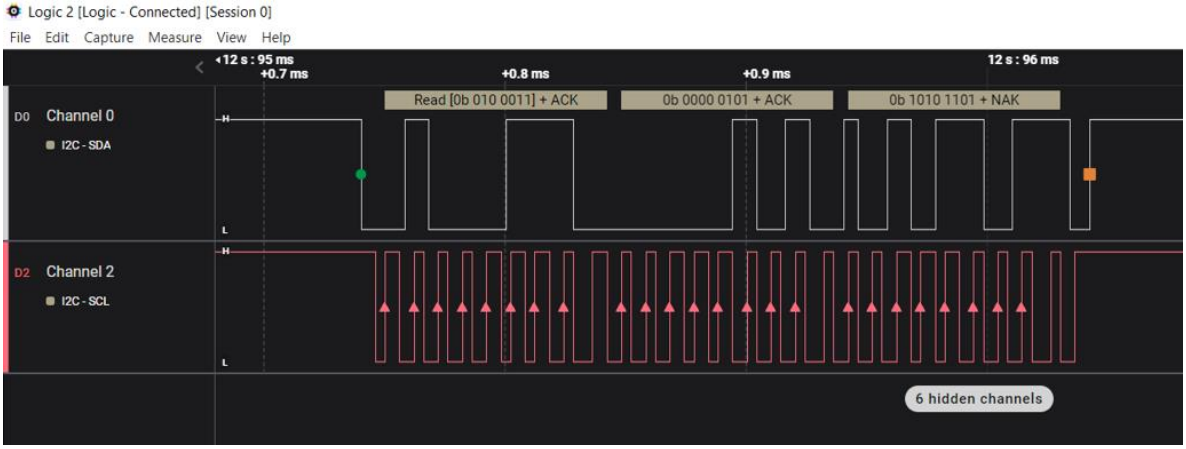
Pela Figura 15 podemos ver que o início da transmissão é sempre dado pelo mestre, iniciando por ele transmitindo um bit de início, sete bits de endereço do dispositivo escravo (A6 a A0), um bit para informar se é leitura ou escrita (R/W'), um bit de reconhecimento pelo dispositivo escravo e, em seguida, é feita a transmissão dos bits de dados a depender da direção que foi dada junto ao endereço do dispositivo escravo, podendo conter no máximo 32 bytes de dados por mensagem. Na sequência, o dispositivo receptor da mensagem envia um bit de reconhecimento da mensagem e o mestre encerra a comunicação mantendo o SCL em nível alto e enviando uma borda de subida no barramento SDA.

Na

Figura 16, pode-se perceber o comportamento do protocolo, o byte 0b0100011 ou 23h indica o endereço do sensor seguido do pulso em nível lógico alto de leitura e um do sinal de reconhecimento (ACK). Na sequência, tem-se dois bytes de dados indicando o dado correspondente a intensidade luminosa captada e transmitida pelo sensor ao microcontrolador.

Para obter o dado da figura abaixo, foi utilizado o analisador lógico e posicionada uma fonte de luz diretamente sobre o sensor, a fim de captar um valor alto de intensidade luminosa.

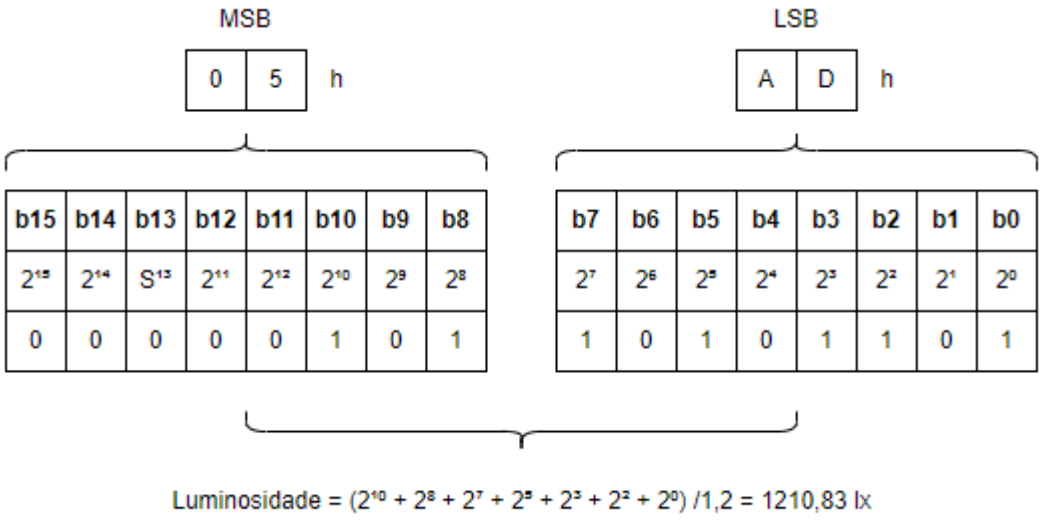
Figura 16 - Transmissão do dado de iluminância pelo sensor BH1750FVI



Fonte: Autoria própria

Mostra-se na Figura 17 como é feita a transformação dos bytes de dados transmitidos para o nível de intensidade luminosa. O byte mais significativo é o que é transmitido primeiro, o 0b0000101, e logo em sequência, o menos significativo, 0b10101101. O peso de cada bit corresponde a sua posição na palavra, indo de 2^0 até 2^{15} e, ao final, é feita a soma seguida da divisão por 1,2 para obter o dado de iluminância em lux.

Figura 17 - Interpretação dos dados de temperatura obtidos na Figura 16



Fonte: Autoria própria

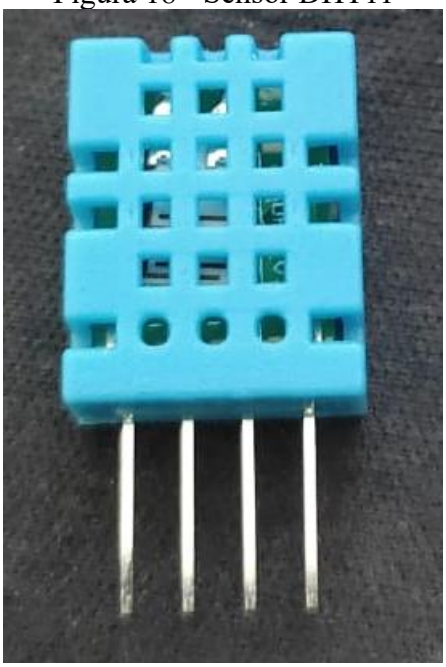
3.1.6 Higrômetro

O higrômetro escolhido para esse projeto é o DHT11, que embora também conte com um sensor de temperatura incluso nele, será considerada no projeto apenas a informação de

umidade ambiente fornecida por ele, sua informação de temperatura será utilizada apenas caso haja perda de comunicação com o DS18B20. Essa escolha foi feita devido ao DS18B20 possuir uma maior precisão na leitura de temperatura, ±0,5 °C, frente aos ±2,0 °C de precisão do DHT11.

Segundo o manual do fabricante, ele pode ser alimentado com uma faixa de tensão de 3 a 5,5 VDC e se utiliza de um protocolo de comunicação próprio, semelhante ao 1-Wire® para transmissão de dados para o microcontrolador. Possui uma faixa de leitura de umidade relativa do ar de 20% a 90%, com resolução de 1% e precisão de ±5%. O sensor DHT11 está ilustrado na Figura 18, abaixo.

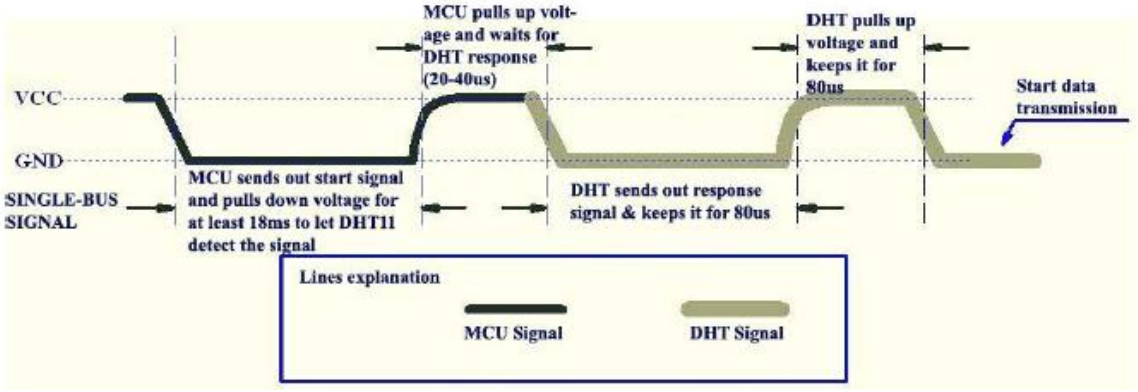
Figura 18 - Sensor DHT11



Fonte: Autoria própria

Consultando sua folha de dados fornecida pelo fabricante, é possível compreender como é feita a transmissão dos dados. O barramento enquanto está livre permanece em nível lógico alto, quando o microcontrolador solicita que a transmissão seja iniciada ele envia um sinal de nível lógico baixo por pelo menos 18ms, depois retorna o barramento para o nível lógico alto e aguarda uma resposta do sensor de 20 µs a 40 µs, o microcontrolador então responderá e manterá o barramento em nível lógico baixo por 80 µs e depois irá enviar um sinal em nível lógico alto por mais 80 µs e então dará início à transmissão dos dados. Na Figura 19, pode-se visualizar como ocorre o início da transmissão.

Figura 19 - Início da comunicação entre o DHT11 e o microcontrolador



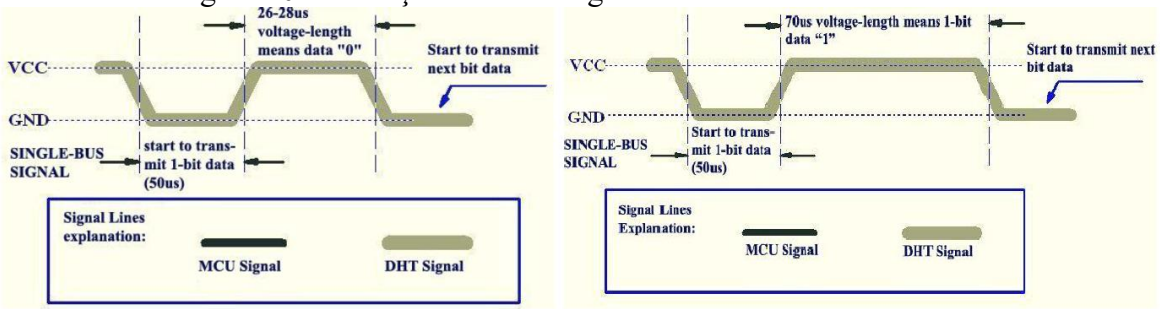
Fonte: Folha de dados do Sensor DHT11

Uma transmissão de dados completa possui 40 bits, sendo dividido em cinco blocos de oito bits, formatados do seguinte modo: primeiro bloco para a parte inteira da umidade relativa, segundo bloco para a parte decimal da umidade relativa, terceiro bloco para a parte inteira da temperatura, quarto bloco para a parte decimal da temperatura e o quinto bloco corresponde ao "*checksum*", que tem que ser igual à soma de todos os blocos anteriores para que o dado seja considerado válido.

Por se tratar de um barramento único de transmissão e recepção de dados, a compreensão do que é nível lógico alto ou baixo é dada pelo tempo em que o pulso permanece no estado alto: a transmissão de cada bit de dados se inicia com 50 µs em nível baixo e depois o tempo que se mantém em nível alto é o que definirá o nível lógico do bit, o nível lógico "0" sendo entendido quando se mantém por 26 a 28 µs, já o nível lógico "1", quando se mantém por cerca de 70 µs.

