UNIVERSIDADE DO SUL DE SANTA CATARINA

EDUARDO THIESEN

SISTEMA AUTOMATIZADO PARA IRRIGAÇÃO EM HIDROPONIA COM TERRA

Palhoça

2020

EDUARDO THIESEN

SISTEMA AUTOMATIZADO PARA IRRIGAÇÃO EM HIDROPONIA COM TERRA

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia Elétrica da Universidade do Sul de Santa Catarina como requisito parcial à obtenção do título de bacharel em Engenharia Elétrica.

Orientador: Prof. Djan de Almeida do Rosário, Esp.

Palhoça

2020

EDUARDO THIESEN

SISTEMA AUTOMATIZADO PARA IRRIGAÇÃO EM HIDROPONIA COM TERRA

Este Trabalho de Conclusão de Curso foi julgado adequado à obtenção do título de Bacharel e aprovado em sua forma final pelo Curso de Engenharia Elétrica da Universidade do Sul de Santa Catarina.

Palhoça, 24 de Novembro de 2020.

_____

Professor e orientador Djan de Almeida do Rosario, Esp.
Universidade do Sul de Santa Catarina

_____

Professor Anderson Soares André, Dr.
Universidade do Sul de Santa Catarina

_____

Professor Jairo Afonso Henkes, Ms.
Universidade do Sul de Santa Catarina

Eu, Eduardo Thiesen, dedico este trabalho aos meus pais, Valdecir Thiesen e Margareth Scheidt Thiesen, as minhas irmãs Karina Thiesen e Letícia Thiesen, e à minha namorada Pâmela Cristina Nunes da Silva.

AGRADECIMENTOS

Agradeço primeiramente a Deus, que sempre me iluminou, me concedeu saúde e abençoou para que eu pudesse superar todas as dificuldades durante toda essa caminhada.

Gostaria de dirigir os meus sinceros agradecimentos aos professores e em especial ao meu orientador Professor Djan de Almeida do Rosário, pelo apoio e disponibilidade prestados durante toda a graduação e para a concretização deste trabalho.

Um agradecimento final à minha namorada e à minha família que sempre estiveram ao meu lado demonstrando seu companheirismo e incentivo para a realização desse sonho.

“O sucesso vem como resultado do desenvolvimento de nosso potencial”. (JOHN MAXWELL).

RESUMO

A agricultura familiar é uma das principais fontes de produção de alimentos do Brasil, gerando cerca de 10 milhões de empregos. Contudo enfrentam dificuldades na produção pois a tecnologia para um aperfeiçoamento da plantação ainda tem um custo muito elevado, onde muitos agricultores possuem dificuldades em adquiri-las. Tendo em vista, este protótipo sugere o desenvolvimento de um sistema de irrigação automatizada no cultivo de morango a um custo acessível para os agricultores, com o intuito de reduzir a mão de obra, na tentativa de melhorar o controle de pragas e doenças. O protótipo foi composto por três modos de operação: o modo manual, semiautomático e automático. No modo manual o operador tem que realizar manualmente a abertura da válvula, no semiautomático define quantos minutos a irrigação ficará ligada por meio de um teclado e no automático a irrigação obedece aos comandos do sensor de umidade e temperatura. No protótipo foi utilizado o microcontrolador Arduino para o controle dos modos operantes.

Palavras-chave: Morango. Irrigação. Automatizado.

ABSTRACT

Family farming is one of the main sources of food production in Brazil, generating about 10 million jobs. However, they face difficulties in production because the technology for improving the plantation still has a very high cost, where many farmers have difficulties in acquiring them. In view, this prototype suggests the development of an automated irrigation system in strawberry cultivation at an affordable cost for farmers, in order to reduce labor, in an attempt to improve pest and disease control. The prototype consist of three modes of operation: manual, semi-automatic and automatic. In manual mode the operator have to manually open the valve, in semiautomatic it defines how many minutes the irrigation is turned on by means of a keyboard and in the automatic irrigation will obey the commands of the humidity and temperature sensor. The prototype use the Arduino microcontroller to control the operating modes.

Keywords: Strawberry. Irrigation. Automatizado.

LISTA DE ILUSTRAÇÕES

LISTA DE GRÁFICOS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Na América do Sul, são estimadas 318 toneladas de morangos em 11.884 hectares, sendo Brasil, Argentina e Chile a maior parte da produção, nos últimos dez anos, esses países não apenas aumentaram a área cultivada, mas também a utilização de novas tecnologias, que aumentaram a produção e a qualidade da fruta (ANTUNES; FAGHERAZZI; VIGNOLO, 2017). “O morangueiro é cultivado, no Brasil, em várias formas: no solo, com ou sem cobertura plástica, em túneis baixos ou em estufas, ou no sistema hidropônico, com ou sem substrato. O sistema hidropônico conduzido em substrato é conhecido no país como semi-hidropônico” (BORTOLOZZO, 2007. p. 1).

O amplo conhecimento no qual o setor de agricultura vem abordando e adquirindo no passar dos anos, beneficia os agricultores com a melhora na produtividade e a sustentabilidade, a informação e o conhecimento desenvolvem um papel essencial, promovendo e adotando inovações e tecnologias mais eficientes e de extrema importância (HU; ZHANG; DUAN, 2016).“A agricultura digital é a prática de tecnologias modernas, como sensores, robótica e análise de dados, para mudar de operações tediosas para processos continuamente automatizados” (SHAMSHIRI, *et. al*., 2018. p. 1).

A irrigação por gotejamento é usada no cultivo protegido de morangos semi-hidropônicos em um substrato artificial. A irrigação localizada oferece as seguintes vantagens: alta eficiência de aplicação, economia de água, energia e mão-de-obra, automação e fertilização e comprometimento do tratamento de proteção de culturas, este sistema aplica água diretamente na região da raiz (BORTOLOZZO, 2007).

Os sistemas embarcados, como por exemplo o Arduino, sofreram mudanças significativas nos últimos anos. Eles eram considerados sistemas complexos e agora estão inseridos em praticamente todos os dispositivos eletrônicos e ajudando no dia a dia das pessoas. Diferente dos computadores convencionais, que executam vários aplicativos ao mesmo tempo, os softwares embarcados são projetados para executar um trabalho específico em uma aplicação. O Arduino possui um microcontrolador que pode desenvolver diversos aplicativos de controle, automação e interatividade, permitindo que usuários comuns criem seus próprios projetos (CUNHA; ROCHA, 2015).

No decorrer deste projeto, o Arduino será utilizado como controlador para ativar os comandos de irrigação na lavoura de morango. O teor de umidade do substrato é aferido utilizando um sensor de umidade e, logo em seguida, é feita uma comparação entre a umidade aferida e a umidade ideal para o cultivo dele. Caso não for encontrada a umidade ideal, será

ativada a bomba que realizará a irrigação deste substrato, assim, alcançando a umidade desejada. Objetivando o aperfeiçoamento e a economia hídrica no cultivo morangueiro.

1.1 PROBLEMA DE PESQUISA

Este projeto de pesquisa questiona se o sistema automatizado da irrigação de plantações de morango, em estrutura semi-hidropônicos, baseado em plataforma Arduino, funciona adequadamente, e se alcançará a umidade ideal para a produção dessa fruta.

1.2 JUSTIFICATIVA

A automação trará inúmeros benefícios ao produtor, podendo aumentar sua produtividade, pois respectivamente quando automatizado o afazer, o produtor terá disponibilidade para exercer outras funções em sua produção. Proporcionando também, ao produtor agilidade, melhor controle da produção e aumento da qualidade do fruto.

O desenvolvimento de um sistema de irrigação automatizado usando a plataforma Arduino será benéfico, pois, além da facilidade de uso e acesso a essa tecnologia, também permitirá o desenvolvimento de um sistema preciso que oferecerá aos agricultores um sistema fidedigno.

A irrigação moderna é bem avançada e possui diferentes tipos de automação. No entanto, pequenos e médios agricultores nem sempre têm acesso total a essas tecnologias, devido a problemas financeiros ou falta de conhecimento (GUIMARÃES, 2011). Pretende-se desenvolver um sistema de baixo custo que seja intuitivo para facilitar a operação aos agricultores, e com facilidade de aquisição.

1.3 OBJETIVOS

1.3.1 Objetivo geral

Elaborar um protótipo automatizado empregando um sistema microcontrolado, para monitorar e controlar a umidade do substrato.

1.3.2 Objetivos específicos

Os objetivos específicos identificam aspectos significativos no desenvolvimento do projeto, listando-se em:

- Desenvolver um *hardware* eletrônico para execução de comandos de controle de irrigação utilizando microcontrolador Arduino;
- Elaborar um *Software* baseado em linguagem C para utilização do controle do Arduino;
- Projetar um circuito capaz de realizar aferições da umidade do substrato e temperatura do ar para promover os controles do microcontrolador;
- Utilizar uma Interface Homem-Máquina (IHM) para o monitoramento do sistema automático, bem como permitir a operação em modo semiautomático e manual;
- Apresentar ao produtor rural que a automação trará maior comodidade, tranquilidade e confiança na realização de seu negócio;
- Permitir a produção de frutos de alta qualidade, evitando o excesso ou insuficiência hídrica que danificam à produção;

2 FUNDAMENTAÇÃO TEÓRICA

2.1 AGRICULTURA BRASILEIRA

A agricultura brasileira, em meados da década de 1960 até o final da década de 1980, passou por um forte processo de transformações, obtendo um grande desenvolvimento, alterando suas fontes de crescimento. Durante o período de transformações, as produtividades da terra e do trabalho, passaram a fazer parte da dinâmica de crescimento deste setor (CONCEIÇÃO; CONCEIÇÃO, 2014).

Após o processo de transformação, a agricultura ainda vem passando por muitas evoluções, principalmente na questão econômica para o país. A revolução agrícola dos últimos 40 anos é certamente o fato mais importante da história econômica recente do Brasil e continua abrindo perspectivas para o desenvolvimento futuro do país. O agronegócio tem sido reconhecido como um vetor importante do crescimento econômico brasileiro. Em 2018, a soma de bens e serviços gerados no agronegócio chegou a 21,1% do PIB brasileiro (CONFEDERAÇÃO DA AGRICULTURA E PECUÁRIA DO BRASIL, 2019).

O desenvolvimento e a introdução de tecnologias contribuíram para o sucesso da agricultura no país, além da efetividade das políticas científicas e tecnológicas desenvolvidas e disseminadas pelas redes de administradores do setor (EKBOIR, 2003). Como um setor-chave da economia em um país em desenvolvimento de grande importância política e estratégica, a agricultura permanecerá no topo da agenda estratégica (SARITAS; KUZMINOV, 2017).

Portanto, a introdução e a difusão de novas tecnologias na agricultura são consideradas de extrema importância, pois oferecem oportunidades para aumentar a produção e a renda e fornecem emprego e meios de subsistência a um grande número de pessoas envolvidas no processo (FEDER; JUST; ZILBERMAN, 1985).

“A produção mundial de morangos vem crescendo em números absolutos nos últimos anos, chegando a 8.114.373 toneladas, para uma área total plantada de 377.435 hectares” (ANTUNES, 2017. p. 96). A produção de morango no Brasil cresce a cada ano e atualmente representa cerca de 40% da área total de produção na América do Sul, o que corresponde a aproximadamente 3.500 hectares. Além disso, essa cultura é de grande importância socioeconômica, uma vez que a maioria das áreas de cultivo de morango está localizada em propriedades com base na agricultura familiar. Isso pode significar maior renda para as

famílias, uma geração maior de empregos e um convite para fixação do homem no campo (EMBRAPA, 2016).

Características do cultivo do morango, o morango é uma planta herbácea, com um ciclo de vida longo, possui um caule subterrâneo conhecido como coroa (caule modificado). A coroa possui um tecido espiral condutivo periférico preso às folhas. A medula é proeminente e muito sensível à geada. À medida que a coroa envelhece, 8 a 10 novas coroas laterais podem aparecer (BORTOLOZZO, 2007).

As folhas surgem da coroa em forma helicoidal, com a forma e a cor dependendo da variedade. Geralmente eles são trifoliados com um par de estípulas na base, às vezes com um par de pequenos folhetos abaixo do normal. Os folhetos são dentados, verde escuro acima e cinza e peludo abaixo. Essa propriedade torna a lavoura muito sensível à falta de água, baixa umidade relativa, alta temperatura e intensidade e duração da luz (BORTOLOZZO, 2007).

Existem três sistemas de cultivo em operações de produção de morango: 1) o convencional (geralmente cultivado ao ar livre), no qual os canteiros são cobertos com filme plástico ou material orgânico (grama, palha ou outros produtos) com o objetivo de proteger o solo da chuva proteger ou manter as frutas limpas; 2) o orgânico; e 3) hidroponia (EMBRAPA, 2016).

O cultivo de morangos no solo está sujeito a várias doenças nas folhas e frutas, causadas em grande parte por organismos que se espalham pelo respingo de gotas de chuva e vento para se iniciar uma infecção. Nessa situação, chuvas frequentes ou irrigação por aspersão são extremamente prejudiciais às lavouras, pois criam condições favoráveis à infecção e lavam os fungicidas usados para a proteção das plantas (EMBRAPA, 2016).

Como alternativa aos problemas citados acima, foi proposto o uso de uma cobertura plástica na forma de túneis baixos, utilizados na maioria das áreas de produção. No entanto, a utilização desse tipo de túnel exige trabalho adicional, no qual a cobertura é aberta e abaixada diariamente para ventilar e cobrir a área plantada, manter a temperatura ambiente e proteger a colheita da chuva. Nesse estado, atividades como a limpeza da colheita para remover folhas e frutos doentes, que são o foco do surto de novas infecções e a detecção pela primeira vez da ocorrência de pragas e doenças, são prejudicadas pela cobertura e não são possíveis em dias de chuva. Essa situação leva a uma maior demanda pelo uso de fungicidas e outros pesticidas para prevenir doenças e pragas. Na figura 1 podemos ver o plantio do morango no solo com cobertura plástica (EMBRAPA, 2016).

Figura 1- Plantação de morango no solo.

Fonte: TV Gazeta (2017).

A agricultura orgânica consiste em um modelo de produção agrícola no qual, não são utilizados produtos ou processos de produção que possam produzir alimentos contaminados e de pouco valor biológico. É baseado no uso racional do solo, em fertilizantes orgânicos de origem vegetal ou animal, em rotações de culturas intermediárias e criações, ao usar produtos de controle alternativos e homeopáticos para pragas, doenças e parasitas. Onde a utilização de produtos pós-colheita que não deixem resíduos em alimentos processados, no comércio justo em respeito aos trabalhadores e suas famílias e a plena conservação dos ecossistemas naturais (SOUZA; RESENDE, 2003).

Para evitar o aumento de perdas e a conscientização sobre o risco do uso frequente de pesticidas, técnicos e produtores de morangos têm procurado novas maneiras de continuar suas atividades. Uma alternativa para contornar esse problema é produzir morangos usando o sistema semi-hidropônico em um ambiente protegido, caracterizado por estufas altas, onde a infestação por pragas e doenças da parte do ar e das raízes é reduzida e o produtor pode realizar tratamentos com maior facilidade quando necessário, pode monitorar a presença de pragas e doenças, eliminando surtos no início de cada ocorrência (EMBRAPA, 2007).

Nesse caso, o morango não é plantado no solo e deve ser produzido em um substrato artificial (inorgânico ou orgânico), que reduz a contaminação por fungos causadores de doenças, possibilitando melhor manejo na fertilização a fim de controlar adequadamente a vitalidade, a produtividade e a qualidade da fruta. Esse sistema alternativo é de grande importância para os produtores, pois garante a rentabilidade da atividade, reduzindo a demanda por pesticidas na lavoura e permite fazer o melhor uso de pequenas áreas. No cultivo protegido, o umedecimento das plantas é mínimo e podem ser desenvolvidas medidas para reduzir os efeitos dos danos causados pelo sol e pela geada, em locais com invernos mais rigorosos (EMBRAPA, 2007). Na figura 2 podemos ver a plantação de morango em sistema semi-hidropônico.

Figura 2- Plantação de morango em sistema semi-hidropônico.

Fonte: Arquivo pessoal do autor (2020).

A plantação de morangos é sustentada por estruturas de madeira, eles também podem ser feitos de concreto ou ardósia. Sacos de plástico são colocados nessa estrutura, que são preenchidos com matéria orgânica para o plantio. A estrutura feita para suportar os sacos é de 80 centímetros, a altura do saco de plantio é de 15 centímetros. Após o preenchimento com o substrato, são feitos buracos em cada saco para receber as mudas de morango. A distância entre cada muda é de 10 a 15 centímetros. A rega é feita por gotas e, juntamente com a água, a planta também recebe os nutrientes necessários para um bom desenvolvimento (SENAR, 2020). Na figura 3 podemos ver a estrutura montada pronta para o plantio das mudas de morango.

Figura 3- Estrutura montada pronta para o plantio das mudas de morango.

Fonte: Arquivo pessoal do autor (2020).

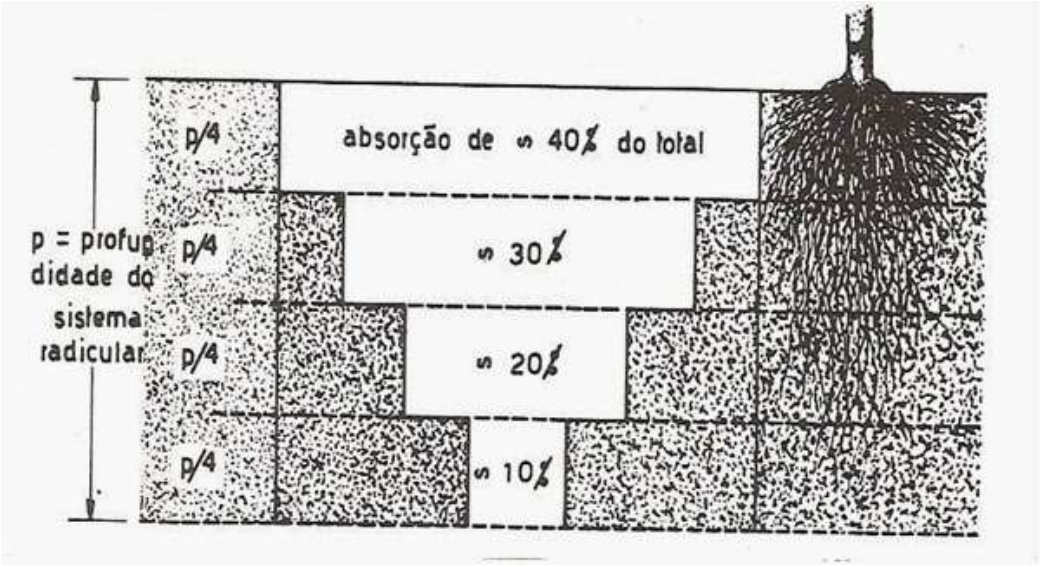
Assim como toda planta necessita de uma temperatura ideal para o seu cultivo, o morango não é diferente, sua temperatura ideal para o desenvolvimento é em média 22º C, sendo uma cultura sensível que necessita de um local correto para ser desenvolvido, variando de cultivos de dias longos e dias curtos. Em dias que a temperatura se encontra muito alta ou muito baixa, a tendência é a diminuição da produção do morangueiro. Para as variedades Aromas, Albion, Camarosa, Oso Grande e Diamante, por exemplo, as temperaturas compreendidas entre 12 e 25ºC não compromete o comportamento do cultivo. As cultivares Oso Grande e Camarosa são consideradas de dias curtos e, por isso, devem ser cultivadas apenas no inverno. Pelas condições climáticas do Brasil, recomenda-se a produção com as cultivares de dias longos (ALMEIDA *et al*., 2009).

