Universidade Federal de Ouro Preto - UFOP
Escola de Minas
Colegiado do curso de Engenharia de Controle e Automação - CECAU

Higor Raniel Ribeiro Garibalde

AUTOMATIZAÇÃO E MONITORAMENTO DE SISTEMA DE ABASTECIMENTO DE ÁGUA

Monografia de Graduação em Engenharia de Controle e Automação

Ouro Preto, Abril de 2021

HIGOR RANIEL RIBEIRO GARIBALDE

AUTOMATIZAÇÃO E MONITORAMENTO DE SISTEMA DE ABASTECIMENTO DE ÁGUA

Trabalho de Conclusão de Curso submetido à Universidade Federal de Ouro Preto, como requisito necessário para obtenção do grau de Bacharel em Engenharia de Controle e Automação

Orientador: Diógenes Viegas Mendes Ferreira

Ouro Preto, Abril de 2021

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
ESCOLA DE MINAS
DEPARTAMENTO DE ENGENHARIA CONTROLE E AUTOMACAO

FOLHA DE APROVAÇÃO

Higor Raniel Ribeiro Garibalde

Automatização e Monitoramento de Sistema de Abastecimento de Água

Monografia apresentada ao Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Engenheiro de Controle e Automação

Aprovada em 15 de abril de 2021

Membros da banca

Ms. Diógenes Viegas Mendes Ferreira - Universidade Federal de Ouro Preto
Dr. Agnaldo Jose da Rocha Reis - Universidade Federal de Ouro Preto
Dr. Paulo Marcos de Barros Monteiro - Universidade Federal de Ouro Preto

Diógenes Viegas Mendes Ferreira, orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 15/04/2021

Documento assinado eletronicamente por **Diogenes Viegas Mendes Ferreira**, **TECNICO DE LABORATORIO AREA**, em 09/06/2021, às 17:41, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0159923** e o código CRC **40261DEF**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35400-000
Telefone: 3135591533 - www.ufop.br

Agradecimentos

Enfim, chegamos à etapa mais esperada da graduação. Com muita luta e determinação, mais um ciclo da minha vida se encerra. Neste momento, é hora de olhar para trás apenas com sensação de dever cumprido e gratidão. Gratidão, esse é o sentimento que quero sempre carregar comigo, pois vivemos em uma sociedade onde não alcançamos nenhum objetivo sozinhos. Hoje agradeço primeiramente à Deus, que sempre me guiou e iluminou nas decisões e momentos de fraqueza. Agradeço aos meus pais Lúcio e Elceia e meu irmão Aller, que formaram meu alicerce e nunca deixaram de acreditar em mim. Agradeço ao meu tio Garibaldi, que além de me ceder moradia e oferecer condições para que eu pudesse estudar com tranquilidade, se tornou um segundo pai e grande amigo. Agradeço à Saozita pelos ensinamentos e lições de vida. Gratidão ao Clebis pela amizade verdadeira e parceria que levarei para o resto da vida. Aos amigos da UFOP, Regina, Luiz, Thiago, Caio, Yuri, Rodolfo e Pedro, os momentos mais divertidos e emocionantes na graduação certamente foram ao lado de vocês. Aos amigos conquistados em Ouro Preto, Marquinhos, Santino, Gabriel e Leleo, obrigado pelos finais de semanas aleatórios. Gratidão à Prof. Francisca pela primeira oportunidade de participar de um projeto de Extensão na UFOP e ao Prof. Alan Kardec pela oportunidade de participação em um projeto de Iniciação Científica. Agradeço ao Prof. Agnaldo pelo apoio em meu início na empresa Automic Jr., experiência que agregou muito em minha vida profissional. Ao Prof. Paulo Monteiro, meu muito obrigado por tornar esse curso possível e por não medir esforços para disseminar seu rico conhecimento aos aluno Agradeço ao meu Orientador Diógenes Viegas, que me direcionou nessa etapa final. Agradeço ao DECAT e seus professores, por dedicarem seu tempo à formação de profissionais e me guiarem até aqui.

Por fim, gratidão à UFOP, instituição que me forneceu estrutura para que meu sonho se realizasse e que me honrarei sempre em dizer que fui graduado na grandiosa Escola de Minas.

Resumo

A precariedade em volta da disponibilidade de água potável no século XXI é um tema em pauta nos Órgãos Ambientais e uma problemática que vêm trazendo transtornos à população. Tanto os fatores de mudanças climáticas, quanto os fatores causados por interferência humana têm contribuído para a situação atual. A gestão responsável da utilização do recurso natural é um dos caminhos para amenizar esse cenário caótico que se desenha para um futuro próximo. A acessibilidade à tecnologia trás algumas alternativas eficazes na gestão dos recursos hídricos. Baseado nisso, o presente projeto propõe um sistema de monitoramento automático de abastecimento capaz de reduzir os desperdícios de água costumeiros em sistemas manuais. Integrando tecnologia de sistemas embarcados e tecnologia Android, o protótipo é capaz de aliar comodidade ao usuário com economia de água. O desenvolvimento do projeto parte do princípio de utilização de componentes financeiramente viáveis e capazes de apresentar resultados consistentes. Basicamente têm se um microcontrolador Arduino recebendo informações de sensores de fluxo, controlando o acionamento de uma bomba d'água e se comunicando com interfaces de interação com o usuário, tudo através de seus periféricos. Vale ressaltar que o protótipo opera em tempo real e possui funções de verificação de possíveis problemas que possam causar desperdícios. Após desenvolvimento de todo o sistema, um passo importante foi a verificação da consistência dos dados coletados, para avaliação da viabilidade e confiabilidade do projeto proposto. Dessa forma, deseja se apresentar uma alternativa viável e que possa ser usada como modelo para desenvolvimento de novos mecanismos de monitoramento de água.

Palavras-chaves:água, microcontrolador, redução de desperdícios, sistemas embarcados, monitoramento, automatização.

Abstract

The precariousness surrounding the availability of drinking water in the 21st century is a topic on the agenda at the Environmental Agencies and a problem that has been causing inconvenience to the population. Both factors of climate change and factors caused by human interference have contributed to the current situation. Responsible management of the use of natural resources is one of the ways to alleviate this chaotic scenario that is being designed for the near future. Accessibility to technology brings some effective alternatives in the management of water resources. Based on this, the present project proposes an automatic supply monitoring system capable of reducing the usual water waste in manual systems. Integrating embedded system technology and Android technology, the prototype is able to combine user convenience with water savings. The development of the project is based on the principle of using components that are financially viable and capable of presenting consistent results. Basically, an Arduino microcontroller has been receiving information from flow sensors, controlling the activation of a water pump and communicating with user interaction interfaces, all through its peripherals. It is worth mentioning that the prototype operates in real time and has functions to check possible problems that may cause waste. After the development of the entire system, an important step was the verification of the consistency of the data collected, in order to assess the feasibility and reliability of the proposed project.

Key-words: water, microcontroller, waste reduction, embedded systems, monitoring, automation.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

PWM	Pulse Width Modulation
ROM	Read Only Memory
RAM	Random Access Memory
E/S	Entrada e Saída
RISC	Reduced Instruction Set Computer
IDE	Integrated Development Environment
CPU	Central Processing Unit;
RISC	Reduced Instruction Set Computer
MIT	Massachusetts Institute of Technology
LCD	Liquid Crystal Display
IHM	Interface Homem Máquina
GND	Graduated Neutral Density
I2C	Inter Integrated Circuit
CI	Circuito Integrado
x	Vaiável de conversão de pulsos para volume
VC	Volume total da caixa
GHz	Giga Hertz
ISM	Industrial Scientific and Medical
IoT	Internet das Coisas

Sumário

1 Introdução

Cerca de 70% da superfície do planeta terra é composta por água (PENA; RODOLFO, 2000). No entanto, apenas 2,6% dessa água é doce, distribuída em sua maioria em geleiras nas calotas polares e em águas subterrâneas, restando uma parcela de 0,3% do total de água do planeta acessível ao ser humano (SOUSA; SARDINHA, 2019) . Além de ser um recurso limitado, a água potável não é bem distribuída ao redor do mundo. Ainda de acordo com Sousa e Sardinha, 60% do recurso disponível está concentrado em apenas 9 países: Canadá, USA, Colômbia, China, Rússia, Indonésia, Índia, Congo e Brasil.

O Brasil, embora contenha 12% de toda a água doce do planeta, também tem essa riqueza mal distribuída em seu território. A maior parte é disposta na região norte, sendo as regiões sudeste, sul e nordeste possuidoras de poucos recursos hídricos. Essa distribuição gera uma problemática, pois o norte possui baixa densidade demográfica e a maior parte da população brasileira se concentra na região sudeste, como mostra a figura 1.

Região	Densidade Demográfica (Hab/Km²)	Concentração dos recursos hidrícos
Norte	4,12	68,50%
Nordeste	34,15	3,30%
Centro-Oeste	8,76	15,70%
Sul	86,92	6,00%
Suldeste	48,58	6,50%
Adaptado de EOS Organização e Sistemas, 2017.		

Figura 1 – Distrubuição de água doce por região no Brasil

Fonte: (EOS, 2017)

Na última década, várias localidades do nosso país tiveram um aumento nos problemas com a falta d'água, onde diversas cidades encaram racionamentos anualmente. Alguns fatores como poluição em rios, desmatamento, alteração do ciclo das chuvas e à má gestão na utilização dos recurso hídricos têm agravado a situação. O saneamento básico do Brasil ainda é bastante ineficiente, por exemplo. Apenas 46% do esgoto do país recebe tratamento, o restante é despejado na natureza, em grande maioria nos rios, aumentando a probabilidade de proliferação de doenças, contaminação da água e consequentemente diminuindo a sua disponibilidade potável (BRKAMBIENTAL, 2020). Outro dado importante que esta fonte traz é sobre o tratamento de esgoto nas 90 maiores cidades do país. Apenas 22 dessas cidades tem mais de 80% da sua população recebendo coleta de esgoto, contribuindo para que 80% dessas grandes cidades tenham uma perda de

cerca de 30% de água potável em sua distribuição.

Outro fator importante nesse processo é o desmatamento, já que as árvores seguram a água das chuvas através de suas folhas e o sombreamento faz com que ela penetre lentamente no solo, abastecendo os mananciais. A alteração no ciclo das chuvas, o tornando bastante imprevisível, com longos períodos de estiagens e pancadas despejadas de uma só vez, também dificulta o abastecimento dos mananciais.

Pode se perceber que a utilização dos recursos hídricos ainda não tem tido uma gestão adequada. Não existe um controle eficiente de utilização da água, com técnicas de reaproveitamento e tratamento, consciência e conhecimento a cerca da quantidade mínima suficiente para cada aplicação, a fim de aumentar a eficiência do uso e diminuir o desperdício.

Ao tentar enxergar possíveis medidas que possam contribuir positivamente para a gestão das águas, várias propostas podem ser apresentadas. Utilização de fontes de energias renováveis, fornecimento de saneamento para uma maior parte da população, implantação de uma política de redução da poluição de rios, preservação de matas e nascentes, controle de uso nas indústrias, atividades agrícolas e uso doméstico são medidas que podem ajudar a amenizar o quadro atual.

Em pequena escala, uma medida muito importante para ajudar na gestão dos recursos hídricos é o monitoramento de consumo de água nas mais diversas aplicações (FILTSOFF; MARTINS, 2018). Através dessa prática, é possível controlar o volume de líquido gasto, cobrar tarifas por utilização, estabelecer limites de uso, etc.

Para monitorar o consumo de H2O pode se usar vários mecanismos. Boias eletromecânicas, controles liga/desliga de bombas, sensores de nível e microcontroladores são componentes capazes de mensurar a quantidade de água consumida em determinado armazenamento, possibilitando o monitoramento e controle desse consumo em tempo real (SANTOS; OLIVEIRA, 2014).

A integração desses componentes compõe o que se denomina sistema embarcado, um sistema computacional construído para atender uma demanda específica desejada. Esses sistemas interagem com o meio externo e são utilizados para controlar ou monitorar processos, associando e manipulando variáveis de entrada e saída (KAELBLING, 1993) .

Neste trabalho, será abordado o monitoramento do consumo de água realizado pela integração de sensores de fluxo com microcontrolador. Através da diferença de vazão medida na saída e entrada de um recipiente, será mensurado o nível de água existente em tempo real e será automatizado o acionamento da bomba de alimentação do sistema, para que a caixa nunca se esvazie completamente ou haja transbordo. Para interagir com usuário, o projeto contará com uma interface disponível para Android e uma interface de campo.

O sistema proposto será abastecido por uma bomba de captação interligada ao recipiente por uma mangueira. O primeiro sensor ficará disposto em série com a mangueira de entrada e o segundo, em série com a mangueira de saída. O controlador será responsável por receber os pulsos dos sensores, convertê-los em nível de água da caixa, acionar e desligar a bomba quando o sistema indicar nível baixo e alto, respectivamente. O usuário terá acesso ao nível d caixa e status da bomba através das interfaces em sistema Android e interface de campo via display LCD.

1.1 Justificativa

Devido à gravidade da diminuição da disponibilidade de água potável no mundo, medidas que possam contribuir para reduzir este cenário devem ser difundidas e apresentadas.

O avanço tecnológico possibilitou a implementação da automação em diversas aplicações físicas. Hoje em dia, vários processos podem ser facilmente monitorados e controlados por sistemas automáticos.

A ideia deste trabalho é justamente unir a necessidade de se monitorar sistemas hídricos, para se reduzir o consumo exagerado e desnecessário de água, com a implementação de um protótipo de automação simples de ser construído e de baixo custo, capaz de apresentar um controle eficaz em tempo real da grandeza monitorada e automatizar o abastecimento, trazendo também comodidade ao usuário.

1.2 Objetivo Geral

A fim de apresentar uma ideia sustentável que pudesse ser elaborada aplicando os conhecimentos adquiridos na graduação de Engenharia de Controle e Automação, o projeto visa desenvolver um sistema de baixo custo de monitoramento de consumo de água em um recipiente e automatização do abastecimento do mesmo.

Para execução da proposta, serão utilizados o microcontrolador Arduino Uno, sensores de pulso que mensuram o fluxo da água, Display LCD, Aparelho Android, módulos de comunicação via Bluetooth e I2C, módulo ponte H ou relé e uma eletrobomba para alimentar a caixa com o líquido controlado.