Na Figura 20, pode-se visualizar como ocorre o reconhecimento do nível lógico dos dados.

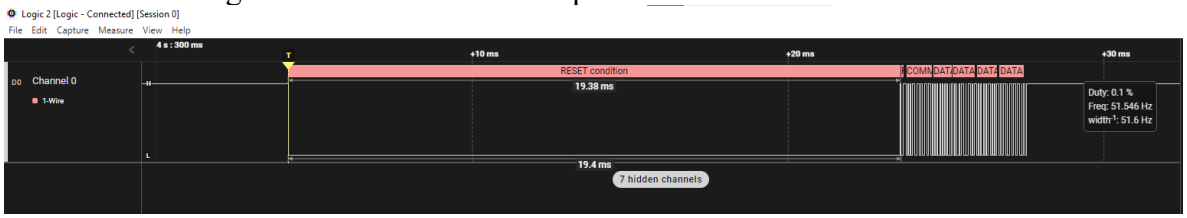
Figura 20 - Detecção do nível lógico dos bits dos dados



Fonte: Folha de dados do Sensor DHT11

Utilizando o analisador lógico, é possível visualizar o seu funcionamento no mundo real. Na Figura 21, tem-se uma visão do aspecto geral de uma transmissão completa.

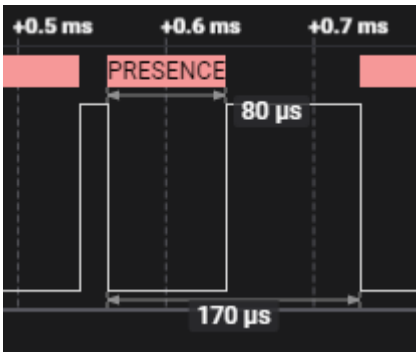
Figura 21 - Transmissão completa do sensor DHT11



Fonte: Autoria própria

Na figura 20, pode-se perceber que o barramento se mantém em nível lógico alto antes do início da transmissão e, para que o microcontrolador inicie essa rotina, ele mantém o barramento em nível lógico baixo por 19,38ms e o sensor responde mantendo o sinal em nível lógico alto por 0,02ms (20µs). A Figura 22, aproxima o sinal obtido na Figura 21, exibindo o sinal de reconhecimento de 80µs em nível baixo e 90µs em nível lógico alto, que precede o início da transmissão dos dados.

Figura 22 - Sinal de reconhecimento da presença do sensor



Fonte: Autoria própria

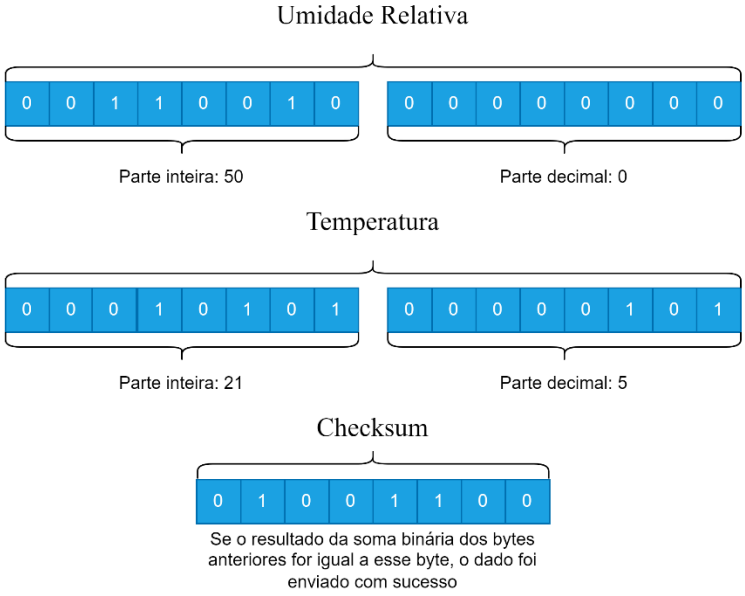
Logo após o sinal de reconhecimento, tem-se a parte principal, que são dados de leitura de umidade relativa e temperatura do sensor DHT11. Na Figura 23, foi feita a “tradução” do sinal para facilitar a transcrição, onde o pulso com um período maior corresponde ao nível lógico 1 e o de menor período ao nível lógico 0.

Figura 23 - Dados de umidade relativa e temperatura do sensor DHT11

Fonte: Autoria própria

O dado de 40 bits obtido nessa leitura foi 0011001000000000000101010000010101001100. A Figura 24 segmenta e mostra a interpretação do dado anterior para uma melhor visualização.

Figura 24 - Interpretação dos dados obtidos na leitura do sensor DHT11



Fonte: Autoria própria

Como a operação binária da equação (3) confirma a verificação do *checksum*, pode-se concluir que a transmissão do dado foi feita com sucesso, obtendo dessa forma, a umidade relativa de 50% e a temperatura de 21,05 °C.

$$00110010 + 00000000 + 00010101 + 00000101 = 01001100 \quad (3)$$

3.1.7 Sensor ultrassônico

Para assegurar que a bomba d'água opere apenas quando estiver submersa a fim de evitar que se estrague, é utilizado um sensor ultrassônico para aferir o nível d'água do recipiente onde a bomba está instalada.

O sensor utilizado é o HC-SR04, como mostrado na Figura 25, cujo princípio de funcionamento consiste na utilização da diferença de tempo obtido entre o sinal transmitido e o sinal de retorno para fazer o cálculo da distância entre objetos, conforme a equação 4.

$$d = v_s \, x \frac{t}{2} \quad (4)$$

Onde:

v_s: velocidade do som, 340[m/s]

t: tempo percorrido entre a emissão do sinal e o seu retorno [s].

d: distância entre o sensor e o obstáculo [m].

Figura 25 - Sensor ultrassônico HC-SR04

Fonte: Eletrogate

3.1.8 Iluminação

Para aumentar o fluxo luminoso no interior do modelo da estufa do projeto, será utilizada uma lâmpada LED de 9W, alimentada por tensão 127 VAC de modelo igual à da Figura 26.

Figura 26 - Lâmpada LED de 9W

Fonte: Amazon

3.1.9 Exaustor

Para a ventilação no interior da estufa, será utilizado um miniventilador de 120mm para exaustão da estufa, a fim de aumentar fluxo de ar para troca de calor com o ambiente externo. O miniventilador utilizado é representado na Figura 27, possuindo capacidade de fluxo de ar de 38,5 CFM e alimentado por uma tensão elétrica de 12 VDC.

Figura 27 – Miniventilador exaustor de 120mm

Fonte: Terabyteshop

3.1.10 Bomba d’água

Para o fornecimento de água a fim de manter a umidade do solo dentro da faixa de umidade desejada, será utilizada uma bomba d’água submersa, como mostrado na Figura 28 a seguir.

Figura 28 - Bomba d'água submersa

Fonte: Amazon

3.1.11 Aquecimento

Para fins de simulação, será utilizado um resistor de chuveiro, posicionado próximo ao sensor de temperatura e alimentada por uma tensão de 12 VDC. O modelo escolhido é mostrado na Figura 29.

Figura 29 – Resistor de chuveiro

Fonte: Amazon

3.1.12 Relés

Para o acionamento das cargas anteriores, será utilizado um módulo contendo quatro relés isolados, individualmente, por optoacopladores, a fim de proteger as portas do microcontrolador contra sobrecorrente. Cada relé é capaz de suportar até 10A, sendo suficiente para o modelo proposto. É mostrado na Figura 30, a seguir, o módulo contendo os quatro relés responsáveis pelo acionamento das cargas utilizadas no projeto.

Figura 30 - Módulo de relés

Fonte: Autoria própria

3.2 FIRMWARE

Tanenbaum e Bos (2015) definem *firmware* como sendo o programa instalado na memória ROM de um microcontrolador que controla dispositivos que não costumam ser vistos como computadores e que não aceitam que sejam instalados softwares pelo usuário. Alguns desses dispositivos são televisores, micro-ondas e carros e o conceito se aplica ao programa do projeto do presente trabalho.

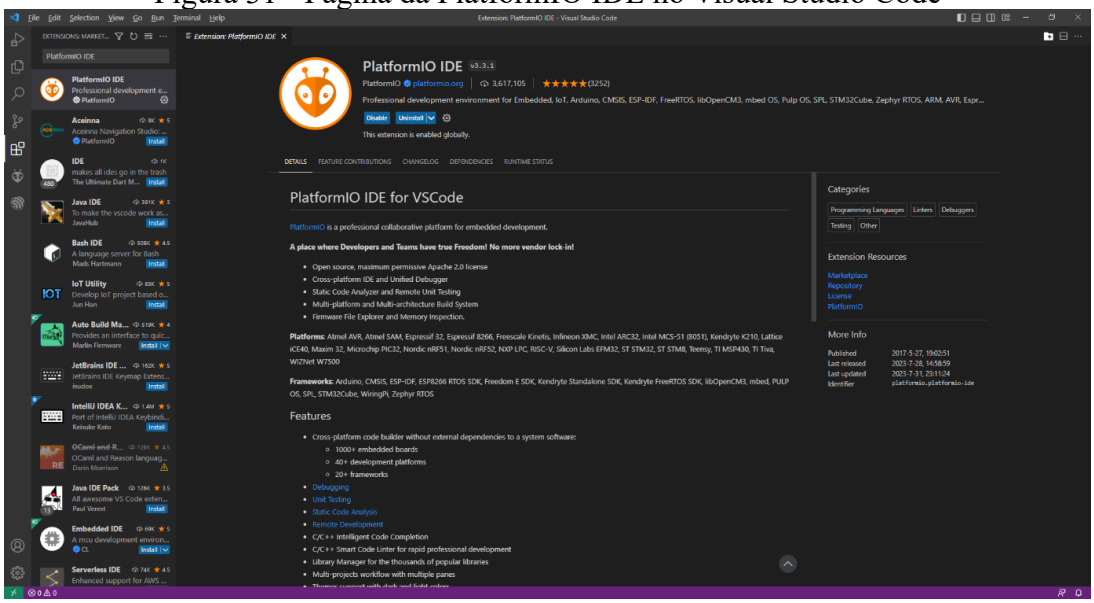
O editor de código utilizado para o desenvolvimento do firmware é o Visual Code Studio, desenvolvido pela Microsoft, que conta com um vasto suporte para extensões que auxiliam no desenvolvimento do código. O *firmware* completo está no Apêndice A.

3.2.1 Platform IO

Uma das extensões citadas do Visual Code Studio é a Platform IO, que conta com diversos frameworks para auxiliar no desenvolvimento de firmware para projetos embarcados, utilizando produtos de empresas como: STMicroelectronics, Microchip, Arduino, Raspberry Pi e Espressif.

Para isso, é necessário instalar a extensão no Visaul Studio Code, que pode ser acessada pressionando as teclas de atalho Ctrl + Shift + X e digitando na barra de pesquisa “PlatformIO IDE” e prosseguindo com a instalação. Na Figura 31, a seguir, mostra-se a tela inicial dessa extensão.

