

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

ENGENHARIA DE COMPUTAÇÃO

OFICINA DE INTEGRAÇÃO I

GABRIEL DE OLIVEIRA FREIRE SILVA

MURILO MASCARIN GUIMARÃES

GABRIEL SAFRANSKI TORRES

**MONITORAMENTO E CONTROLE DE TEMPERATURA ATRAVÉS DE SENSORES
CONTROLADOS POR ARDUINO**

RELATÓRIO FINAL DO PROJETO

**CURITIBA
2019**

GABRIEL DE OLIVEIRA FREIRE SILVA

WhatsApp: (41) 991986147

E-mail: gabrielsilva.1998@alunos.utfpr.edu.br

MURILO MASCARIN GUIMARÃES

WhatsApp: (14) 998933556

E-mail: mumascarin@yahoo.com.br

GABRIEL SAFRANSKI TORRES

WhatsApp: (41) 992770041

E-mail: gabrieltorres@alunos.utfpr.edu.br

**MONITORAMENTO E CONTROLE DE TEMPERATURA ATRAVÉS DE SENSORES
CONTROLADOS POR ARDUINO**

**Projeto elaborado para a obtenção de nota
na disciplina Oficina de Integração I.**

**CURITIBA
2019**

1 INTRODUÇÃO

Este protótipo simula um sistema de monitoramento e controle de temperatura de dois reservatórios de água, com a utilização de sensores conectados a um *shield* Arduino. As leituras captadas pelos sensores serão enviadas a um computador, onde serão apresentados graficamente por meio do software *MATLAB*. Com o uso de comandos preestabelecidos, o microcontrolador realizará instruções para estabilização da temperatura em ambos os reservatórios.

1.1 MOTIVAÇÃO

O projeto tem como inspiração o trabalho de conclusão de curso *Aplicação de sensores capacitivos para monitoramento da formação de parafina em oleodutos*¹, neste foi exposto um estudo sobre a formação de parafina em oleodutos, que muitos sabem são as tubulações usadas na extração de óleo bruto em poços de petróleo, esse óleo bruto extraído é composto por várias substâncias, dentre elas diversos tipos de parafinas.

Quando o óleo é extraído, está a uma temperatura aproximada de 65°C, e escoar por um oleoduto que encontra-se em uma temperatura menor, principalmente em operações submarinas, essa diferença de temperatura é extremamente relevante para a deposição de parafinas, pois a deposição de parafinas está associada ao equilíbrio de fases.

Quando esse equilíbrio é quebrado, ocorre o aparecimento de depósitos parafínicos, que são provocados pelo resfriamento do petróleo e/ou desprendimento das frações mais leves originalmente dissolvidas neste petróleo.

Em decorrência da exposição da parafina a baixas temperaturas, elas tendem a ser precipitarem em forma de cristais e caracterizam dessa forma uma fase sólida.



Figura 1 – Parafina despositada na parede interna de oleoduto
Fonte: STATOIL (2011)

Assim esses cristais parafínicos, depositados na superfície da parede interna da tubulação acabam diminuindo a intensidade do fluxo de óleo bruto, que pode ser transportado pelo oleoduto, acarretando em uma sobrecarga do sistema entre outros problemas que resultariam em paradas não programadas na produção, para a execução de limpeza nos dutos ou até a substituição dos mesmos.

Tendo em vista a seriedade do problema gerado pelo depósito de parafinas em oleodutos, os alunos acima citados, desenvolveram um sistema capaz de medir a deposição de parafina que ocorre em tubulações durante o processo de exploração de petróleo.

O sistema é composto de um sensor capacitivo e uma eletrônica associada para medição da variação de capacitância.