No cultivo das plantas, a umidade do solo é de extrema importância. Segundo Lucietti (2014) quando começa a faltar água, algumas plantas apresentam alguns sinais, como as folhas que ficam enroladas e amareladas. Se a planta mostrar esses sinais por um determinado período, é provável que a produção cai acentuadamente, porém, o agricultor não pode esperar por sinais de falta de água antes de regar, a água deve ser reabastecida várias vezes antes que a cultura comece a murchar.

O tempo de irrigação, no sistema semi-hidropônico, normalmente é em torno de 2 a 5 minutos, sendo ministrado até 1 L de água por saco, por irrigação, dependendo da época do ano e da condição climática (HIDROPONIA BRASIL, 2020).

Um dos aspectos básicos num projeto de irrigação é quantidade de água a ser aplicada. A lâmina da água a ser usada deve ser a necessária para aumentar a umidade do solo para a capacidade de campo na camada do solo, que corresponde à profundidade efetiva da raiz. Essa lâmina depende do tipo de solo e da forma de cultivo (LUCIETTI, 2014). Na figura 4 podemos observar a profundidade da raiz no solo e a absorção de água.

Figura 4 - Crescimento das raízes das plantas no solo e absorção de água



Fonte: Lucietti (2014).

O desenvolvimento do sistema radicular é muito diferente para cada cultura e tipo de solo. Para uma melhor compreensão da profundidade efetiva do solo, é aconselhável avaliar isso no local da instalação. Essa determinação deve ser feita com o apoio do técnico do município, que também coleta o solo para determinar em laboratório a capacidade de retenção de água no solo e, consequentemente, o volume de água a ser utilizado, a cultura e em todas as etapas de seu desenvolvimento (LUCIETTI, 2014). Na tabela 1 podemos observar algumas profundidades efetivas do sistema radicular de algumas hortaliças.

Tabela 1 - Profundidade efetiva do sistema radicular (Z) de algumas hortaliças

Hortaliça	**Z (cm)**	**Hortaliça**	**Z (cm)**
Alface	20 a 30	Melancia	60 a 80
Alho	20 a 40	Morango	25 a 50
Batata	30 a 75	Pepino	45 a 60

Batata-doce	60 a 120	Pimentão	49 a 90
Beterraba	60 a 90	Repolho	40 a 50
Cenoura	45 a 75	Tomate	30 a 90
Couve-flor	30 a 60	Vagem	40 a 60

Fonte: Lucietti (2014).

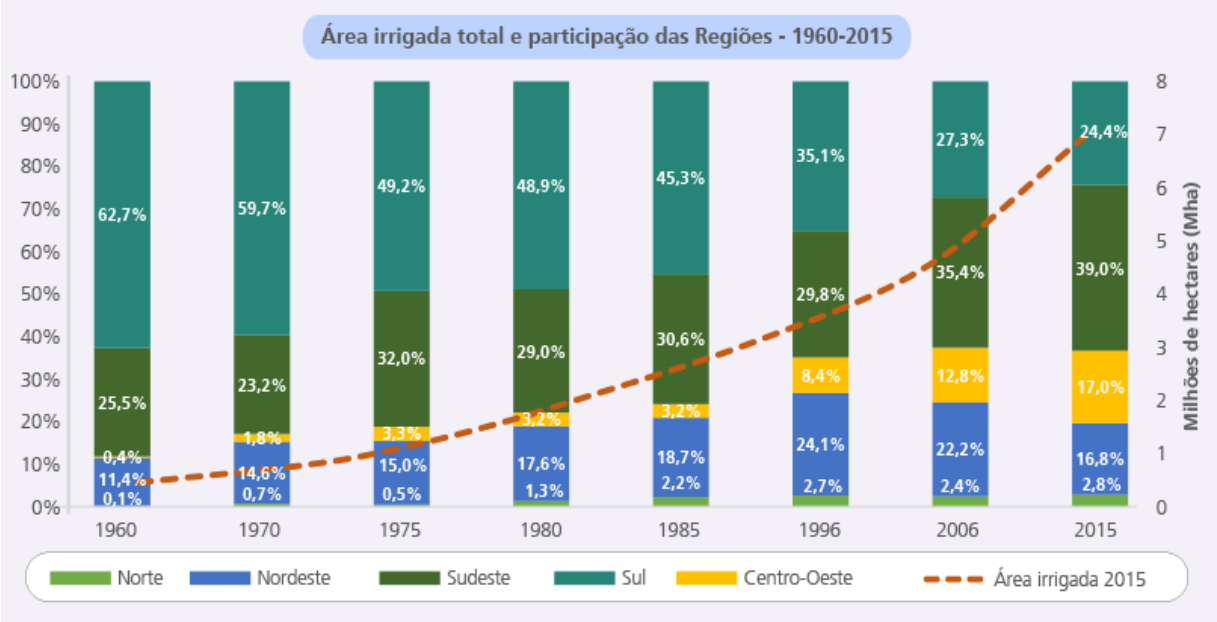
A irrigação deve ser realizada quando a disponibilidade de água no solo for reduzida a um nível que não prejudique a cultura. Com a água facilmente disponível a colheita não tem que trabalhar muito para tirar a água do solo (LUCIETTI, 2014).

2.2 SISTEMA DE IRRIGAÇÃO

Há muitas variáveis, etapas, equipamentos e aspectos de gestão para cuidar quando o assunto é a produção agrícola. O sistema de irrigação é um sistema de grande importância, capaz de fornecer um elemento indispensável para a planta, principalmente em períodos de estiagem. Por isso, manejar a água não é um trabalho simples, podendo ser abranger em: definir quando irrigar e quanto de água será necessário. Tornando assim, a capacidade de monitorar e alterar os parâmetros da irrigação uma tarefa de suma importância para a produção (GUIMARÃES, 2011).

A irrigação no Brasil começou nas plantações de arroz no Rio Grande do Sul entre o final do século 19 e início do século 20 (AGÊNCIA NACIONAL DE ÁGUAS, 2017). No gráfico 1 podemos ver a evolução da área irrigada no Brasil.

Gráfico 1 - Evolução da área irrigado do Brasil



Fonte: AGÊNCIA NACIONAL DE ÁGUAS (2017).

Segundo Melo e Silva (2007) os sistemas de irrigação são divididos em três grupos:

- Irrigação por superfície: Consiste em um método de irrigação feita diretamente no solo, em que a condução da água ocorre por meio de sistema de distribuição até o ponto de infiltração conforme a parcela desejada a ser irrigada.

- Irrigação por aspersão: é um método que procura ser semelhante a uma chuva, no qual a água é aspergida sobre a superfície do terreno por meio do fracionamento do jato da água em gotas.

- Irrigação localizada: é um método em que a água é aplicada com pequena intensidade e alta frequência diretamente sobre a região radicular.

2.2.1 Irrigação por superfície

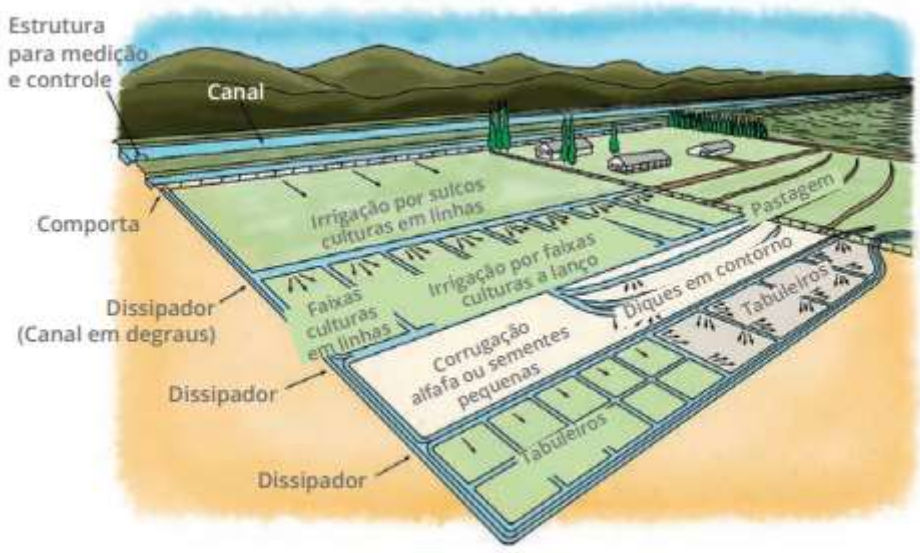
Segundo Cuenca (1989) a irrigação por superfície é o método mais utilizado e o mais antigo em todo o mundo. A história da irrigação começa com a aplicação de água ao solo e aproveitando sua superfície para o escoamento por gravidade.

> A irrigação por superfície é caracterizada pela aplicação de água diretamente sobre a superfície do solo da área a ser irrigada. No método de irrigação por superfície, a distribuição da água se dá por gravidade através da superfície do solo. Na maioria dos casos, esta técnica apresenta menor custo de implantação em relação aos demais métodos de irrigação. Este método é mais indicado para solos de textura fina a média (argilo-arenosos), com declividade relativamente pequena e uniforme. "A maior despesa inicial se refere ao nivelamento (dar formato adequado ao terreno) - investimento que só se faz uma vez. Se a declividade original do terreno se aproximar da declividade final projetada, um movimento de terra mínimo será necessário, para se conseguir o formato desejado (FERENC, 2020. p. 1).

Com o método de irrigação de superfície, a água pode ser aplicada usando os seguintes sistemas: sulcos, ondulações, faixas ou inundações. No sistema de irrigação por inundação, a água é aplicada ao solo através de bacias, e as áreas são quase planas e delimitadas por diques. Independentemente do sistema utilizado, a irrigação por superfície é, por um lado, um método mais barato, por outro, torna-se menos eficiente, pois se perde um grande volume de água devido a infiltrações e drenagens profundas, específicas desse método (FERENC, 2020).

Devido ao alto consumo de água para a aplicação em sistemas de irrigação de superfície, essa tecnologia exerce uma grande influência na disponibilidade de água das fontes de água. A definição de irrigação de superfície inclui sistemas de irrigação que distribuem a água diretamente sobre a superfície do solo a partir de uma extremidade da colheita e cobrem gradualmente a área (SENAR, 2019). A Figura 5 mostra um sistema de irrigação de superfície.

Figura 5 - Irrigação por superfície



Fonte: SENAR (2019).

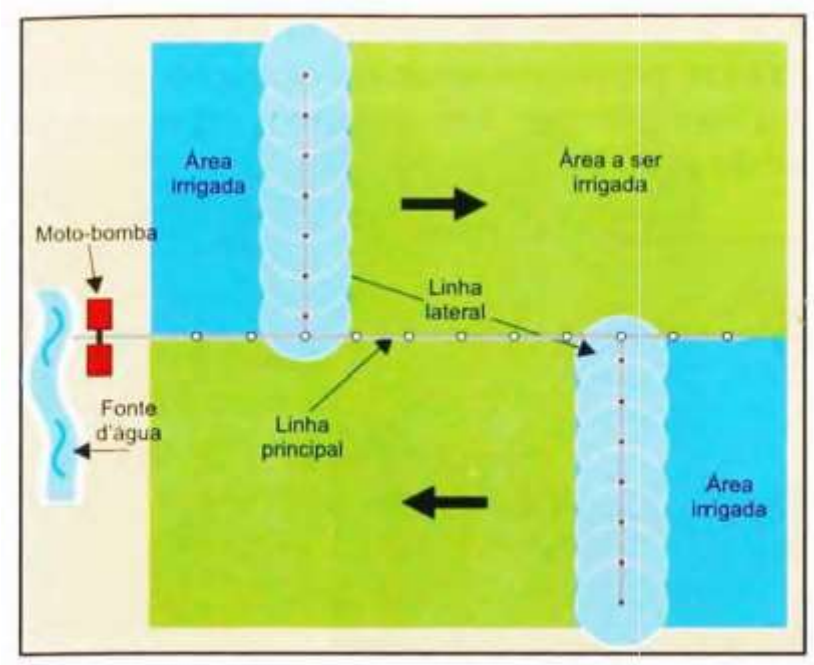
2.2.2 Irrigação por aspersão

No Brasil, mais de 90% da área total de hortaliças irrigadas são realizadas por irrigação por aspersão. A aspersão, por mais que seja o método mais utilizado, não deve ser considerado ideal para todas as condições de produção e capaz de atender a todos os interesses envolvidos. Ao escolher o sistema, deve-se basear em uma análise de viabilidade técnica e econômica para cada situação (MAROUELLI; SILVA; SILVA, 2008).

O sistema de irrigação por aspersão leva a água por meio de dutos até a área a ser irrigada e a distribui por aspersores, semelhante à chuva. É o sistema mais comum usado na agricultura, mas os aspersores utilizados devem ser adequados (LUCIETTI, 2014).

São classificados os sistemas de irrigação por aspersão convencional em portáteis, semi-portáteis e fixos, dependendo de como o sistema é manuseado no local. Eles geralmente consistem em uma linha adutora, um conjunto de bombas, uma linha principal, uma ou mais linhas laterais e aspersores, com os quais áreas de qualquer formato podem ser irrigadas. A Figura 6 mostra o esquema de um sistema de irrigação por aspersão semi-portátil convencional com dois lados móveis (MAROUELLI; SILVA; SILVA, 2008).

Figura 6 - Sistema de irrigação por aspersão



Fonte: MAROUELLI; SILVA; SILVA (2008).

Ao longo da área a ser irrigada, os componentes são deslocados manualmente por meio do sistema convencional portátil. A aquisição inicialmente é de baixo custo, mas requer grande demanda de mão-de-obra para as mudanças dentro da área. No sistema fixo, há um aumento do custo do sistema devido todos os componentes serem fixos, porém há uma redução significativa o uso de mão de obra e permite automatizar a irrigação. Já no sistema semi-portátil, as linhas laterais e os aspersores são deslocados no interior da área, enquanto os demais permanecem fixos (MAROUELLI; SILVA; SILVA, 2008).

No que se trata em cultivo de morango esta técnica, caiu em desuso na grande maioria das plantações, por ser ineficiente ao desperdiçar muita água e possibilitar o surgimento de fungos nas folhas e nos frutos do morangueiro (EMBRAPA, 2016).

2.2.3 Irrigação localizada

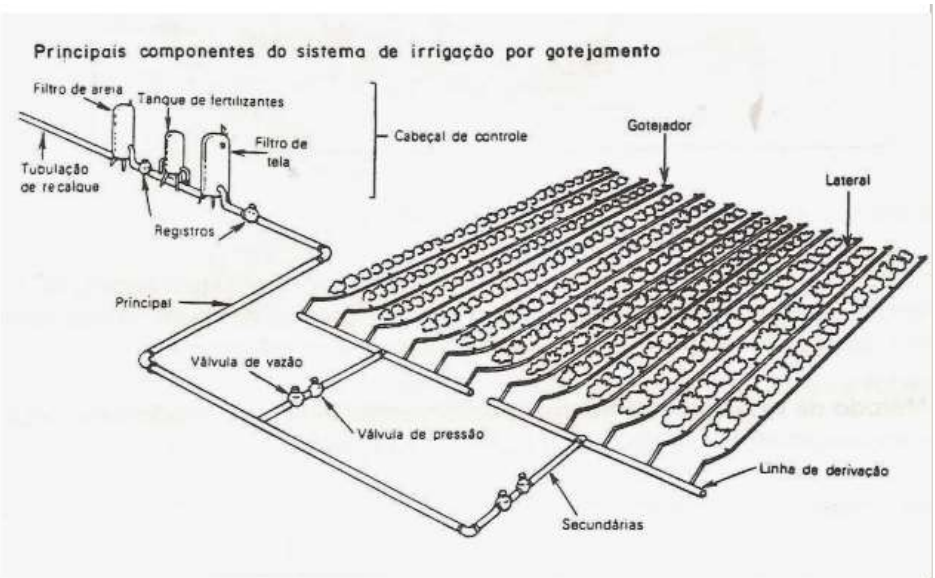
A irrigação por gotejamento ou localizada é utilizada no cultivo protegido do morango semi hidropônico em substrato artificial. Este tipo de irrigação apresenta algumas vantagens, sendo elas: alta eficiência em aplicação, economia hídrica, energia e mão-de-obra, permite automatização do sistema, fertirrigação (técnica utilizada para levar nutrientes até a planta por meio hídrico) e não interfere no desenvolvimento da planta (BORTOLOZZO, 2007).

A irrigação por gotejamento ocorre através da aplicação de água em pequena quantidade e alta frequência diretamente na região da raiz da planta. Está irrigação nos traz inúmeras vantagens em comparação ao sistema convencional de aspersão, pois não se aplica a água sobre toda área irrigada. Além do mais, encontramos na irrigação por gotejamento um potencial para atingir uma irrigação uniforme ao decorrer da plantação, possibilitando a aplicação de adubos por meio do gotejamento (EMBRAPA, 2016). De acordo com Lucietti (2014,) é amplamente utilizado para cultivo com espaçamento maior, como pepinos, feijões verdes, melancias, abóboras e, especialmente, para lavouras de alta densidade econômica, como tomates, pimentões e morangos, este sistema economiza água e tem uma eficiência de aplicação de até 95%.

O gotejamento tem como vantagens também a redução da mão de obra, melhor controle fitossanitário e a ausência da interferência nas práticas culturais. Contudo, este sistema de irrigação apresenta alto custo inicial, tendo como exigência água de qualidade e um sistema eficiente sistema de filtragem para diminuir os problemas de obstrução dos gotejadores (EMBRAPA, 2016).