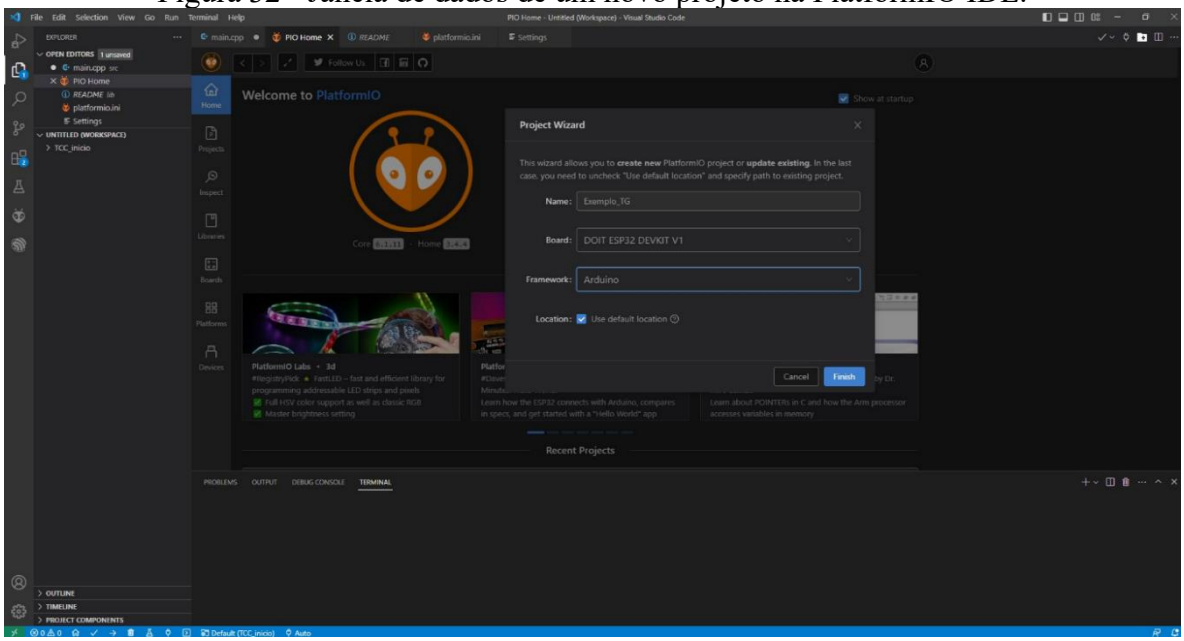
Figura 31 - Página da PlatformIO IDE no Visual Studio Code



Fonte: Autoria própria

Já com a extensão instalada, é possível criar um projeto acessando o menu principal da PlatformIO IDE clicando em “New Project”, nisso se abrirá uma janela para dar nome ao projeto e selecionar qual placa está sendo usada para executar o projeto, bem como o *framework* a ser utilizado. No caso do presente projeto, está sendo utilizada a placa DOIT ESP32 DEVKIT V1 e o *framework* Arduino, conforme mostrada na Figura 32.

Figura 32 - Janela de dados de um novo projeto na PlatformIO IDE.

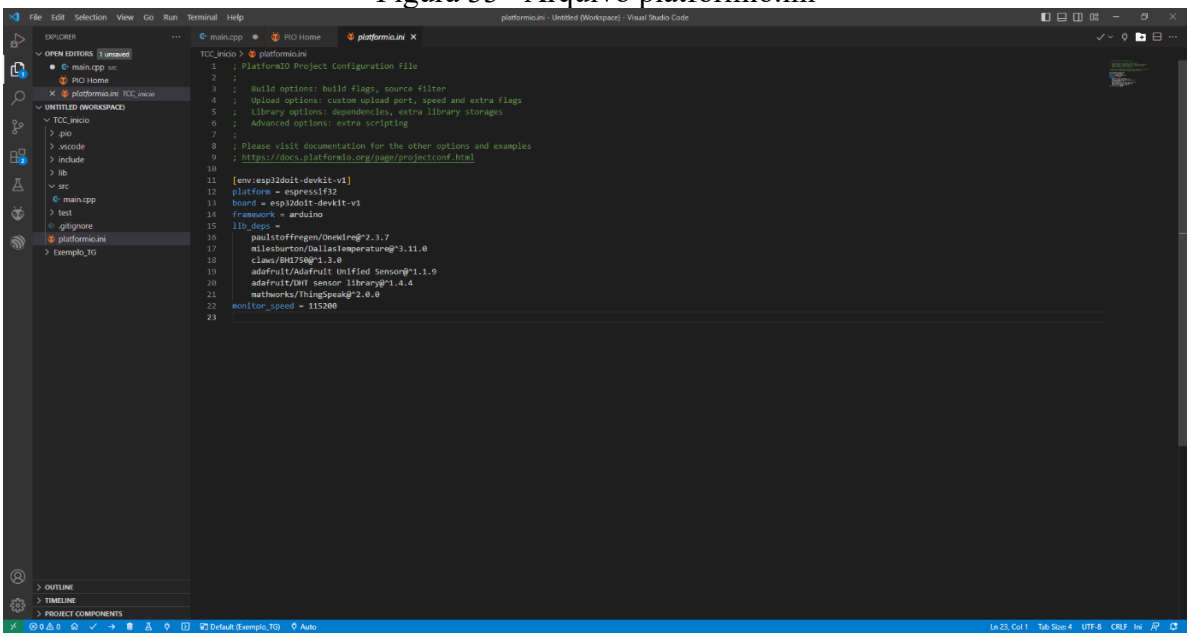


Fonte: Autoria própria

Uma vez criado o projeto nessa aba, serão criadas também as pastas e arquivos do projeto na área de trabalho do Visual Studio Code. Os dois principais arquivos criados são o

"main.cpp", onde foi escrito todo o *firmware* do projeto e o "platformio.ini", onde estão as características do projeto no que tange à plataforma, à placa, às bibliotecas, ao *baud rate* e aos *frameworks* utilizados, como pode ser visto na Figura 33, abaixo.

Figura 33 - Arquivo platformio.ini

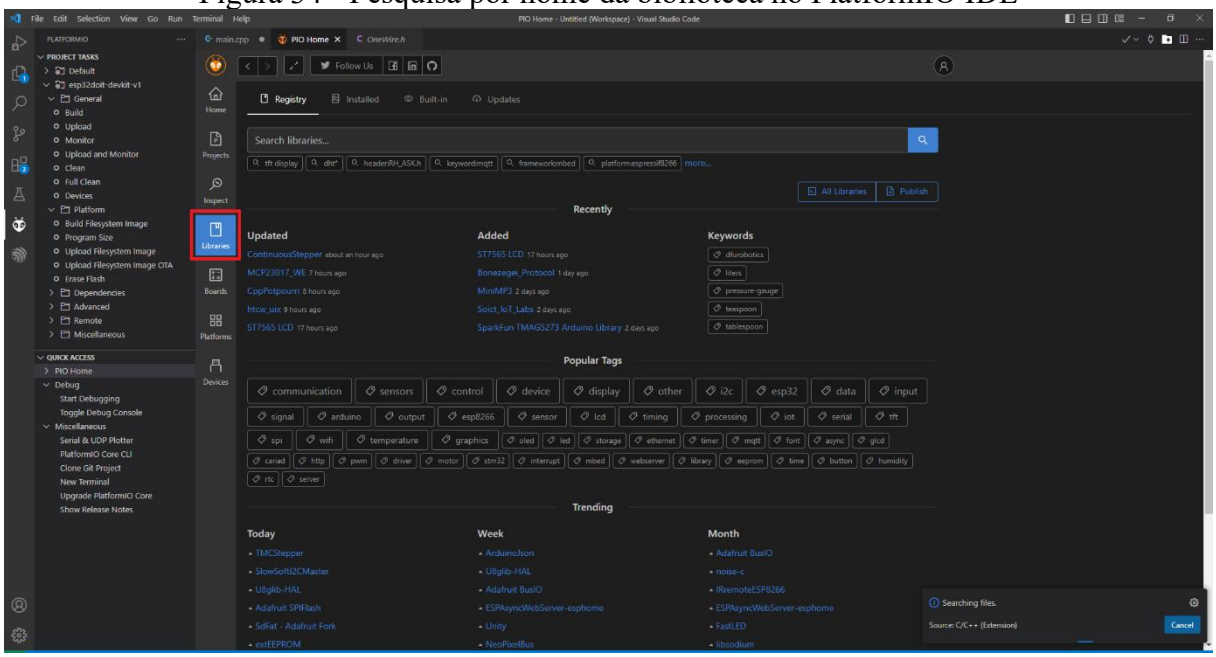


Fonte: Autoria própria

3.2.2 Bibliotecas

Para agilizar o desenvolvimento do *firmware*, foram utilizadas algumas bibliotecas para fazer a interface entre os sensores e protocolos utilizados. Para instalar é preciso acessar o menu principal da extensão "PlatformIO IDE", clicar em "Libraries", conforme mostra a Figura 34 e realizar a busca pelo nome da biblioteca desejada.

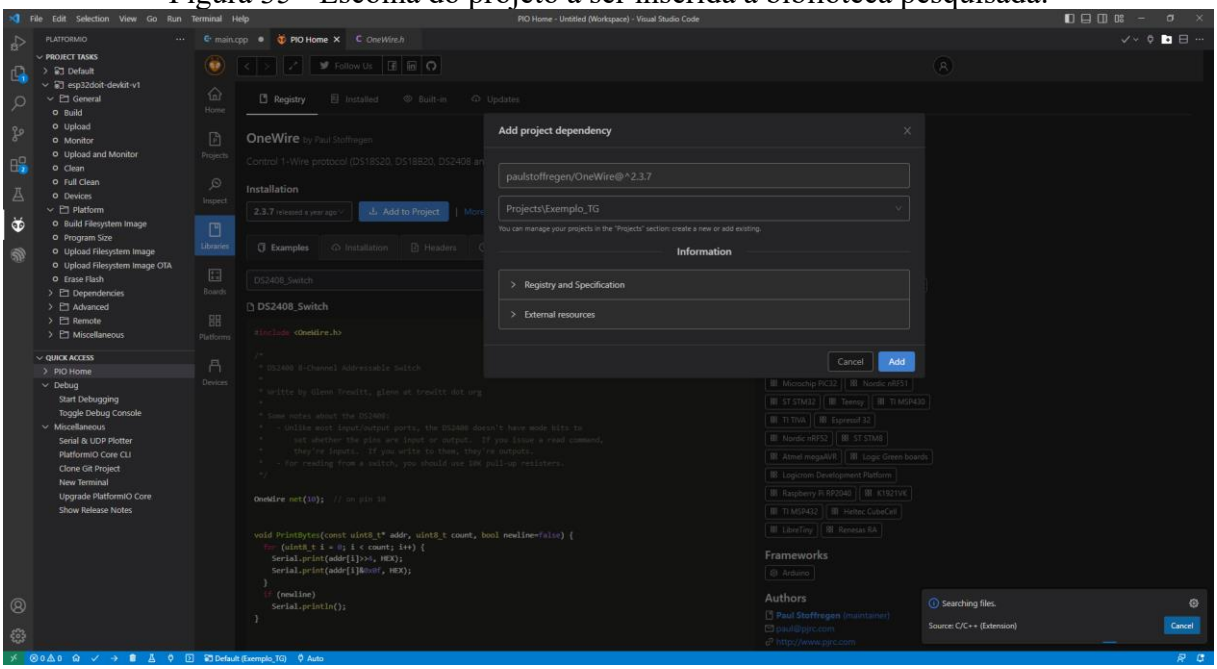
Figura 34 - Pesquisa por nome da biblioteca no PlatformIO IDE



Fonte: Autoria própria

Assim que encontrada a biblioteca escrita pelo autor desejado, é feita a sua inserção no projeto de destino clicando no botão "*Add to Project*" e selecionado o projeto desejado conforme mostrado na Figura 35.

Figura 35 - Escolha do projeto a ser inserida a biblioteca pesquisada.