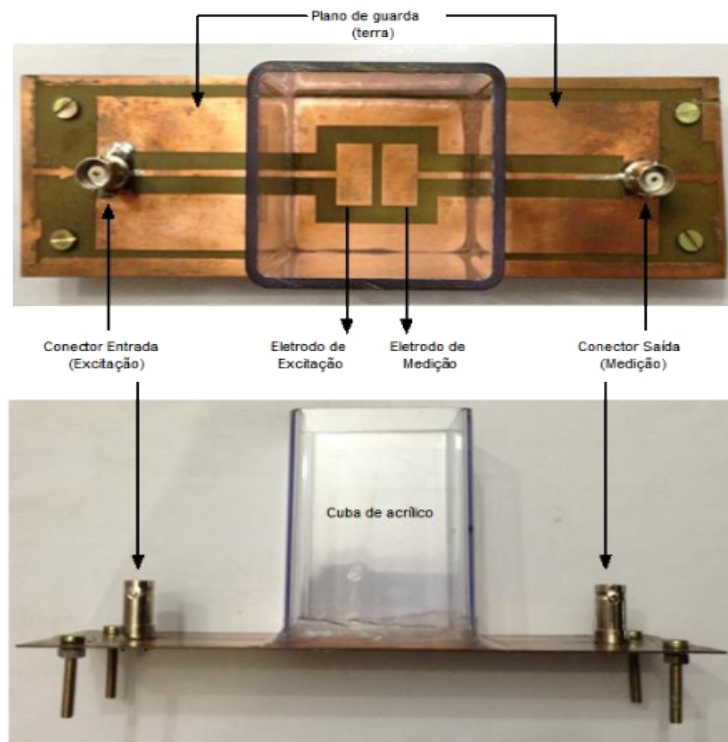


Figura 2 – Sensor capacitivo

Fonte: Aplicação de sensores capacitivos para monitoramento da formação de parafina em oleodutos (2013)

1.2 OBJETIVOS

Com base no sistema exposto acima, desenvolvemos um sistema de monitoramento e controle de temperatura através de sensores de temperatura DS18B20 controlados através do Arduino Uno.

O projeto consiste em dois reservatórios contendo água, um dos reservatórios simula uma tubulação -como um oleoduto ou outra similar- a água neste reservatório sofre variações de temperatura, inicialmente ela será resfriada até uma temperatura previamente estipulada -simulando a formação de cristais ou gelo dentro da tubulação- a partir do momento que a água no primeiro reservatório atingir a temperatura programada, o Arduino irá ativar o aquecedor de água -instalado junto com a bomba d'água no segundo reservatório, que representa o sistema de aquecimento- que irá aquecer a água até 50°C, quando a água atingir a temperatura de 50°C, o Arduino ativará a bomba d'água que tem como função fazer com que a água circule através da serpentina de cobre instalada entre os dois reservatórios, aumentando assim a temperatura no primeiro reservatório.