"Na irrigação localizada, os componentes do sistema geralmente são fixos, sendo constituídos de: motobomba, cabeçal de controle, linha principal, linha secundária, linha lateral, válvulas, acessórios e gotejadores" (EMBRAPA, 2016, p. 299). Na figura 7 podemos ver um exemplo de sistema de irrigação por gotejamento.

Figura 7 - Sistema de irrigação por gotejamento



Fonte: (LUCIETTI, 2014).

Atualmente, a irrigação por gotejamento vem sendo muito utilizada no cultivo de morangos, principalmente porque pode ser facilmente combinada com outras práticas culturais (ambiente protegido e cobertura do solo). Essas práticas reduzem o consumo de água, mantém a umidade e as frutas do solo limpas, controlam as ervas daninhas e melhoram a qualidade e o tamanho das frutas (YUAN, 2004). Segundo esses autores, a irrigação leva a um aumento na produtividade do morango, o que leva a frutos mais pesados e a uma quantidade maior (EMBRAPA, 2016). Na figura 8 podemos ver a aplicação da irrigação localizada no cultivo do morango.

Figura 8 - Sistema de irrigação localizada.

Fonte: Bortolozzo (2007).

"A automação de sistemas de gotejamento atende a várias necessidades: incrementa a produção agrícola com o uso eficiente dos recursos hídricos e energéticos, reduz a mão de obra e facilita a aplicação de fertilizantes através da água de irrigação" (EMBRAPA, 2016, p. 300).

2.3 COMPONENTES CONSTITUINTES DE UMA AUTOMATIZAÇÃO DE IRRIGAÇÃO

2.3.1 Sistema (*on/off*)

Em automação, o interesse focaliza em sistemas que são dinâmicos em um sentido especial. A palavra "dinâmico" é entendida em geral como relativa a forças e energias produzindo movimento". No entanto, um segundo significado tornou-se essencial nas últimas décadas, devido a inúmeros e importantíssimos outros tipos de sistemas, tais como os de chaveamento manual ou automático. Sua estrutura impõe principalmente em regras lógicas, de causa e efeito, para eventos; seus sinais são números naturais representados estados lógicos (*on/off*) (MORAES; CASTRUCCI, 2010).

O sistema proposto caracteriza-se de um sistema *on/off* controlado por um microcontrolador Arduino, em que o sistema automático depende do sensor de umidade para ser ativado. Este sensor de umidade será colocado no substrato, que constantemente fará a leitura da umidade, certificando-se se há necessidade de irrigar.

O sistema semiautomático depende da entrada de dados que será feita pelo operador através da IHM, onde o operador irá escolher qual será o tempo de irrigação. Já o sistema manual, o operador terá que abrir o registro da válvula manualmente ligar e desligar o sistema de irrigação.

2.3.2 Microcontrolador

Entre as décadas 60 e 70 após o surgimento do circuito integrado e do microprocessador, teve um aumento na quantidade de inteligência que pode ser embutida em uma máquina com um custo baixo. Houve o crescimento no número de tarefas que podem ser feitas automaticamente. No momento atual, indica-se o computador pessoal para realizar funções simples e complexas, com um custo acessível. A automação requer um operador, mas pode reduzir a mão de obra. O operador consegue controlar a tarefa através de uma máquina, não necessitando fazê-la diretamente (RIBEIRO, 1999).

O microcontrolador é um computador pequeno (SoC) em um único circuito integrado que contém um núcleo de processador, memória e periféricos de entrada e saída programáveis. A memória de programação pode ser RAM, NOR *flash* ou PROM, que geralmente está incluída no chip. Os microcontroladores são desenvolvidos para aplicações embarcadas, em contraste com os microprocessadores usados em computadores pessoais ou outros aplicativos de uso geral (GOILAV; LOI, 2016).

Conforme o estudo feito por Marsczaokoski, Cruz e Silva (2013), em que se empregou o microcontrolador PIC para desenvolver um sistema de irrigação automatizado, com o intuito de atender pequenos cultivadores de morango, havia a necessidade de um controle eficaz na irrigação que é um fator essencial para a produtividade. Para a alimentação de dados do sistema de controle são utilizados sensores de umidade e de temperatura responsáveis por definir quando o aplicativo deverá atuar em modo automático, e um display de duas linhas e 16 colunas usado para a visualização de informações.

No presente estudo foi realizado a aplicação do microcontrolador Arduino para desenvolver um sistema automatizado para irrigação de morango sob plantio semi hidropônico. Para o sistema automático foram necessários sensores de temperatura e de umidade para a alimentação de dados do sistema de controle, onde o microcontrolador irá aferir os dados obtidos através dos sensores, assim comparando com os dados de entrada definido pelo programador, que verificou qual a melhor temperatura e umidade do solo para o plantio do morango e deixou definido no sistema. Também foi considerado o uso de uma IHM onde poderá ser monitorado a funcionamento do sistema, verificando se a bomba e

válvula estão ligadas no modelo automático. Através da IHM o operador define os segundos em que a válvula ficará ligada para o sistema semiautomático, e em modo manual o ligamento e desligamento da bomba é realizado de modo como o operador almejar.

2.3.3 Arduino

No ano de 2005, em uma cidade da Itália chamada Ivrea, deu-se início ao Arduino através do *Interaction Design Institute*, onde Massimo Banzi, professor, buscava um método barato e fácil para os estudantes de design operar a tecnologia. Ele debateu sobre o assunto com o David Cuartielles, um pesquisador de uma universidade da Suíça, no qual estava em busca por solução parecida, onde criou-se o Arduino (EVANS; NOBLE; HOCHENBAUM, 2013).

Naquela época, existiam produtos caros e relativamente difíceis de manusear. As principais requisições eram que fosse barato e que fosse uma plataforma que qualquer pessoa pudesse manusear. A placa foi desenhada por David Cuartielles, David Melllis que era um aluno de Massimo, programou o *software* para executar a placa. Quando o público percebeu que era fácil de ser utilizado, e de baixo custo se tornou muito popular entre os alunos, podendo ser empregado em seus próprios projetos bem como era excelente introdução para programação de microcontroladores (EVANS; NOBLE; HOCHENBAUM, 2013).

O objetivo da criação do Arduino foi desenvolver uma plataforma muito mais fácil em comparação às demais já existentes, tornando-o ideal para desenvolvedores e iniciantes com experiência, podendo agora desenvolver projetos de uma forma mais rápida e fácil. Outro motivo pelo qual torna o Arduino mais cativante é a sua ideologia de *Software* Livre, melhor dizendo, pode ser utilizado para criar inúmeros projetos, isento de custos, podendo ser distribuídos gratuitamente pelo uso da plataforma, conforme a necessidade do desenvolvedor. Trazendo inúmeras vantagens, além de criar e distribuir variadas bibliotecas e ferramentas para contribuir no desenvolvimento de projetos diariamente, ele possui uma coletividade abundante de pessoas que desvendam detalhes e informações sobre como funciona o Arduino. Estes são uns dos motivos pelo qual o Arduino vem se tornando cada vez mais popular e crescendo em meios aos desenvolvedores (PEDRERA, 2017).

“O Arduino é uma plataforma de computação física de fonte aberta, com base em uma placa simples de entrada/saída (*input/output*, ou I/O). O Arduino pode ser utilizado para desenvolver objetos interativos independentes, ou conectado a *softwares* de seu computador (BANZI, 2011, p. 17).”

> Um microcontrolador é uma simplificação de um computador em um circuito impresso. Normalmente, um computador é composto de vários elementos que trabalham juntos em torno de um elemento central, o processador. Esses elementos são divididos em quatro partes principais: um processador, uma memória RAM, uma memória ROM e uma série de conectores que garantem a interface com o mundo exterior. Essas quatro partes são diferentes umas das outras, mas devem trabalhar juntas para garantir o bom funcionamento de um computador. A interface homem-máquina do microcontrolador ou interface de entrada e saída é uma série de conexões que permitem a comunicação com o seu computador, seja através de teclado, mouse ou tela (GOILAV; LOI, 2016, p. 9).

O microcontrolador comumente presente nas placas Arduino é alimentado por uma tensão de 5V. De acordo com o modelo da placa, essa voltagem pode ser fornecida pela conexão USB usada para conectá-la ao computador ou por uma das tomadas de energia da placa. As tomadas devem fornecer uma voltagem entre 7 e 12V, mas essa voltagem não precisa ser estável, graças à existência de um regulador de voltagem na placa. Pode haver duas tomadas de energia em uma placa Arduino. O primeiro tipo normalmente é o mais simples de executar. Este é um soquete padrão simples presente nas placas Arduino Uno e Mega onde você só precisa ligar uma fonte de alimentação que forneça a voltagem necessária para iniciar a placa. Essa tomada não está presente em todos os modelos, principalmente em placas menores, onde não são vitais. Por outro lado, o segundo tipo está presente em todas as placas. É um conector denominado de Vin, que sem encontra em um lado da placa. Basta conectar polo positivo da fonte de alimentação neste conector, como se fosse uma bateria, e o polo negativo a um conector GND (GOILAV; LOI, 2016).

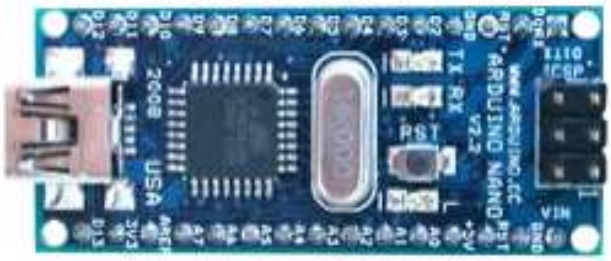
Há muitos tipos de Arduino que podem ser utilizados, dependendo do objetivo, com inúmeras formas e configurações de *hardware*. O Arduino Uno é o de maior uso, mas o Arduino Mega, tendo como exemplo, possui maior quantidade de portas de entrada, concedendo a criação de maiores e complexos dispositivos. O Arduino Nano é uma versão sucinta de um Arduino comum, com a finalidade de criar objetos eletrônicos menores (PEDRERA, 2017).

2.3.4 Arduino NANO

O Arduino NANO é uma versão pequena do Arduino semelhante ao Arduino UNO, pois também possui um *chip* ATMega328, na versão SMD. Possui um conector para cabos Mini-USB para gravação. Uma das diferenças entre esta placa e o Arduino UNO ou Arduino 2009 é que esta placa possui mais 2 entradas analógicas e um jumper + 5V AREF. Esta placa

não possui um conector para uma fonte externa, mas é possível alimentá-la através do pino Vin. O Arduino NANO seleciona automaticamente a maior potência fornecida. Na figura 9 ilustra a placa de um Arduino NANO e na tabela 2 podemos ver as principais característica do Arduino NANO (GRUPO DE ROBÓTICA, 2012).

Figura 9 - Arduino NANO.



Fonte: GRUPO DE ROBÓTICA (2012).

Tabela 2 - Principais característica do Arduino NANO

Microcontrolador	Atmel ATMEGA168 ou ATMEGA328
Tensão de operação	5 V
Tensão de entrada (recomendado)	7-12 V
Limites de tensão de entrada	6-20 V
Pinos de I/O digital	14 (dos quais 6 dispõem de saídas PWM)
Pinos de entrada analógica	8
Nível de corrente por Pino I/O	40 mA
Memória de programa – Flash	16 KB(ATMEGA168) ou 32 KB (ATEMEGA328) (2 KB para bootloader)
Memória SRAM	1 KB (ATEMEGA168) ou 2 KB (ATEMEGA328)
Memória EEPROM	512 bytes (ATEMEGA168) ou 1 KB (ATEMEGA328)
Frequência de clock	16 Mhz

Fonte: Stevan, Silva (2015).

2.3.5 Arduino UNO

O Arduino UNO é a versão mais difundida da família Arduino, pois possui um microcontrolador de entrada, bons recursos e um número suficiente de interfaces para muitos projetos simples. Possui algumas versões que diferem principalmente pelo modo de colagem do microcontrolador, além de melhorias no circuito de *reset* (STEVAN; SILVA, 2015).

É uma placa com microcontrolador Atmega328. Contém 14 entradas/saídas digitais, 6 entradas analógicas, um cristal oscilador de 16MHz, conexão USB, uma entrada para fonte, soquetes para ICSP, e um botão de *reset*. Para alimentá-la conecte-a a um computador com o cabo USB ou ligue a placa com uma fonte AC-DC. O UNO seleciona automaticamente a

fonte de alimentação. A figura 10 abaixo ilustra um Arduino UNO, na tabela 3 podemos ver as principais características do Arduino UNO, e na figura 11 mostra mais detalhadamente as posições das interfaces de comunicação e alimentação (GRUPO DE ROBÓTICA, 2012).

Figura 10 - Arduino UNO.

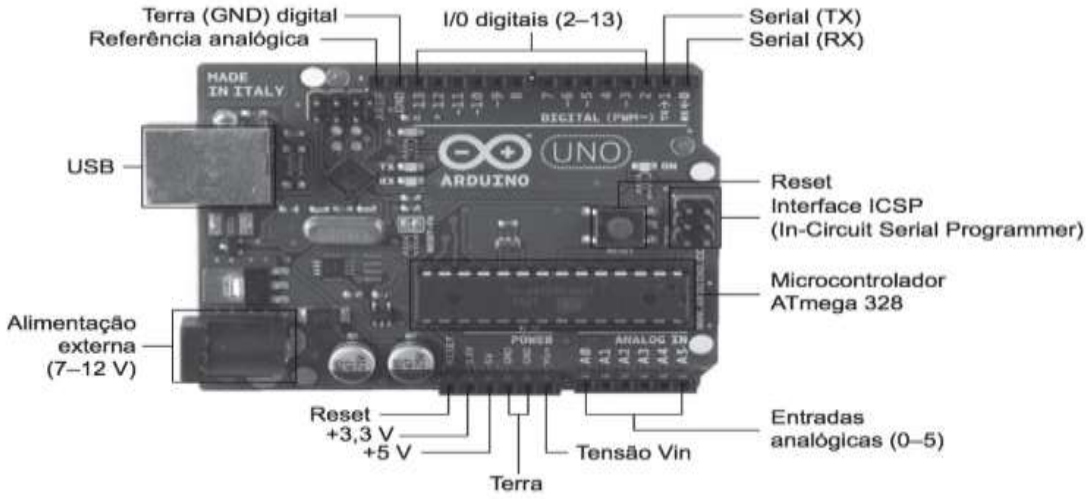
Fonte: Grupo de Robótica (2012).

Tabela 3 - Principais característica do Arduino UNO

Microcontrolador	ATMEGA328
Tensão de funcionamento	5 V
Tensão de entrada fonte externa (recomendado)	7-12 V
Tensão de entrada (limites)	6-20 V
Digital pinos de I/O	14 (dos quais 6 fornecem uma saída PWM)
Entrada Analógica	6
Corrente DC para pinos I/O	40 mA
Corrente DC para Pino 3,3 V	50 mA
Memória Flash	32KB(ATEMEGA328), dos quais 0,5 KB utilizado pelo bootloader
SRAM	2 KB (ATEMEGA328)
EEPROM	1 KB (ATEMEGA328)
Velocidade de clock	16 Mhz

Fonte: Stevan, Silva (2015).

Figura 11 - Interface de comunicação e alimentação do Arduino UNO



Fonte: Stevan, Silva (2015).

2.3.6 Arduino MEGA

O Arduino MEGA 2560 é uma placa de microcontrolador baseada no ATmega2560. Possuindo 54 pinos de entradas/saídas digitais, 16 entradas analógicas, 4 UARTs (portas seriais de *hardware*), um oscilador de cristal de 16MHz, uma conexão USB, uma entrada de alimentação, uma conexão ICSP e um botão de *reset*. Esta placa contém o necessário para dar suporte ao microcontrolador, basta conectar a um computador com um cabo USB ou a uma fonte de alimentação. O Arduino MEGA é compatível com a maioria dos *shields* desenhados para os Arduino Uno, *Duemilanove* e para o *Diecimila*. Na figura 12 ilustra o Arduino MEGA (GRUPO DE ROBÓTICA, 2012).

Figura 12 - Arduino MEGA

Fonte: Grupo de Robótica (2012).

De forma semelhante ao Arduino UNO, no MEGA a interface USB é controlada por um microcontrolador auxiliar (ATMEGA16U2), que está conectado ao microcontrolador principal (ATMEGA2560) através do canal serial. E com o cristal oscilador possibilita 16 MIPS (a execução de 16 milhões de instruções por segundo) (STEVAN; SILVA, 2015).

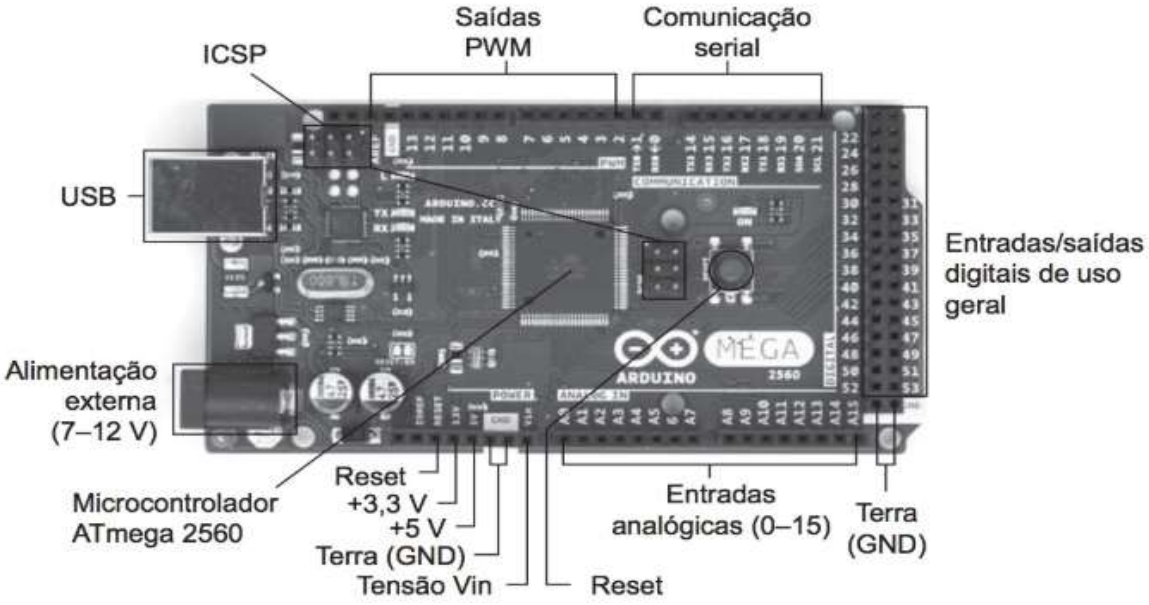
O Arduino MEGA destaca-se por possuir um multiplicador por *hardware*, 4 canais de comunicação serial, 16 entradas analógicas e 15 saídas PWM. Possui ainda comunicação SPI, I²C e 6 pinos de interrupções externas. Os 54 pinos de entrada e saída atuam igual o UNO, com tensões de 0V para nível baixo e de 5V para nível lógico alto, podendo produzir ou drenar até 40 mA por pino. Na tabela 4 podemos ver as principais características do Arduino MEGA, e na figura 13 mostra mais detalhadamente as posições das interfaces de comunicação e alimentação (STEVAN; SILVA, 2015).