Fonte: Autoria própria

Para o desenvolvimento desse projeto foram utilizadas ao todo seis bibliotecas da PlatformIO IDE, que são as seguintes:

- OneWire, fornecida por Paul Stoffregen, versão 2.3.7, responsável por fazer a leitura do protocolo OneWire do sensor de temperatura DS18B20;
- DallasTemperature, fornecida por Miles Burton, versão 3.11.0, responsável por fazer a conversão dos dados de temperatura DS18B20;
- BH1750, fornecida por Claws, versão 1.3.0, responsável por fazer a leitura do protocolo I²C e conversão dos dados de iluminância do sensor BH1750;
- Adafruit Unified Sensor, fornecida por Adafruit, versão 1.1.9, responsável por fazer a leitura do protocolo do sensor de temperatura DHT11;
- DHT sensor library, fornecida por Adafruit, versão 1.4.4, responsável por fazer a conversão dos dados de temperatura DHT11;
- ThingSpeak, fornecida por Math Works, versão 2.0.0, responsável por fazer a inserção de valores das informações coletadas pelos sensores no banco de dados da plataforma *ThingSpeak*, descrita a seguir.

3.2.3 ThingSpeak

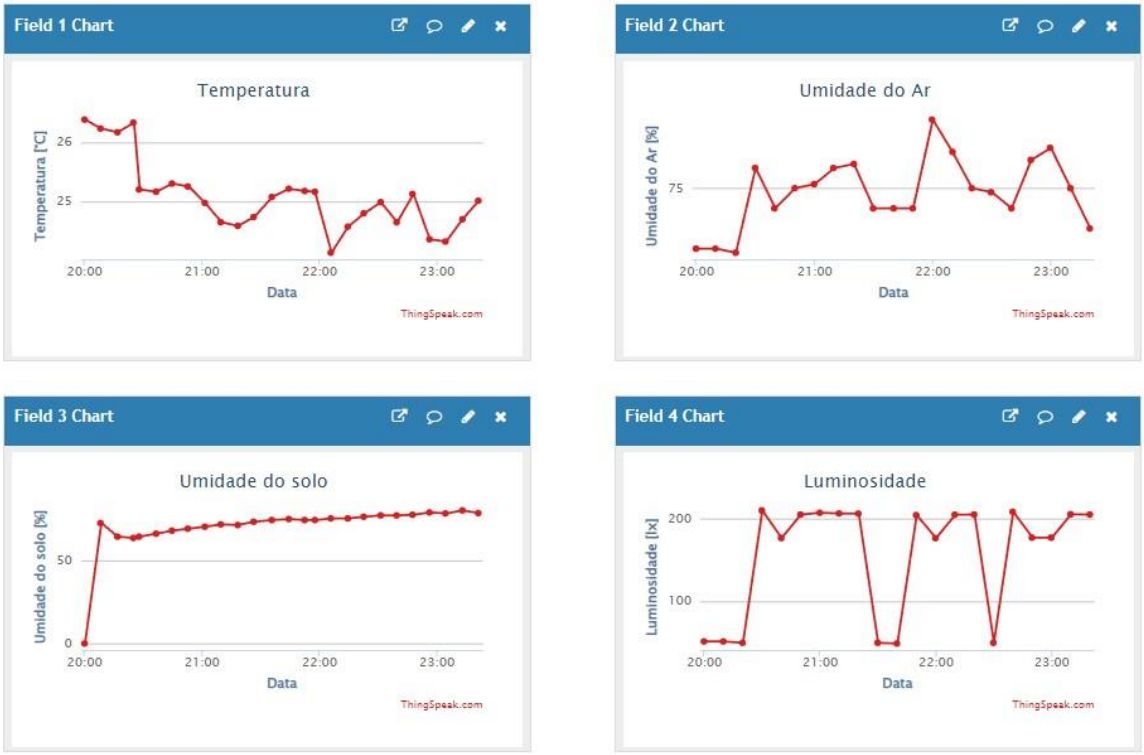
A "ThingSpeak" é definida em seu *website* como sendo uma plataforma de serviço de análise de dispositivos de Internet das Coisas que permite agregar, visualizar e analisar dados em tempo real na nuvem.

A plataforma permite essa comunicação utilizando-se do protocolo MQTT, que funciona no modelo de publicação/assinatura, onde desacopla o remetente (publicador) e o destinatário (assinante), tendo um agente MQTT para intermediar a comunicação, autorizando e autenticando clientes MQTT, recebendo e filtrando mensagens e identificando os clientes que estão inscritos em cada mensagem.

Para a realização do projeto, foram inseridos na plataforma as quatro variáveis monitoradas em função do tempo, temperatura do ar, umidade do ar, iluminância e umidade do solo, enviando dados a cada intervalo de tempo, como pode ser visualizado na Figura 36, e o estado lógico das quatro saídas, a lâmpada, o ventilador, o aquecedor e a bomba d'água.

Foi inserido ainda um indicador luminoso para o estado atual de cada saída, que ao ser ativada, acende a sua respectiva lâmpada. O estado lógico dessas saídas, bem como o indicador luminoso do estado de cada saída podem ser visualizadas na Figura 37.

Figura 36 - Visualização das variáveis monitoradas em função do tempo na plataforma *ThingSpeak*



Fonte: Autoria própria

Figura 37 - Visualização do estado de duas das saídas ao longo do tempo e de seus respectivos indicadores luminosos na plataforma

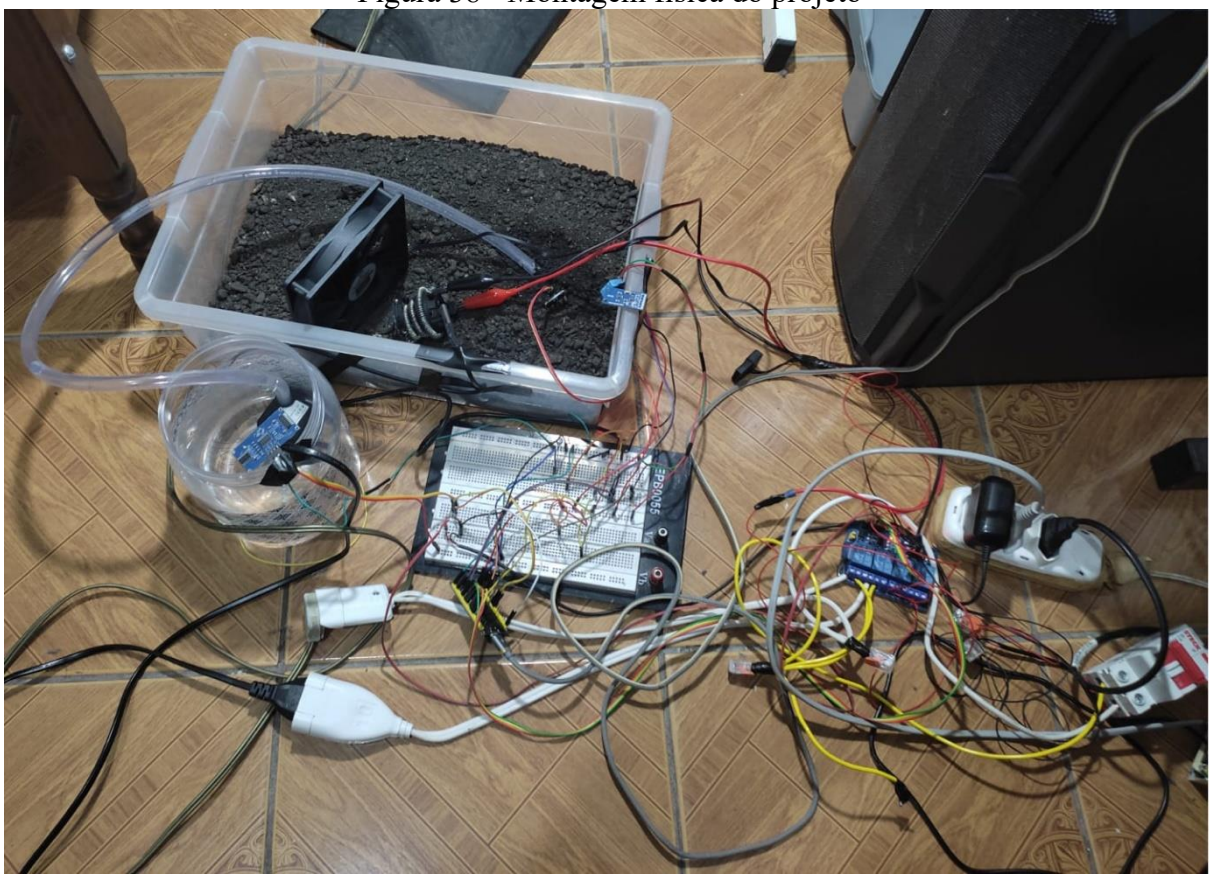
Fonte: Autoria própria

3.3 MONTAGEM

A montagem do projeto, objetivou-se ser mais uma prova de conceito do que a realização de uma estufa em versão reduzida, dando uma ênfase maior para a funcionalidade. O diagrama de blocos da Figura 6, descreve todas as conexões realizadas para a execução do projeto.

A montagem física do projeto, é apresentada na Figura 38, abaixo, foi feita em uma caixa plástica contendo 2 kg de terra, com os sensores posicionados próximos aos atuadores, com exceção da lâmpada. A saída da bomba ficou próxima ao sensor de umidade do solo, e o ventilador e o aquecedor, próximos ao sensor de temperatura primário. Embora a lâmpada tenha saído do enquadramento da fotografia, está posicionado de forma a incidir diretamente sobre o sensor de iluminância.

Figura 38 - Montagem física do projeto



Fonte: Autoria própria

Na Tabela 4 - Lista de material do projeto é listado o material que foi comprado, especificamente para a utilização no projeto; não foram contabilizados os custos com cabos, conectores de emenda, disjuntor, extensões, caixa plástica e *protoboard*.

Tabela 4 - Lista de material do projeto

Material	Quantidade	Preço
Sensor HL-69	1 un.	R$ 6,99
Sensor DHT11	1 un.	R$ 13,90
Sensor DS18B20	1 un.	R$ 14,90
Sensor HC-SR04	1 un.	R$ 15,38
Sensor BH1750	1 un.	R$ 22,68
Módulo de relés	1 un.	R$ 23,99
Lâmpada 9 W	1 un.	R$ 7,71
Bomba d'água submersa	1 un.	R$ 36,29
Resistor 4400 W	1 un.	R$ 14,41
Exaustor 12 V	1 un.	R$ 16,35
ESP32	1 un.	R$ 22,95
Fonte 12 V	1 un.	R$ 24,51
Mangueira	1 m	R$ 5,00
Total		R$ 225,06

Fonte: Autoria própria

No apêndice B, foi desenvolvido um circuito para ser feito em placa de circuito impresso utilizando componentes disponíveis atualmente no mercado. Sendo assim, foi trocado o módulo do ESP32-WROOM-32 pelo ESP32-S3-WROOM-1-N8R8 e adicionada a possibilidade de ser alimentado diretamente na rede elétrica ou por uma fonte de 5V com conector micro-USB.

4 RESULTADOS

Nesse capítulo serão apresentados os resultados de alguns dos testes realizados, ao colocar em funcionamento simultâneo todos sensores e atuadores por um determinado período.