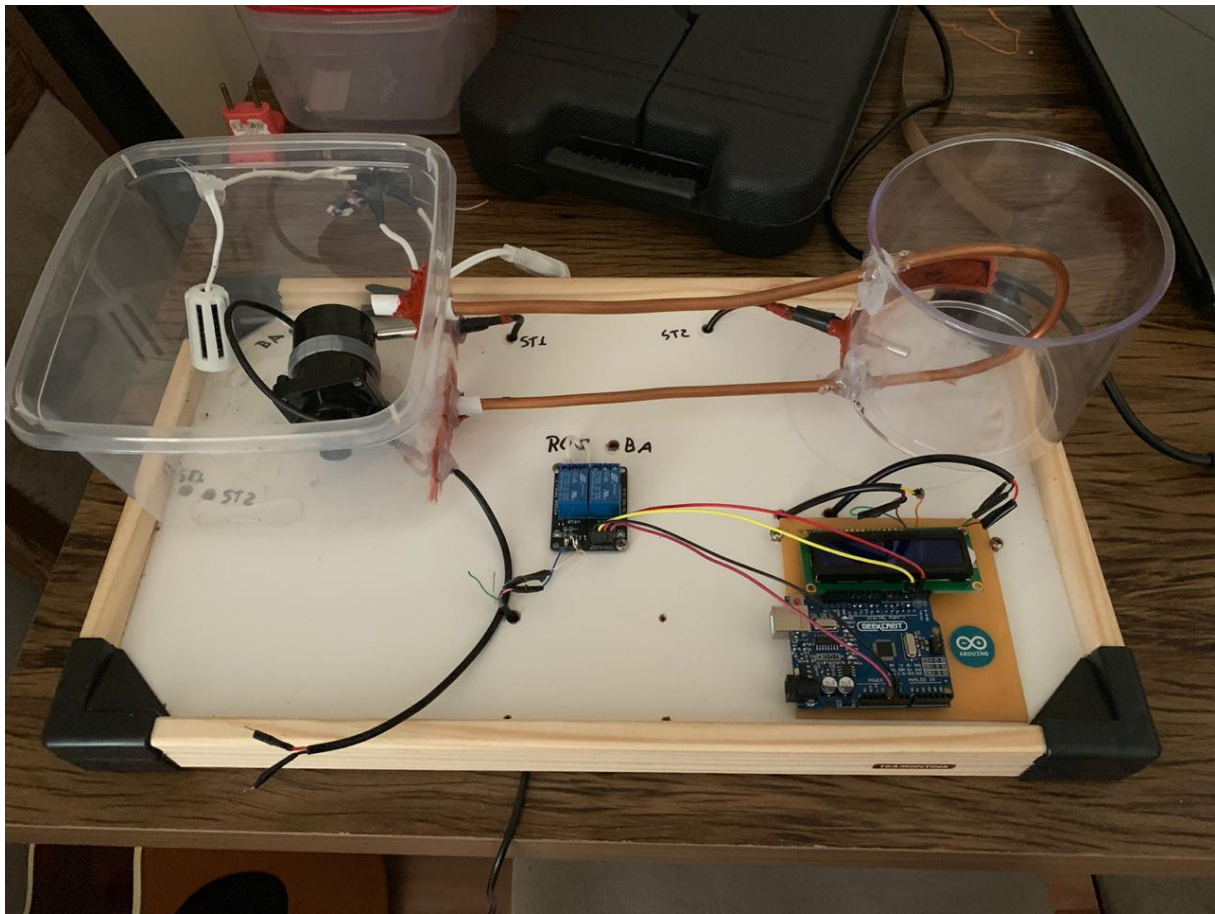


Figura 3 – Protótipo do projeto
Fonte: Autoria própria

2 HARDWARE

2.1 DIAGRAMA DO PROJETO

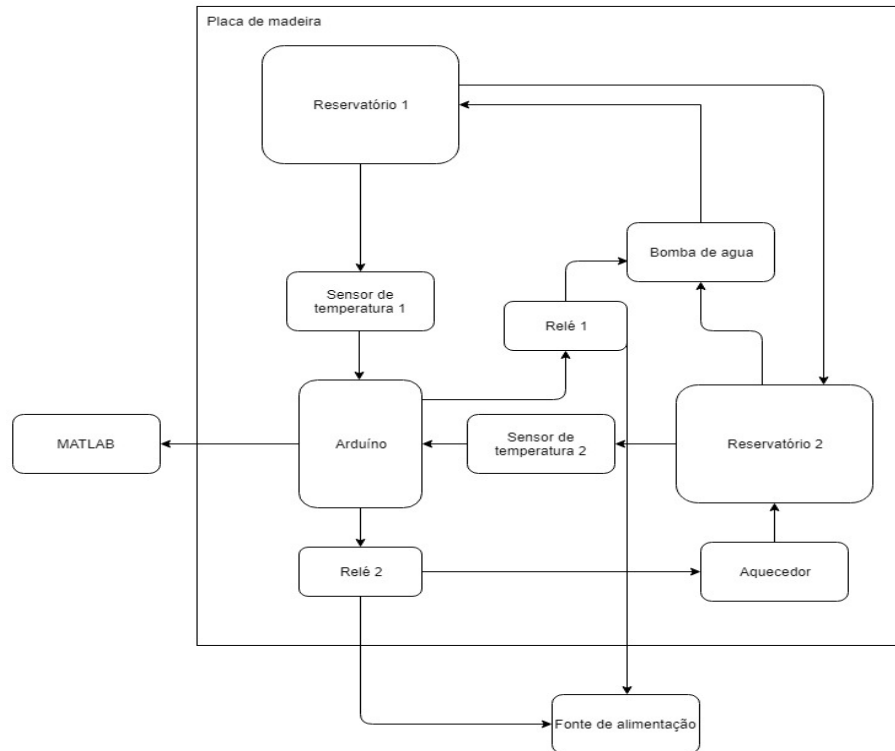


Figura 4 – Diagrama Ilustrativo feito no Draw.io
Fonte: Autoria própria

2.2 ESQUEMÁTICO ELETRÔNICO

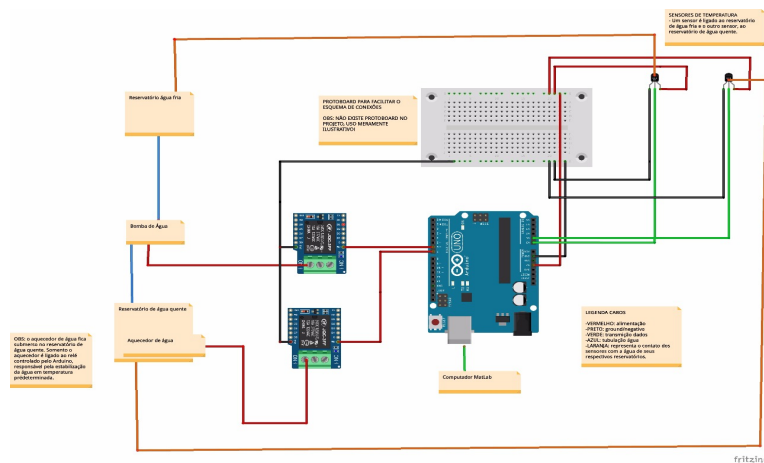


Figura 5 – Esquemático Eletrônico Ilustrativo feito no Fritzing
Fonte: Autoria própria

2.3 PROCESSO DE CRIAÇÃO DA PLACA DE CIRCUITO IMPRESSO

2.3.1 LAYOUT

Primeiramente foi necessário elaborar o layout do circuito em uma placa, essa etapa foi realizada utilizando o software de design de PCB Eagle, que disponibilizado gratuitamente pela Autodesk para alunos e educadores.