1.3 Objetivos Específicos

- Realizar um estudo sobre cada componente utilizado no desenvolvimento do trabalho;

- Construir um sistema de monitoramento de baixo custo que entregue a grandeza medida em tempo real;
- Automatizar o sistema de abastecimento, para trazer comodidade ao usuário e reduzir o tempo em que a bomba fica ligada desnecessariamente;
- Desenvolver um protótipo que possa ser utilizado como base para diversas outras aplicações de monitoramento de nível de fluidos.

1.4 Estrutura do trabalho

O presente trabalho está organizado em 5 capítulos. O Capítulo 1 apresenta uma breve introdução aos problemas com falta d'água enfrentados nos últimos anos e mostra possíveis medidas para solucionar a questão. Este capítulo também descreve o objetivo do trabalho, que é propor um sistema de automação no abastecimento de uma caixa d'água de uma propriedade. O Capítulo 2 apresenta uma revisão sobre os componentes e softwares utilizados nas etapas de desenvolvimento do trabalho, bem como uma revisão teórica dos temas relacionados. O desenvolvimento detalha a metodologia utilizada e as etapas de construção do trabalho, apresentadas no capítulo Capítulo 3. Os testes realizados e análise de dados estão no Capítulo 4. O Capítulo 5 traz a conclusão e considerações finais do protótipo.

2 Revisão Teórica

Sistemas de monitoramento de fluidos líquidos em recipientes são amplamente utilizados e desenvolvidos em larga escala. Desde PLCs robustos a microcontroladores, sensores a laser, ultrassônicos, capacitivos e fluxostatos, sistemas supervisórios, interfaces de display e aplicativos são todos usuais para a realização do controle de volumes.

2.1 Revisão Bibliográfica

A Internet das Coisas (IoT) veio para estabelecer uma relação entre os objetos e os humanos. Através da IoT, é possível haver uma comunicação entre o mundo físico e o virtual, onde pode se realizar transmissão de dados, processos e redes de pessoas, conforme abordado por (DORNELAS; OLIVEIRA, 2017).

No artigo de (DORNELAS; OLIVEIRA, 2017), foi apresentado um protótipo bastante parecido com o proposto neste trabalho. Nele, foi desenvolvido um sistema de monitoramento de consumo de água utilizando o próprio Arduino com um sensor de fluxo, mas a plataforma de comunicação para acompanhamento em tempo real do consumo foi desenvolvida utilizando se um Raspberry Py e um rádio nRF24L01+, que fazem a coleta dos dados e enviam para o software desenvolvido em KNoT, onde o usuário possui acesso em tempo real às informações.

Em outra aplicação semelhante, (OLIVEIRA; SANTOS; RODRIGUES, 2014) utilizou um Arduino Mega 2560, um sensor de pressão MPX5010dp e um módulo bluetooth RS 232 HC-05 para monitorar o nível de água de uma caixa. O sensor de pressão é responsável pela leitura dos dados e envio ao Arduino. No programa do microcontrolador, um software foi desenvolvido para fazer a leitura da pressão recebida e por meio de cálculos transformá-la em volume de água. Esse volume é enviado para um aplicativo desenvolvido em plataforma Android, através da conexão do dispositivo com o módulo Bluetooth, que está interligado ao controlador.

As propostas de monitoramento também se expandem para outros fluidos além da água. Em seu trabalho de Monografia, (NIDEJELSKI, 2018) projetou o controle de combustíveis em tanques de armazenamento. Recorrendo também ao microcontrolador Arduino, o sensor ultrassônico HC-SR04 foi o responsável por realizar a medição do nível do reservatório. Posicionado exatamente no meio da parte superior do recipiente, o HC-SR04 foi direcionado na posição vertical com sentido para baixo, conseguindo ler da extremidade

inferior até ao topo do tanque. A distância medida pelo sensor é enviada ao Arduino e convertida no nível do combustível.

Os projetos acima citados, inspiram e auxiliam a execução do modelo apresentado neste trabalho. O dispositivo de medição, sensor de fluxo, é de um modelo parecido com o utilizado no projeto de (DORNELAS; OLIVEIRA, 2017). O desenvolvimento do software e a comunicação entre microcontrolador e ele foram baseados no trabalho de (OLIVEIRA; SANTOS; RODRIGUES, 2014), utilizando se uma plataforma de construção de aplicativos e um sensor bluetooth de fácil configuração e alcance necessário a este projeto, respectivamente.

Para o desenvolvimento do sistema proposto neste trabalho, os componentes escolhidos serão apresentados abaixo, destacando se suas características principais para que se tenha uma introdução de como poderão ser aplicados para alcançar o objetivo do trabalho.

2.2 Sensor de fluxo

Sensores de fluxo ou fluxostatos, são dispositivos capazes de mensurar o fluxo de líquidos, ar ou gases dentro de tubulações. Estes componentes variam por tamanho, modelo, método de aferição, dentre outras características.

No Brasil, os tipos mais comercializáveis de medidores de fluxo são os sensores a pressão, calorimétricos, placas de orifícios, medidores de velocidade do fluxo e medidores de fluxo via turbina. Este medidor de fluxo geralmente é do tipo magnético, constituído de pás rotativas que têm sua velocidade de giro determinada pela quantidade de fluxo que passa por elas.

(MUNDIM, 1999), o Efeito Hall, descoberto por Edwin H. Hall em 1879, é o fenômeno que acontece em condutores percorridos por corrente elétrica, que quando submetidos á um campo magnéticos têm suas cargas deslocadas ao longo do condutor, criando um campo magnético perpendicular ao campo gerado pela corrente inicial, como mostra a figura 2.

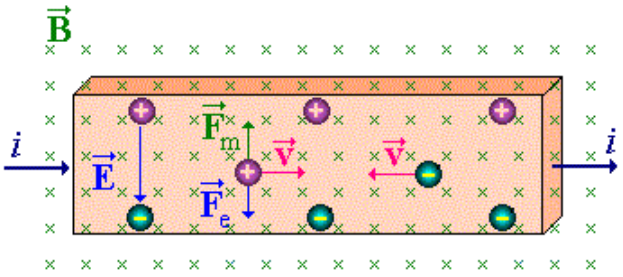

Figura 2 – Efeito Hall

Fonte: (MUNDIM, 1999)

O fenômeno do Efeito Hall é muito utilizado para sensoriamento, devido ao fato de se poder detectar campos magnéticos externos à cargas em movimento em determinado material semicondutor. A presença deste campo faz com que as cargas tenham sua trajetória alterada, acumulando se em uma determinada região do material, onde pode se detectar uma tensão gerada.

Um sensor muito utilizado na instrumentação e que utiliza o princípio magnético do Efeito Hall é o sensor de fluxo YF-S403, mostrado na figura 3.

Figura 3 – YF-S403

Fonte: Autor

Constituído por uma estrutura de plástico, um rotor formado por pás com um ímã acoplado e um sensor de efeito Hall conectado paralelamente, o YF-S403 é um dispositivo capaz de mensurar o fluxo de fluido que passa por ele. A medida que as hélices vão girando com a entrada do fluxo, seja água ou ar, por exemplo, o sensor vai detectando a quantidade de giros, através do ímã existente nas pás. Os pulsos computados são enviados para o microcontrolador e é feito o cálculo da quantidade de fluido que passa pelo sensor em cada giro do rotor.

Desenvolvido para atuar com microcontroladores, o YF-S403 possui tensão de funcionamento de 5 a 24VDC, vazão de água de 1 a 60 L/min,temperatura de operação menor ou igual a 80 °C, pressão de água menor ou igual a 1.75MPa, diâmetro de 26mm (3/4") de entrada e saída (YF-S403. . . ,).

De acordo com o fabricante, a vazão do YF-S403 é dada pela relação da quantidade de pulsos enviada pelo sensor dividida pela constante 4,5. Essa expressão foi obtida através de calibração e fornecida no datasheet do componente, com margem de erro de 3%.

2.3 Arduino

Desenvolvidos por diversos fabricantes e famílias diferentes, os microcontroladores são muito utilizados na área da eletrônica. Os microcontroladores são chips contendo núcleo de processamento, memórias RAM, ROM e FLASH, periféricos de I/O, conversores de sinal e geradores de clock. Embora se pareçam muito com microcomputadores, possuem limitações em relação a eles, com memórias e frequências de clock menores, por exemplo. Abaixo segue o exemplo do diagrama (Figura 4) do microcontrolador AVR ATmega328, contendo todos os recursos nele existentes.

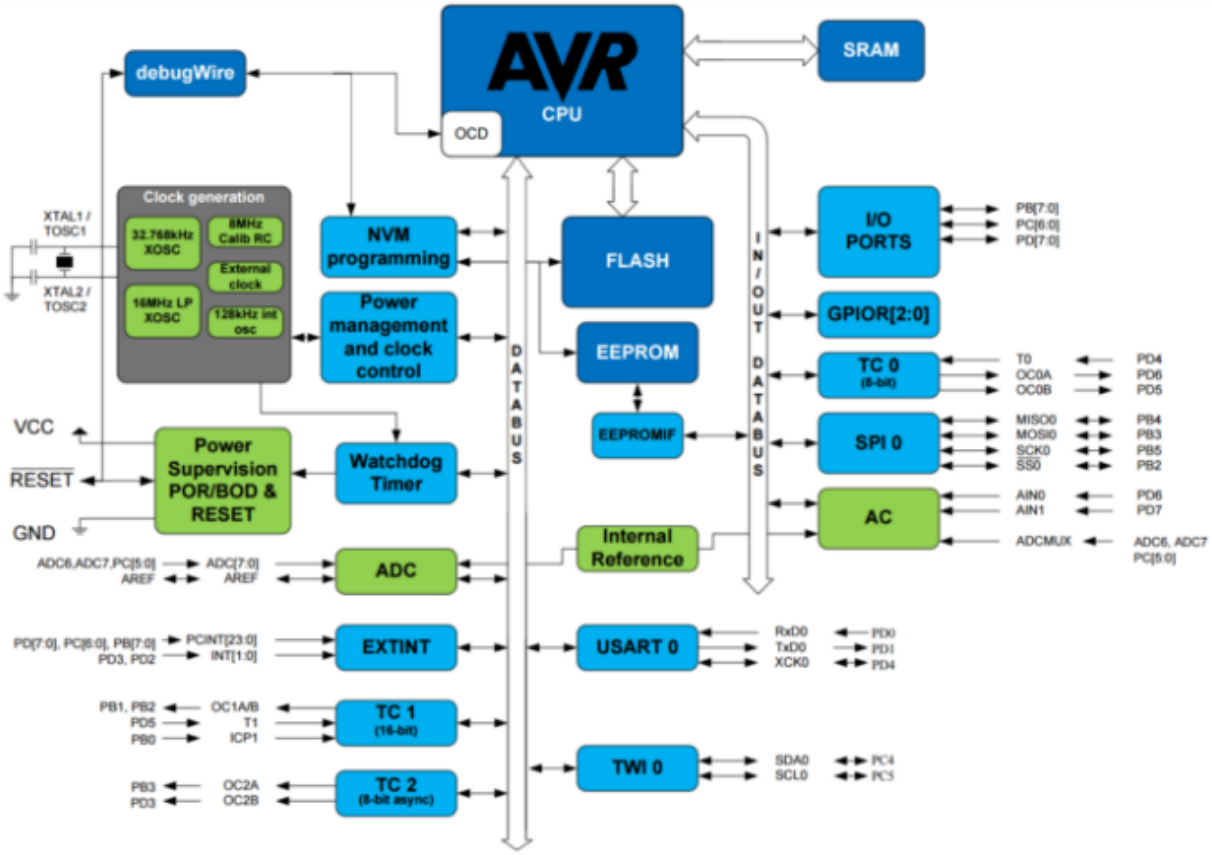

Figura 4 – Diagrama de blocos do microcontrolador AVR ATmga328

Fonte: Aureliano, 2017

Os microcontroladores são amplamente utilizados em aplicações de controle e monitoramento de processos, desenvolvimento de pequenos robôs e automação em geral. A escolha do chip ideal para uma aplicação qualquer, se da através de uma análise das suas características construtivas, preço agregado e disponibilidade no mercado, haja visto a grande variedade de famílias de placas de desenvolvimento existentes, como Arduino, ESP e PIC, as mais utilizadas em aplicações de pequena escala eletrônica.

Contando com uma interface livre e programável por computador, o Arduino é

uma placa baseada no ATmega328, um microcontrolador de arquitetura RISC AVR da Atmel, com ótimo desempenho e eficiente para compiladores (PéREZ, 2019). O Arduino Uno possui catorze pinos de entrada e saída digital, sendo seis podendo ser utilizados como saídas PWM e dois podendo ser utilizados como interrupção, contém 6 entradas analógicas, um cristal de quartzo de 16 MHz e um conector USB, utilizado na comunicação com o computador para a programação e gravação dos códigos na placa (SOUZA, 2013).

O Arduino Uno é um microcontrolador muito utilizado na eletrônica. Possuidor de uma IDE simples para desenvolvimento de software, modelada na linguagem de programação "Wiring", baseada nas linguagens C e C++, basta conectar a placa ao computador e apertar o botão carregar (botão dourado da figura 5 da IDE) que o código é transmitido para o compilador do microcontrolador (SOUZA, 2013).

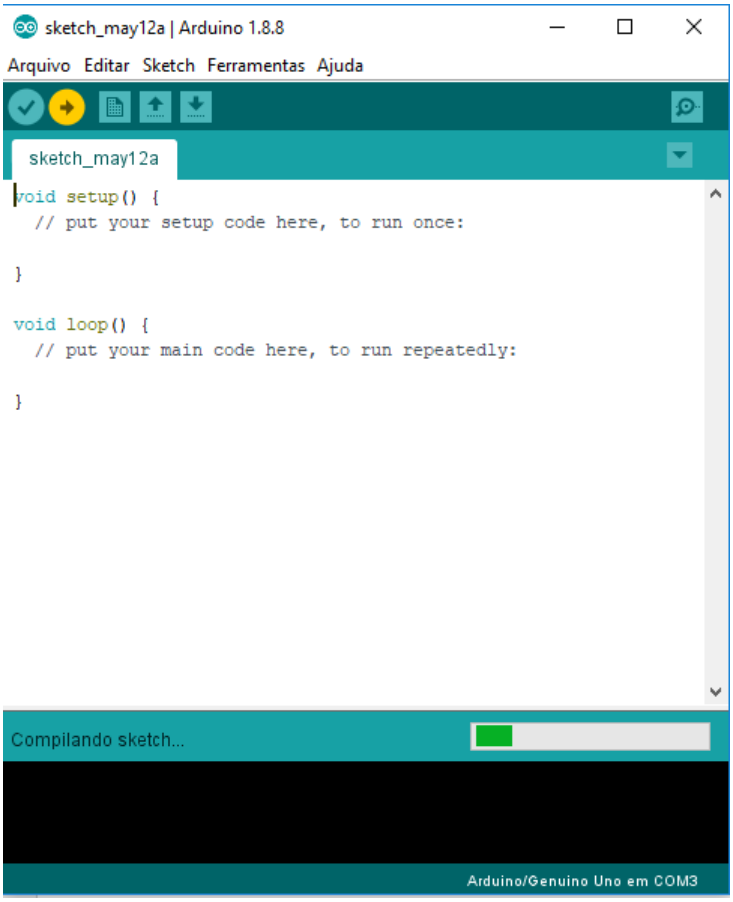

Figura 5 – IDE Arduino

Além do fácil desenvolvimento de software, o Arduino Uno possui baixo valor de mercado, é desenvolvido em uma pequena dimensão, se comunica com diversos componentes de monitoramento físico e possui os periféricos bem identificados e de fácil conexão, como mostra a figura 6.