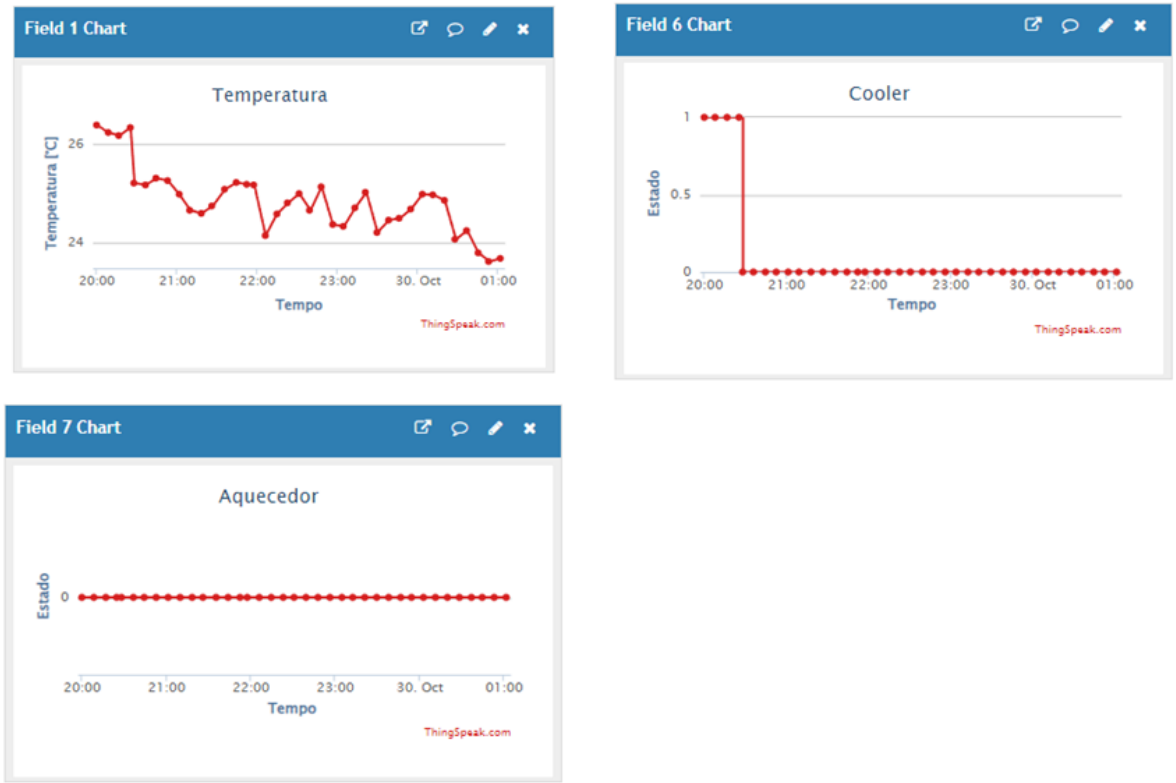
4.1 PRIMEIRO TESTE

As medições foram feitas em ambiente interno, iniciando-se no dia 29/10/2023, às 20:00h, e finalizando no dia 30/10/2023, às 01:00h.

Para esse teste foram configurados os seguintes parâmetros:

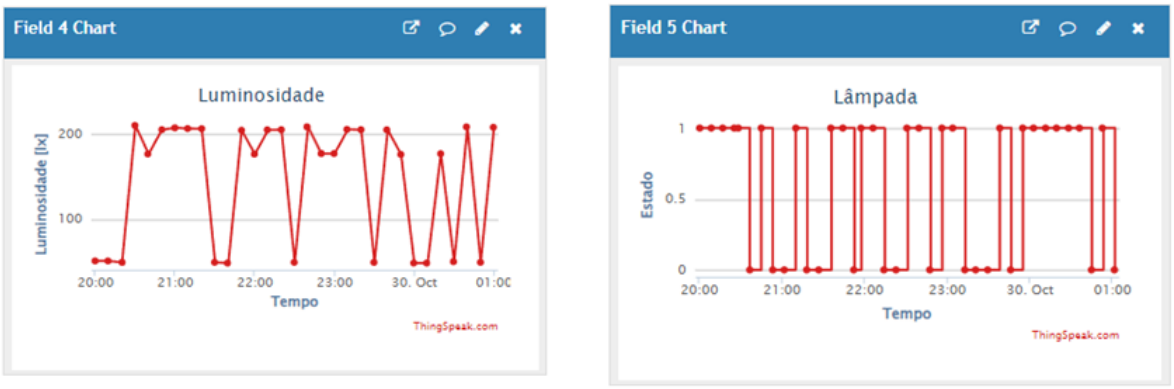
- Limite de iluminância para atuar a lâmpada: 180 lux;
- Limite de temperatura mínima para atuar o cooler: 26 ºC;
- Limite de temperatura máxima para atuar o aquecedor: 20 ºC;
- Limite de umidade do solo máxima para atuar a bomba: 60%;
- Tempo de atualização da plataforma ThingSpeak: 8min 20s (500000 ms);
- Tempo entre leituras dos sensores: 10s (10000 ms).

Figura 39 – Temperatura e estados do ventilador e aquecedor no primeiro teste



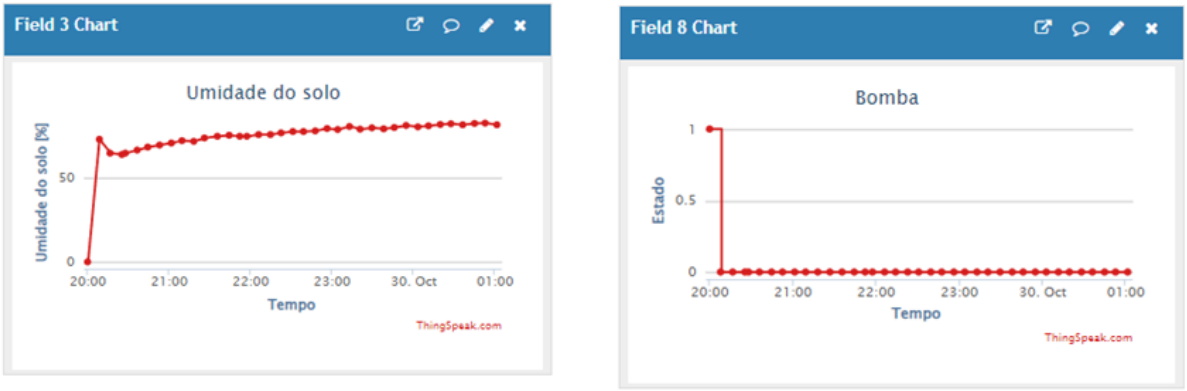
Fonte: Autoria própria

Figura 40 - Iluminância e estado da lâmpada no primeiro teste



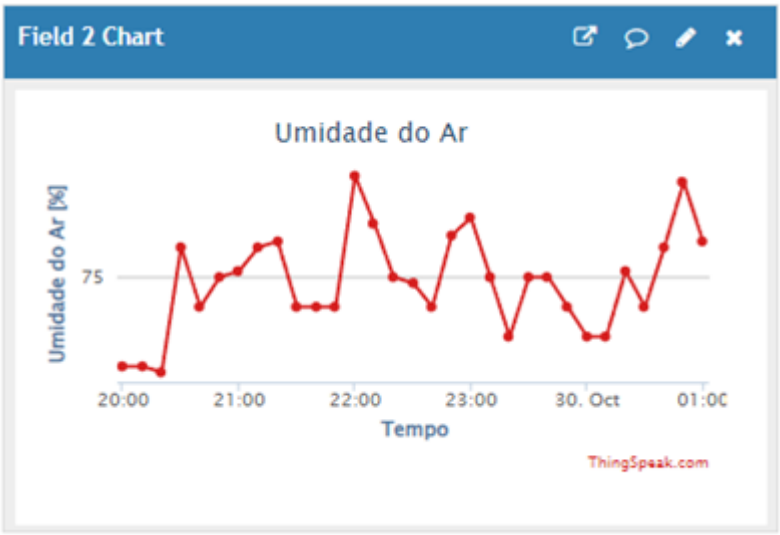
Fonte: Autoria própria

Figura 41 - Umidade do solo e estado da bomba d'água no primeiro teste



Fonte: Autoria própria

Figura 42 – Umidade relativa do ar em relação ao tempo no primeiro teste



Fonte: Autoria própria

Como se pode ver nas figuras 39 e 40, os resultados obtidos no primeiro teste não foram satisfatórios. Na Figura 39, a ação do ventilador não reduziu a temperatura ambiente durante o

tempo que esteve ligado, a queda abrupta na temperatura ambiente ocorreu devido à abertura da janela de onde foi feito o ensaio e não foi reduzida ao ponto de ser necessária a atuação do aquecedor.

O valor esperado de iluminância configurado no firmware foi colocado abaixo do valor em que a lâmpada estava fornecendo, logo a lâmpada não se manteve acesa ao longo de todo o tempo, conforme mostrado na Figura 40. Tal comportamento não seria o ideal em um caso real, onde a planta precisasse da incidência luminosa por um período constante.

A ação da bomba em resposta ao sensor de umidade do solo foi condizente com o esperado, o solo estava inicialmente seco, provocando o ligamento da bomba d'água e assim que a umidade do solo atingiu o valor configura de 60% ela foi desligada, essas ações são mostradas na Figura 41.

O valor de umidade relativa do ar, indicado na Figura 42, é de caráter informativo apenas, não é responsável pela mudança de estado de nenhum atuador do projeto.

4.2 SEGUNDO TESTE

As medições foram feitas em ambiente interno, iniciando-se no dia 30/10/2023, às 20:23h, e foram finalizadas no mesmo dia às 22:13h.

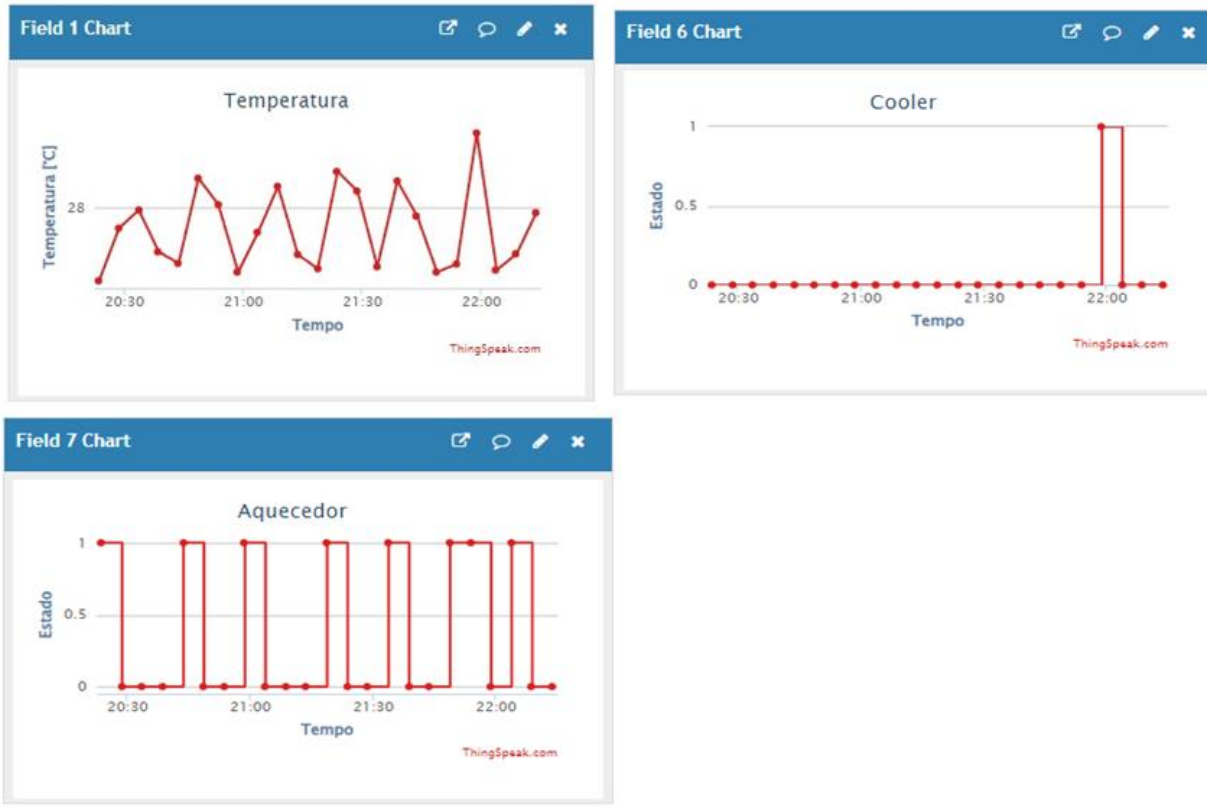
Para uma melhor avaliação do sistema, foram configurados parâmetros diferentes, conforme se apresenta a seguir:

- Limite de iluminância para atuar a lâmpada: 250 lux;
- Limite de temperatura mínima para atuar o cooler: 29 ºC;
- Limite de temperatura máxima para atuar o aquecedor: 27 ºC;
- Limite de umidade do solo máxima para atuar a bomba: 80%;
- Tempo de atualização da plataforma ThingSpeak: 5min (300000 ms);
- Tempo entre leituras dos sensores: 30s (30000 ms);

Além das alterações de *firmware* relatadas acima, também foi retirada toda a água do reservatório e modificado a posição do sensor ultrassônico às 21:30 para pode fazer o seu teste de funcionamento.