Tabela 4 - Principais característica do Arduino MEGA

Microcontrolador	ATMEGA2560
Tensão de funcionamento	5 V
Tensão de entrada (recomendado)	7-12 V
Tensão de entrada (limites)	6-20 V
Digital pinos de I/O	54 (dos quais 15 podem operar com saídas PWM)
Entrada Analógica pinos	16
DC Current per I/O Pin	40 mA
Corrente DC para Pino 3,3 V	50 mA
Memória Flash	256 KB dos quais 8 KB usados pelo bootloader
SRAM	8 KB
EEPROM	4 KB
Velocidade de clock	16 Mhz

Fonte: Stevan, Silva (2015).

Figura 13 - Interface de comunicação e alimentação do Arduino MEGA



Fonte: Stevan, Silva (2015).

2.3.7 IHM

Segundo Moraes e Castrucci (2010, p. 119), “IHM é um *hardware* industrial composto normalmente por uma tela de cristal líquido e um conjunto de teclas para navegação ou inserção de dados que utiliza um *software* proprietário para sua programação”. Na figura 14 visualizamos uma IHM.

Figura 14 - IHM

Fonte: Elaboração do autor (2020).

A IHM será utilizada para monitorar o modo automático, para entrada de dados onde o operador irá definir o tempo em que a irrigação ficará ligada quando está em modo semiautomático e para ligar e desligar a bomba quando estiver em modo manual.

2.3.8 Válvula solenoide

"O solenoide é um dispositivo usado para traduzir sinais elétricos *ON/OFF* em movimentos mecânicos *ON/OFF*. Válvula é um dispositivo mecânico projetado para controlar a vazão de fluidos" (RIBEIRO, 1999. p. 8). Na figura 15 visualizamos uma válvula solenoide.

Figura 15 - Válvula solenoide

Fonte: Válvula (2020).

A válvula solenoide é utilizada na irrigação para diferenciar áreas onde recebera o comando do microcontrolador Arduino para saber qual válvula deve abrir e qual válvula deve fechar.

2.3.9 Sensor de temperatura

"O sensor de temperatura DS18B20 possui sua própria inteligência. Ele é capaz de ler a temperatura, interpretá-la e enviar a informação do valor de temperatura em graus Celsius para o microcontrolador usando um barramento de apenar um fio" (MADEIRA, 2018, p. 1).
Um sensor DS18B20 possui um endereço exclusivo de 64 bits no qual o desenvolvedor pode adicionar inúmeros sensores em somente um barramento, onde irá utilizar somente uma porta do microcontrolador e obter valores de temperatura de cada sensor particularmente. Esse

sensor pode medir temperaturas entre -55 ºC e 125 ºC com uma precisão de cerda de 0,5°C na faixa de -10°C e 85°C. Na figura 16 visualizamos um sensor DS18B20 (MADEIRA, 2018).

Figura 16 - Sensor DS18B20

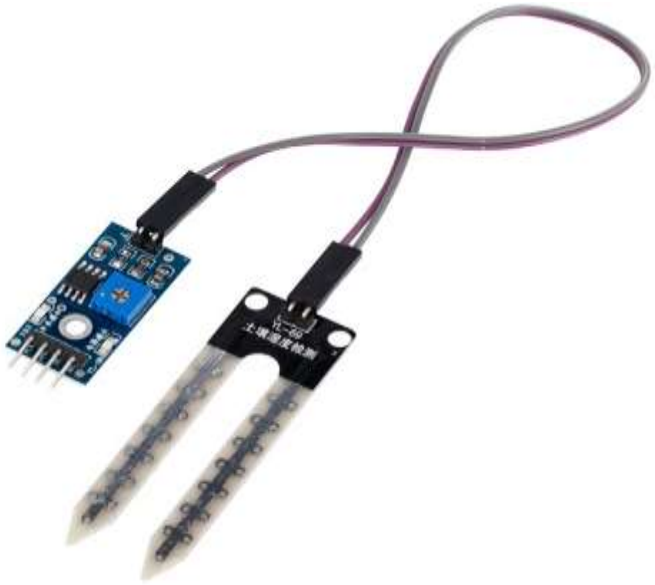
Fonte: Madeira (2018).

Para o cultivo do morango, a temperatura é um fator de muita importância pois pode influenciar na qualidade do fruto. O sensor irá aferir a temperatura enviando os dados para o microcontrolador no qual irá comparar com a temperatura ideal. Assim, se houver alguma alteração na temperatura a bomba será acionada.

2.3.10 Sensor de umidade do solo

"Este sensor faz a leitura da umidade do solo em que está inserido e transmite o valor da umidade lida para outros sensores que fazem uso desta para tomar decisões no acionamento da irrigação autônoma"(PEÑARANDA, 2018, p. 21). Na figura 17 visualizamos um sensor de umidade do solo Higrômetro.

Figura 17 - Sensor de umidade do solo Higrômetro



Fonte: (PEÑARANDA, 2018).

Manter a umidade ideal do solo no plantio do morango é de extrema importância para uma excelente produção. Para isso, serão utilizados sensores de umidade que estarão constantemente conferindo a umidade do solo, e realizando o acionamento da irrigação se necessário.

2.3.11 Fonte chaveada 24V

As fontes de alimentação modernas podem ser divididas em dois grupos principais: com controle linear ou com controle de chaveada. O que ele simplesmente chama de fonte de comutação é, na verdade, um conversor estático de CA para CC com regulação por comutação (MEHL, 2004).

Neste protótipo será necessário a utilização de uma fonte chaveada de 24V, ou seja, a fonte terá uma entrada 220V e irá comutar para que tenha uma saída de 24V, que será utilizado para a alimentação da IHM e dos LEDs. Na figura 18 podemos observar uma fonte chaveada 24v.

Figura 18 - Fonte chaveada 24V

Fonte: Elaboração do autor (2020).

2.3.12 LEDs

O desenvolvimento dessa tecnologia ampliou suas aplicações e se tornou fontes de luz que podem substituir as lâmpadas em diversas aplicações. Por seu pequeno porte e eficiência, a iluminação semicondutora tem atraído empresas e pesquisadores que conseguiram desenvolver produtos empregando esta tecnologia, tornando possível sua utilização em vários projetos (MARTELETO, 2011).

Os LEDs serão usados para sinalização facilitando o entendimento do operador. Quando o LED azul estiver aceso está indicando que o protótipo está operando em modo automático, quando o LED amarelo estiver aceso o protótipo está operando em modo manual, se o LED verde estiver aceso o protótipo está operando em modo semiautomático e caso o LED vermelho estiver aceso estará sinalizando que o botão de emergência está apertado. Na figura 19 podemos ver os LEDs.

Figura 19 - LEDs

Fonte: Elaboração do autor (2020).

2.3.13 Botão de emergência

Os botões de parada de emergência são dispositivos de acionamento que ficam fixados em um ponto visível da máquina ou em suas proximidades, estão sempre ao alcance do operador e, quando acionados, servem para interromper o movimento da máquina e desativar seu comando (MUNDO DA ELÉTRICA, 2020).

O botão de emergência está sendo previsto no protótipo para quando acontecer alguma irregularidade na irrigação. Quando o operador apertar o botão, a bomba de irrigação deverá parar seu funcionamento. Para liberar o funcionamento da bomba o botão terá que ser girado para direita. Na figura 20 podemos ver o botão de emergência.

Figura 20 - Botão de emergência

Fonte: Elaboração do autor (2020).

2.3.14 Chave seletora 3 posições

"É normalmente utilizada para selecionar entre acionamento automático ou manual de partida de motores, máquinas e equipamentos e para evitar acionamento indevido (acidental ou deliberado) de equipamentos, em casos de para de manutenção, por exemplo" (CETTI, 2020, p.1).

A chave seletora será utilizada para acionamento de modo automático quando ela estiver girada para a direita, se estiver centralizada indica que está em modo manual e caso esteja girada a esquerda indica que está em modo semiautomático. Na figura 21 ilustra chave seletora 3 posições.

Figura 21 - Chave seletora 3 posições

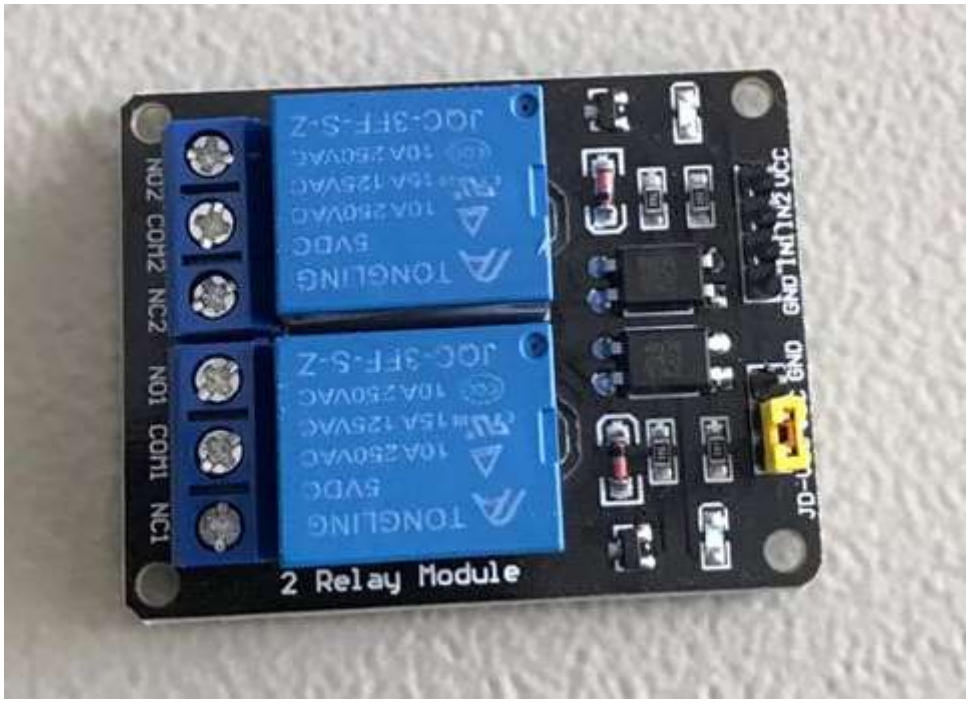
Fonte: Elaboração do autor (2020).

2.3.15 Módulo relé 5V

"O módulo relé é uma placa que pode ser utilizado para controlar dispositivos de corrente alternada 110V/220V de até 10 Ampères acionados por sinais digitais de 5V, pode servir para ativar lâmpadas, motores, ventiladores etc. A partir daí é ligado um relé a um Arduino" (ROBÔ LIVRE, 2020. p.1).

O módulo relé 5V estará conectado ao Arduino, e terá como função o acionamento de bomba de irrigação. Na figura 22 ilustra Módulo relé 5V.

Figura 22 - Módulo relé 5V



Fonte: Elaboração do autor (2020).

2.3.16 Dispositivo de proteção contra surtos (DPS)

DPS são dispositivos protetores de surto desenvolvidos para limitar sobretensões transientes e desviar as altas correntes vindas de descargas atmosféricas. Os dispositivos protetores de surto são essenciais em qualquer instalação que tenha riscos de sofrer danos por sobretensões (NBR-5410,2004).

Devido a entrada de energia possuir um sistema trifásico, serão utilizados 4 DPSs, sendo um para cada fase e um para o neutro. Assim tentamos evitar que os equipamentos não queimam devido a algum surto que possa ocorrer. Na figura 23 ilustra os DPS.

Figura 23 - DPS

Fonte: Elaboração do autor (2020).

2.3.17 Disjuntor

Os disjuntores são definidos como dispositivos de manobra e proteção que são capazes de produzir, conduzir e interromper correntes elétricas em condições normais de operação de circuitos elétricos, assim como estabelecer, conduzir por tempo definido e interromper correntes elétricas em condições anormais de funcionamento de tais circuitos, como por exemplo, em situações de sobrecarga e curto-circuito (FERNADES FILHO e DIAS,2014).

Será utilizado um disjuntor trifásico de modo que possamos proteger os equipamentos caso haver algum curto-circuito ou sobrecargas na rede. Na figura 24 podemos ver o disjuntor.

Figura 24 - Disjuntor

Fonte: Elaboração do autor (2020).

2.3.18 Relé falta de fase

É um dispositivo eletrônico que monitora a presença e o valor da tensão elétrica nas três fases, e em alguns modelos também monitora as tensões entre fase e neutro. Diante do exposto, detectar a ausência de uma das fases e desligar imediatamente o motor são essenciais com este tipo de proteção (FERNADES FILHO e DIAS,2014).

Este relé é muito importante no caso de motores, pois um motor trifásico queima se rodar com duas fases. Então utilizaremos um relé falta de fase para proteção do sistema, assim se houver alguma anomalia o relé possa atuar antes que danifique algum equipamento. Na figura 25 ilustra relé falta de fase.

Figura 25 - Relé falta de fase

Fonte: Elaboração do autor (2020).

2.3.19 Contatora

Contator é um dispositivo eletromecânico que permite, a partir de um circuito de comando, efetuar o controle de cargas num circuito de potência. Este dispositivo possibilita realizar a chamada isolação galvânica, adequando valores de tensão e corrente elétrica entre circuitos de comando e de potência (FERNADES FILHO e DIAS,2014).

O protótipo contara com duas contatoras, uma será utilizada para o botão de emergência, quando o botão for acionado a contatora irá atuar assim paralisando o sistema e a outra contatora será utilizada para a atuação da bomba de irrigação. Na figura 26 ilustra as contatoras.

Figura 26 - Contatoras

Fonte: Elaboração do autor (2020).

2.3.20 Disjuntor motor

É um dispositivo de manobra e proteção de motores elétricos. O dispositivo garante a proteção de circuitos elétricos e motores contra correntes de sobrecarga e curtos-circuitos, os quais são dotados de relés térmicos e magnéticos que podem ser calibrados de acordo com as características da carga e também podem reagir com sensibilidade à falta de fase (FERNADES FILHO e DIAS,2014).

O disjuntor motor foi adotado no protótipo a fim de ter mais uma proteção contra correntes de sobre cargas e curtos-circuitos para a bomba de irrigação tentando evitar que o equipamento danifique. Na figura 27 ilustra o disjuntor motor.

Figura 27 - Disjuntor motor

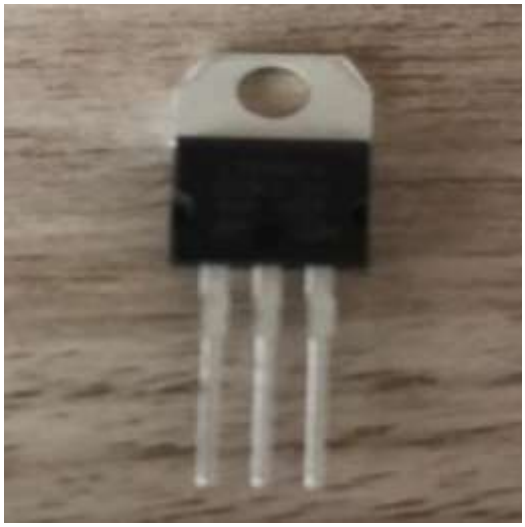
Fonte: Elaboração do autor (2020).

2.3.21 Regulador de tensão 7805

O regulador de tensão 7805, é capaz de regular a tensão de saída em seu terminal. O terminal de entrada do regulador de tensão pode receber tensão de 7,5 a 24V, entretanto, oferecerá em seu terminal de saída 5V estabilizado com corrente máxima de 1A (FILIPEFLOP, 2020).

O regulador de tensão 7805, o protótipo utiliza 3 reguladores uma para cada posição da chave seletora, alimentado por uma fonte de 24V, e obtende 5V na saída que servira para alimentação do Arduino e assim o Arduino identificará em qual modo operará. Na figura 28 ilustra um regulador de tensão 7805 (FILIPEFLOP, 2020).

Figura 28 - Regulador de tensão 7805.



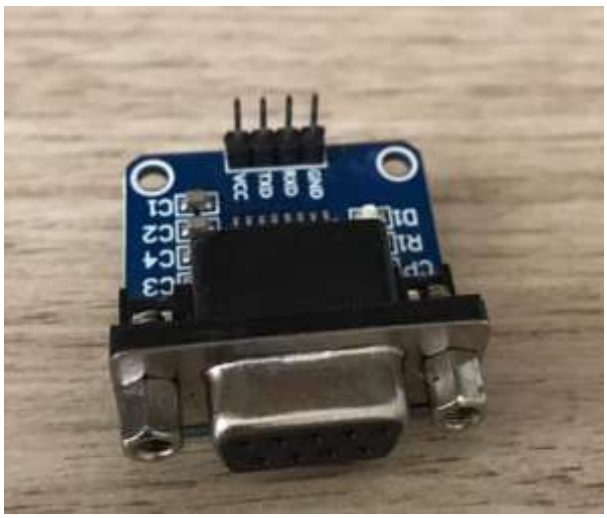
Fonte: Elaboração do autor (2020).

2.3.22 Conversor RS232

O conversor RS232 é um modulo eletrônico conversor que possui a capacidade de converter sinas do tipo RS232 em sinais de nível TTL para facilitar a comunicação entre computadores e plataformas microcontroladoras (WEIS, 2020).

Conta com uma porta RS232 do tipo DB9 macho para conexão com o computador ou outro tipo de equipamento e na outra extremidade apresenta 4 pinos (GND/RXD/TXD/VCC) para fácil conexão. O conversor RS232 será responsável pela comunicação entre a IHM e o Arduino, na figura 29 ilustra o Conversor RS232 (WEIS, 2020).

Figura 29 - Conversor RS232.



Fonte: Elaboração do autor (2020).