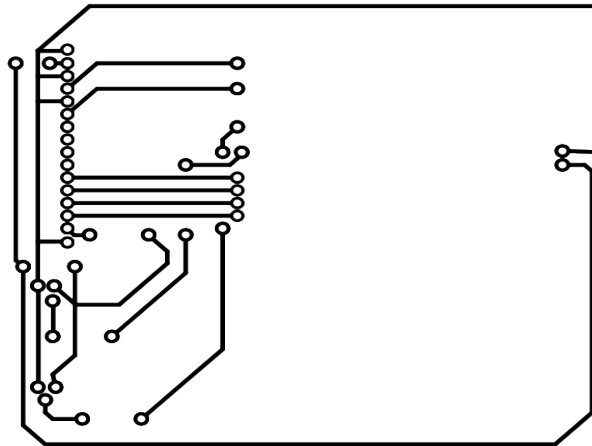


Figura 6 – Placa de circuito impresso feita no Eagle
Fonte: Autoria própria

2.3.2 IMPRESSÃO DO LAYOUT E TRANSFERÊNCIA PARA A PLACA

Após a criação do layout, o mesmo foi impresso em papel fotocópia e sobreposto a placa, assim foi iniciado o processo de fixação das trilhas e ilhas sobre a placa, através da transferência de calor gerada pelo ferro de passar, colocado sobre a placa.

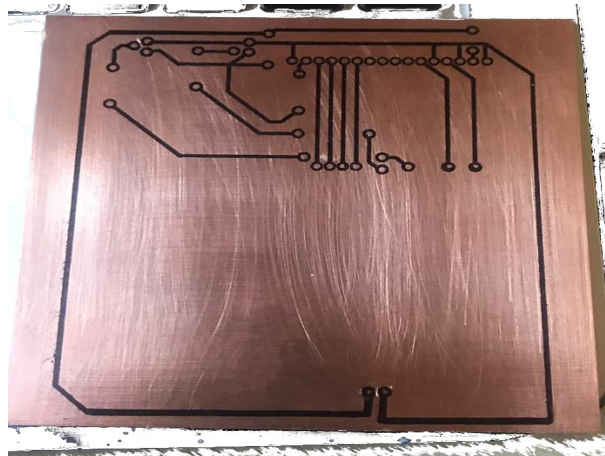


Figura 7 – Layout aplicado a placa
Fonte: Profº Rubens Farias

2.3.3 CORROSÃO DA PLACA

Essa é a etapa final na construção da placa de circuito impresso.