A temperatura ambiente no início do teste era de 26,65 °C, então para verificar o correto funcionamento dos atuadores, foi definido um pequeno intervalo de temperatura que se buscava manter o sistema entre 27 °C a 29 °C.

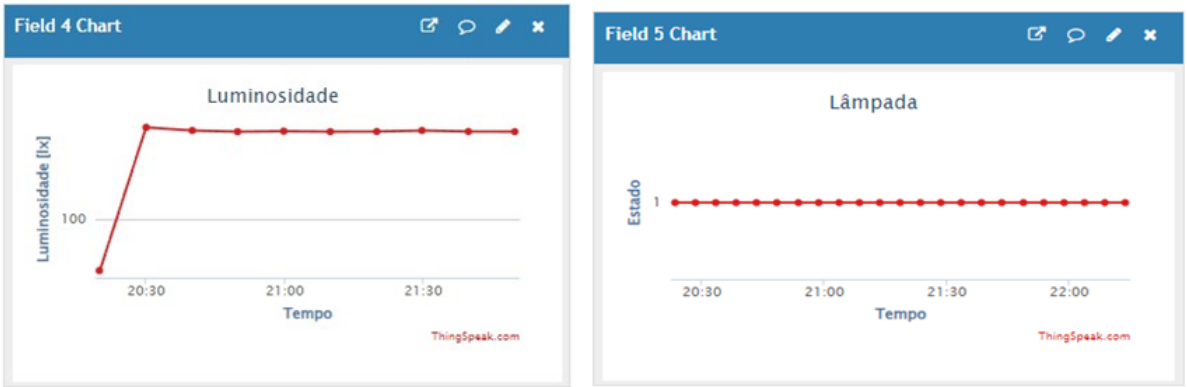
Figura 43 - Temperatura e estados do ventilador e aquecedor no segundo teste



Fonte: Autoria própria

Na Figura 43 pode-se verificar que o sistema conseguiu se manter na faixa de temperatura desejada, com os atuadores conseguindo regular a temperatura, uma vez que ela está totalmente acima da temperatura ambiente. Os dados exibidos na plataforma não estão em sincronia com o acionamento dos relés, porque a atualização de seus estados é feita a cada 5 leituras dos sensores (150s) e o envio de dados para a plataforma é feito a cada 300s.

Figura 44 - Iluminância e estado da lâmpada no segundo teste

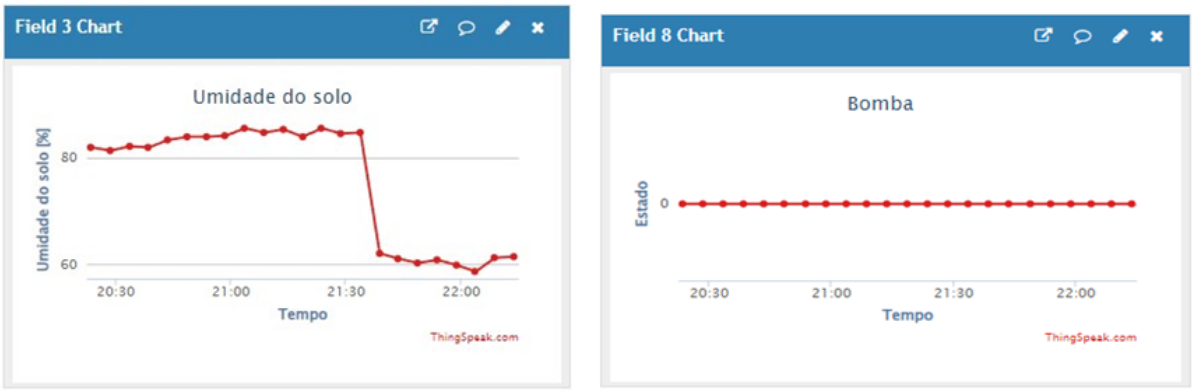


Fonte: Autoria própria

A alteração do limite de iluminância foi para haver um nível iluminação constante sobre o sistema, sendo configurado um pouco acima do que estava sendo medido. Na Figura 44 está indicado que a lâmpada ficou acesa durante todo o período, conforme esperado.

O intervalo menor de tempo de envio dos dados à plataforma, foi para uma coleta de um número maior de pontos e o tempo do intervalo entre leituras dos sensores foi alterado pelo seguinte motivo: cada dado que é enviado para a plataforma é primeiro processado, fazendo-se a média móvel das últimas cinco amostras. Desse modo, cada amostra enviada não sofre interferência de nenhum valor do pacote contendo as amostras anteriores, possibilitando, assim, uma melhor visualização dos efeitos causados pela ação dos atuadores.

Figura 45- Umidade do solo e estado da bomba d'água no segundo teste



Fonte: Autoria própria

Na Figura 45 está mostrada a confirmação do funcionamento do sensor ultrassônico, que fez com que a bomba d´água não fosse acionada com o reservatório de água vazio, prolongando, dessa maneira, a sua vida útil.

Figura 46 – Umidade relativa do ar em relação ao tempo no segundo teste

Fonte: Autoria própria

O valor de umidade relativa do ar, indicado na Figura 46, é de caráter informativo apenas, não é responsável pela mudança de estado de nenhum atuador do projeto.

5 CONCLUSÃO

A automação na agricultura é um fenômeno crescente e irreversível, que traz benefícios tanto para os produtores quanto para os consumidores e para o meio ambiente. Ela permite aumentar a eficiência, a qualidade, a segurança, a sustentabilidade da produção agrícola, promove a redução do trabalho humano, os custos e os riscos, além de facilitar a adaptação e a mitigação das mudanças climáticas, que afetam a agricultura de forma significativa.

A aquisição de dados pelos sensores e a exibição dos dados pela plataforma *ThingSpeak* ocorreram conforme o programado. A utilização da média móvel, na etapa de processamento de dados no *firmware*, foi uma boa solução para mitigar os efeitos causados pelo limite de inserção de entrada de dados máxima permitida pela plataforma, sem perder a precisão nos dados.

O controle da temperatura com recursos limitados, provou-se ser uma tarefa difícil, mas factível quando a temperatura a ser alcançada está superior a ambiente. O controle de iluminância, uma vez que estava em ambiente fechado e iluminado, diferiu do que seria uma aplicação prática, mas cumpriu com o esperado. O controle de umidade do solo utilizou de um sensor de baixo custo, e consequente, de baixa precisão, que inclusive apresentou corrosão ao longo dos testes e precisou ser substituído, mas foi o suficiente para solucionar as ações que eram esperadas, a leitura de um sinal analógico e o controle da bomba d´água, que por sua vez, contou com um sensor ultrassônico, para medir o nível do reservatório, o que foi primordial para evitar a queima do motor assim que o nível abaixasse do limite.

Mesmo com a capacidade limitada do protótipo, foi possível verificar que com os atuadores apropriados, é possível fazer a aplicação de um projeto similar numa estufa de médio porte. Para isso ser possível, no entanto, deve-se fazer primeiramente o estudo das condições necessárias para o tipo do vegetal a ser cultivado e da localização geográfica em que será construída a estufa, bem como o da sua estrutura.

Há a sugestão de dois trabalhos futuros: um visando o acionamento de máquinas elétricas e as instalações elétricas para a atuação numa estufa de tamanho médio e um outro tratando de técnicas de controle multivariáveis numa estufa.

REFERÊNCIAS

AMAZON. **O que é MQTT?**. [s.d]. Disponível em: https://aws.amazon.com/pt/what-is/mqtt/. Acesso em: 27 set. 2023.

ANDRIOLO, J. L. **Olericultura geral**: princípios e técnicas. 2. ed. Santa Maria: UFSM, 2002.

BEZERRA, F. C. **Produção de mudas de hortaliças em ambiente protegido**. Fortaleza: Embrapa Agroindústria Tropical, 2003.

BILHALVA, C. D. *et al*. **Olericultura uma proposta sustentável**: a percepção do agricultor de sistemas de base familiar. Pelotas, v. 1, 2011. Disponível em: https://periodicos.ufpel.edu.br/ojs2/index.php/seur/article/view/5303. Acesso em: 15 ago. 2023

BRITISH COLUMBIA MINISTRY OF AGRICULTURE. **Understanding humidity control in greenhouses**. 2015. Disponível em: https://www2.gov.bc.ca/assets/gov/farming-natural-resources-and-industry/agriculture-and-seafood/animal-and-crops/crop-production/understanding_humidity_control.pdf/. Acesso em: 10 set. 2023.

CURTO CIRCUITO. Disponível em: https://curtocircuito.com.br. Acesso em: 30 set. 2023.

ELETRODEX. Disponível em: https://www.eletrodex.net. Acesso em: 30 set. 2023.

ELETROGATE. Disponível em: https://www.eletrogate.com. Acesso em: 30 set. 2023.

ELETROPECAS. Disponível em: https://www.eletropecas.com. Acesso em: 30 set. 2023.

ESPRESSIF SYSTEMS. **ESP32-WROOM-32**. [s.d]. Disponível em: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf. Acesso em: 17 set. 2023.

HOROWITZ, P.; HILL, W. **The arts of electronics**. 3rd ed. Estados Unidos: Cambridge University Press, 2015.

MAXIM INTEGRATED. **DS18B20:** 1-Wire digital thermometer. [s.d]. Disponível em: https://www.analog.com/media/en/technical-documentation/data-sheets/ds18b20.pdf. Acesso em: 15 set. 2023.

MAZOYER, M.; ROUDART, L. **Histórias das agriculturas no mundo:** do neolítico à crise contemporânea. São Paulo: Editora UNESP; Brasília, DF: NEAD, 2010.

MELO, P. C. T.; ARAÚJO, T. H. **Olericultura**: planejamento da produção, do plantio à comercialização. Curitiba: SENAR - Pr., 2016.

MOUSER. **DHT11**: Humidity & temperature sensor. [s.d]. Disponível em: https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf. Acesso em: 19 set. 2023.

NEMALI, K. **Temperature control in greenhouses**. Purdue Education Store, [*S.l*], 2021. Disponível em: https://www.extension.purdue.edu/extmedia/HO/HO-327-W.pdf 1. Acesso em: 15 ago. 2023.

OSEPP ELECTRONICS. **HC-SR04**: Ultrasonic Sensor Module. [s.d]. Disponível em: https://www.alldatasheet.com/datasheet-pdf/pdf/1132203/ETC2/HC-SR04.html . Acesso em: 01 out. 2023.

PAULILO, M. T. S. *et al*. **Fisiologia vegetal**. Florianópolis: Universidade Federal de Santa Catarina, 2015.

RODRIGUES, R. A. S. **Olericultura**. Londrina: Educacional S.A, 2019.

ROHM SEMICONDUCTOR. **BH1750FVI:** digital 16bit serial output type ambient light sensor IC. [s.d]. Disponível em: https://www.mouser.com/datasheet/2/348/Rohm_11162017_ROHMS34826-1-1279292.pdf. Acesso em: 17 set. 2023.

SEEMANN, J. *et al*. **Agrometeorology**. Berlin, Heidelberg: Springer, 1979.