2.3.23 Conector DB9

O conector DB9 é uma variante analógica com 9 pinos da família dos conectores D-Subminiaturas. Ele serve essencialmente para as conexões em série, que permitem uma transmissão de dados assíncrona de acordo com o padrão RS232. Os plugues contêm soquetes e pinos, com cada pino numerado e etiquetado. Na tabela abaixo podemos ver a pinagem serial (WEIS, 2020).

Tabela 5 - Pinagem Serial DB9

Pin	Signal Direction	Signal Name	Signal Function
1	→	CD	Carrier Detected
2	←	RxD	Receive Data
3	→	TxD	Transmit Data
4	→	DTR	Data Terminal Ready
5	—	GND	Ground
6	←	DSR	Data Set Ready
7	←	RTS	Request To Send
8	→	CTS	Clear to Send
9	→	RI	Ring Indicator

Fonte: (WEIS, 2020).

→ *Transmitted From* DTE *Device*

← *Receive by* DTE *Device*

O conector DB9 será utilizado para comunicação da IHM com o com o Arduino, ele é conectado a IHM e ao conversor RS232, na figura 30 ilustra o conector DB9.

Figura 30 - Conector DB9

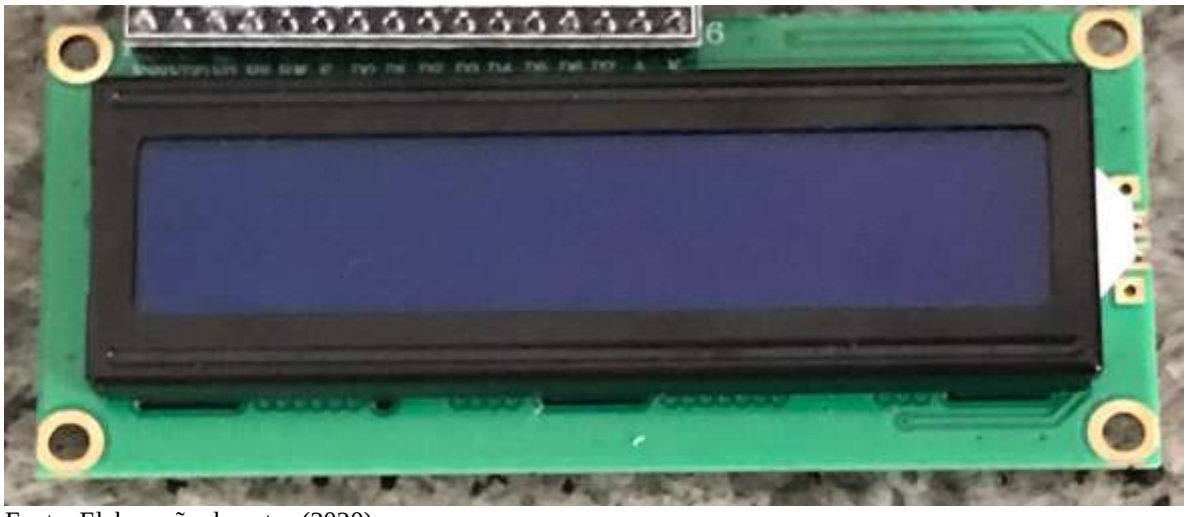
Fonte: Elaboração do autor (2020).

2.3.24 *Display* LCD (16x2)

Existem diversas maneiras de apresentarmos para um usuário as informações por via eletrônica, como texto, imagens e vídeos geradas pelos nossos projetos. Uma delas é a utilização do *display* de cristal líquido (LCD) (OLIVEIRA e ZANETTI, 2015).

O *display* LCD será utilizado para que o operador possa acompanhar o que está acontecendo, se está em modo manual, semi automático ou automático, também poderá monitorar se a bomba está em estado *ON* (ligado) ou *OFF* (desligado). Na figura 31observa-se um *display* LCD (16x2).

Figura 31 - *Display* LCD (16x2).



Fonte: Elaboração do autor (2020).

2.3.25 Módulo I2C

O protocolo I2C foi desenvolvido, visando conectar diversos dispositivos (periféricos) utilizando apenas as duas linhas de dados a DAS e a SCL, serial Data e serial *Clock*. A ideia principal é definir um endereço hexadecimal para cada dispositivos e no momento de comunicação somente o dispositivo solicitado responderá (ROBOCORE, 2020).

O Módulo I2C será utilizado para a comunicação entre o Arduino e o *display* LCD. Na figura 32 pode-se observar o modulo I2C.

Figura 32 - Módulo I2C:

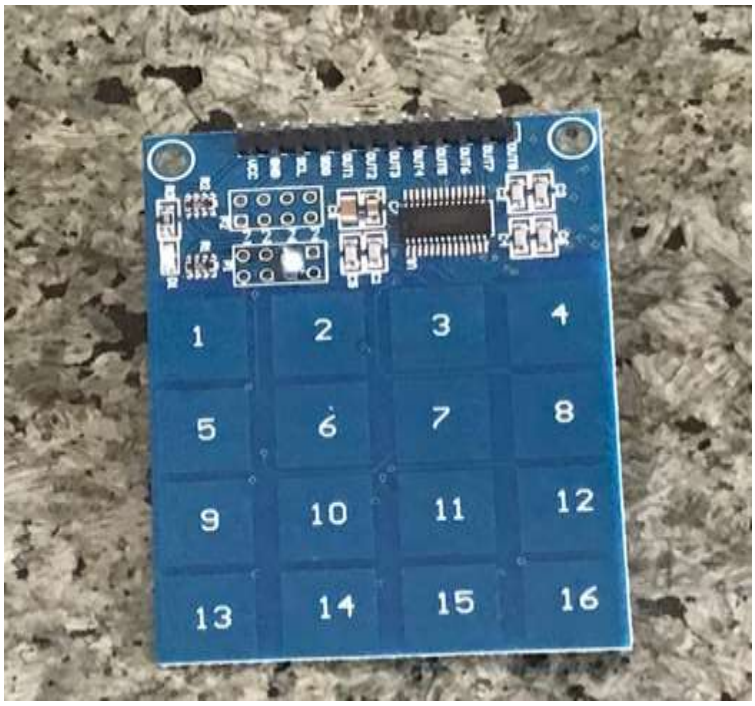
Fonte: Elaborado pelo autor (2020).

2.3.26 Teclado Capacitivo

O teclado capacitivo *touch* TTP239 com 16 teclas é capaz de detectar toques em cada uma das superfícies numéricas indicadas na placa, o teclado é baseado no circuito integrado TTP229-BSF com alta sensibilidade ao toque (OLIVEIRA, 2020).

O teclado capacitivo será utiliza no protótipo para a entrada de informação, quando está operando em estado semi automático. Cada tecla pressionada corresponde ao minuto. Na figura 33 observa-se o teclado capacitivo.

Figura 33 - Teclado Capacitivo.



Fonte: Elaborado pelo autor (2020).

3 DESENVOLVIMENTO DO PROTÓTIPO E RESULTADOS

O sistema de irrigação proposto neste trabalho deve suprir as necessidades hídricas do plantio de morango. O plantio em substrato requer um maior cuidado, visto que sua drenagem é maior, necessitando, portanto, de maiores pulsos de água durante o dia. A irrigação automatizada por gotejamento permitirá um maior controle da faixa de água. Dessa forma, a utilização de um sensor de umidade proposto no protótipo seria responsável por fazer a análise da necessidade da irrigação e enviar um pulso para que a bomba seja acionada. Esse sensor de umidade torna dispensável a presença de um operador para controlar a necessidade hídrica da planta. Logo, há uma redução na mão de obra, atendendo a um dos objetivos propostos pelo trabalho.

É imprescindível que esse sistema de irrigação automatizado, apresente um modo de operação em que o produtor possa realizar a irrigação mesmo em casos de falha de algum dos sensores, como por exemplo, o sensor de umidade, funcionando como uma segunda opção. Para atender a essa necessidade, o sistema proposto apresenta um modo de operação manual e um semiautomático, onde o produtor é responsável por controlar a necessidade hídrica da planta e a quantidade de água que ele deseja colocar. Portanto, devido ao tempo limitado não foi possível realizar uma avaliação de economia hídrica no cultivo do morango.

Para a ativação da bomba, independentemente do modo de operação utilizado, foi aplicado o microcontrolador Arduino para realizar essa função. Esse microcontrolador receberá um comando do sensor de umidade para o modo automático ou do próprio produtor quando o modo operado for o semiautomático ou o manual. A fim de que o produtor possa ter um monitoramento de quando a bomba está ou não ligada, foi incluído no protótipo um *Display* LCD.

Neste capítulo serão apresentados todos os resultados obtidos com a implementação do protótipo de irrigação automatizada. Testaram-se todos os sensores utilizados, a comunicação entre Arduino e *display* LCD, com o objetivo de avaliar se eles apresentam resultados satisfatórios.

O objetivo inicial era de implementar uma IHM onde o operador poderia monitorar o funcionamento do sistema. Em modo semiautomático forneceria informação do tempo em que a irrigação se manterá ligada, e no modo manual iria ligar e desligar a bomba. O proposto inicialmente sofreu alteração devido a um problema técnico com a IHM. Buscou-se uma nova solução para poder atender a todos os objetivos iniciais.

O *display* LCD e o teclado capacitivo foram adotados para suprir a ausência da IHM, onde o *display* servirá para monitoramento do sistema e o teclado capacitivo fornecera a

informação de quantos minutos a irrigação permanecerá ligada. Com o teclado capacitivo temos uma limitação, pois possui 16 teclas então o sistema poderá ficar ligado no máximo 16 minutos.

Esses testes têm o objetivo de avaliar todo o desempenho do protótipo e analisar se tudo que foi proposto está sendo atendido.

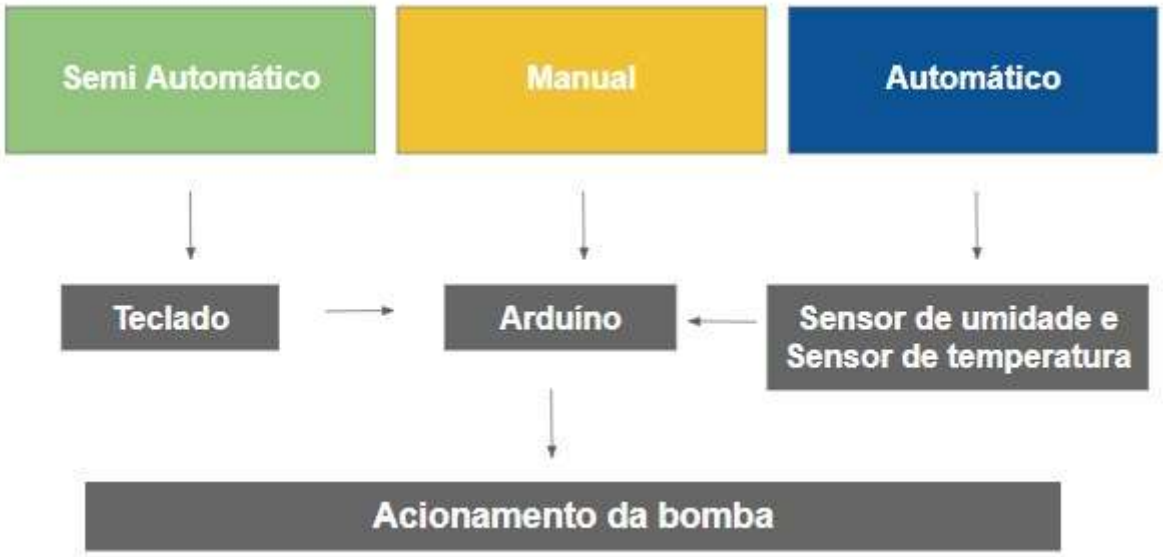
3.1 DESCRIÇÃO DO SISTEMA DE IRRIGAÇÃO PROPOSTO

A proposta deste projeto é desenvolver um sistema de irrigação automatizada, de forma a reduzir a mão de obra do agricultor e melhorar a qualidade dos frutos, visto que com um controle de umidade e temperatura correto pode-se amenizar várias doenças. Segue abaixo esquema demonstrativo do funcionamento do protótipo.

Para realizar o controle da automação foram utilizados 3 microcontroladores Arduino UNO um para cada modo operante. Poderia ser utilizado apenas 1 microcontrolador, como o Arduino MEGA, que possui mais portas analógicas e digitais. Devido a disponibilidade das placas de Arduinos UNO, optou-se por empregá-los no protótipo.

A utilização de 3 Arduinos UNO pode nos trazer uma grande vantagem, pois se algum deles parar de funcionar temos mais dois modos para operar, assim garantido que a plantação não fique sem ser irrigada. Ou seja, com a utilização de três Arduino a probabilidade de a plantação fique sem irrigar e a planta prejudicada é menor.

Figura 34 - Esquema do protótipo.



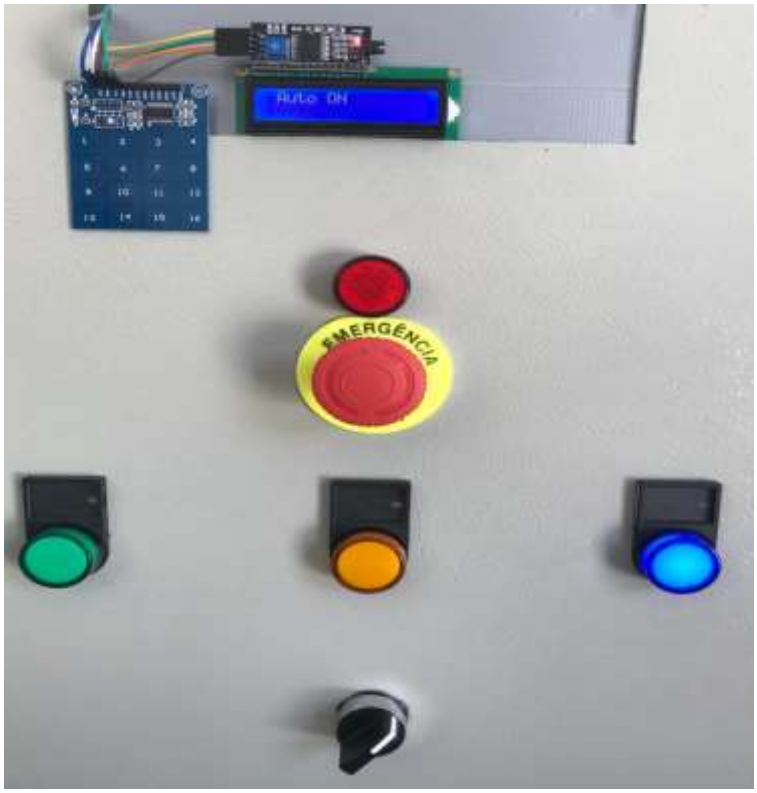
Fonte: Elaboração do autor (2020).

3.2 MODO AUTOMÁTICO

Quando a chave seletora estiver girada em modo automático acenderá o Led azul assim o operador identificará o modo operante. Automaticamente o sistema começa a fazer a leitura do sensor de umidade, se haver a necessidade de irrigar o Arduino irá acionar o módulo relé que ligará a bomba de irrigação e o *display* LCD irá descrever (Auto *ON*). Se o sistema não necessitar de irrigação aparecerá no *display* LCD (Auto *OFF*). Nas figuras abaixo observa-se o modo automático em modo *ON/OFF*.

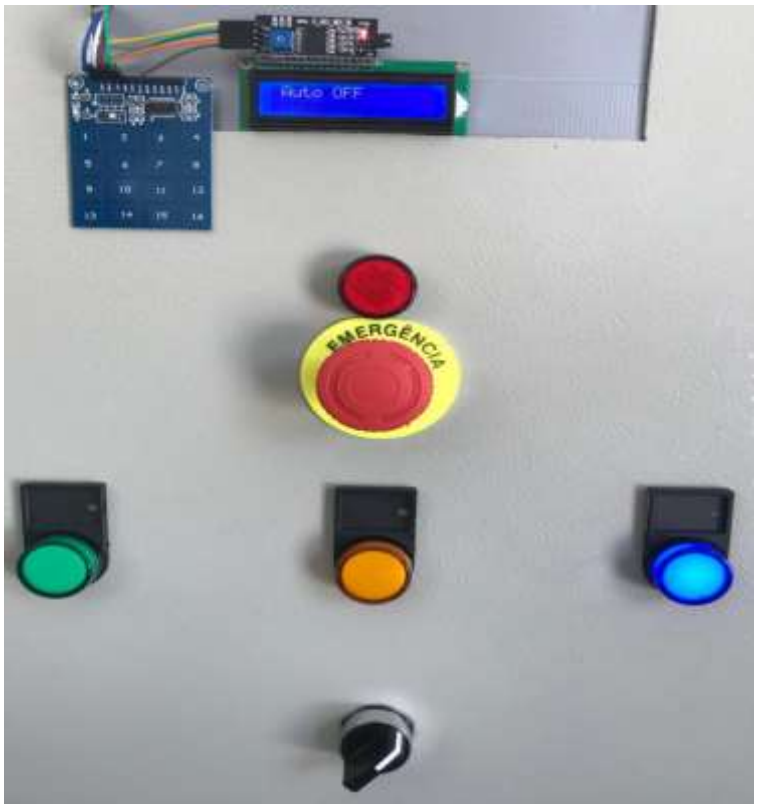
O funcionamento do modo automático dependerá da leitura dos sensores de umidade e temperatura que estrão alocados no campo. Na figura 35 e 36 mostra o funcionamento do *display* LCD no modo automático.

Figura 35 - *Display* Auto *ON*



Fonte: Elaboração do autor (2020).

Figura 36 - *Display* Auto *OFF*



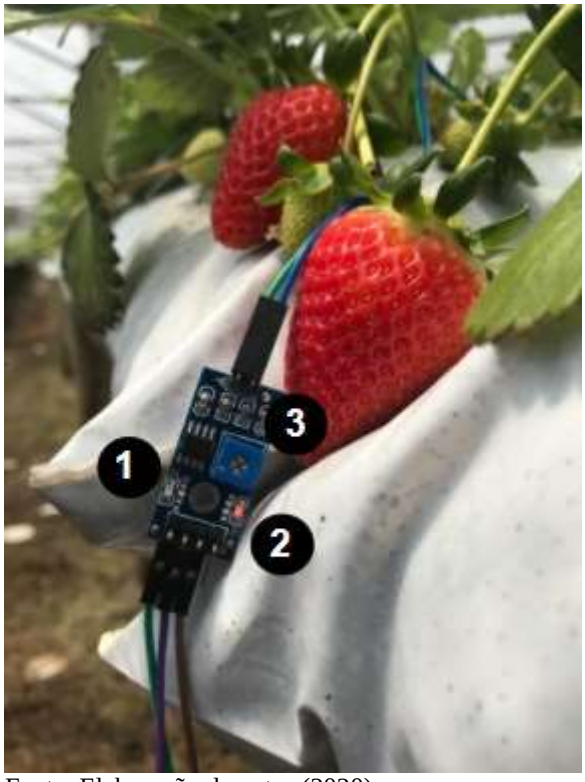
Fonte: Elaboração do autor (2020).