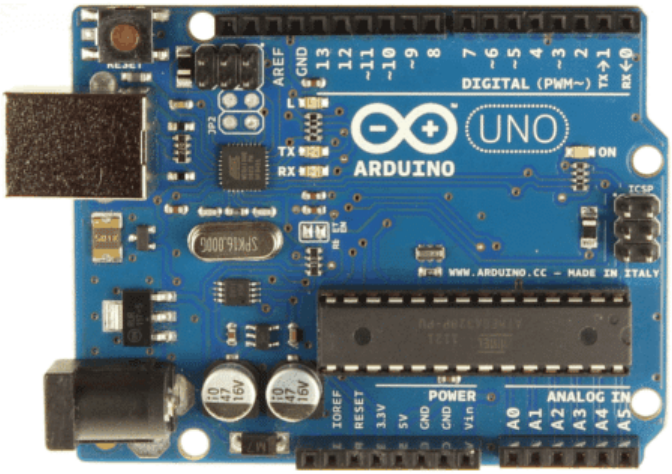

Figura 6 – Arduino UNO

2.4 Display LCD 16x2

Os Displays LCD são componentes constituídos de cristais líquidos entre placas de vidro, recebendo luminosidade de fontes fluorescentes ou de LEDs (MURTA, 2019). Esses componentes são utilizados em diversas aplicações como interface gráfica.

O Display LCD é muito utilizado em projetos associados a microcontroladores. O fato de imprimir números, letras, símbolos e caracteres especiais, além de possuir baixo custo e necessitar de pouca energia para funcionar, é condicionado a ótimo meio de Interface Homem Máquina IHM.

Disponibilizados em grande variedade de tamanhos, a escolha do Display se dá de acordo com a necessidade de cada projeto. O mercado oferece Displays de tamanhos *8x2, 20x4, 20x2, 40x4, 40x2, 24x4 e 16x2, etc* (*número de caracteres por linha x número de linhas*), cores variadas das impressões, funções gráficas avançadas (impressões de imagens e gráficos), etc.

Os Displays 16x2 1602A, muito recorridos à aplicações de coletas e exibição de dados simples, possuem 2 linhas com disponibilidade de exibição de 16 caracteres em cada uma na tela. Esse LCD dispõe de 16 pinos onde são interligadas a comunicação e alimentação do componente, conforme figura 7.

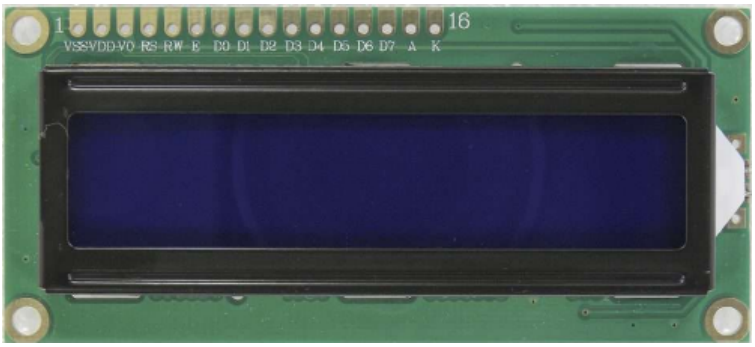

Figura 7 – Display LCD 1602A

Fonte: Autor

Abaixo segue a função da pinagem do Display LCD 16x2 modelo 1602A, mostrada por (MURTA, 2019).

Pino 01 - VSS - Pino de Alimentação (GND)

Pino 02 - VDD - Pino de Alimentação (5 Volts)

Pino 03 - VO – Pino de ajuste do contraste do LCD (Conectado ao potenciômetro para controlar a luminosidade da tela)

Pino 04 - RS – Seleção de Comandos (nível 0) ou Dados (nível 1)

Pino 05 - R/W – Read(leitura – nível 1) / Write (escrita – nível 0)

Pino 06 - E – Enable (Ativa o display com nível 1 ou Desativa com nível 0)

Pino 07 - D0 – data bit 0 (usado na interface de 8 bits)

Pino 08 - D1 – data bit 1 (usado na interface de 8 bits)

Pino 09 - D2 – data bit 2 (usado na interface de 8 bits)

Pino 10 - D3 – data bit 3 (usado na interface de 8 bits)

Pino 11 - D4 – data bit 4 (usado na interface de 4 e 8 bits)

Pino 12 - D5 – data bit 5 (usado na interface de 4 e 8 bits)

Pino 13 - D6 – data bit 6 (usado na interface de 4 e 8 bits)

Pino 14 - D7 – data bit 7 (usado na interface de 4 e 8 bits)

Pino 15 - A – Anodo do LED de iluminação (+5VCC)

Pino 16 – K – Catodo do LED de iluminação (GND)

2.5 Circuito Integrado PCF8574AT

O protocolo de comunicação I2C, criado pela Philips nos anos 90, representa uma comunicação serial utilizando apenas 2 fios. O I2C é um meio de comunicação que possui interação entre os dispositivos seguindo duas premissas: Mestre e Escravo. O dispositivo Mestre coordena os periféricos, ou seja, controla o envio e recebimento de dados do dispositivo Escravo.

Vários componentes têm sido desenvolvidos para realizar essa comunicação I2C entre dispositivos. Exemplo disso é o CI CI PCF8574, um expansor de 8 bits de (E/S), comandados pelo barramento serial I2C, que pode acionar dispositivos diversos e até realizar leitura de botões, por exemplo.

O PCF8574 é muito utilizado para controlar LCDs, pois realiza a comunicação utilizando apenas 4 fios, diferentemente da comunicação direta entre algum dispositivo e LCDs, que geralmente utiliza mais de 8 fios.

A Figura 8 mostra a placa PCF874AT, usada para a comunicação I2C.

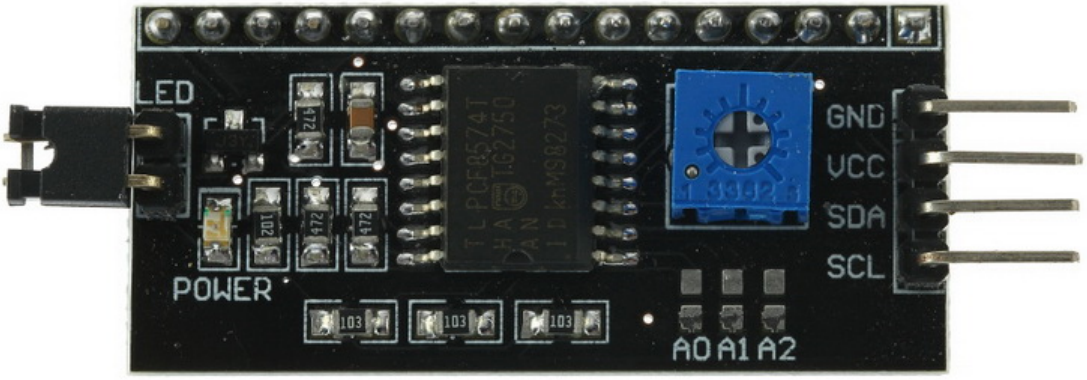

Figura 8 – CI PCF8574AT

Fonte: Autor

O pino SDA da figura é responsável pela transmissão de dados e o SLC pelo clock do circuito. As conexões VCC e GND são responsáveis pela alimentação, geralmente 3,3V ou 5V. Do lado oposto, as duas conexões LED são para definir o estado da luz de fundo do display, caso estejam “jumpeadas11, luz acesa, caso contrário luz apagada. O componente quadrado azul, no meio da placa é responsável pelo ajuste no contraste da tela.

2.6 Módulo Bluetooth HC-05

O sistema de conectividade Bluetooth tem como principal característica a comunicação sem fio. Essa tecnologia permite que diversos dispositivos se comuniquem entre

si e troquem dados utilizando o método de rádio frequência, bastando apenas estarem próximos uns aos outros.

O alcance de um sistema bluetooth é baseado em sua potência.Uma potência máxima de 100mW (miliWatts) permite um alcance de até 100 metros (ALECRIM, 2019). Já a potência de 2,5mW permite alcance de até 1 metro e um dispositivo com potência de até 1mW disponibiliza alcance de no máximo 1 metro.

Sensores, controladores e diversos outros dispositivos necessitam se comunicarem para trocar dados em diversas aplicações. Um módulo Bluetooth muito utilizado para interfacear a comunicação desses dispositivos é o HC-05. Este módulo é da classe 02, ou seja, possui alcance de até 10 metros.

O HC-05 possui tensão de alimentação entre 3.3 e 6V, porém a tensão de transmissão de dados é de 3.3V (VIDAL, 2019). O módulo possui frequência de 2,4GHz, banda ISM e protocolo v2.0+EDR.

O Módulo HC-05 está disposto nas figuras 9 e 10. O pino STATE é utilizado para conectar algum LED para monitorar o estado do módulo. Já os conectores VCC e GND recebem a alimentação do módulo. RX e TX são responsáveis pela recepção e transmissão de dados. Por fim o Pino EN, que alterna entre o recebimento de informações e o modo de configuração.

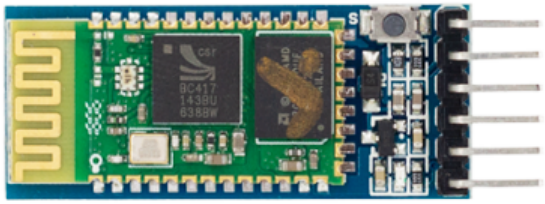

Figura 9 – Módulo HC-05

Fonte: Autor

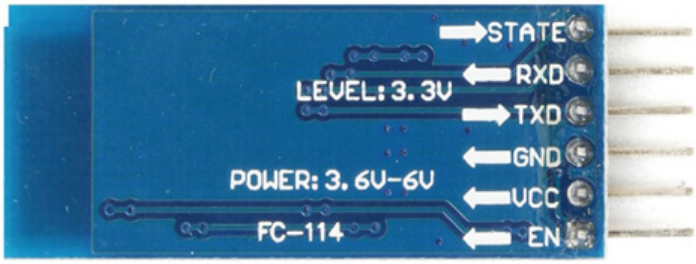

Figura 10 – Parte de trás do HC-05

Fonte: Autor

A frequência de pulsos do LED do Módulo indica seu estado. Caso não esteja se comunicando com nenhum dispositivo, ele recebe pulsos alternados, piscando intermitentemente. Quando está conectado, o LED não recebe pulsos e fica apagado.

2.7 Módulo relé FC-65

Os relés, muito utilizados na eletrônica, são dispositivos eletromecânicos que possuem um magneto móvel, que comuta e se desloca abrindo e fechando contatos metálicos (SANTOS, 2020).

Os relés são aplicados para abrir disjuntores, ativar alarmes, servir de proteção para sobrecorrente, controlar cargas, etc. A estrutura do relé é formada por uma bobina, um circuito magnético e contatos. Quando a bobina é energizada, é capaz de gerar um campo magnético que movimenta uma armadura nos contatos do relé eletromecânico.

A vantagem da utilização de relés é que eles separam o circuito de controle do circuito de carga, protegendo assim o agente controlador, caso sua tensão de operação seja diferente da tensão da carga controlada.

Nos sistemas embarcados, módulos relé são frequentemente utilizados para acionar circuitos. Estes módulos são ligados á microcontroladores que enviam sinais para controlar cargas como lâmpadas, LEDs, motores diversos, bombas, etc. Exemplo disso é o FC-65, um módulo relé capaz de controlar cargas de até 30A/250VAC e 30A/30VDC, com tensão de operação de 5V e possuindo 1 canal de nível lógico alto ou baixo.

A figura 11 ilustra a estrutura física do módulo FC-65 que contém o relé SLA-05VDC-SL-C, composto também pelos pinos DC+, DC- e IN, que são alimentação positiva, alimentação negativa e entrada de sinal, respectivamente. Do outro lado conecta se a carga, NC, NO e COM, pino do contato normalmente fechado, pino do contato normalmente aberto e pino comum, respectivamente.

Figura 11 – Módulo relé FC-65

Fonte: Autor

O módulo FC-65 possui um diferencial que é a existência de um optoacoplador em sua constituição. Os componentes optoacopladores operam por meio de feixe de luz, transmitindo sinais de um circuito para outro sem ligação elétrica, através de uma fonte emissora de luz (LED) e um foto sensor de silício (fototransistor), sensível às variações espectrais da fonte emissora. Essa característica torna o módulo altamente seguro, pois isola completamente o circuito da carga com o circuito de acionamento, protegendo o microcontrolador contra surtos de sobrecorrente e sobretensão.

2.8 Contator de Potência

Diversos equipamentos elétricos precisam ser acionados por dispositivos de comando, seja por questões de segurança, economia, controle de algum parâmetro ou eficiência.

Dentre os diversos tipos de acionamentos, destacam se acionamentos via inversores de frequência, soft starters, relés de partida e contatores. Para equipamentos de partida direta, relés e contatores são os mais adequados, devido ao menor custo em relação aos outros componentes.

Nesta seção, serão apresentadas as principais características dos contatores, que serão os dispositivos utilizados no projeto deste trabalho.

Contatores nada mais são do que chaves liga/desliga, onde os contatos mudam de estado quando as bobinas são energizadas. Estes componentes são encontrados em diversos modelos diferentes. Existem contatores com acionamento 220VAC, 11VAC, 24VDC, monofásicos, trifásicos, contatos NA, NF, correntes elétricas variadas, entre outras características.