TANENBAUM, A.; BOS, H. **Sistemas operacionais modernos**. 4. ed. São Paulo: Pearson, 2015.

TERABYTESHOP. Disponível em: https://www.terabyteshop.com.br/. Acesso em: 14 set. 2023.

VALVANO, J. **Embedded systems:** introduction to Arm® Cortex™-M Microcontrollers. 5th ed. Estados Unidos: CreateSpace, 2012.

VAMUINO. Disponível em: https://lojinha.vamuino.com.br. Acesso em: 14 set. 2023.

APÊNDICE A – Firmware do projeto

```
// Firmware do TG
// Autor: Rodrigo Guerra

#include <Arduino.h> // Biblioteca IDE do Arduino
#include "freertos/FreeRTOS.h" // Biblioteca FreeRTOS
#include "freertos/task.h" // Biblioteca de Task do FreeRTOS

// Biblioteca sensores
#include <OneWire.h>
#include <DallasTemperature.h>
#include <BH1750.h>
#include <Wire.h>
#include "DHT.h"
#include <WiFi.h>

// Biblioteca ThingSpeak
#include <ThingSpeak.h>

// GPIO DHT11
#define DHTPIN 15

// Definir tipo do sensor de umidade/temperatura entre DHT11 e DHT22
#define DHTTYPE DHT11

// GPIO DS18B20
static int oneWireBus = 4;

//GPIO HC_SR04
const int trigPin = 26;
const int echoPin = 27;

// ADC pin sensor de umidade do solo
```

```
#define soilSensor 34

//0% e 100% do sensor de umidade do solo
#define lowSoilHumidity 1489
#define highSoilHumidity 3723

//Parametros sensor ultrassonico
#define SOUND_SPEED 0.034
#define levelLimitPump 10.0

//GPIOs saída
#define OUTPUTLIGHT 19 //IN4 relé Luz
#define OUTPUTHEATER 18 //IN3 resistencia
#define OUTPUTCOOLERS 17 //IN2 relé coolers
#define OUTPUTPUMP 16 //IN1 relé bomba

#define TEMPOSENSOR 30000 //Tempo entre leituras dos sensores
#define TEMPOSENSORNIVEL 400 //Tempo de leitura do sensor de nível de água para atuar proteção
#define TEMPOTS 300000 //Tempo de atualização ThingSpeak - 5min

//Variaveis globais dos sensores
static float temperatureC = 0.0; //Temperatura DS18B20
static float lux = 0.0; //Luminosidade
static float humidity = 0; //Umidade do ar DHT11
static float temperature2 = 0; //Temperatura DHT11
static int  moistureValue = 0; //Umidade do solo
long duration; //Sensor de nível
float distanceCm; //Sensor de nível

//Limites das variáveis dos sensores
static float luxLimit = 250.0; //Limite  de luminosidade para atuar a lâmpada
static float tempLimitSup = 29.0; //Limite de temperatura para atuar o cooler
static float tempLimitInf = 27; //Limite de temperatura para atuar o aquecedor
```

```
static float moistureLimit = 80; //Limite de umidade para atuar a bomba d'água

static bool flagErroSensorTemp = 0; //Flag de redundancia do sensor de temperatura

//Flags para indicação das saídas na plataforma ThingSpeak
static bool outLight = 0;
static bool outHeater = 0;
static bool outPump = 0;
static bool outCooler = 0;

//Variáveis da Média móvel
#define NUM_VARIAVEIS 5   // Número de variáveis
#define TAM_JANELA 5      // Tamanho da janela da média móvel
float valores[NUM_VARIAVEIS][TAM_JANELA] = {0};  // Matriz para armazenar os valores
float medias[NUM_VARIAVEIS] = {0};            // Array para armazenar as médias
float novosValores[NUM_VARIAVEIS] = {0};
int indice = 0;                           // Índice atual da janela
int contador = 0;
static bool mediaOk = 0;

#define TEMPERATURA 0
#define LUMINOSIDADE 1
#define TEMPERATURADHT11 2
#define UMIDADEAR 3
#define UMIDADESOLO 4

//Handles das tasks FREERTOS
TaskHandle_t xTaskTemperaturaHandle = NULL;
TaskHandle_t xTaskLuminosity = NULL;
TaskHandle_t xTaskHumidity = NULL;
TaskHandle_t xTaskSoilHumidity = NULL;
TaskHandle_t xTaskCheckSensors = NULL;
TaskHandle_t xTaskMediaMovel = NULL;
TaskHandle_t xTaskNivel = NULL;
```

```
TaskHandle_t xTaskThingSpeak = NULL;

// Protótipos das funções
void vTaskTemperatura(void *pvParameters);
void vTaskLuminosity(void *pvParameters);
void vTaskHumidity(void *pvParameters);
void vTaskSoilHumidity(void *pvParameters);
void vTaskCheckSensors(void * pvParameters);
void vTaskMediaMovel(void *pvParameters);
void vTaskNivel(void *pvParameters);
void vTaskThingSpeak(void *pvParameters);
void DS18B20_readings();
void BH1750_readings();
void DHT11_readings();
void soilHumidity_readings();
void level_readings();
void checkSensors();
void atualizaMedia();
void calculoMedia();
void thingSpeak();

const char *ssid = "********";       // Nome da rede Wi-Fi
const char *pass =  "********";   //Senha da rede Wi-Fi
const char* server = "api.thingspeak.com";
unsigned long myChannelNumber = *******;  //Canal do Thingspeak
const char *myWriteAPIKey = "****************"; //Chave da API do Thingspeak
WiFiClient client;

// Setup a oneWire instance to communicate with any OneWire devices
OneWire oneWire(oneWireBus);
// Pass our oneWire reference to Dallas Temperature sensor
DallasTemperature sensors(&oneWire);
BH1750 lightMeter;
DHT dht(DHTPIN, DHTTYPE);
```

```
void setup() {
  Serial.begin(115200); //baud rate da comunicação serial 115200
  sensors.begin(); //start do sensor DS18B20
  sensors.setResolution(12); //configuração DS18B20 12 bits de resolução
  Wire.begin(); //start i2c
  lightMeter.begin(); //start sensor BH1750
  dht.begin(); //start sensor DHT11
  analogReadResolution(12); // Configuração da resolução do adc do ESP32 para 12 bits

  pinMode(OUTPUTLIGHT, OUTPUT); //Saída Luz 19
  pinMode(OUTPUTHEATER, OUTPUT); // Saída resistencia 18
  pinMode(OUTPUTCOOLERS, OUTPUT); // Saída coolers 17
  pinMode(OUTPUTPUMP, OUTPUT); //Saída bomba 16
  pinMode(trigPin, OUTPUT); //  Trig como saída - sensor ultrassonico
  pinMode(echoPin, INPUT); // Echopin como entrada - sensor ultrassonico

  //Inicialização das saídas desligadas
  digitalWrite(OUTPUTPUMP, !LOW);
  digitalWrite(OUTPUTCOOLERS, !LOW);
  digitalWrite(OUTPUTHEATER, !LOW);
  digitalWrite(OUTPUTLIGHT, !LOW);

 /*Criação das tasks
  Tasks de aquisição de dados pelos sensores na CPU 0
  Tasks de processamento de dados na CPU 1
*/
  xTaskCreatePinnedToCore(vTaskTemperatura, "Task
Temperatura",configMINIMAL_STACK_SIZE + 5000, NULL, 5,
&xTaskTemperaturaHandle,0);
  xTaskCreatePinnedToCore(vTaskLuminosity,"Task
Luminosidade",configMINIMAL_STACK_SIZE + 5000, NULL, 4, &xTaskLuminosity,0);
  xTaskCreatePinnedToCore(vTaskHumidity,"Task
umidade",configMINIMAL_STACK_SIZE + 5000, NULL, 3, &xTaskHumidity,0);
```

```
 xTaskCreatePinnedToCore(vTaskSoilHumidity, "Task Umidade do solo", configMINIMAL_STACK_SIZE + 5000, NULL, 2, &xTaskSoilHumidity,0);
 xTaskCreatePinnedToCore(vTaskNivel, "Task Nivel Agua", configMINIMAL_STACK_SIZE + 5000, NULL, 10, &xTaskSoilHumidity,0);
 xTaskCreatePinnedToCore(vTaskCheckSensors,"Task Check Sensors",configMINIMAL_STACK_SIZE + 5000, NULL, 10, &xTaskCheckSensors,1);
 xTaskCreatePinnedToCore(vTaskMediaMovel,"Task Média Móvel",configMINIMAL_STACK_SIZE + 5000, NULL, 20, &xTaskMediaMovel,1);
 xTaskCreatePinnedToCore(vTaskThingSpeak,"Task Thing Speak",configMINIMAL_STACK_SIZE + 5000, NULL, 1, &xTaskThingSpeak,1);

 WiFi.begin(ssid, pass);

    while (WiFi.status() != WL_CONNECTED)
    {
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.println("WiFi connected");
    delay(500);
    ThingSpeak.begin(client);
}
void loop() {
}

//Task temperatura DS18B20
void vTaskTemperatura(void *pvParameters){
 while(1){
  DS18B20_readings();
  vTaskDelay(pdMS_TO_TICKS(TEMPOSENSOR));
  }
}
```

```
//Task luminosidade BH1750
 void vTaskLuminosity(void *pvParameters){
  while(1){
   BH1750_readings();
   vTaskDelay(pdMS_TO_TICKS(TEMPOSENSOR));
  }
}

//Task umidade ambiente DHT11
void vTaskHumidity(void *pvParameters){
  while(1){
   DHT11_readings();
   vTaskDelay(pdMS_TO_TICKS(TEMPOSENSOR));
  }
}

//Task umidade do solo
void vTaskSoilHumidity(void *pvParameters){
  while(1){
   soilHumidity_readings();
   vTaskDelay(pdMS_TO_TICKS(TEMPOSENSOR));
  }
}

//Task Nivel da Agua
void vTaskNivel(void *pvParameters){
  while(1){
   level_readings();
   vTaskDelay(pdMS_TO_TICKS(TEMPOSENSORNIVEL));
  }
}

//Task média móvel
 void vTaskMediaMovel(void *pvParameters){
```

```
  while(1){
  vTaskDelay(pdMS_TO_TICKS(5000));
   atualizaMedia();
   vTaskDelay(pdMS_TO_TICKS(TEMPOSENSOR + 100));
   if (contador == TAM_JANELA){
    calculoMedia();
    contador = 0;
    xTaskNotifyGive(xTaskCheckSensors);
   }
  }
}

//Task processamento dos dados
 void vTaskCheckSensors(void * pvParameters){
  while(1){
   ulTaskNotifyTake(pdTRUE, portMAX_DELAY);//Espera o cálculo das médias dos valores
dos sensores para atualizar as saídas
   checkSensors();
   xTaskNotifyGive(xTaskThingSpeak);
  }
}

void vTaskThingSpeak(void * pvParameters){
  while(1)
  {
  ulTaskNotifyTake(pdTRUE, portMAX_DELAY);
  thingSpeak();
  }
}

 //Função de leitura da temperatura ambiente DS18B20
 void DS18B20_readings()
{
  sensors.requestTemperatures();
```

```
  temperatureC = sensors.getTempCByIndex(0);
  Serial.print("Temperatura: ");
  Serial.print(temperatureC);
  Serial.println("ºC");
  novosValores[0] = temperatureC;
}

//Função de leitura da luminosidade BH1750
void BH1750_readings(){
  lux = lightMeter.readLightLevel();
  Serial.print("Luminosidade: ");
  Serial.print(lux);
  Serial.println(" lx");
  novosValores[1] = lux;
}

//Função de leitura umidade ambiente e temperatura - DHT11
void DHT11_readings(){
  temperature2 = dht.readTemperature();
  humidity = dht.readHumidity();
  /**/  if (isnan(humidity) || isnan(temperature2)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;}
  Serial.print("Umidade: ");
  Serial.print(humidity);
  Serial.println(" %");
  Serial.print("Temperatura DHT11: ");
  Serial.print(temperature2);
  Serial.println(" ºC");

  novosValores[2] = temperature2;
  novosValores[3] = humidity;
}
```

```
//Função de leitura da umidade do solo
void soilHumidity_readings(){
  int sensorValue = 0;
  sensorValue = analogRead(soilSensor);
   int sensorValueConstrain = constrain(sensorValue,lowSoilHumidity,highSoilHumidity);
  int sensorvalueNormalized =
map(sensorValueConstrain,lowSoilHumidity,highSoilHumidity,0,100);
  moistureValue = 100 - sensorvalueNormalized;
  Serial.print("Umidade do solo: ");
  Serial.print(moistureValue);
  Serial.println(" %");
  novosValores[4] = moistureValue;
  Serial.print("Distancia tanque: ");
  Serial.println(distanceCm);
}

//-------------Função do processamento dos sensores e controle das saídas----------
void checkSensors(){

  //Luminosidade
  if(medias[LUMINOSIDADE] > luxLimit){
    digitalWrite(OUTPUTLIGHT, !LOW);
    outLight = 0;
  }
  else if(medias[LUMINOSIDADE]<=luxLimit){
    digitalWrite(OUTPUTLIGHT, !HIGH);
    outLight = 1;
  }

  //Temperatura ambiente
  if(medias[TEMPERATURA] <= -100){
    flagErroSensorTemp = 1; //Quando há perda de comunicação com sensor DS18B20, ele é
ignorado e será considerado a temperatura obtida pelo DHT11
    medias[TEMPERATURA] = medias[TEMPERATURADHT11];
```

```
}
else{
  flagErroSensorTemp = 0;
}

if(medias[TEMPERATURA] >= tempLimitSup){ //
  digitalWrite(OUTPUTCOOLERS, !HIGH);
  outCooler = 1;
  digitalWrite(OUTPUTHEATER, !LOW);
  outHeater = 0;
}
else if(medias[TEMPERATURA] <= tempLimitInf){ //
  outCooler = 0;
  digitalWrite(OUTPUTHEATER, !HIGH);
  outHeater = 1;
}
else{
  digitalWrite(OUTPUTCOOLERS, !LOW);
  outCooler = 0;
  digitalWrite(OUTPUTHEATER, !LOW);
  outHeater = 0;
}

//Bomba
if(medias[UMIDADESOLO] < moistureLimit && distanceCm < levelLimitPump){ //
  digitalWrite(OUTPUTPUMP, !HIGH);
  outPump = 1;
}
else{
  digitalWrite(OUTPUTPUMP, !LOW);
  outPump = 0;
}

//Print serial para debug
```

```
  Serial.println("Saídas: ");
  Serial.print("Lampada: ");
  if(outLight==1)
    Serial.println("Ligado ");
    else
    Serial.println("Desligado ");
  Serial.print("Aquecedor: ");
    if(outHeater==1)
    Serial.println("Ligado ");
    else
    Serial.println("Desligado ");
  Serial.print("Cooler: ");
      if(outCooler==1)
    Serial.println("Ligado ");
    else
    Serial.println("Desligado ");
  Serial.print("Bomba: ");
      if(outPump==1)
    Serial.println("Ligado ");
    else
    Serial.println("Desligado ");
}
//------------Fim processamento e controle da saída----------

void atualizaMedia(){

        for (int i = 0; i < NUM_VARIAVEIS; i++) {
          for (int j = TAM_JANELA - 1; j > 0; j--) {
             valores[i][j] = valores[i][j - 1];
          }
          valores[i][0] = novosValores[i];
       }
       contador++;
}
```