3.2.1 Sensor de umidade

O sensor de umidade é introduzido no substrato, onde fica constantemente fazendo a leitura da umidade, assim averiguando se há necessidade irrigar ou não. Na figura 37 pode-se observar o sensor: quando ela está com os pontos 1 e 2 acesos necessita a irrigação, então enviará para o Arduino que comandará a ligação da bomba. No ponto 3 se encontra um potenciômetro onde o operador regula sensibilidade do sensor.

Recomenda-se o encapsulamento do componente eletrônico onde se encontra o potenciômetro que regula a sensibilidade do sensor, pois ele se encontra em uma área onde está sujeito a molhar, podendo então danificar o componente.

Figura 37 - Sensor com solo úmido



Fonte: Elaboração do autor (2020).

3.2.2 Código sensor de umidade

No cultivo do morango utilizou-se sensor de umidade para que o protótipo se torne totalmente automático, sem dependência de um operador, onde o sensor constantemente faz a leitura da umidade. Quando houver a necessidade de irrigação, o sensor enviará para o Arduino que irá ligar a bomba de irrigação. O código abaixo representa o trabalho realizado pelo sensor de umidade.

```
#include <LiquidCrystal_I2C.h>
#define rele 12 // define o pino do rele no Arduino
#define sens 8 // define o pino do sensor no Arduino
#define endereco  0x27 // Endereços comuns: 0x27, 0x3F
#define colunas   16
#define linhas    2
LiquidCrystal_I2C lcd(endereco, colunas, linhas);
void setup() {
Serial.begin(9600); //inicializa comunicação serial
pinMode(rele, OUTPUT);//define o rele como saída
pinMode(sens, INPUT);
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
```

```
lcd.print(" Auto OFF"); // escreve no display
}
void loop() {
if(!digitalRead (sens)) // leitura do sensor
digitalWrite(rele, LOW); //liga rele se a umidade estiver baixa
else digitalWrite(rele,HIGH); //desliga rele quando alcançar umidade desejada
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); //liga a iluminação do display
lcd.clear(); //limpa o display
lcd.print(" Auto ON"); // escreve no display
}
```

3.2.3 Sensor de temperatura

O sensor de temperatura fica constantemente fazendo a leitura da temperatura, quando a temperatura for maior e igual a 28°C irá ligar a bomba que, por meio de aspersores, tentará amenizar a temperatura, ajudando assim no controle de pragas e doenças. Na figura 38 podemos observar o funcionamento do programa, onde para temperaturas maior ou igual a 28ºC aciona a bomba para o funcionamento dos aspersores. Na figura 39 mostra o sensor de temperatura alocado no campo, e na figura 40 se observa o funcionamento dos aspersores.

Figura 38 - Monitoramento sensor de temperatura

COM6

```
Sensor DS18B20
22 graus C DESLIGADO
Sensor DS18B20
22 graus C DESLIGADO
Sensor DS18B20
22 graus C DESLIGADO
Sensor DS18B20
26 graus C DESLIGADO
Sensor DS18B20
27 graus C DESLIGADO
Sensor DS18B20
28 graus C LIGADO
Sensor DS18B20
28 graus C LIGADO
Sensor DS18B20
28 graus C LIGADO
Sensor DS18B20
29 graus C LIGADO
Sensor DS18B20
28 graus C LIGADO
Sensor DS18B20
27 graus C DESLIGADO
Sensor DS18B20
27 graus C DESLIGADO
```

Fonte: Elaboração do autor (2020).

Figura 39 - Sensor de temperatura no campo

Fonte: Elaboração do autor (2020).

Figura 40 - Aspersores

Fonte: Elaboração do autor (2020).

3.3.4 Código sensor de temperatura

Para controle de temperatura se utilizou sensor de temperatura DS18B20, onde, em modo automático, fica constantemente fazendo a leitura da temperatura. Quando atingir 28°C, envia o comando para o Arduino ligar a bomba que, por meio de aspersores, amenizará a temperatura. O código abaixo representa a rotina de controle pelo sensor de temperatura.

```
#include <Wire.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS 10
#define sens 7 // define o pino do sensor
int rele = 5; // define o pino do rele
int rele2 =6; // define o pino do rele2
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
DeviceAddress insideThermometer = { 0x28, 0xB9, 0xC5, 0x16, 0xA8, 0x01, 0x3C,
0x44 };
void printTemperature(DeviceAddress deviceAddress);
void setup()
{
Serial.begin(9600);   //inicializa comunicação serial
sensors.begin();      //inicializa sensores
 sensors.setResolution(insideThermometer, 10); //configura para resolução de 10
bits
```

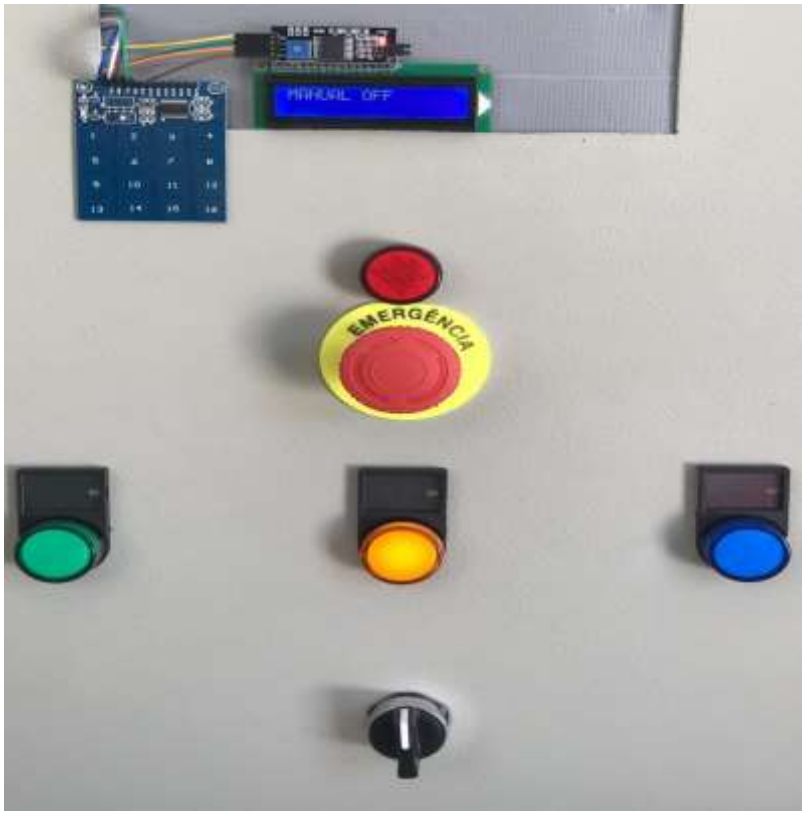
```
 pinMode (rele, OUTPUT); //define rele como saída
 pinMode (rele2,OUTPUT); //define rele como saída
 pinMode(sens, INPUT); //define sensor como entrada
 digitalWrite(rele2, LOW);
} //end setup
void loop(void)
{
  if(!digitalRead(sens))  digitalWrite(rele2,HIGH);  //  faz  comparação  com  a
temperatura do     ambiente e liga a bomba caso necessário
 else digitalWrite(rele2,LOW);
 Serial.println("Sensor DS18B20");
 sensors.requestTemperatures();
 printTemperature(insideThermometer);
}
void printTemperature(DeviceAddress deviceAddress)
{
 int tempC = sensors.getTempC(deviceAddress);
 if (tempC == -127.00)
  {
 Serial.print("Erro de leitura");
  }
 else
 {
  Serial.print(tempC);
  Serial.print(" graus C ");
}
if (tempC >= 28) // Escolha a temperatura de referência para ligar e desligar
{
digitalWrite (rele, 0);
Serial.println ("LIGADO"); // escreve na serial
}
else
{
digitalWrite (rele, 1);
Serial.println ("DESLIGADO"); // escreve na serial
}
delay(3000);
}
```

3.3 MODO MANUAL

Quando a chave seletora estiver posicionada no meio acenderá a luz amarela, assim o operador identificará que o modo manual está em operação. O *display* LCD irá descrever (MANUAL *OFF*). Sua programação foi feita para respeitar um lapso de 5 segundos, para que não atue quando a chave seletora passar pelo modo manual. Se o operador desejar passar do modo semi automático para o modo automático, a chave seletora obrigatoriamente passa pelo modo manual.

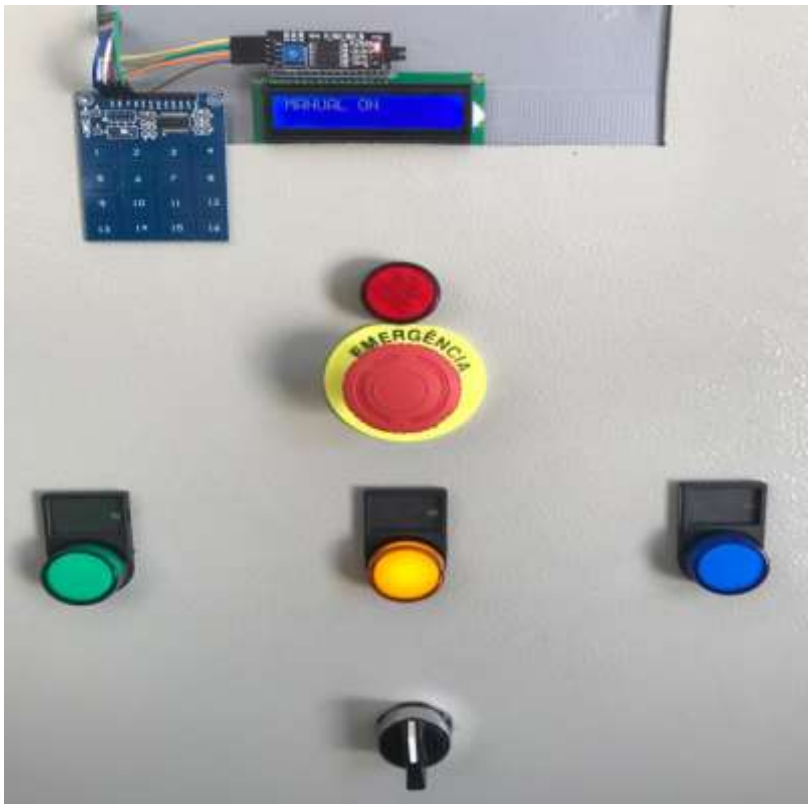
Esta precaução foi tomada para que não houvesse nenhum dano ao sistema de encanamento, visto que em modo manual o operador terá que abrir através do manuseio da válvula. Então sempre que o operador desejar irrigar em modo manual primeiro terá que abrir a válvula solenoide, para depois girar a chave seletora para o modo manual. Após os 5 segundos o *display* LCD irá descrever (MANUL *ON*). O código a ser implementado no microcontrolador Arduino encontra-se no Apêndice C. Nas figuras 41 e 42 observamos o funcionamento do modo manual. E nas figuras 43 e 44 pode-se observar como operar para abrir manualmente a válvula solenoide.

Figura 41 - Manual *OFF*



Fonte: Elaboração do autor (2020).

Figura 42 - Manual *ON*



Fonte: Elaboração do autor (2020).

Figura 43 - Solenoide fechada

Fonte: Elaboração do autor (2020).

Figura 44 - Solenoide aberta

Fonte: Elaboração do autor (2020).

3.3.1 Código modo manual

Para controle manual do protótipo o operador terá que abrir válvula solenoide antes de posicionar a chave seletora no modo manual, a fim de não danificar nenhuma tubulação. O código abaixo representa o trabalho realizado no modo manual, onde após 5 segundos da chave seletora posicionada no respectivo modo irá ligar a bomba de irrigação.

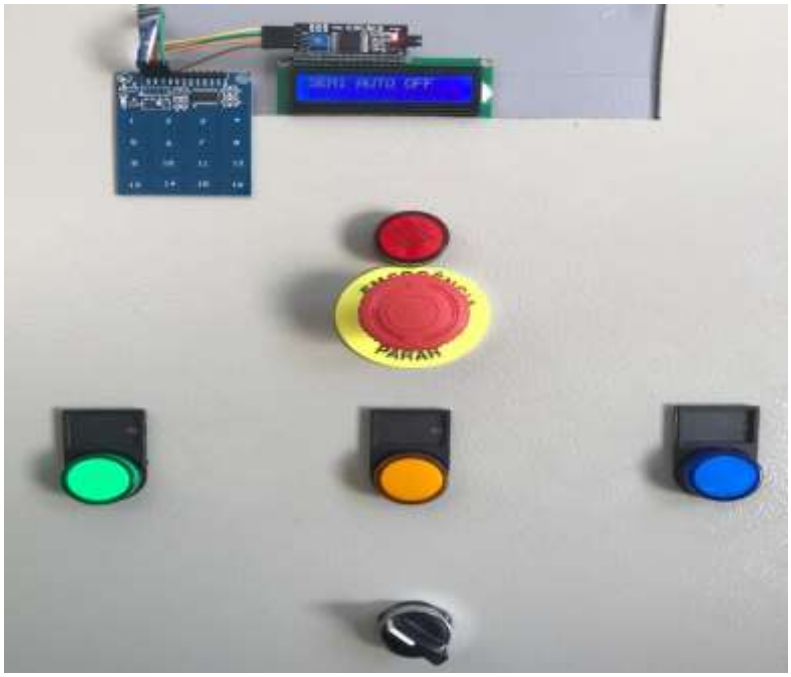
```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#define rele 10 // define pino do rele no Arduino
#define manual 13 // define pino manual
#define endereco  0x27 // Endereços comuns: 0x27, 0x3F
#define colunas   16
#define linhas    2
LiquidCrystal_I2C lcd(endereco, colunas, linhas);
void setup() {
 lcd.init(); // inicia a comunicação com o display
 lcd.backlight(); // liga a iluminação do display
 lcd.clear(); // limpa o display
 lcd.print("MANUAL OFF"); // escreve no display
 delay(5000);// delay de 5 segundos
 lcd.setCursor(0, 1);
 lcd.noBacklight(); // desliga a iluminação do display
 delay(2000); // delay de 2 segundos
 lcd.backlight(); // liga a iluminação do display
delay(2000); // delay de 2 segundos
pinMode(rele, OUTPUT);
pinMode (manual, INPUT); // recebe manual como entrada
digitalWrite(rele, LOW);
 lcd.clear(); // limpa o display
}
void loop() {
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
 lcd.print("MANUAL OFF"); // escreve no display
 lcd.setCursor(0, 1);
 if(!digitalRead(manual)) digitalWrite(rele, HIGH); // se manual for ativado liga
rele
 else digitalWrite(rele, LOW);
 lcd.init(); // inicia a comunicação com o display
 lcd.backlight(); // liga a iluminação do display
 lcd.clear(); // limpa o display
 lcd.print("MANUAL ON"); // escreve no display
}
```

3.4 MODO SEMI AUTOMATICO

Quando a chave seletora estiver girada em modo semi automático o *display* LCD irá descrever (SEMI AUTO *OFF*), indicando que o sistema está desligado. Nos objetivos iniciais o operador iria entrar com as informações pela IHM para que a irrigação ficasse ligada por um determinado tempo ligado. Como no decorrer do desenvolvimento houve um problema técnico com a IHM, buscou-se uma nova solução.

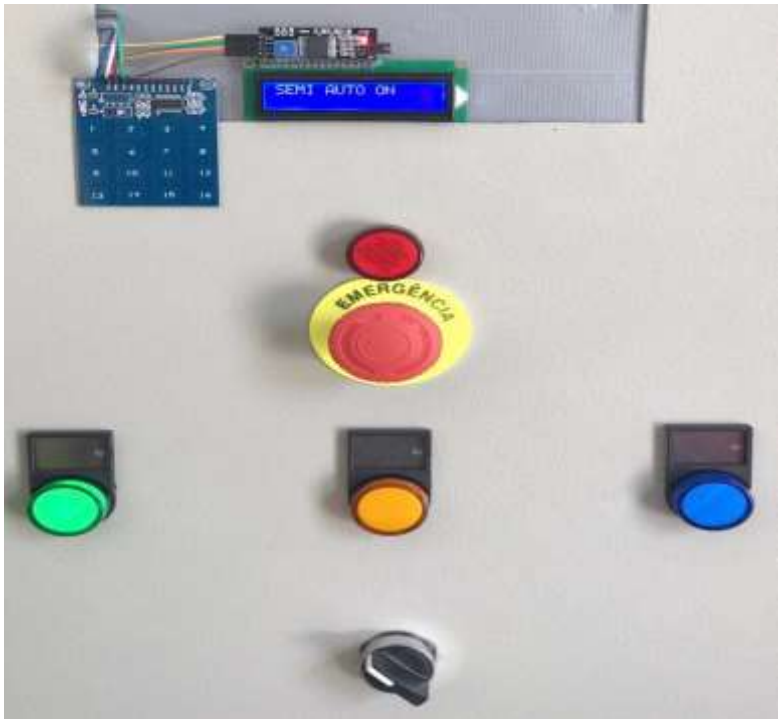
A solução encontrada foi a utilização de um teclado capacitivo que possui 16 teclas onde cada tecla corresponde ao minuto desejado, podendo manter a irrigação ligada até 16 minutos. O operador irá definir quanto minutos manterá o sistema de irrigação em funcionamento, e o *display* LCD descreverá (SEMI AUTO *ON*). Nas figuras 45 e 46 pode-se observar o funcionamento do sistema.

Figura 45 - Semi auto *OFF*



Fonte: Elaboração do autor (2020).