O princípio de funcionamento geralmente é o mesmo. Quando a bobina do contator é energizada, os contatos mudam de estado. Ou seja, se é um contato NA, eles se fecham e interligam o circuito, se é contato NF, eles se abrem e interrompem o circuito. Quando desenergizados, voltam ao estado natural.

Estes dispositivos podem ser acionados por temporizadores, sensores, módulos de comando, microcontroladores e relés. A única restrição é a tensão de acionamento, que deve ser a mesma especificada na bobina do contator.

A figura 12 ilustra um contator trifásico, 220V, 25A, 3 NA, modelo CWM25 do fabricante WEG.

Figura 12 – Contator WEG CWM25

Fonte: (**??**)

2.9 MIT App Inventor

O MIT App Inventor é uma plataforma aberta, que possibilita o desenvolvimento de aplicativos para Android. A aplicação, criada pela Google, hoje é mantida pelo Massachusetts Institute of Technology (MIT).

O App Inventor possui uma plataforma simples e didática para a criação de aplicativos. Porém, ele se limita a pequenas aplicações, não contendo recursos para desenvolvimentos profissionais.

Para desenvolver aplicativos nessa plataforma, primeiramente basta se cadastrar no site, baixar gratuitamente o aplicativo na loja de um celular android e iniciar o projeto.

Para conectar o celular ao projeto, basta abrir o App Inventor no navegador do computador, clicar sobre os ícones *"Connect"* e *"Connect Companion"*. Irá aparecer um QR code, onde se usará o aplicativo no Android para realizar a leitura do código e se conectar.

O aplicativo de desenvolvimento possui duas seções, uma voltada para a criação da interface, o *"App Inventor Designer"*, que é a tela principal do projeto, onde se adiciona botões, imagens e outros componentes disponíveis para a criação da interface gráfica; e a seção *"App Inventor Blocks Editor"*, onde se define as ações dos itens inseridos na interface e ativa os comandos de execução de tarefas.

2.9.1 App Inventor Designer

O *"App Inventor Designer"* é a tela principal do aplicativo. Nela se desenvolve toda a interface do programa de maneira fácil e intuitiva. Através de blocos já configurados, o usuário os escolhe para a aplicação e os arrasta até a tela *"Viewer"*, podendo alterar suas dimensões, conteúdos, nomes, etc. A figura 13 ilustra a tela do *"App Inventor Designer"*, onde a janela de exibição simula a interface de um smartphone com sistema operacional Android.

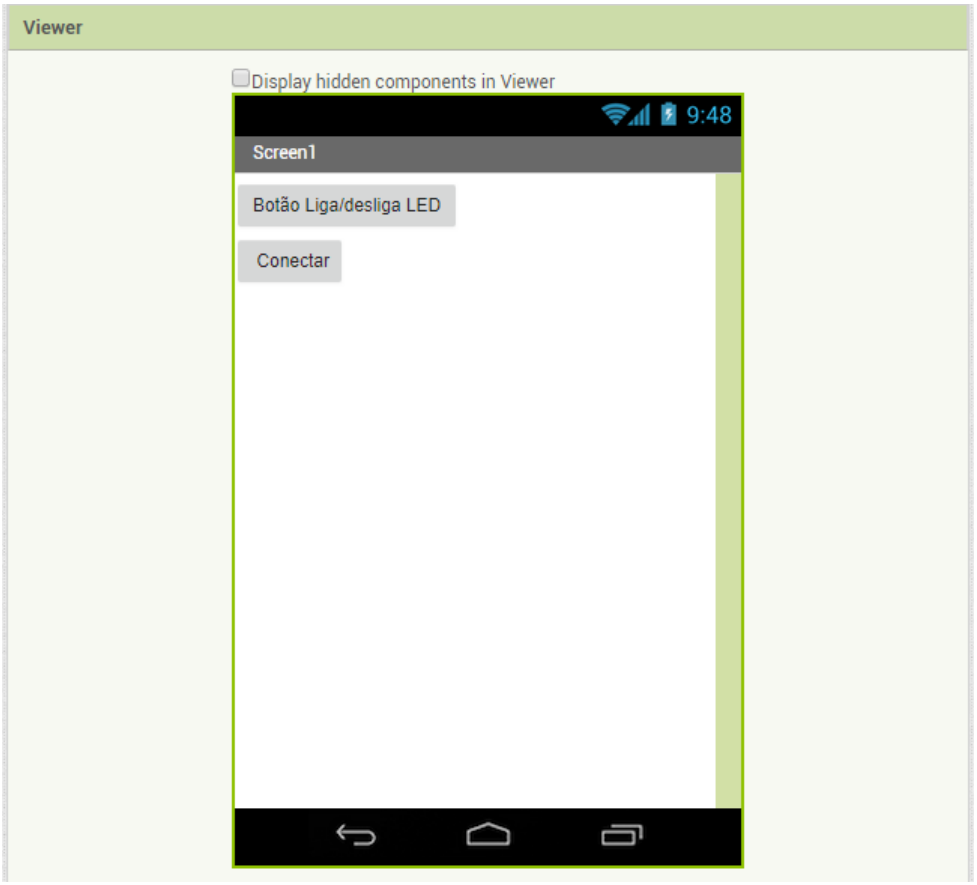

Figura 13 – Simulação de Tela no Viewer do aplicativo

Fonte: Autor

Os componentes que serão utilizados na Tela Principal, são encontrados na aba *"Pallet"* , coluna que fica à esquerda do *"Viewer"*. Na Paleta são encontrados todos os componentes utilizáveis no aplicativo como botões, imagens, textos, modos de conectividade, layout, etc. Os componentes são dispostos em seções, afim de tornar o ambiente mais organizado, conforme figura 14.

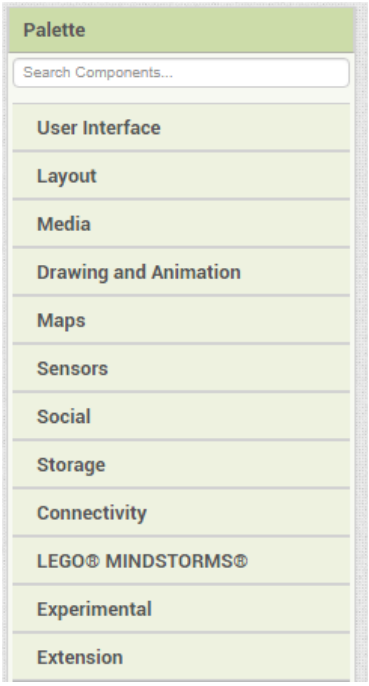

Figura 14 – Coluna Palette do App Inventor

Fonte: Autor

Os itens adicionados à Tela podem ser visíveis ou não visíveis. Eles ficam disponíveis na coluna *"Components"*, localizada à direita do *"Viewer"*. Nela é possível identificar e editar o nome dos componentes adicionados ao projeto, conforme exemplo da figura 15.

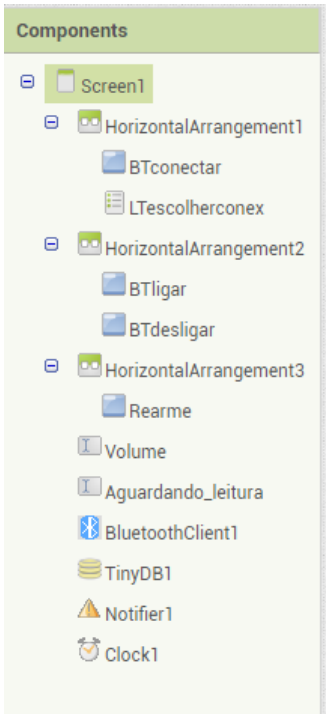

Figura 15 – Coluna Components do APP Inventor

Fonte: Autor

Ao lado da Coluna *"Components"*, está a aba *"Properties"*, onde se pode modificar as configurações e editar as propriedades dos objetos e essas alterações são atualizadas instantaneamente no *"Viewer"*. A figura 16 demonstra um exemplo desta aba.

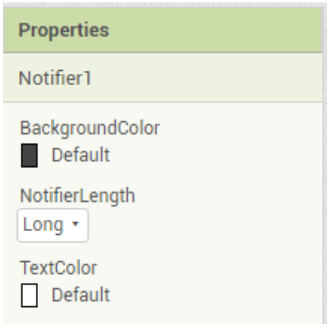

Figura 16 – Coluna Properties do APP Inventor

Fonte: Autor

2.9.2 App Inventor Blocks Editor

No *"App Inventor Blocks Editor"* se define a ação de cada componente adicionado no *"App Inventor Designer"*. Também se arrastando blocos já estruturados, vai se formando a lógica do aplicativo.

No lado esquerdo da Tela do *"Blocks Editor"*, ao lado do *"Viewer"*, há a coluna *"Blocks"*, onde os blocos lógicos são encontrados. Nesta aba, estão dispostos os componentes adicionados ao projeto e clicando com o botão esquerdo do mouse sobre eles, aparece a lista de blocos com funções lógicas disponíveis para cada um deles. A figura 17 ilustra uma coluna *"Blocks"* com componentes adicionados, além dos componentes lógicos disponíveis em todas as aplicações, na partição *"Built-in"*.

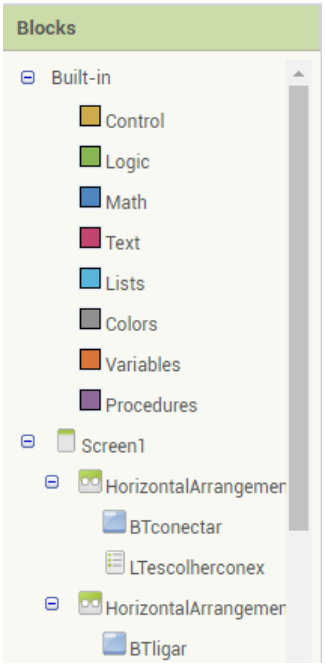

Figura 17 – Coluna de blocos na tela do Blocks Editor

Fonte: Autor

Após adicionar os componentes no *"Designer"*, os blocos podem ser acessados e a cada função adicionada, o sistema responde imediatamente e atualiza a nova configuração definida.

3 Metodologia

O sistema proposto objetiva automatizar o processo de abastecimento de água de determinado recipiente, rastrear possíveis falhas na bomba ou tubulação de entrada, possibilitar ao usuário controle manual do sistema e que ele possua interfaces onde consiga acompanhar nível da caixa e situação da bomba em tempo real.

Para este projeto será utilizado o sistema de abastecimento de uma propriedade rural para a realização de testes. Uma bomba de água de 3cv alimenta uma caixa de 1000 litros, e este sistema será automatizado para verificação do funcionamento do protótipo desenvolvido.

O protótipo apresentado funciona da seguinte maneira: O controlador se comunica com sensores através de seus periféricos. Assim que os sensores indicarem os níveis estabelecidos como limites mínimos e máximos para acionamento da bomba, sinais são enviados para que o controlador faça a bomba operar ou parar.Todo esse controle terá uma interface gráfica onde o usuário acessará e poderá acompanhar em tempo real, via aplicativo ou através de um display LCD.

desenvolvimento do projeto seguiu as seguintes etapas:

1. Desenvolvimento da lógica de leitura dos sensores;
2. Desenvolvimento da lógica de acionamento da bomba;
3. Desenvolvimento da lógica de comunicação com o display;
4. Desenvolvimento da lógica de acionamento manual através de botões;
5. Desenvolvimento da comunicação via bluetooth;
6. Desenvolvimento da lógica de cálculo de volume do recipiente;
7. Desenvolvimento da lógica que imprime o valor do volume no display LCD e no aplicativo;
8. Desenvolvimento do aplicativo;
9. Desenvolvimento da lógica de intertravamento do sistema;

As seções abaixo apresentarão a construção do protótipo, desde as ligações dos componentes com o microcontrolador, desenvolvimento da lógica que determina o volume,

lógica de programação do Arduino, contemplando todos os passos e funções do sistema, e lógica de desenvolvimento do aplicativo e demonstração de sua interface.

3.1 Leitura dos sensores

Com a finalidade de automatizar o abastecimento da caixa e medir o volume de água existente em tempo real na caixa, utilizou se dois sensores de fluxo. O primeiro sensor, conectado em série com a tubulação de abastecimento para medir o fluxo de entrada e o segundo sensor conectado em série com a mangueira de saída da caixa, para mensurar o volume gasto. Para mensurar o nível da caixa em tempo real, uma operação algébrica foi estabelecida entre os sensores de entrada e saída do sistema, totalizando o volume existente no reservatório.

Os sensores de fluxo 01 e 02 foram ligados aos pinos digitais 02 e 03, respectivamente, do microcontrolador. A escolha destes pinos se deve ao fato deles poderem ser utilizados com interrupção, a fim de priorizar as tarefas de contagem de pulsos do sistema. A alimentação dos sensores é realizada pelo próprio microcontrolador, já que possuem tensão de alimentação de 5V. Os fios VCC dos sensores foram conectados ao pino 5V e os fios GND ao GND do microcontrolador, conforme esquema da figura 18.

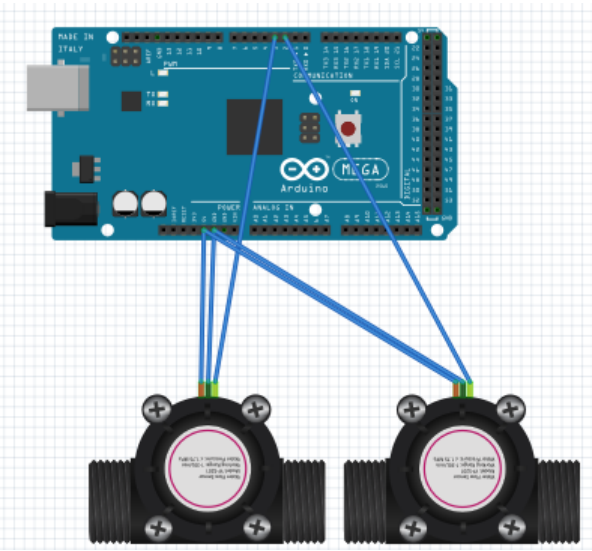

Figura 18 – Esquema da ligação dos sensores de fluxo

Fonte: Autor

Os dois pinos foram configurados como entrada, para receberem os dados dos sensores. Com as funções de interrupção ativadas no programa, sempre que os pinos 02 e 03 receberem algum sinal, o programa interrompe a tarefa que está executando, exceto se estiver em um *loop*, e executa a tarefa de contagem dos pulsos recebidos implementada

nas funções de interrupção dos dois pinos, respeitando a prioridade do pino 02 sobre o pino 03 estabelecida pelo próprio microcontrolador.