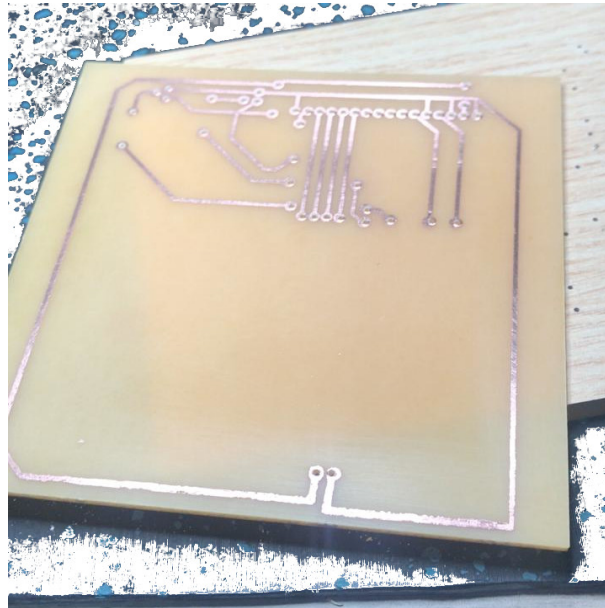


Figura 8 – Placa de circuito impresso após a corrosão
Fonte: Profº Rubens Farias

2.4 COMUNICAÇÃO ENTRE O ARDUINO, LCD E SENSORES

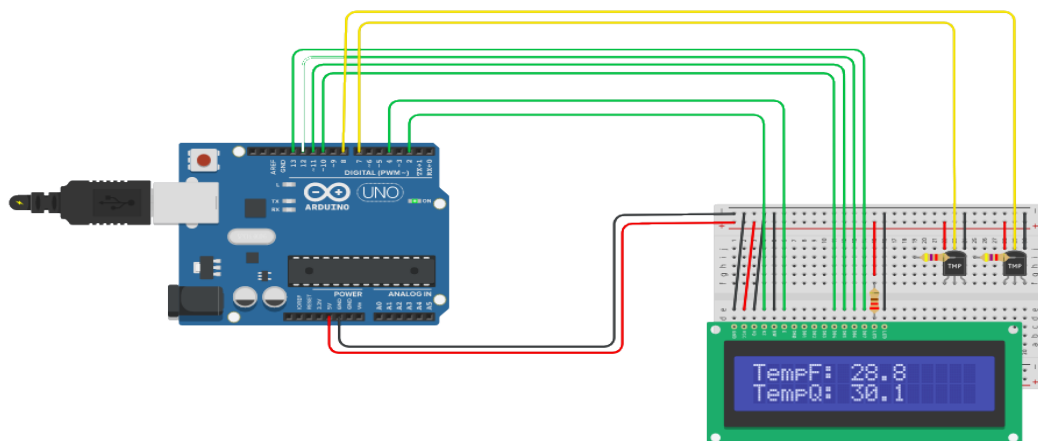


Figura 9 – Simulação realizada no Tinkercad.com
Fonte: Autoria própria

2.5 MONTAGEM DO PROTÓTIPO

2.5.1 PROTÓTIPO INICIAL

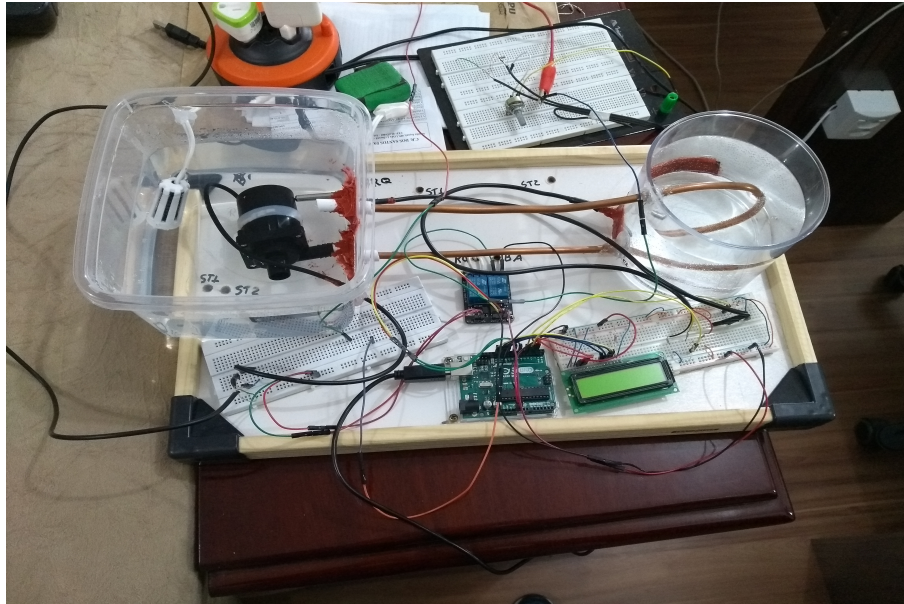


Figura 10 – Protótipo inicial do projeto
Fonte: Autoria própria

2.5.2 ORGANIZAÇÃO DO PROTÓTIPO INICIAL

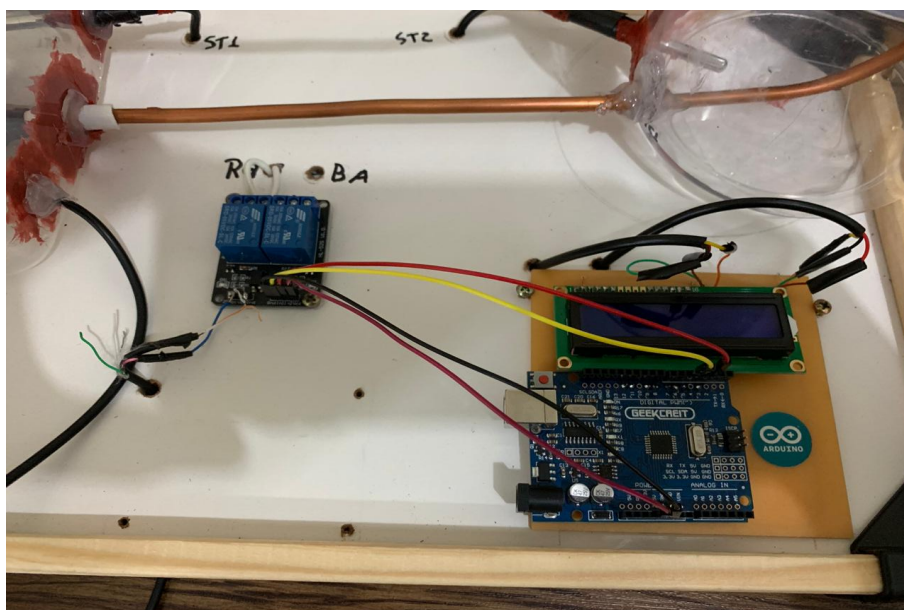


Figura 11 – Protótipo inicial empregado na placa
Fonte: Autoria própria

3 SOFTWARE

3.1 SOFTWARES UTILIZADOS NO DESENVOLVIMENTO DO PROJETO

- *IDE Arduino*: É uma aplicação multiplataforma escrita em Java derivada dos projetos Processing e Wiring. Foi utilizado para o desenvolvimento do código(.ino), que passa as instruções de execução para o Arduino Uno.
- *Octave*: É uma linguagem computacional, desenvolvida para computação matemática. Foi utilizado na tentativa de transmissão gráfica em tempo real das leituras captadas pelos sensores.
- *MATLAB*: O software integra análise numérica, cálculo com matrizes, processamento de sinais e construção de gráficos em ambiente fácil de usar onde problemas e soluções são expressos somente como eles são escritos matematicamente, ao contrário da programação tradicional. Foi utilizado para a tentativa de gerar os gráficos em tempo real, das leituras captadas pelos sensores.
- *ProjectLibre*: É uma versão de código aberto do software para gerenciamento de projetos OpenProj. Foi utilizado na elaboração do cronograma do projeto.
- *Draw.io*: Editor gráfico online no qual é possível desenvolver desenhos, gráficos e outros. Foi utilizado na criação do diagrama do projeto.