Figura 46 - Semi auto *ON*



Fonte: Elaboração do autor (2020)

3.4.1 Código modo semi automático

Para controle semi automático do protótipo o operador terá que informar quantos minutos deseja que a irrigação permaneça ligada, para isso colocou-se um teclado capacitivo de 16 teclas onde cada número corresponde ao minuto. O código representado abaixo faz o controle do modo semi automático.

```
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
#include <TTP229.h>
#define rele 12 // define pino do rele no Arduino
#define SCL_PIN 8 // define pino do SCL no Arduino
#define SDO_PIN 9 // define pino do SDO no Arduino
#define endereço 0x27 // endereços comuns: 0x27
#define colunas   16
#define linhas    2
byte key;
LiquidCrystal_I2C lcd(endereco, colunas, linhas);
 void setup() {
 Serial.begin(9600); //inicializa comunicação serial
 pinMode (SCL_PIN, OUTPUT); // define SCL como saída
 pinMode (SDO_PIN, INPUT); // define SDO como entrada
 pinMode (rele, OUTPUT); // define rele como saída
 lcd.init(); // inicia a comunicação com o display
```

```
 lcd.backlight(); // liga a iluminação do display
 lcd.clear(); /limpa o display
 lcd.print("SEMI AUTO OFF");// escreve no display
 delay(5000);// delay de 5 segundos
 lcd.setCursor(0, 1);
 lcd.noBacklight(); // desliga iluminação do display
 delay(2000); // delay de 2 segundos
 lcd.backlight(); // liga a iluminação do display
 digitalWrite (led, HIGH);
}
void loop() {
 key = Read_keypad();
 if (key ==1){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 1 minuto
  delay (60000);
  digitalWrite (led, HIGH);
  }
  if (key ==2){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 2 minuto
  delay (120000);
  digitalWrite (led, HIGH);
  }
  if (key ==3){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 3 minuto
  delay (180000);
  digitalWrite (led, HIGH);
  }
  if (key == 4){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 4 minuto
```

```
delay (240000);
digitalWrite (led, HIGH);
}
if (key == 5){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 5 minuto
delay (300000);
digitalWrite (led, HIGH);
}
 if (key == 6){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 6 minuto
delay (360000);
digitalWrite (led, HIGH);
}
if (key == 7){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 7 minuto
delay (420000);
digitalWrite (led, HIGH);
}
if (key == 8){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 8 minuto
delay (480000);
digitalWrite (led, HIGH);
}
if (key == 9){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 9 minuto
```

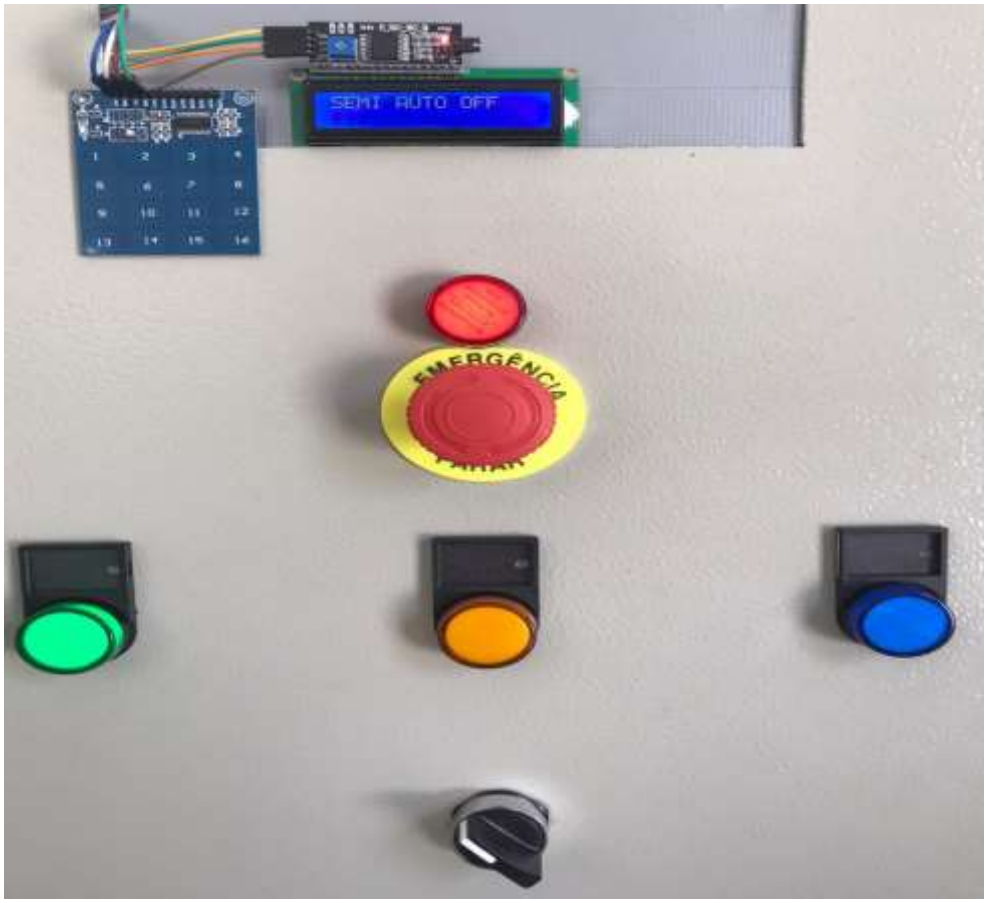
```
  delay (540000);
  digitalWrite (led, HIGH);
  }
  if (key == 10){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 10 minuto
  delay (600000);
  digitalWrite (led, HIGH);
  }
  if (key == 11){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 11 minuto
  delay (660000);
  digitalWrite (led, HIGH);
  }
  if (key == 12){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON");//escreve no display
  digitalWrite(led,LOW);// liga a bomba por 12 minuto
  delay (720000);
  digitalWrite (led, HIGH);
  }
  if (key == 13){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 13 minuto
  delay (780000);
  digitalWrite (led, HIGH);
    }
  Serial.println(key);
  delay(200);
  if (key == 14){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
```

```
  lcd.print("SEMI AUTO ON");//escreve no display
  digitalWrite(led,LOW);// liga a bomba por 14 minuto
  delay (840000);
  digitalWrite (led, HIGH);
    }
  if (key == 15){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 15 minuto
  delay (900000);
  digitalWrite (led, HIGH);
    }
  if (key == 16){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 16 minuto
  delay (960000);
  digitalWrite (led, HIGH);
}
 Serial.println(key);
 delay(200);
}
 byte Read_keypad(void)
 {
  byte Count;
  byte key_State = 0;
  for (Count = 1; Count <= 16; Count++)
  {
   digitalWrite(SCL_PIN,LOW);
   if (!digitalRead(SDO_PIN))
   key_State = Count;
   digitalWrite (SCL_PIN,HIGH);
  }
        return key_State;
 }
```

3.5 BOTÃO DE EMERGENCIA

O botão de emergência serve para o operador interromper o funcionamento da irrigação caso haja alguma irregularidade e necessite de uma parada rápida, basta apertar o botão que irá interromper o funcionamento, o botão de emergência se aplica para todos os modos de operação. Quando pressionado o botão fica retido e acenderá a luz vermelha, para o sistema voltar a operar, basta girar o botão e a luz vermelha se apagará. Na figura 47 abaixo se pode observar o botão pressionado.

Figura 47 - Botão de emergência pressionado



Fonte: Elaboração do autor (2020).

3.6 CUSTO DO PROTÓTIPO

A escolha por este protótipo teve como objetivo uma melhor relação entre custo e benefício, de modo que pequenos produtores pudessem adquiri-los. A seguir está sendo representada a tabela 6 com os componentes necessário para elaboração do protótipo juntamente com o preço de cada produto.

Tabela 6 - Custo dos componentes

Descrição	**Valor (R$)**	**Quantidade**	**Total (R$)**
DPS	40	4	160
Disjuntor Trifásico	64	1	64
Relé Falta de Fase	74	1	74
Contatora Botão Emergência	62	1	62
Fonte Chaveada (24V)	49,10	1	49,10
Disjuntor Motor	109,99	1	109,99
Contator motor	48	1	48
Módulo Relé	15,90	1	15,90
Borne Terra	15,50	1	15,50
Borne Neutro	15,50	1	15,50
Teclado Capacitivo	19,90	1	19,90
Display LCD	18,90	1	18,90
Módulo IC2	11,90	1	11,90
Sensor de Umidade	9,90	1	9,90
Sensor de Temperatura	16,90	1	16,90
Led	5,95	4	23,8
Botão de Emergência	14,87	1	14,87
Chave Seletora	12,22	1	12,22
Arduino	43,90	3	131,7
Protoboard	16,90	1	16,90
Trilho DIN	9,86	2	19,72
Canaleta	46,53	1	46,53
Caixa Quadro Comando	360	1	360
Fio	42,90	1	42,90
			Total = 1360,13

Fonte: Elaboração do autor (2020).

O propósito inicial era a implementação de uma IHM no protótipo que o deixaria mais robusto. No entanto, por conta de problemas técnicos, tornou-se necessário substituí-la por um teclado capacitivo, *display* LCD e o módulo de comunicação I2C, que se mostrou muito interessante pelo custo pagos nos mesmos. A implementação da IHM acresceria um valor de

R$ 1350,00 no protótipo final, este valor se torna significante pelo fato de ser maior do que o valor gasto em todo o protótipo.

Este valor se mostra satisfatório comparado com controladores prontos encontrados na *internet*, pois o protótipo alcançou o mesmo valor dos encontrados, mas com uma diferença, os encontrados na *internet* não possuem as proteções adotadas neste protótipo para os componentes eletrônicos e a bomba.

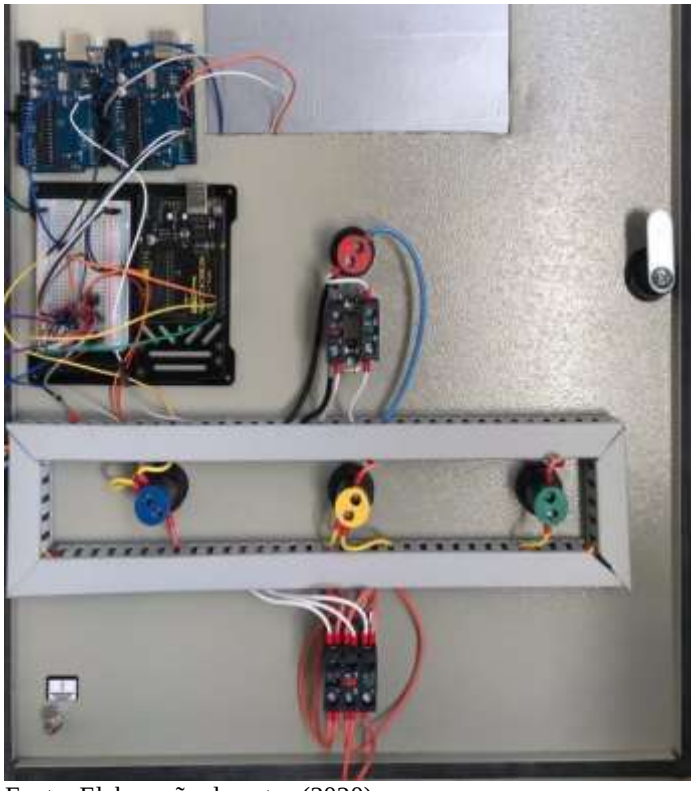
3.7 PROTÓTIPO FINAL

Na elaboração do projeto foi desenvolvido um protótipo para automatizar a irrigação e controlar a temperatura, com inúmeros benefícios ao produtor, pois quando automatizado o afazer, o produtor terá a disponibilidade para exercer outras funções em sua produção. Podendo também reduzir pragas e doenças com o manejo correto da umidade e temperatura e aumentando a produção de sua plantação. Nas figuras 48, 49 e 50 abaixo pode-se observar a versão final do protótipo.

Figura 48 - Protótipo Final

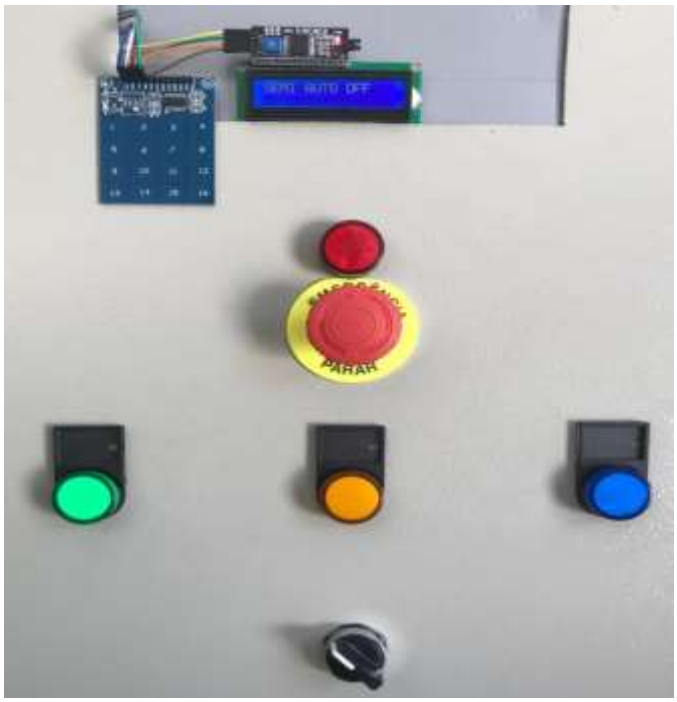
Fonte: Elaboração do autor (2020).

Figura 49 - Protótipo Final



Fonte: Elaboração do autor (2020).

Figura 50 - Protótipo Final



Fonte: Elaboração do autor (2020).

4 CONCLUSÃO

Visto a importância do setor agrícola na economia do país e como fonte de sustento para muitas famílias brasileiras, a implementação de tecnologias nesse âmbito traz inúmeras vantagens e benefícios ao produtor, facilitando a mão de obra e cultivo morangueiro, visando uma melhor produtividade.

O sistema de irrigação automatizado é uma das tecnologias que vem sendo aplicada, permitindo um melhor controle hídrico da plantação, além de otimizar o tempo, possibilitando que o produtor realize outras funções. O mesmo pode ser utilizado de forma totalmente automática, sendo baseado em um monitoramento feito por um sensor de umidade que dirá quando o sistema de irrigação deve ser ativado. Adicionalmente, o modo semi automático confere ao operador capacidade para decidir quantos minutos a irrigação ficará ligada, e no modo manual o operador terá que abrir manualmente a válvula solenoide e ativar a bomba.

Dentre as possíveis formas de automatizar esse sistema, a placa Arduíno é o que apresentou um bom custo benefício, viabilizando a instalação para pequenos produtores, permitindo, portanto, que estes também tenham acesso a um sistema de eficiência, que lhes garantirá um acréscimo na produtividade.

Foram desenvolvidos *softwares* em linguagem em C, para a utilização de controle dos três Arduinos aplicados no projeto. Para cada modo de operação foi criado um *software* específico e aplicado em um determinado Arduino, sendo capaz de atender os objetivos propostos no projeto.

O sistema proposto foi submetido a diversos testes por meio da aplicação do protótipo no campo, nos quais foi possível verificar a eficiência dos sensores e o controle por parte do Arduino em modo automático. Também foi possível certificar o teclado capacitivo que programa o Arduino em modo semi automático e o modo manual operando após cinco segundos a chave seletora devidamente posicionada. Em todos os testes submetidos o sistema funcionou corretamente, atendendo assim o propósito inicial de automatizar o sistema de irrigação de cultivo de morango.

Com isso o objetivo de implementar um sistema de irrigação automatizado a custos acessíveis para pequenos agricultores se realizou com sucesso. O objetivo inicial era de implementação de uma IHM para monitoramento, entrada de informação para o sistema semiautomático e manual. Mas por problemas técnicos teve que se buscar outra solução, que foram encontradas com a aplicação do *display* LCD e o teclado capacitivo para realizar estas

funções. Os substitutos apresentaram o resultado desejado, capaz de cumprir com os objetivos propostos inicialmente, com um custo relativamente menor.

4.1 TRABALHOS FUTUROS

Pesquisar um sensor de umidade que seja mais eficaz, assim trazendo um melhor desempenho. Utilizar uma resistência de modo a aquecer o ambiente de cultivo, na incidência de temperaturas mais baixas, quando necessário, assim mantendo um controle de temperatura apropriado para cultivar o morango. Aplicar um *tablet* no lugar da IHM para a entrada de informações, visando um custo mais baixo do projeto. Realizar um *software* para aplicação do projeto de automação para a utilização apenas do Arduino Mega.

REFERÊNCIAS

AGÊNCIA NACIONAL DE ÁGUAS (Brasil). **Atlas irrigação**: uso da água na agricultura irrigada. Brasília: ANA, 2017.

ALMEIDA, I. R. de; ANTUNES, L. E. C.; JUNIOR, C. R.; STEINMETZ, S.; CARVALGO, F. L. C. **Potenciais regiões produtoras de morango durante a primavera e verão e riscos de ocorrência de geada na produção de inverno no Estado do Rio Grande do Sul**. Pelotas, RS: Embrapa Clima Temperado, 2009. (Comunicado Técnico 229).

ANTUNES, L. E. C.; FAGHERAZZI, A. F.; VIGNOLO, G. K. Morangos tem produção crescente. **Campo & Lavoura**, n. 1, p. 96-102, 2017.

BANZI, M. **Primeiros passos com Arduino**. Tradução Rafael Zanolli. São Paulo: Novatec Editora, 2011.

BORTOLOZZO, A. R Produção de morangos no sistema semi-hidropônico. **Embrapa Uva e Vinho**, Bento Gonçalves, 2007. Disponível em: https://www.infoteca.cnptia.embrapa.br/infoteca/bitstream/doc/541435/1/cir062.pdf. Acesso em: 23 out. 2020.

CETTI. **Chave seletora três posições**. [2020]. Disponível em: encurtador.com.br/ktBV6. Acesso em: 2 out. 2020.

CONCEIÇÃO, J. C. P. R; CONCEIÇÃO, P. H. Z. **Agricultura**: evolução e importância para a balança comercial brasileira. Brasília: IPEA, 2014. Disponível em: https://www.ipea.gov.br/portal/images/stories/PDFs/TDs/td_1944.pdf. Acesso em: 24 out. 2020.

CONFEDERAÇÃO DA AGRICULTURA E PECUÁRIA DO BRASIL (CNA). **Panorama do agro**. [2019]. Disponível em: https://www.cnabrasil.org.br/cna/panorama-do-agro. Acesso em: 3 set. 2020.

CUENCA, H. R. **Irrigation system design**: an engineering approach. New Jersey: PrenticeHall, 1989.

CUNHA, K. C. B. da; ROCHA, R. V. Automação no processo de irrigação na agricultura familiar com plataforma Arduíno. **RECoDAF**: Revista Eletrônica Competências Digitais para Agricultura Familiar, Tupã, v. 1, n. 2, p. 62-74, jul./dez., 2015.

EKBOIR, J. M. Research and technology policies in innovation systems: zero tillage in Brazil. **Research policy**, v. 32, n. 4, p. 573-586, abr. 2003. DOI: https://doi.org/10.1016/S0048-7333(02)00058-6. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S0048733302000586?via%3Dihub. Acesso em: 23 set. 2020.