3.2 Acionamento da bomba

A bomba d'água, composta por um motor monofásico de 3cv e alimentada por uma tensão bifásica 220V alternada, tem seu acionamento interligado com o microcontrolador via módulo FC-65 e contator de potência. Conforme apresentado anteriormente, esse relé suporta uma corrente de 30A e uma tensão de 250VAC, tornando o adequado à ligação da bomba d'água.

O contator terá a bobina A1 ligada direto em uma fase 127V e a alimentação da bobina A2 passará pelo relé, ligado ao pino digital 9 do Mega. Quando o microcontrolador enviar 5V pela saída do relé, este fechará seus contatos e a tensão da fase conectada a ele passará para a bobina do relé, fechando também seu contato e alimentando a bomba.

A utilização deste circuito se deve ao fato dele separar os circuitos da bomba e do Arduino, pois o microcontrolador não suporta a carga de acionamento da bomba, podendo ser danificado caso os circuitos se intercedam diretamente.

Um esboço do diagrama da ligação do circuito da bomba é apresentado na figura 19.

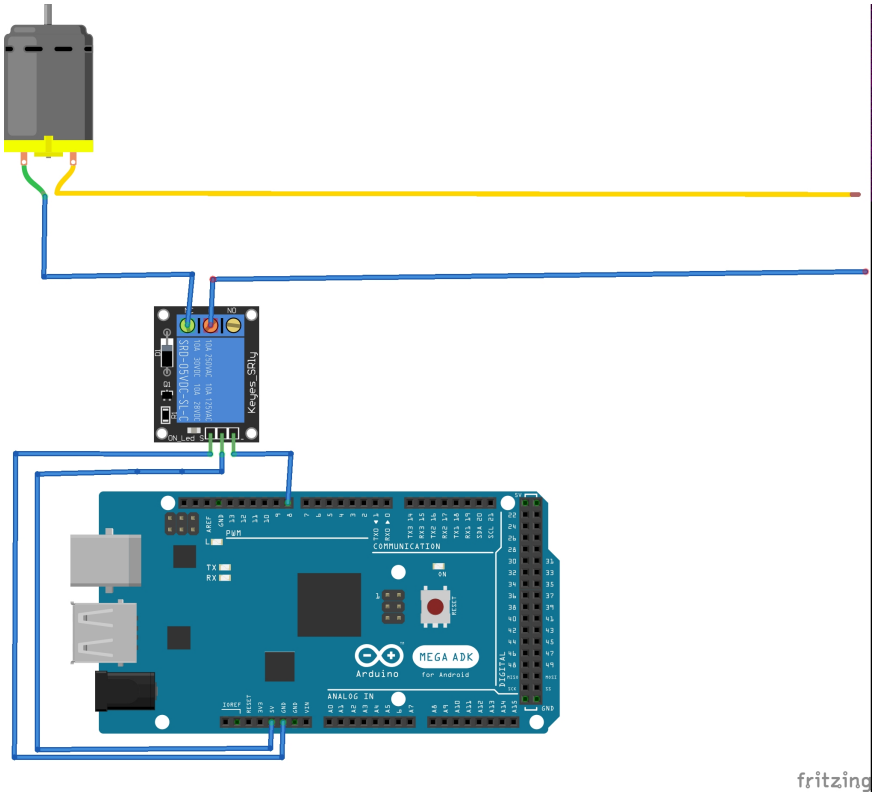

Figura 19 – Ligação de acionamento da bomba

Fonte: Autor

Apenas quando a ponte recebe o sinal *HIGH* do microcontrolador, através do pino 8, é que o circuito de alimentação da eletrobomba se fecha e ela é acionada. Esse sinal é enviado quando a caixa atinge 50% de sua capacidade ou quando o botão liga é acionado manualmente.

3.3 Comunicação com o Display

Para a indicação do volume e estado da bomba, é utilizado o módulo I2C integrado a um Display LCD 16x2, onde o usuário é capaz de visualizar as informações do sistema. O modo de comunicação escolhido para demonstração é a comunicação via Bluetooth, mas também podendo ser utilizado outros métodos, como a comunicação via Ethernet, por exemplo.

O módulo de comunicação entre microcontrolador e Display é ligado ao Arduino através dos pinos SDA, SCL, VCC e GND, conectados aos pinos A4, A5, 5V e GND, respectivamente, do Arduino. Os pinos 5V e GND recebem a alimentação do CI e os pinos SDA e SCL são responsáveis pelas trocas de informações no tempo correto.

A utilização do PCF874AT é motivada pela sua capacidade de realizar comunicação

com apenas 4 fios conectados aos pinos citados acima. Esta característica diminui a utilização de pinos digitais do microcontrolador, possibilitando a inserção de outras funções no sistema ou utilização de um microcontrolador de poucas saídas digitais.

Os pinos do módulo I2C são conectados ao Display 1602A, conforme ilustra a figura 20.

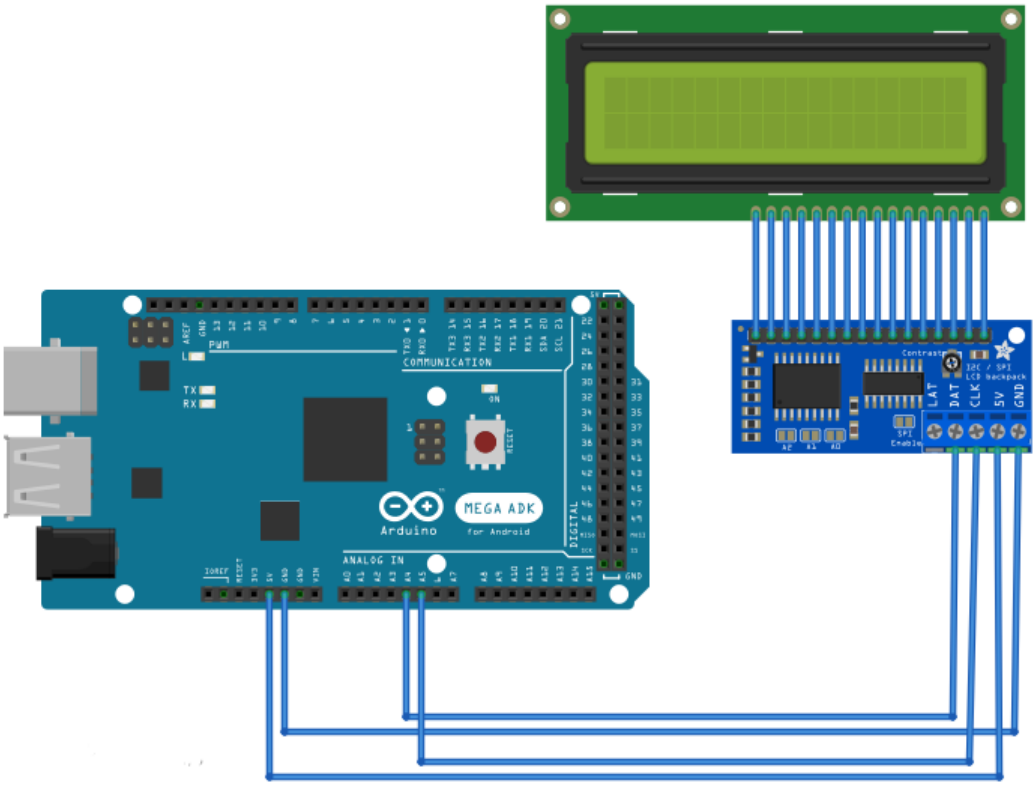

Figura 20 – Ligação do Display LCD

Fonte: Autor

No display as informações de volume, vazão de entrada de água e situação da bomba são recebidas pelo I2C e impressas no LCD, sendo as duas primeiras informações na linha 1 e a segunda na linha 2.

3.4 Botões de acionamento manual

Outra característica do sistema é que através de botões no campo ou no aplicativo, é possível transferir o sistema para o modo local e controlar o acionamento da bomba manualmente. No entanto, quando este recurso é utilizado, o sistema automático dependente do nível baixo da caixa é desabilitado, sendo necessário utilizar o botão de “rearme”, seja pelo aplicativo ou no campo, para que a função automática de acionamento da

bomba e verificação de possíveis falhas volte a operar. Já para o nível alto, a condição de desligamento da bomba quando se atingir nível alto da caixa se mantém, para cercar possível esquecimento do usuário para desligar a bomba e evitar desperdícios. Porém, o sistema também só voltará ao modo automático após utilização do botão "Rearme".

Quando o sistema apresentar algum indício de falha, como será abordado mais adiante, uma mensagem de alerta é mostrada na tela do display, com a finalidade de mostrar ao usuário que há alguma anormalidade que precisa ser corrigida.

3.5 Conexão via Bluetooth para o aplicativo

Para comodidade do usuário será desenvolvido um aplicativo Android, onde as informações de volume, status da bomba, do sistema e acionamento manual da mesma bomba, serão disponibilizadas.

A comunicação do do Android com o microcontrolador será via bluetooth, através do Módulo HC-05. Basta ao usuário ativar a função Bluetooth no aparelho celular, entrar no aplicativo, selecionar a opção de se comunicar com o dispositivo HC-05, inserir a senha que será requerida (para este módulo a senha é 1234), e o aparelho já recebe as informações do microcontrolador.

O módulo é conectado ao Arduino conforme figura 21. Os pinos 10 e 11 recebem TX e RX do HC-05, responsáveis por realizarem a transmissão dos dados. VCC e GND recebem a alimentação de 3.3V do microcontrolador. Com essa configuração, o Módulo já se torna capaz de se comunicar via bluetooth com o dispositivo Android.

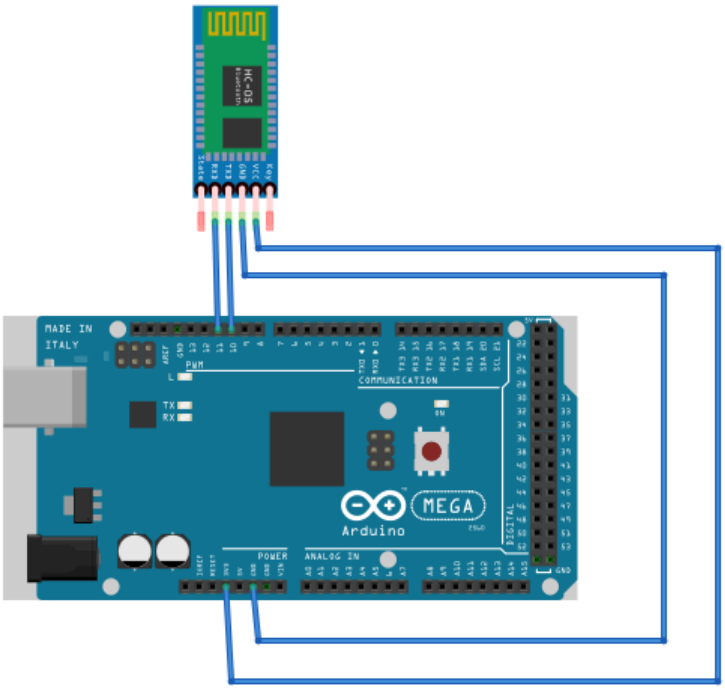

Figura 21 – Ligação do Módulo HC-05

Fonte: Autor

3.6 Desenvolvimento da lógica do sistema

Como a finalidade do trabalho é desenvolver um projeto autônomo e seguro, essas características foram tratadas na lógica de programação do sistema.

Como apresentado na subseção dos sensores de fluxo, eles enviam a quantidade de pulsos referentes ao volume de água que passa por eles, para o microcontrolador. Esses pulsos são transformados em volume, através da relação definida no datasheet do sensor ou estabelecida através de calibrações dos dispositivos sensores, onde é definido o valor da variável x.

$$V = p/x \tag{3.1}$$

Em que,

V = Volume;

p = quantidade de pulsos enviados pelo sensor;

x = quantidade de líquido em *ml* que passa no sensor a cada pulso.

As vazões de entrada e saída do sistema são dadas em *L/min* e calculadas através da divisão d volume pelo tempo.

$$Q = V/60 \tag{3.2}$$

Onde,

Q = Vazão.

Com a caixa iniciando completamente vazia, é realizado a operação algébrica de diferença entre os volumes de água que passam pelo sensor 01 e 02, para estimar o volume total da caixa. A fórmula para os volumes de água que passam pelos dois sensores é a mesma da equação 4.1.

$$VC = VS1 - VS2 \tag{3.3}$$

VC = Volume total da caixa;

VS1 = Volume de água que passa pelo sensor 01;

VS2 = Volume de água que passa pelo sensor 02.

Através das informações de volume, assim que a caixa atingir nível igual ou inferior a 50 % de sua capacidade, a porta 9 envia um sinal de 5V para a ponte. Dessa forma, o driver fecha o circuito de alimentação da bomba, habilitando seu acionamento. Quando a caixa atingir 100% de sua capacidade de armazenamento, valor definido de acordo com suas dimensões, o microcontrolador envia 0V para o driver de controle da bomba e ela para de receber alimentação.

Devido à bomba não poder operar sem água, como medida de segurança para garantir a integridade do equipamento, foi implementada uma condição de operação. A cada 5 segundos o programa entra numa rotina de verificação do sistema de abastecimento. Estando a bomba com o acionamento habilitado e o sensor de fluxo 01 detectar um fluxo inferior a 10ml durante 60 segundos, o programa entra em um *Loop*, o acionamento é desabilitado e o software de controle recebe uma mensagem de alerta. O processo só será reestabelecido após o acionamento local de um botão.

Através da porta serial do microcontrolador, a cada 2 segundos são transmitidas as informações de vazão dos dois sensores, nível de água na caixa e situação de operação da bomba e a cada um minuto transmite se a informação de consumo médio.

3.7 Código de programação do microcontrolador

O sistema foi programado na Linguagem Arduino, onde as interações entre microcontrolador, sensores, bomba e aplicativo foram definidas. Abaixo será mostrado a ideia de cada função separadamente e como o algoritmo foi desenvolvido. No anexo I será apresentado todo o código de maneira sequencial.

Inicialmente, são adicionadas as bibliotecas especiais usadas pelos componentes conectados ao microcontrolador. Neste sistema, serão utilizadas as bibliotecas *LiquidCrystal* e *SoftwareSerial*, responsáveis pelas rotinas do Display LCD e comunicação com o aplicativo através do Módulo Bluetooth, respectivamente.