- *Fritzing*: Software para montagem de protótipos de circuitos elétricos. Foi utilizado na criação do esquemático eletrônico do projeto.
- *Eagle*: É um software de design de PCB com um editor de esquemático. Foi utilizado na elaboração da placa de circuito impresso do projeto.
- *Tinkercad.com*: Software para modelagem 3D online que incorpora a possibilidade de criação e simulação de circuitos eletrônicos digitais, incluindo o uso do Arduino. Sua utilização no projeto se deu através da realização de algumas simulações de comunicação entre o Arduino, LCD e sensores de temperatura.
- *Overleaf.com*: Overleaf é uma ferramenta colaborativa de escrita online em LaTeX e Rich Text. Foi utilizado para a elaboração do relatório final do projeto.

3.2 CÓDIGO(.INO) DESENVOLVIDO PARA O PROJETO

CodigoGeral §

```

1 //Bibliotecas
2 #include <OneWire.h>
3 #include <DallasTemperature.h>
4 #include <LiquidCrystal.h>
5
6 //Pinos digitais .
7 #define ONE_WIRE_BUS_1 4 //Sensor Fonte fria.
8 #define ONE_WIRE_BUS_2 5 //Sensor Fonte quente.
9 #define pin_1 7 //Rabo quente.
10 #define pin_2 8 //Bomba d'água.
11
12 //OneWire objetos.
13 OneWire oneWire_1(ONE_WIRE_BUS_1);
14 OneWire oneWire_2(ONE_WIRE_BUS_2);
15
16 //Dallas Temperature objetos.
17 DallasTemperature sensors_1(&oneWire_1);
18 DallasTemperature sensors_2(&oneWire_2);
19
20 //LiquidCrystal objetos.
21 LiquidCrystal LCD(2, 3, 10, 11, 12, 13);
22
23 void setup(void)
24 {
25     //Inicialização da porta serial.
26     Serial.begin(9600);
27
28     //Inicialização dos sensores.
29     sensors_1.begin();
30     sensors_2.begin();
31
32     //Inicialização do relê.
33     pinMode(pin_1, OUTPUT);
34     pinMode(pin_2, OUTPUT);
35
36     //Inicialização do LCD.
37     LCD.begin(16, 2);
38 }