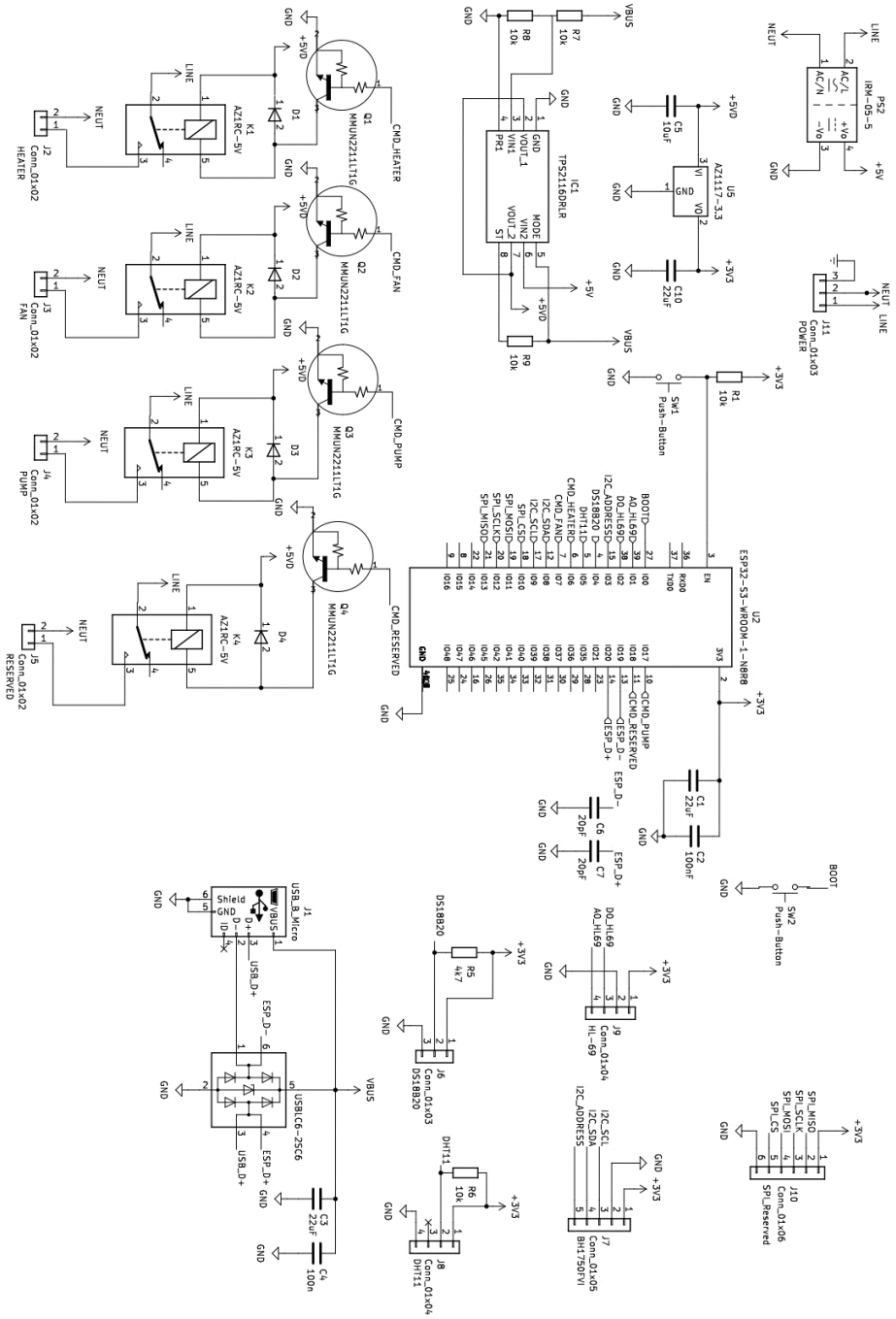
```
void calculoMedia(){

   for (int i = 0; i < NUM_VARIAVEIS; i++) {
         double soma = 0.0;
         for (int j = 0; j < TAM_JANELA; j++) {
            soma += valores[i][j];
         }
         medias[i] = soma / TAM_JANELA;

   //Print serial para debug
     Serial.println(" ");
     Serial.println("Medias");
     Serial.print("Temperatura DS1820: ");
     Serial.println(medias[0]);
     Serial.print("Luminosidade: ");
     Serial.println(medias[1]);
     Serial.print("Temperatura DHT11: ");
     Serial.println(medias[2]);
     Serial.print("Umidade do Ar: ");
     Serial.println(medias[3]);
     Serial.print("Umidade do solo: ");
     Serial.println(medias[4]);
  }
}

void level_readings(){

  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);

  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
```

```
  duration = pulseIn(echoPin, HIGH);

  distanceCm = duration * SOUND_SPEED/2;

  if(distanceCm >= levelLimitPump){ //Para a bomba assim que detectar o nível de água perto da altura da bomba
    digitalWrite(OUTPUTPUMP, !LOW);
    outPump = 0;
  }
}

void thingSpeak(){
  //Temperatura DS1820[0] - Luminosidade[1] - Temperatura DHT11[2] - Umidade DHT11[3] - Umidade do Solo[4]
  ThingSpeak.setField(1, medias[TEMPERATURA]);
  ThingSpeak.setField(2, medias[UMIDADEAR]);
  ThingSpeak.setField(3, medias[UMIDADESOLO]);
  ThingSpeak.setField(4, medias[LUMINOSIDADE]);
  ThingSpeak.setField(5, outLight);
  ThingSpeak.setField(6, outCooler);
  ThingSpeak.setField(7, outHeater);
  ThingSpeak.setField(8, outPump);
  ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);
  delay(TEMPOTS);
}
```

APÊNDICE B – Circuito Esquemático

Na Figura 47, está apresentado o circuito esquemático proposto, utilizando o software KiCad 7.0.

Figura 47: Circuito esquemático proposto



Fonte: Autoria própria

Figura 48: Relação de componentes importados

Referência	Código do fabricante	Fabricante	Qtd.	Preço ($)	Preço Total ($)
C1, C3, C10	GRM187R61A226ME15D	Murata Electronics	3	0,32	0,96
C2, C4	C0603C104K8RAC	KEMET	2	0,11	0,22
C5, C5	GRM188C81A106MA73D	Murata Electronics	2	0,24	0,48
C6, C7	CL10C200JB8NNNC	Samsung	2	0,1	0,2
R1, R6, R7, R8, R9	RT0603DRE1310KL	YAGEO	5	0,1	0,5
R5	RT0603FRE134K75L	YAGEO	1	0,1	0,1
D1, D2, D3, D4	1N4148W-E3-18	Vishay Semiconductors	4	0,16	0,64
U1	USBLC6-2SC6	STMicroelectronics	1	0,61	0,61
U2	ESP32-S3-WROOM-1-N8R8	Espressif Systems	1	3,62	3,62
SW1, SW2	TS02-66-70-BK-100-SCR-D	CUI Devices	2	0,1	0,2
IC1	TPS2116DRLR	Texas Instruments	1	0,87	0,87
PS2	IRM-05-5	MEAN WELL	1	7,69	7,69
Q1, Q2, Q3, Q4	MMUN2211LT1G	onsemi	4	0,12	0,48
					16,57

Fonte: Mouser

Figura 49: Relação de componentes do mercado interno

Referência	Código do fabricante	Fabricante	Qtd.	Preço (R$)	Preço Total (R$)
J11	BR903V	Metaltex	1	8,04	8,04
J2, J3, J4, J5	BR902V	Metaltex	4	5,3	21,2
K1, K2, K3, K4	AZ1RC-5V	Metaltex	4	5,01	20,04
J1,J6,J7,J8,J9,10	Barra de pinos 01x40 - 2,54mm - 180°		1	1,2	1,2
					50,48

Fonte: Eletrodex, Eletropecas e Curtocircuito

As figuras 48 e 49 trazem os preços dos componentes importados e não-importados, respectivamente. Considerando o valor dólar do dia 30/11/2023 de R$4,93, excluindo os sensores e atuadores que já foram mencionados os preços e as taxas de importação, temos o valor de R$132,17 em componentes eletrônicos para a montagem do projeto.