Após a inclusão das bibliotecas, os pinos ligados ao Módulo Bluetooth e ao Display LCD foram definidos e declarados na IDE, utilizando as funções *SoftwareSerial bluethooth (TX, RX)*, onde os pinos 10 e 11 foram usados como TX e RX, respectivamente, e a função *LiquidCrystal lcd (13, 12, 7, 6, 5 4)* utilizada para o display, onde os números entre parêntese representam os pinos digitais utilizados no Arduino.

A função *Void Setup*, chamada para iniciar variáveis e bibliotecas, é usada para se definir a a condição e configuração I/O dos pinos utilizados e realizar as definições iniciais do programa.

Neste projeto, definiu se o Display como sendo de 16 colunas e 2 linhas, através da função *lcd.begin* encontrada na biblioteca "LiquidCrystal". Outras definições exibidas nesta função são as taxas de Comunicação Serial em bits por segundo do Módulo Bluetooth e do Display, através das funções *bluetooth.begin* e *Serial.begin*, respectivamente.

O pinos conectados à bomba e ao botão de rearme, são definidos através da função *PinMode(Variavel, Configuração I/O)*, onde utiliza se *"INPUT"* para entrada e *"OUTPUT"* para saída. Já os pinos 2 e 3, conectados aos sensores de fluxo, são definidos e têm a função de interrupção habilitada em cada um através do *AttachInterrupt*, sendo acionada quando o estado do pino passa de *"LOW"* para *"HIGH"*, configuração determinada pelo *"RISING"*.

Dentra da função *Void Loop* é onde as tarefas do programa são executadas sequencialmente enquanto a placa estiver ligada, repetindo o processo a cada fim de execução.

No *Loop* do sistema proposto, definiu se inicialmente a bomba em estado desligado, através da função *DigitalWrite* recebendo o valor "LOW".

As variáveis que recebem a quantidade de pulsos dos sensores iniciam todas as

rotinas recebendo o valor “0”. Isso se faz para que se tenha uma nova contagem a cada execução da função e a contagem antiga não interfira nos números novos.

Dentro do *Loop* as interrupções foram habilitadas, para que toda vez que os sensores receberem algum pulso, o sistema interrompa a tarefa que estiver executando e execute a função incrementada na interrupção. A interrupção ficará habilitada por 1000 mili segundos e será desabilitada.

O incremento das interrupções foi realizado de uma maneira diferente do habitual em programas que as utilizam. Elas iniciaram desabilitadas, imediatamente foram habilitadas, deu se um “DELAY” de 1000 mili segundos, e foram desabilitadas novamente. Toda a lógica aritmética desenvolvida para achar o volume do recipiente foi implementada e antes de implementar as funções de comunicação com o aplicativo e Display, a interrupção foi habilitada novamente. Esse procedimento foi adotado para que o microcontrolador conseguisse se comunicar normalmente com Display e Aplicativo, pois essas rotinas não são executadas corretamente com as interrupções desabilitadas, conforme será mostrado mais adiante. As interrupções são habilitadas através da macro “SEI” e desabilitadas pela macro “CLI”.

Foram criadas variáveis para receber a quantidade de pulsos recebidos pelos sensores 01 e 02 divididos pelos seus fatores de conversão para vazão, respectivamente.

Outras variáveis foram adicionadas para atuar como contadores, recebendo seu valor antigo adicionado do novo valor das variáveis de vazão, sendo atualizadas toda vez que o programa repete o *loop*.

O cálculo do volume se dá, então, através da diferença algébrica entre as variáveis de vazão de entrada e saída.

Caso o sistema apresente alguma falha, como por exemplo, a tubulação de entrada ou saída se solte, e a diferença entre os valores médios das vazões de entrada e saída se torne negativa ou ultrapasse o volume suportado pelo recipiente, foi desenvolvida uma condição que limita o volume mínimo e máximo da caixa, onde através da condição *if* estabeleceu se que todo volume computado abaixo de 0 litro receberá o valor “0” e maior que 1000 litros receberá o valor de “1000”.

As rotinas de exibição do volume foram desenvolvidas em sequência, utilizando funções simples das bibliotecas “SoftSerial” e “LiquidCrystal”. Habilitou se a interrupção para que a comunicação com os módulos que recebem os dados fosse efetiva. Logo após, desenvolveu se as rotinas de impressão para o display e para o aplicativo, respectivamente.

A bomba também poderá ser acionada de maneira manual pelo aplicativo ou pela interface colocada em campo. Caso os botões de acionamento da bomba sejam utilizados pelo usuário, o programa verifica o estado da bomba e altera o estado. Se a bomba estiver desligada e não esteja em estado de falha, ela é ligada e entra em um *Loop* até que o

botão de rearme do campo ou do aplicativo seja acionado. No laço acima, a bomba se desligará quando a caixa atingir seu volume máximo pré estabelecido, porém não retornará ao estado automático caso o botão de rearmar o sistema não seja acionado. O código desenvolvido para o estado manual é basicamente o mesmo do laço de repetição para falhas, com a diferença de a bomba ficar ligada apenas até atingir o nível máximo, função copiada do modo automático e já exibida acima.

Para o caso de a bomba estar ligada e o botão de desliga do aplicativo ou botão de acionamento do campo sejam acionados, a bomba irá se desligar e o programa só retornará ao estado automático após acionamento de um dos botões de rearmar do sistema. Dentro do laço se repetirão as características do laço de falha que serão descritas abaixo.

Como as bombas d'água geralmente não podem operar sem água, foi criada uma condição para que sistema desligue a bomba, caso o sensor de fluxo de entrada não detecte vazão durante 60 segundos, tempo médio que a água chega no sensor após o acionamento da bomba. Esta condição também consegue cercar possíveis falhas na alimentação da bomba ou rompimento da mangueira que abastece a caixa, indicando ao usuário que há uma falha no sistema de abastecimento.

Após as condições acima, o sistema entrará em um *Loop* até que o botão de rearme do aplicativo ou em campo seja acionado. Dentro deste *Loop* a alimentação da bomba será cortada e o Aplicativo e Display emitirão a mensagem de texto "Falha no sistema". Ainda dentro do *Loop*, o programa executará a lógica que determina o volume, tendo as interrupções habilitadas e a lógica que faz a leitura dos dados recebidos pelo aplicativo. Após sair do laço, o programa volta a operar em modo automático.

Como os sensores coletam dados externamente à caixa, caso haja alguma falha em algum deles ou no microcontrolador, como um corte de alimentação, por exemplo, eles perderão a referência do volume real. Para isso, criou se uma adequação na lógica, onde o usuário terá que acionar a bomba manualmente e encher a caixa em seu volume máximo. Logo após, apertará o botão de *reset* e o programa começará a rodar contabilizando o volume inicializado com o volume máximo da caixa.

Por fim, as funções de interrupção foram acrescidas e são chamadas sempre que houver uma interrupção nos sensores. As duas funções são muito parecidas, basicamente contam a quantidade de pulsos que o sensor recebe a cada vez que há a interrupção.

O fluxograma da figura 22 ilustra a interação entre as funções desenvolvidas no código e descritas acima.

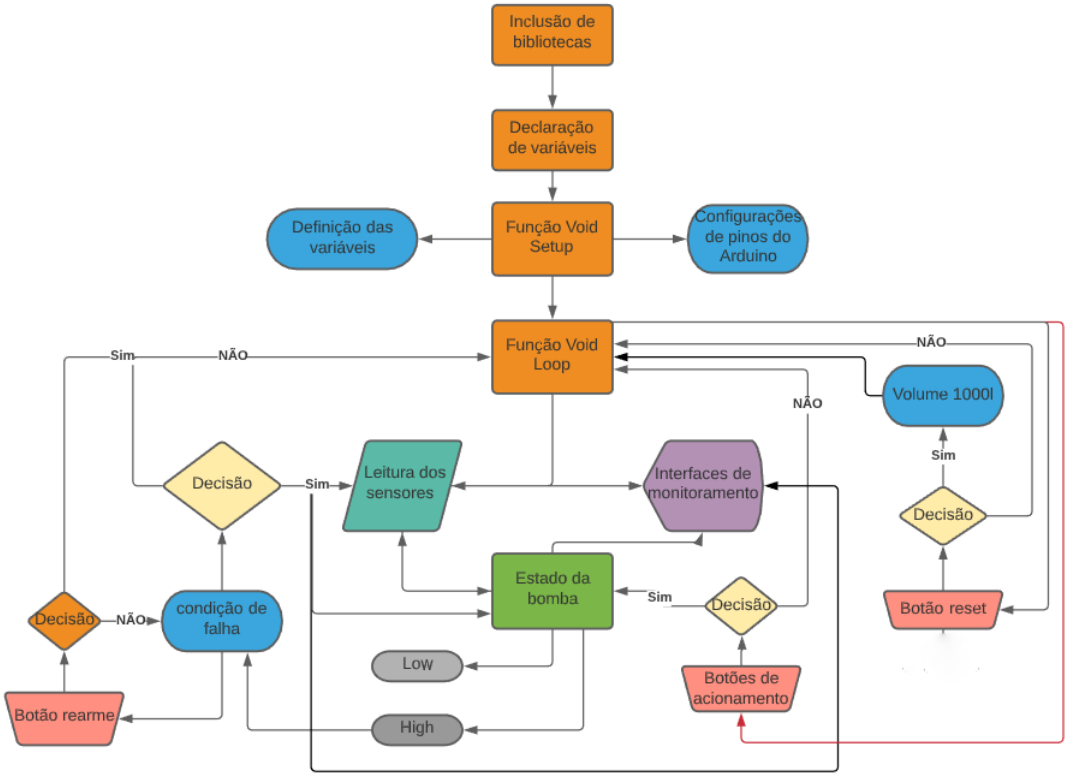

Figura 22 – Fluxograma do código desenvolvido

Fonte: Autor

3.8 Desenvolvimento do Aplicativo

O aplicativo escolhido para interagir com o sistema foi desenvolvido na plataforma *MIT App Inventor*, descrita no capítulo 2. A escolha dessa plataforma se deve à sua interface de desenvolvimento ser bastante intuitiva e de fácil construção. Embora não apresente recursos para aplicações robustas, a utilizamos devido ao sistema proposto não exigir propriedades complexas.

No *App Inventor Blocks Editor* , inicialmente foi configurado a conexão do aplicativo com o módulo *Bluetooth* associado ao microcontrolador. A construção dessa parte do aplicativo foi baseada em (GUIMARãES, 2019).

Na Função *Variáveis* (Figura 23), será utilizado o bloco da figura 24, uma variável global denominada “dispositivo” que iniciará com um texto vazio, onde será guardado o dispositivo que será conectado.

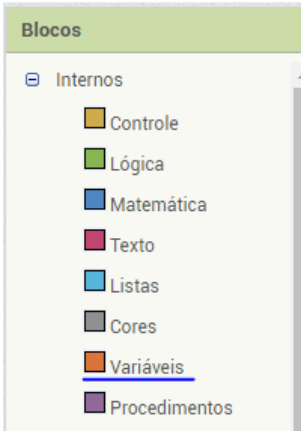

Figura 23 – Função Variável

Fonte: Autor

Figura 24 – Bloco Variável Global

Fonte: Autor

O próximo passo é criar uma rotina que procura dispositivos na base de dados quando o aplicativo inicializa, e caso encontre algum, testa se este está pareado com o aparelho. Na figura 25, temos o bloco final que testa se o dispositivo encontrado não está pareado.

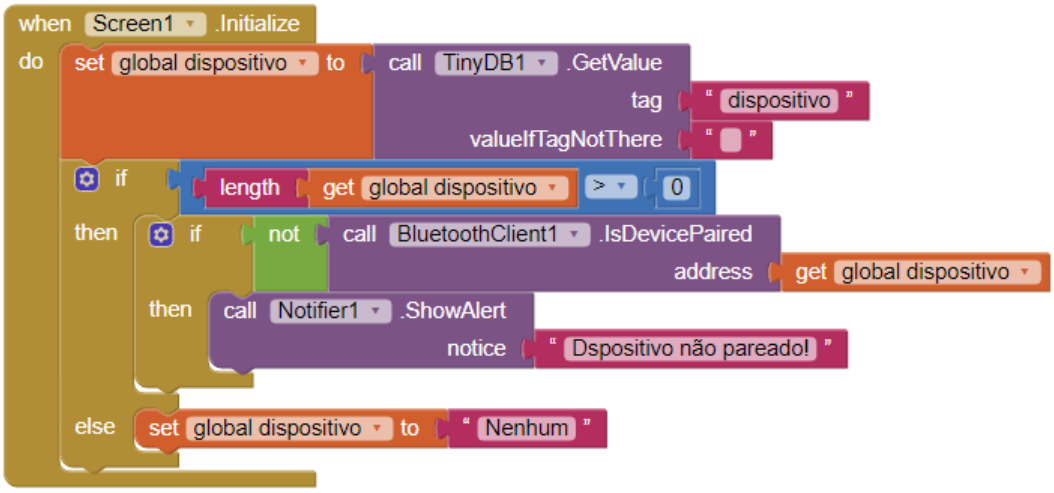

Figura 25 – Bloco Teste de pareamento

Fonte: Autor

No bloco acima, dentro da função “When Screen1 Initialize”, primeiramente é gravado na variável “dispositivos” o último dispositivo utilizado pelo aplicativo. O bloco “call TinyDB1 GetValue” é adicionado na frente do bloco “set global dispositivo to” e define o valor a variável “dispositivo” como sendo o valor da tag “dispositivo” no banco de dados TinyDB1. Caso não haja valor no banco de dados, a variável é definida como um valor vazio.

Abaixo da função “When Screen1 Initialize”, é utilizado uma condição “if”, que testa se há algum dispositivo gravado no banco de dados. O programa testa a quantidade de caracteres da variável “dispositivo”, se for maior que zero, entra na condição, caso contrário, define a variável com o valor “Nenhum”.

Dentro do laço de condição descrito no parágrafo anterior, é criado outro laço onde é testado se o dispositivo não está pareado. Se não estiver, é exibido a notificação "Dispositivo não pareado!".

Abaixo do bloco de inicialização, é adicionado o bloco “when ListPicker BeforeWicking”, que lista todos os dispositivos bluethooth disponíveis no aplicativo, conforme figura 26.