```

Figura 12 – Bibliotecas e Setup do projeto desenvolvido na IDE Arduino
Fonte: Autoria própria

```

44 void loop()
45 {
46     //Leitura das temperaturas.
47     Serial.print("Aguardando a leitura...");
48
49     //Envia um comando requisitando a temperatura.
50     sensors_1.requestTemperatures();
51     sensors_2.requestTemperatures();
52
53     Serial.print("DONE");
54     Serial.println();
55
56     //Cria variáveis e inicializa com os valores retornados pelos sensores.
57     float tempF = sensors_1.getTempCByIndex(0);
58     float tempQ = sensors_2.getTempCByIndex(0);
59
60     //Printa no monitor serial.
61     Serial.print("TempF: ");
62     Serial.print(tempF);
63     Serial.println();
64     Serial.print("TempQ: ");
65     Serial.print(tempQ);
66     Serial.println();
67
68     //Printa no LCD.
69     LCD.clear();
70     LCD.setCursor(0, 0);
71     LCD.print("TempF: ");
72     LCD.print(tempF);
73     LCD.setCursor(0, 1);
74     LCD.print("TempQ: ");
75     LCD.print(tempQ);
76     delay(2000);
77
78     //Referente ao Relê.
79     Serial.println("Relê.....");
80
81     //Verificação das leituras captadas pelo sensor 1.
82     if(tempF != DEVICE_DISCONNECTED_C){
83         Serial.println();
84         Serial.print("Temperatura Fonte Fria: ");
85         Serial.println(tempF);
86
87         if(tempF <= 18.0){
88             digitalWrite(pin_2, LOW); //Caso a temperatura na fonte fria seja menor que 12.0°C, a bomba d'água é ativada.
89             delay(10000);
90         } else {
91             digitalWrite(pin_2, HIGH); //Caso contrário, a bomba d'água é desativada ou permanece desativada.
92         }
93         delay(1000);
94     } else {
95         Serial.println("Erro: Não foi possível fazer a leitura da temperatura na Fonte Fria.");
96     }
97     //Verificação das leituras captadas pelo sensor 2.
98     if(tempQ != DEVICE_DISCONNECTED_C){
99         Serial.print("Temperatura Fonte Quente: ");
100         Serial.println(tempQ);
101         Serial.println();
102
103         if(tempQ <= 20.0){
104             digitalWrite(pin_1, LOW); //Caso a temperatura na fonte quente seja menor que 20.0°C, o rabo quente é ativado.
105             delay(10000);
106         } else {
107             digitalWrite(pin_1, HIGH); //Caso contrário, o rabo quente é desativado ou permanece desativado.
108         }
109         delay(1000);
110     } else {
111         Serial.println("Erro: Não foi possível fazer a leitura da temperatura na Fonte Quente.");
112     }
113 }

```

Figura 13 – Loop do projeto desenvolvido na IDE Arduino
Fonte: Autoria própria

4 RESULTADOS

4.1 CRONOGRAMA DO PROJETO

4.1.1 Planilha de Atividades:

Nome	Início	Fim
Etapa 1	11/03/19 12:00	01/04/19 15:30
Definição da equipe.	11/03/19 12:00	18/03/19 13:00
Início das discussões sobre o projeto.	18/03/19 13:00	25/03/19 13:00
Definição e discussão com os professores sobre o projeto a ser desenvolvido.	25/03/19 13:00	01/04/19 13:00
Desenvolvimento do esboço do projeto.	01/04/19 13:00	01/04/19 13:00
Elaboração da proposta de projeto.	25/03/19 13:00	01/04/19 13:00
MARCO 0 - Apresentação da proposta de projeto escolhida pela equipe, para os professores e a turma.	01/04/19 13:00	01/04/19 15:30
Etapa 2	08/04/19 13:00	29/04/19 15:30
Ajustes a proposta de projeto e elaboração do orçamento inicial do projeto.	08/04/19 13:00	15/04/19 13:00
Compra dos materiais necessários para a elaboração do projeto.	15/04/19 13:00	22/04/19 13:00
Montagem da estrutura física do projeto na base de madeira.	22/04/19 13:00	29/04/19 13:00
MARCO 1 - Apresentação da estrutura física do projeto aos professores e a turma.	29/04/19 13:00	29/04/19 15:30
Etapa 3	06/05/19 13:00	27/05/19 15:30
Testes com os componentes elétricos, para se ter uma melhor compreensão acerca do funcionamento dos mesmos.	06/05/19 13:00	13/05/19 13:00
Pausa no projeto.	13/05/19 13:00	20/05/19 13:00
Início do desenvolvimento do projeto na IDE Arduino.	20/05/19 13:00	27/05/19 13:00
Primeiros testes com os displays contidos em cada reservatório.	20/05/19 12:00	27/05/19 13:00
MARCO 2 - Apresentação das primeiras leituras captadas pelos sensores e mostradas nos displays, para os professores e a turma.	27/05/19 13:00	27/05/19 15:30
Etapa 4	03/06/19 13:00	17/06/19 15:30
Transmissão dos dados do Arduino para o computador, para que as leituras captadas pelos sensores sejam esboçadas graficamente através do MATLAB.	03/06/19 13:00	10/06/19 13:00
Primeiros testes com o MATLAB.	03/06/19 13:00	10/06/19 13:00
Continuação do desenvolvimento do projeto no MATLAB.	10/06/19 13:00	17/06/19 13:00
MARCO 3 - Apresentação dos primeiros gráficos gerados através do MATLAB, para os professores e a turma.	17/06/19 13:00	17/06/19 15:30
Etapa 5	24/06/19 13:00	08/07/19 15:30
Verificação do funcionamento de cada componente elétrico e manutenção, caso, seja necessário.	24/06/19 13:00	01/07/19 13:00
Elaboração dos testes finais com o MATLAB.	01/07/19 13:00	08/07/19 13:00
Revisão da documentação do projeto.	01/07/19 13:00	08/07/19 13:00
MARCOS 4 e 5 - Apresentação final e entrega do relatório final do projeto.	08/07/19 13:00	08/07/19 15:30

Monitoramento de temperatura de reservatório de água, através de sensores microcontrolados por Arduino.