EMBRAPA. **Morangueiro**. Brasília, DF: Embrapa, 2016. *E-book*. Disponível em: https://www.embrapa.br/busca-de-publicacoes/-/publicacao/1092843/morangueiro. Acesso em: 3 set. 2020.

EMBRAPA. *IV* Seminário Brasileiro Sobre Pequenas Frutas. Vacaria RS, 2007. Disponível em: https://ainfo.cnptia.embrapa.br/digital/bitstream/CNPUV/9582/1/doc059.pdf#page=63. Acesso em: Maio de 2020.

EVANS, M.; NOBLE, J.; HOCHENBAUM, J. **Arduino em ação**. Tradução Camila Paduan. São Paulo: Novatec Editora, 2013.

FEDER, G.; JUST, R. E.; ZILBERMAN, D. Adoption of agricultural innovations in developing countries: a survey. economic development and cultural change. **The University of Chicago Press Journals**, v. 32, n. 2, 1985. DOI: https://doi.org/10.1086/451461. Disponível em: https://www.journals.uchicago.edu/doi/10.1086/451461. Acesso em: 4 out. 2020.

FERENC, C. H. R. **Irrigação por superfície**: entenda como funciona. [2020]. Disponível em: https://www.cpt.com.br/dicas-cursos-cpt/irrigacao-por-superficie-entenda-como-ela-funciona. Acesso em: 23 set. 2020.

FERNANDES FILHO, G. E. F; DIAS, R.A. Comandos elétricos. – 1.ed. – São Paulo: Érica, 2014.

FILIPEFLOP. Regulador de Tensão 7805. [2020]. Disponível em: https://www.filipeflop.com/produto/regulador-de-tensao-7805-5v/. Acesso em: 28/11/2020.

GOILAV, N.; LOI, G. **Arduino**: aprender a desarrollar para crear objetos inteligentes. Cornellà de Llobregat, Barcelona: Eni Ediciones, 2016.

GRUPO DE ROBÓTICA. **Introdução ao Arduino**: notas de aula. Universidade Federal do Mato Grosso do Sul, 2012.

GUIMARÃES, Vinícius Galvão. **Automação e monitoramento de sistema de irrigação na agricultura**. Monografia (Graduação, Engenharia Mecatrônica) - Universidade de Brasília, Brasília, 2011.

HIDROPONIA BRASIL. Produzir morango em hidroponia. [**Blog**]. 2020. Disponível em: https://www.hidroponiabrasil.com/post/produzir-morango-em-hidroponia. Acesso em: 4 out. 2020.

HU, Y.; ZHANG, Y.; DUAN, Y. Improving Agricultural Information and Knowledge Transfer in Cambodia - Adopting Chinese Experience in Using Mobile Internet Technologies. *In*: LI D.; LI Z. (eds.) COMPUTER AND COMPUTING TECHNOLOGIES IN AGRICULTURE - CCTA, 9., 2015. **Anais** [...] IFIP Advances in Information and Communication Technology, v. 479. Springer, Cham, 2016.

LUCIETTI, Donato**. Irrigação das hortaliças**: cultivo orgânico. Santa Catarina, 2014. Disponível em: https://cultivehortaorganica.blogspot.com/2014/01/irrigacao-das-hortalicas.html. Acesso em:3 out. 2020.

MADEIRA, D. **DS18B20**: sensor de temperatura inteligente. Portal vida de silício, 25 jun. 2018. Disponível em: https://portal.vidadesilicio.com.br/sensor-de-temperatura-ds18b20/#O sensor de temperatura DS18B20. Acesso em: 23 out. 2020.

MAROUELLI, W. A.; SILVA, W. L. de C. e; SILVA, H. R. da. **Irrigação por aspersão em hortaliças**: qualidade da água, aspectos do sistema e método prático de manejo. Brasília: Embrapa, 2008. Disponível em: https://www.embrapa.br/busca-de-publicacoes/-/publicacao/762590/irrigacao-por-aspersao-em-hortalicas-qualidade-da-agua-aspectos-do-sistema-e-metodo-pratico-de-manejo. Acesso em: 3 out. 2020.

MARSCZAOKOSKI, F. G; CRUZ, R. P. da; SILVA, W. D. A. E. **Sistema microcontrolado de irrigação aplicado a morangueiros**. Monografia (Graduação em Industrial Elétrica - Ênfase em Automação) - Universidade Tecnológica Federal do Paraná, UTFPR, Curitiba, 2013. Disponível em: https://nupet.daelt.ct.utfpr.edu.br/tcc/engenharia/docequipe/2012_1_07/2012_1_07_final.pdf. Acesso em: 23 set. 2020.

MARTELETO, DOUGLAS. C. Avaliação do diodo emissor de luz (LED) para iluminação de interiores, 2011. Disponível em: https://pantheon.ufrj.br/bitstream/11422/8094/1/monopoli10003763.pdf Acesso em Novembro de 2020.

MEHL, EWALDO L. M. Fonte Chaveada. Universidade Federal do Paraná. 2004. Disponível em: http://www.cricte2004.eletrica.ufpr.br/mehl/downloads/FontesChaveadas.pdf Acesso em novembro.

MELLO, Jorge Luiz Pimenta; SILVA, Leonardo Duarte Batista da. **Apostila de irrigação**. Rio de Janeiro, 2007.

MORAES, C. C. de.; CASTRUCCI, P. de L. **Engenharia de automação industrial**. 2. ed. Rio de Janeiro: LTC, 2010.

MUNDO DA ELÉTRICA. **Botão emergência**. [2020]. Disponível em: https://www.mundodaeletrica.com.br/botoeira-de-emergencia-caracteristicas-e-aplicacoes/. Acesso em: 2 out. 2020.

NBR-5410. Normas Brasileiras, ABNT NBR 5410 ano 2004. Disponível em: < https://docente.ifrn.edu.br/jeangaldino/disciplinas/2015.1/instalacoes-eletricas/nbr-5410\> Acesso em Novembro de 2020. (NBR-5410, 2004)

OLIVEIRA, C. L. V, ZANETTI, H. A. P. Arduino Descomplicado: como elaborar de eletrônica – São Paulo: Érica, 2015.

OLIVEIRA, Euler. Como usar com Arduino: teclado capacitivo touch TTP229 com 16 teclas. **Blog MasterWalker**, [2020]. Disponível em: https://blogmasterwalkershop.com.br/arduino/como-usar-com-arduino-teclado-capacitivo-touch-toque-ttp229-com-16-teclas/. Acesso: 09 nov. 2020.

PEDRERA, A. C. **Arduino para principiantes**. [*S. l.*]: CreateSpace Independent Publishing Platform, 2017.

PEÑARANDA, R. E. C. Sistema de irrigação automatizado pela umidade do solo. Curitiba 2018. Disponível em: http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/13882/1/CT_CEIOT_I_2018_13.pdf Acesso em Novembro de 2020.

RIBEIRO, M. A. A**utomação industrial**. 4. ed. Salvador: Tek Treinamento & Consultoria, 1999.

ROBÔ LIVRE. Módulo relé 5V. [2020]. Disponível em: http://www.roboliv.re/conteudo/modulo-rele-automacao. Acesso em: 12 set. 2020.

ROBOCORE. Módulo I2C: primeiros passos. **Robocore**, São Paulo, [2020]. Disponível em: https://www.robocore.net/tutorials/primeiros-passos-com-modulo-i2c#:~:text=O%20M%C3%B3dulo%20LCD%20I%C2%B2C%20ou,apenas%20duas%20linhas%20de%20dados. Acesso: 09 nov. 2020.

SANHUEZA, R. M. V. Produção de morangos no sistema semi-hidropônico. *In*: SEMINÁRIO BRASILEIRO SOBRE PEQUENAS FRUTAS, 4., 2007. **Anais** [...]. Vacaria, RS: Embrapa, jul. 2007. Disponível em: https://ainfo.cnptia.embrapa.br/digital/bitstream/CNPUV/9582/1/doc059.pdf#page=63. Acesso em: 4 jul. 2020.

SARITAS, O.; KUZMINOV, I. Global challenges and trends in agriculture: impacts on Russia and possible strategies for adaptation. **Foresight**, v. 19, n. 2, p. 218-250, 2017. DOI: https://doi.org/10.1108/FS-09-2016-0045. Disponível em: https://www.emerald.com/insight/content/doi/10.1108/FS-09-2016-0045/full/html. Acesso em: 4 out. 2020.

SENAR (Santa Catarina). **Morango**: lavouras suspensas garantem mais benefícios ao produtor. Florianópolis: Senar, 2020. Disponível em: http://www2.senar.com.br/Noticias/Detalhe/8627. Acesso em: 4 out. 2020.

SERVIÇO NACIONAL DE APRENDIZAGEM RURAL (SENAR). **Irrigação**: gestão de sistemas por superfície. Brasília: Senar, 2019. [*E-book*]. Disponível em: https://www.cnabrasil.org.br/assets/arquivos/253-IRRIGA%C3%87%C3%83O.pdf. Acesso em: 23 set. 2020.

SHAMSHIRI, R.; WELTZIEN, C.; HAMEED, I.; YULE, I.; GRIFT, T.; BALASUNDRAM, S.; PITONAKOVA, L.; AHMAD, D.; CHOWDHARY, G. Research and development in agricultural robotics: a perspective of digital farming. International. **Journal of Agricultural and Biological Engineering**, v. 11, n. 4, p. 1-14, 2018. DOI: 10.25165/j.ijabe.20181104.4278. Disponível em: https://www.researchgate.net/publication/326929441_Research_and_development_in_agricultural_robotics_A_perspective_of_digital_farming. Acesso em: 15 out. 2020.

SOUZA, J. L.; RESENDE, P. **Manual de horticultura orgânica**. Viçosa: Aprenda fácil, 2003.

STEVAN JR., S. L. SILVA, R. A. **Automação** instrumentação industrial com Arduino: teoria e projetos. São Paulo: Érica, 2015.

VÁLVULA solenoide. [2020]. Disponível em: https://www.rainbird.com.br/paisagismo.php?page=produtos&cat=vavulas&produto=HV. Acesso em: 23 set. 2020.

WEIS, Olga. Interface de comunicação serial pinagem RS232. **Eltima Publishing**, [*S. l.*], 2020.

YUAN, B. Z.; SUN, J.; NISHIYAMA, S. Effect of drip irrigation on strawberry growth and yield inside a plastic greenhouse. **Biosystems Engineering**, v. 87, n. 2, p. 237-245, 2004.

APÊNDICES

APÊNDICE A – SENSOR TEMPERATURA

```
#include <Wire.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS 10
#define sens 7 // define o pino do sensor
int rele = 5; // define o pino do rele
int rele2 =6; // define o pino do rele2
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
DeviceAddress insideThermometer = { 0x28, 0xB9, 0xC5, 0x16, 0xA8, 0x01, 0x3C,
0x44 };
void printTemperature(DeviceAddress deviceAddress);
void setup()
{
Serial.begin(9600);   //inicializa comunicação serial
sensors.begin();      //inicializa sensores
 sensors.setResolution(insideThermometer, 10); //configura para resolução de 10
bits
 pinMode (rele, OUTPUT); //define rele como saída
 pinMode (rele2,OUTPUT); //define rele como saída
 pinMode(sens, INPUT); //define sensor como entrada
 digitalWrite(rele2, LOW);
} //end setup
void loop(void)
{
  if(!digitalRead(sens))  digitalWrite(rele2,HIGH);  //  faz  comparação  com  a
temperatura do     ambiente e liga a bomba caso necessário
 else digitalWrite(rele2,LOW);
 Serial.println("Sensor DS18B20");
 sensors.requestTemperatures();
 printTemperature(insideThermometer);
}
void printTemperature(DeviceAddress deviceAddress)
{
 int tempC = sensors.getTempC(deviceAddress);
 if (tempC == -127.00)
  {
 Serial.print("Erro de leitura");
  }
 else
 {
  Serial.print(tempC);
  Serial.print(" graus C ");
```

```
}
if (tempC >= 28) // Escolha a temperatura de referência para ligar e desligar
{
digitalWrite (rele, 0);
Serial.println ("LIGADO"); // escreve na serial
}
else
{
digitalWrite (rele, 1);
Serial.println ("DESLIGADO"); // escreve na serial
}
delay(3000);
}
```

APÊNDICE B – SENSOR DE UMIDADE

```
#include <LiquidCrystal_I2C.h>
#define rele 12 // define o pino do rele no Arduino
#define sens 8 // define o pino do sensor no Arduino
#define endereco  0x27 // Endereços comuns: 0x27, 0x3F
#define colunas   16
#define linhas    2
LiquidCrystal_I2C lcd(endereco, colunas, linhas);
void setup() {
Serial.begin(9600); //inicializa comunicação serial
pinMode(rele, OUTPUT);//define o rele como saída
pinMode(sens, INPUT);
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print(" Auto OFF"); // escreve no display
}
void loop() {
if(!digitalRead (sens)) // leitura do sensor
digitalWrite(rele, LOW); //liga rele se a umidade estiver baixa
else digitalWrite(rele,HIGH); //desliga rele quando alcançar umidade desejada
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); //liga a iluminação do display
lcd.clear(); //limpa o display
lcd.print(" Auto ON"); // escreve no display
}
```

APÊNDICE C – MODO MANUAL

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#define rele 10 // define pino do rele no Arduino
#define manual 13 // define pino manual
#define endereco  0x27 // Endereços comuns: 0x27, 0x3F
#define colunas   16
#define linhas    2
LiquidCrystal_I2C lcd(endereco, colunas, linhas);
void setup() {
 lcd.init(); // inicia a comunicação com o display
 lcd.backlight(); // liga a iluminação do display
 lcd.clear(); // limpa o display
 lcd.print("MANUAL OFF"); // escreve no display
 delay(5000);// delay de 5 segundos
 lcd.setCursor(0, 1);
 lcd.noBacklight(); // desliga a iluminação do display
 delay(2000); // delay de 2 segundos
 lcd.backlight(); // liga a iluminação do display
delay(2000); // delay de 2 segundos
pinMode(rele, OUTPUT);
pinMode (manual, INPUT); // recebe manual como entrada
digitalWrite(rele, LOW);
 lcd.clear(); // limpa o display
}
void loop() {
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
 lcd.print("MANUAL OFF"); // escreve no display
 lcd.setCursor(0, 1);
 if(!digitalRead(manual)) digitalWrite(rele, HIGH); // se manual for ativado liga
rele
 else digitalWrite(rele, LOW);
 lcd.init(); // inicia a comunicação com o display
 lcd.backlight(); // liga a iluminação do display
 lcd.clear(); // limpa o display
 lcd.print("MANUAL ON"); // escreve no display
}
```

APÊNDICE D – MODO SEMIAUTOMÁTICO

```
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
#include <TTP229.h>
#define rele 12 // define pino do rele no Arduino
#define SCL_PIN 8 // define pino do SCL no Arduino
#define SDO_PIN 9 // define pino do SDO no Arduino
#define endereço 0x27 // endereços comuns: 0x27
#define colunas   16
#define linhas    2
byte key;
LiquidCrystal_I2C lcd(endereco, colunas, linhas);
 void setup() {
 Serial.begin(9600); //inicializa comunicação serial
 pinMode (SCL_PIN, OUTPUT); // define SCL como saída
 pinMode (SDO_PIN, INPUT); // define SDO como entrada
 pinMode (rele, OUTPUT); // define rele como saída
 lcd.init(); // inicia a comunicação com o display
 lcd.backlight(); // liga a iluminação do display
 lcd.clear(); /limpa o display
 lcd.print("SEMI AUTO OFF");// escreve no display
 delay(5000);// delay de 5 segundos
 lcd.setCursor(0, 1);
 lcd.noBacklight(); // desliga iluminação do display
 delay(2000); // delay de 2 segundos
 lcd.backlight(); // liga a iluminação do display
 digitalWrite (led, HIGH);
}
void loop() {
 key = Read_keypad();
 if (key ==1){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 1 minuto
  delay (60000);
  digitalWrite (led, HIGH);
  }
  if (key ==2){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 2 minuto
```

```
delay (120000);
digitalWrite (led, HIGH);
}
if (key ==3){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 3 minuto
delay (180000);
digitalWrite (led, HIGH);
}
if (key == 4){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 4 minuto
delay (240000);
digitalWrite (led, HIGH);
}
if (key == 5){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 5 minuto
delay (300000);
digitalWrite (led, HIGH);
}
 if (key == 6){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 6 minuto
delay (360000);
digitalWrite (led, HIGH);
}
if (key == 7){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 7 minuto
```

```
delay (420000);
digitalWrite (led, HIGH);
}
if (key == 8){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 8 minuto
delay (480000);
digitalWrite (led, HIGH);
}
if (key == 9){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 9 minuto
delay (540000);
digitalWrite (led, HIGH);
}
if (key == 10){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 10 minuto
delay (600000);
digitalWrite (led, HIGH);
}
if (key == 11){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON"); //escreve no display
digitalWrite(led,LOW);// liga a bomba por 11 minuto
delay (660000);
digitalWrite (led, HIGH);
}
if (key == 12){
lcd.init(); // inicia a comunicação com o display
lcd.backlight(); // liga a iluminação do display
lcd.clear(); // limpa o display
lcd.print("SEMI AUTO ON");//escreve no display
digitalWrite(led,LOW);// liga a bomba por 12 minuto
```

```
  delay (720000);
  digitalWrite (led, HIGH);
  }
  if (key == 13){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 13 minuto
  delay (780000);
  digitalWrite (led, HIGH);
    }
  Serial.println(key);
  delay(200);
  if (key == 14){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON");//escreve no display
  digitalWrite(led,LOW);// liga a bomba por 14 minuto
  delay (840000);
  digitalWrite (led, HIGH);
    }
  if (key == 15){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o display
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 15 minuto
  delay (900000);
  digitalWrite (led, HIGH);
    }
  if (key == 16){
  lcd.init(); // inicia a comunicação com o display
  lcd.backlight(); // liga a iluminação do display
  lcd.clear(); // limpa o
  lcd.print("SEMI AUTO ON"); //escreve no display
  digitalWrite(led,LOW);// liga a bomba por 16 minuto
  delay (960000);
  digitalWrite (led, HIGH);
}
 Serial.println(key);
 delay(200);
}
 byte Read_keypad(void)
```

```
{
 byte Count;
 byte key_State = 0;
 for (Count = 1; Count <= 16; Count++)
 {
  digitalWrite(SCL_PIN,LOW);
  if (!digitalRead(SDO_PIN))
  key_State = Count;
  digitalWrite (SCL_PIN,HIGH);
 }
      return key_State;
}
```