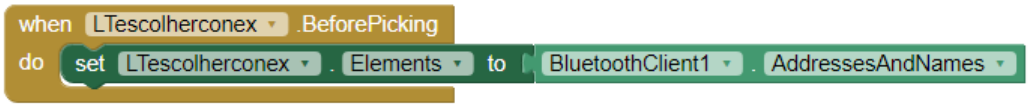

Figura 26 – Bloco para criação da Lista de Dispositivos

Fonte: Autor

Para selecionar o dispositivo desejado na lista, foi criado o bloco da figura 27. O bloco “when ListPicker AfterPicking” da guia ListPicker é selecionado e dentro dele adicionado o bloco de definir variável ligado ao bloco de ’Seleção’. Abaixo dele é colocado o bloco “TinyDB StoreValue” na guia “TinyDB”, responsável por gravar valores dentro do banco de dado. À frente deste bloco é colocado o tag “dispositivo” e o valor a ser gravado como sendo o valor da variável dispositivo. Sendo assim, sempre que selecionado um dispositivo na lista, ele é gravado na variável ’dispositivo’ e o valor dela também é gravado no banco de dados.

Figura 27 – Bloco para gravar seleção de dispositivos

Fonte: Autor

Para a conexão do dispositivo selecionado, é criada uma variável denominada "Conectado" e é definida como falsa. O dispositivo irá tentar uma conexão sempre que o Botão "Conectar" for acionado. Para isso é utilizado o bloco de ação "When Click To" do botão. Neste bloco é testado se o dispositivo está pareado. Caso não esteja, uma notificação é exibida através do bloco "call Notifier ShowAlert", informando que o aparelho não está pareado.

Caso o dispositivo esteja pareado, é tentado a conexão através do bloco "call Bluetooth Client Connect" e esta conexão é testada. Se sim, uma notificação com o texto "Dispositivo Conectado" é exibida na tela. Caso contrário, exibe se "Erro na conexão". A figura 28 apresenta a configuração dos blocos responsáveis pela conexão.

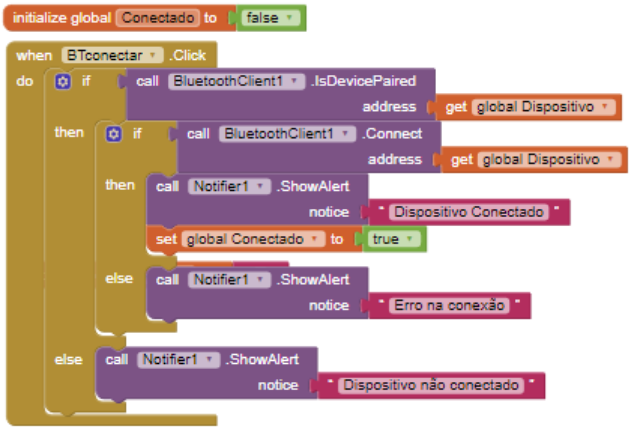

Figura 28 – Blocos para conectar o dispositivo

Fonte: Autor

Para definir as funções do aplicativo, utilizou se o bloco "When Timer" do Clock. Nele, há a verificação da ativação do bluetooth através da condição 'if' e dentro dela verifica se o dispositivo está conectado a outro. Caso esteja, uma variável denominada "Receber" armazenará os dados recebidos através da conexão bluetooth.

Foram criados dois “TextBox” para imprimirem as informações recebidas pelo Aplicativo. O primeiro “TextBox” exibe a situação do sistema, se está ativo ou "Aguardando Leitura". O segundo “TextBox” exibe as informações de volume, situação da bomba e situação do sistema.

Para imprimir o texto recebido do microcontrolador, foi implementada uma função que verifica se no texto enviado pelo microcontrolador há a palavra “Volume”. Se positivo, é impresso na caixa “Volume"todas as variáveis enviadas pela função *“printf”* do Arduino.

A situação do sistema é impressa de maneira semelhante ao volume, o aplicativo verifica a existência da palavra “Falha” nas informações recebidas e se positivo, imprime tudo o que recebe do dispositivo conectado.

Caso o dispositivo não esteja conectado, a caixa “Volume” não imprime nenhum caractere e o “TextBox Aguardando Leitura” imprime exatamente o texto de seu título.

A figura 29 apresenta a estrutura no sistema dos blocos responsáveis por imprimir as informações na tela do Aplicativo.

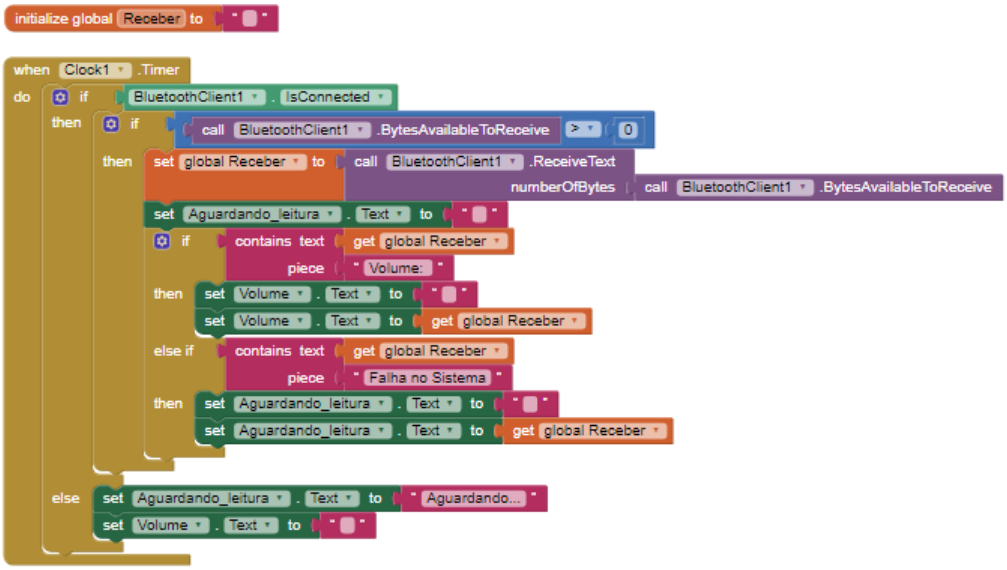

Figura 29 – Blocos para imprimir as informações recebidas

Fonte: Autor

Para dar praticidade ao sistema, foi desenvolvida uma lógica em que o usuário consegue enviar comandos de LIGA/DESLIGA para a bomba e de reestabelecer o modo automático do protótipo através do aplicativo. Foram utilizados três botões, um com a função LIGA, outro com a função DESLIGA e o botão de REARME. Apenas com um click consegue se acionar a bomba manualmente ou reestabelecer o processo natural após entrada em algum laço de repetição.

Os blocos para habilitar estas funções são simples, basta criar um “When Click” para os três botões e com o “Call Send Text” do “Bluetooth Client” enviar uma mensagem de texto para o controlador sempre que o botão for acionado. Para o Botão LIGA, o aplicativo envia o caractere “b” ao microcontrolador e o Botão desliga envia o caractere “a”, já o Botão de Rearme enviará o caractere “c”, conforme figura 30.

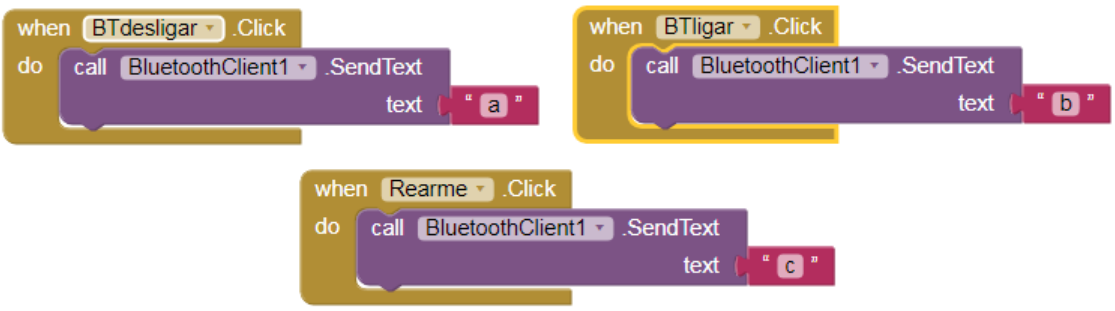

Figura 30 – Blocos para acionar a bomba d’água e rearmar o sistema

Fonte: Autor

4 Testes

Utilizando uma caixa d'água de 1000 litros, um recipiente graduado em 40 litros e uma bomba d'água de 3cv, os componentes do sistema foram integrados e foi dado início aos testes de funcionamento e confiabilidade.

4.1 Testes dos componentes

Após as interligações do dos componentes, iniciaram se os testes das funcionalidades do sistema proposto.

4.1.1 Acionamento da Bomba

Isoladamente, desenvolveu se uma lógica apenas para testar o acionamento da bomba d'água através do relé. O código apenas aciona a bomba por 30 segundos e desliga por 30 segundos.

No relé foi realizado a ligação de uma fase de alimentação da bomba como proposto no protótipo, e quando este recebia um sinal, a bomba ligava, quando o sinal era cortado, a bomba desligava. Este teste foi realizado para verificar se as conexões físicas estavam corretas e para testar também o funcionamento do relé e contator.

4.1.2 Teste do sistema completo

Após a certificação de que o acionamento elétrico da bomba estava funcionando sem anomalias, interligou se todo o sistema e iniciou se o teste com o código principal.

A conexão do aplicativo Android com o microcontrolador foi a primeira situação verificada. Ao abrir o aplicativo, o usuário acionou o botão "Escolher conexão", onde abriu se a aba 31, contendo todos os dispositivos bluetooth disponíveis no momento. O dispositivo selecionado é o que contém o endereço do módulo “HC-05”.

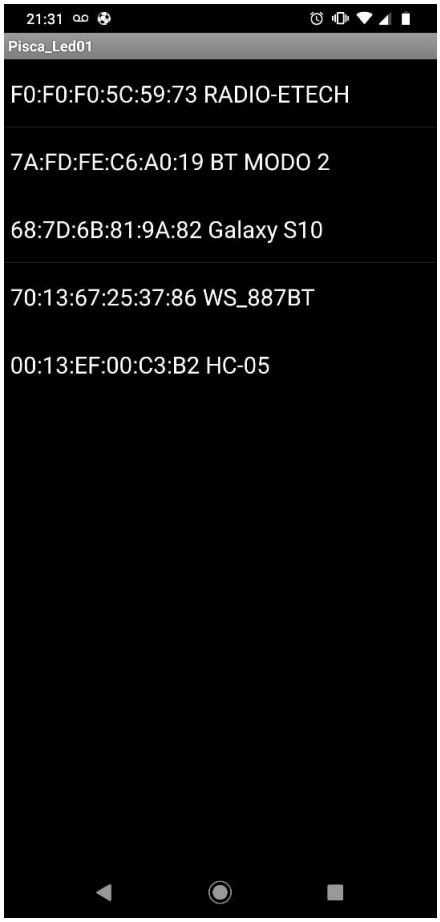

Figura 31 – Lista de dispositivos bluetooth disponíveis

Fonte: Autor

Após a escolha do dispositivo, o aplicativo volta para a interface interativa, onde se conecta clicando no Botão “Conectar”, mostrado mais a frente.

Inicialmente, teve se um desvio desconhecido, pois com os componentes integrados, o microcontrolador não enviava as informações para as interfaces do Aplicativo e Display, ou enviava em formato desconhecido e de maneira inconsistente.

Após verificações no código de programação e ajuda de colegas, foi identificado que o problema estava na função *Interrupt* utilizada para os sensores de fluxo. No programa inicial, a função era habilitada, permanecia assim por 1000 Millisegundos e imediatamente era desabilitada. Esta desabilitação da função não permitia a comunicação do Arduino com o Aplicativo e com o Display, pois as bibliotecas destinadas a isso também necessitam de interrupção.

Para sanar o problema, o programa principal inicia com a função desabilitada, repete se o processo para leitura dos sensores e após a execução da função de leitura dos sensores, a função é habilitada novamente , permitindo a comunicação Serial e do protocolo

I2C.

Posterior a essas mudanças, o sistema operou normalmente, enviando a leitura dos sensores, acionando de maneira automática a bomba, enviando as informações para as interfaces de monitoramento e obedecendo aos comandos manuais nos botões de campo e do aplicativo.

A figura 32 ilustra a interface do aplicativo desenvolvido e funcionando perfeitamente. Nela, pode se ver os botões de escolher o dispositivo que deseja se conectar (Escolher Conexão), o botão para se conectar ao módulo bluetooth escolhido (Conectar), o volume em tempo real, o estado da bomba e os botões “LIGAR BOMBA”,“DESLIGAR BOMBA” e “REARMAR” criados para a interação com o usuário.

Figura 32 – Interface do aplicativo desenvolvido

Fonte: Autor

Em um dado momento, retirou se a mangueira da captação da água para simular uma possível falha do sistema. Com a bomba ligada e sem leitura de vazão por 60 segundos consecutivos, o sistema obedeceu a lógica e logo desligou a bomba, apresentando nas interfaces a mensagem de "Falha"e mostrando o estado da bomba desligada, conforme

ilustra a figura 33.

Figura 33 – Interface apresentando o sistema em falha

Fonte: Autor

Com o sistema operando da maneira desejada, o passo seguinte foi a coleta de amostras para verificar a confiabilidade do protótipo proposto.

4.2 Coleta de amostras

Em geral, sistemas de medição são considerados aceitáveis quando possuem um erro de dispersão inferior a 10%, de acordo com o Manual de MSA da QS-9000 (1997). Baseado nessa informação, foram coletadas amostras do sistema proposto para avaliação de sua confiabilidade e aceitação.

Realizou se dez amostras de um volume conhecido e após a coleta dos dados foram empregados os cálculos média e desvio padrão do sistema. Em cada amostra definiu se que o volume a ser medido seria de 40l. Para obter esse valor e verificar o erro do sistema, acrescentava se 40l de água através do sensor de entrada e retirava se 40l através do sensor

de saída. Esses valores foram medidos em um recipiente graduado de 40 litros. Abaixo segue a tabela com os valores medidos pelos sensores e o volume indicado pelo sistema.

Tabela 1: Valores obtidos em medições dos sensores de entrada e saída.

Sensor entrada (80l) /	Sensor de saída (40l) /	Volume da caixa medido pelo sistema
82l	37l	45l
79l	36l	43l
80l	41l	39l
82l	42l	40l
83l	40l	43l
78l	39l	39l
81l	38l	43l
84l	39l	45l
81l	42l	39l
79l	44l	35l

4.2.1 Média aritmética e Erro Sistemático

Com os dados da tabela acima, foi extraído a média dos volumes de água medidos e realizado o cálculo do erro sistemático estimado do sistema, através da fórmula 4.1 mostrada abaixo.

$$Es = X - V_x \tag{4.1}$$

em que,

Es - Erro sistemático estimado;

X - média dos valores de volume obtidos nas dez medições;

V_x - volume real que o sistema deveria apresentar.

A média dos valores medidos nas 10 amostras foi de 41,1l. Sendo assim, o valor encontrado para *Es* foi de 1,1l.

O *erro sistemático estimado* ocorre em processos sujeitos à variações ambientais, como pressão e temperatura, por exemplo. Ele deve ser minimizado na calibração do sistema, e seu resultado é um dos indicadores de confiabilidade de um instrumento ou sistema de medição.

4.2.2 Desvio Padrão e Erro Padrão

Após calcular o *erro sistemático* do sistema de medição de nível proposto, realizou se o cálculo do desvio padrão da amostra, a fim de obter o erro padrão estimado e analisar a confiabilidade das medidas. O desvio padrão é calculado de acordo com a fórmula 4.2, que é dada pela raiz quadrada da variância amostral.

$$\sigma = \sqrt{\sum \frac{(x_i - \bar{X})^2}{n-1}}$$

(4.2)discriminada abaixo,

σ - desvio padrão;

x_i - valor individual de cada medição

X - média dos valores de volume obtidos nas dez medições;

n - tamanho amostral.

Para a amostra coletada, o valor de σ é de 3,21l, o que indica uma dispersão de 8% em relação ao valor real de medição.

O *desvio padrão* indica o índice de dispersão de uma amostra, e através dele podemos calcular o *erro padrão* do sistema, que é a medida que verifica a confiabilidade da média amostral, dividindo o desvio padrão pela raiz quadrada do tamanho amostral, dado pela equação 4.3.

$$Er = \frac{\sigma}{\sqrt{n}} \tag{4.3}$$

onde,

- Er - Erro padrão do sistema.

Substituindo os valores do desvio padrão e do tamanho amostral, tem se um Erro Padrão estimado de 1,01l para o problema proposto.

4.2.3 Intervalo de Confiança

Enfim, com os resultados acima é possível estabelecer um intervalo de confiança para a amostra utilizada.

Com uma margem de 95% de probabilidade de acerto, o percentil de 1,96 é utilizado para se calcular o Intervalo de Confiança da amostra. A equação 4.4 estabelece os valores mínimo e máximo deste intervalo.

$$[X - Erx1,96; X - Erx1,96] \tag{4.4}$$

Dessa forma, o Intervalo de Confiança da amostra apresentada é [39,12l ; 42,98l].

5 Conclusão

O protótipo desenvolvido possui a finalidade de ser um sistema dinâmico e prático, de fácil instalação, baixo custo e eficiente, onde se consegue automatizar o abastecimento, receber as informações de volume em tempo real e indicar possíveis falhas.

O aplicativo desenvolvido para a interação do usuário com o sistema foi bem sucedido, respeitando as particularidades do método utilizado. A troca de informações entre o sistema e a interface funcionou satisfatoriamente, respondendo aos comandos realizados pelo usuário e enviando às informações necessárias de maneira consistente. Mantendo as características da interface proposta, talvez uma comunicação via Ethernet possa deixar o sistema mais eficaz, pois o usuário não terá a restrição de distância para monitorá-lo ou controlá-lo.

O circuito de acionamento da bomba também trouxe uma resposta positiva. Os componentes utilizados para isolar a alimentação dos circuitos do microcontrolador e da bomba cumpriram o objetivo e responderam aos sinais de acionamento sem falhas.

A utilização de sensores de fluxo, se deu pela possibilidade de mapear possíveis problemas na bomba de abastecimento e de se evitar danos à ela e desperdícios de água, além da instalação não necessitar de uma calibração diferente para cada tipo e formato de caixa d'água. Em contrapartida, caso haja algum vazamento na caixa, este passará despercebido no programa, pois os sensores de fluxo são instalados nas tubulações de entrada e saída. O erro geralmente apresentado pelos sensores de fluxo é relativamente grande e pode comprometer a confiabilidade do sistema. Neste caso, fizemos uma pequena análise de dados para avaliar a confiabilidade do protótipo, que nos apresentou valores aceitáveis. Talvez a utilização de algum sensor ultrassônico ou a laser em redundância com os sensores de fluxo seja uma alternativa mais eficaz, tendo como contrapartida o fato de terem que ser calibrados para cada tipo e tamanho de caixa diferentes.

Dentro do objetivo proposto, o trabalho atendeu às expectativas, pois foi capaz de automatizar o processo de abastecimento, dando ao usuário algumas opções de monitoramento e abrindo o leque para outras propostas similares que possam ser apresentadas, utilizando diversos conceitos de automação e controle aprendidos durante a graduação.

Para trabalhos futuros, fica como sugestão um foco na análise dos dados mensurados pelos sensores utilizados. Estimar o erro através um número superior de amostras e realizar mais pontos de calibração, trarão mais profundidade e confiabilidade na análise, de acordo com (ALBERTAZZI; SOUZA, 2017).

Referências

ALBERTAZZI, A.; SOUZA, A. *Fundamentos de Metrologia Científica e Industrial*. [S.l.]: Manole, 2017. Citado na página 58.

ALECRIM, E. *Tecnologia Bluetooth: o que é e como funciona?* 2019. Url https://www.infowester.com/bluetooth.php. Citado na página 25.

BRKAMBIENTAL. *Saneamento básico no Brasil: conheça os números das regiões do país*. 2020. Disponível em: <https://blog.brkambiental.com.br/saneamento-basico-no-brasil/>. Citado na página 12.

DORNELAS, E.; OLIVEIRA, C. S. Monitoramento de consumo doméstico de Água utilizando uma meta-plataforma de iot. *Revista de Engenharia e Pesquisa Aplicada*, Revista de Engenharia e Pesquisa Aplicada, v. 2, n. 2, 2017. Citado 2 vezes nas páginas 16 e 17.

EOS. *AS GRANDES DIFICULDADES DA DISTRIBUIÇÃO DA ÁGUA NO BRASIL*. 2017. Http://www.eosconsultores.com.br/dificuldades-distribuicao-da-agua-no-brasil/. Citado na página 12.

FILTSOFF, R. B.; MARTINS, D. M. S. *Sistema de monitoramento e controle do consumo de água residencial*. 20 p. Monografia (Monografia) — CENTRO DE ENSINO SUPERIOR DE JUIZ DE FORA, Juiz de Fora, 2018. Citado na página 13.

GUIMARãES, F. *Módulo Bluethooth - Criando Aplicativo*. 2019. `http://mundoprojetado.com.br/modulo-bluetooth-criando-aplicativo-parte-2/`. Acessado em 09/09/2019. Citado na página 45.

KAELBLING, P. L. *Learning in embedded Systems*. [S.l.]: Massachusetts Institute Of Technology, 1993. Citado na página 13.

MUNDIM, K. C. *O Efeito Hall*. 1999. Disponível em: <http://ensinoadistancia.pro.br-/EaD/Eletromagnetismo/EfeitoHall/EfeitoHall.html>. Citado 2 vezes nas páginas 17 e 18.

MURTA, G. *Guia completo do Display LCD- Arduino*. 2019. Https://blog.eletrogate.com/guia-completo-do-display-lcd-arduino/. Citado 2 vezes nas páginas 22 e 23.

NIDEJELSKI, D. M. *Projeto de um sistema de controle de combustíveis em tanques de armazenamento utilizando microcontrolador Arduino*. 65, f. Monografia (Monografia) — UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL, Porto Alegre, 2018. Citado na página 16.

OLIVEIRA, H. R. I.; SANTOS, R. B. C.; RODRIGUES, L. A. M. Desenvolvimento de um aplicativo android para monitoramento microcontrolado do nÍvel de um reservatÓrio de Água residencial em tempo real. *ConferÊncia de Estudo em Engenharia Elétrica*, 2014. Citado 2 vezes nas páginas 16 e 17.

PENA, A. F.; RODOLFO. Água potável no mundo. *Nature*, Nature Publishing Group, v. 408, n. 6810, p. 361–365, 2000. Citado na página 12.

PéREZ, A. *Conhecendo o cerne do microcontrolador ATmega328p Arduino Uno*. 2019. Disponível em: <http://www.newtoncbraga.com.br/index.php/microcontrolador-/138-atmel/14863-conhecendo-o-cerne-do-microcontrolador-atmega328p-arduino-uno-mic165>. Citado na página 21.

SANTOS, D. *Relê*. 2020. `https://www.infoescola.com/eletronica/rele/`. Acessado em 09/02/2020. Citado na página 26.

SANTOS, M. P.; OLIVEIRA, C. K. J. Automação de baixo custo para reservatórios de água. *Revista de Engenharia e Pesquisa Aplicada*, Revista Principia, n. 25, p. 1–64, 2014. Citado na página 13.

SOUSA, R.; SARDINHA, V. M. *Água*. 2019. Disponível em: <https://brasilescola.uol-.com.br/geografia/agua.htm>. Citado na página 12.

SOUZA, F. *Arduino Uno*. 2013. Access date: 19 março 2019. Disponível em: <https://www.embarcados.com.br/arduino-uno/>. Citado na página 21.

VIDAL, V. *Módulos bluetooth HC-05 e HC-06 para comunicação com dispositivos móveis com arduino*. 2019. Url https://blog.eletrogate.com/modulos-bluetooth-hc05-e-hc06-para-comunicacao-com-dispositivos-moveis-com-arduino/. Citado na página 25.

YF-S403 3/4. Citado na página 19.

Apêndices

APÊNDICE A – Código desenvolvido na IDE do Arduino para o sistema proposto

include <LiquidCrystal$_I$2$C.h$ >

include <Wire.h>

include <SoftwareSerial.h> //Comandos Seriais

LiquidCrystal$_I$2$Clcd(0x3F, 16, 2)$;

SoftwareSerial bluetooth (10, 11); // TX-RX

//Define os pinos que serão utilizados para ligação ao display

const int bomba = 8;//Variável que recebe o pino digital onde a ponte H da bomba d'água é conectada

int volume=0; //VAriável que armazena o volume da caixa

int volume1 = 0;

int Pulso2; //Variável para a quantidade de pulsos sensor 2

int Pulso1; //Variável para a quantidade de pulsos sensor 1

int j; //Variável para contagem sensor 1

int i; //Variável para contagem sensor 2

int vazaoagua1; //Variável para armazenar o valor em L/min

int vazaoagua2; //Variável para armazenar o valor em L/min

float vazaomin2 = 0;

float vazaointerr2 = 0; // variavel que armazena a vazao em 5 segundos, para verificar se a bomba está mandando água

int valormedia1 = 0; //Variável para tirar a média a cada 1 minuto

int valormedia2 = 0; //Variável para tirar a média a cada 1 minuto

int estadobomba;

int botaofalha = 13; //botão para rearmar o sistema, caso haja alguma falha

```
int estadofalha = 0; //variável que armazena o estado do botao rearme
int falhaapp = 0;
int funcX = 0; //valor enviado pelo botão rearme
int funcZ = 0;//valor armazenado para o botão rearme
int botaoliga = 12; //botão liga/desliga para acionamento manual da bomba
int botreset = 7; //botão para resetar o sistema, desligando a bomba ezerando o volume
int estadobot = 0; //variável que armazena o estado do botao liga/desliga
int estadoreset = 0; //variável que armazena o estado do botão reset
int funcA = 0; //valor enviado pelo botão liga/desliga
int funcB = 0; //valor armazenado para o botão liga/desliga
int ligaapp = 0;
int desligaapp = 0;
int readBluetooth = 0;
void setup()
{
lcd.begin(16, 2);//Inicia o LCD definindo o com 16 colunas e 2 linhas
Serial.begin(9600);//
bluetooth.begin(9600);
pinMode(bomba, OUTPUT);//Define o pino onde a ponte H é conectada como uma saída
pinMode(botaoliga, INPUT); //Define o pino do botão liga/desliga como entrada
pinMode(botaofalha, INPUT); //define o pino do botão de rearmar o sistema como entrada
pinMode(2, INPUT);
attachInterrupt(0, incrpulso1, RISING);//Configura a porta digital 2, para interrupção sensor 1
pinMode(3, INPUT);
attachInterrupt(1, incrpulso2, RISING);//Configura a porta digital 3, para interrupção sensor 2
lcd.init();
```

```
void loop () {
void vol () {valormedia1 = 0;
valormedia2 = 0;
estadobot = 0;
estadofalha = 0;
ligaapp = 0;
desligaapp = 0;
Pulso1 = 0; //Começa do 0 variável para contar os giros das pás internas,em segundos
Pulso2 = 0; //Começa do 0 variável para contar os giros das pás internas,em segundos
delay(50);
cli();
delay(1);
sei();
delay(1000);
cli();
delay(1);
vazaoagua1 = Pulso1 / 5,88; //Converte para Litro/minuto - Mudar valor com a calibração
vazaoagua2 = Pulso2 / 5,88; //Converte para Litro/minuto - Mudar valor com a calibração
//vazaomin2 = vazaomin2 + vazaoagua2;
valormedia1 = valormedia1 + vazaoagua1; //Soma a vazão para o calculo da valormedia
valormedia2 = valormedia2 + vazaoagua2; //Soma a vazão para o calculo da valormedia
volume = valormedia2 - valormedia1;
volume1 = volume1 + volume;
j++;
```

```
i++;
if (volume1 < 0) volume1 = 0; // garante que caso haja alguma falha no sistema, o volume não fique negativo
if (volume1 > 1000) volume1 = 1000; // garante que caso a caixa encha e a bomba não desligue, o volume máximo da caixa seja o limite contabilizado
sei();
lcd.setBacklight(HIGH);
lcd.clear();
lcd.setCursor(0,0);//Posiciona o cursor na coluna 1, linha 0;
lcd.print("Volume: ");
lcd.print(volume1);
bluetooth.println("Volume: ");
bluetooth.println(volume1);
if (volume1 <= 500) {digitalWrite(bomba, HIGH); //Seacaixaestivercom500mloumenos, lig
if(volume1 >= 1000)
{
estadobomba = digitalRead(bomba);
if (estadobomba == LOW) {
lcd.setCursor(0, 1);
lcd.print("Bomba desligada");
bluetooth.println("Bomba desligada");
if (estadobomba == HIGH) {
lcd.setCursor(0, 1);
lcd.print(- Bomba ligada");
bluetooth.println(- Bomba ligada");

voidincrpulso1()
{
Pulso1++;
```

```
void incrpulso2 ()
{
Pulso2++;
```