Figura 14 – Planilha de Atividades feita no ProjectLibre
Fonte: Autoria própria

4.1.2 Diagrama de Gantt:

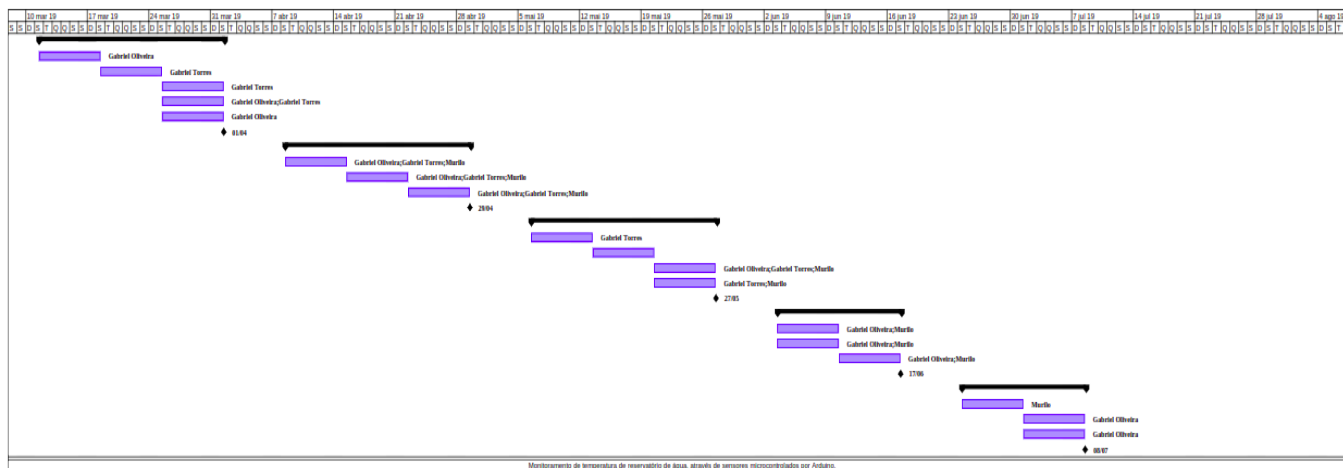


Figura 15 – Diagrama de Gantt feito no ProjectLibre
Fonte: Autoria própria

4.2 LISTA DE MATERIAIS:

QUANTIDADE	MATERIAIS	PREÇO(R\$)
1 Un	Placa Arduino Uno	//
1 Un	Mini Bomba D'água 110v	20,00
2 Un	Shield Módulo Relé 5v - 2 Canais	32,76
2 Un	Shield Sensor de Temperatura DS18B20 - à prova d'água	40,39
1 Un	Aquecedor de água 110v (Rabo Quente)	19,90
1 Un	Chapa de Madeira	24,90
2 Un	Pote Redondo Médio de Acrílico	19,90
1 Un	Pote Quadrado Médio de Plástico	9,90
1 Un	Tubo de Silicone	10,00
Aprox. 1mt	Tubulação de Cobre	10,00

Total: R\$ 187,75

5 CONCLUSÕES

5.1 DIFICULDADES ENCONTRADAS

As maiores dificuldades encontradas estão no código do Arduino, na vedação dos recipientes e na montagem do PCB para deixar o projeto mais organizado. A primeira tentativa de vedar foi com cola quente, porém só uma das saídas vedou, a segunda envolvia silicone, essa se saiu melhor que a cola quente mas ainda vazava água em alguns pontos, então combinamos silicone e fita isolante para melhorar a vedação, um dos recipientes acabou ficando muito danificado após todas tentativas, e houve a necessidade de trocar ele, desta vez optamos por um recipiente de plástico, deste modo todas as saídas foram vedadas. A dificuldade da PCB, foi montar o layout da PCB, que precisou de adaptações desde sua primeira versão, e achar alguém que montaria a PCB a partir desse layout no tempo necessário. Após a impressão da PCB, houve a dificuldades no processo de soldagem dos componentes na mesma, e outro problema foi que a impressão acabou saindo espelhada, o que levou a um atraso no processo, pois algumas trilhas acabaram se invertendo, tornando assim o processo mais longo. Sobre o código do Arduino, tivemos que entender o funcionamento de sua linguagem, achar as bibliotecas necessárias e a escrita do código.

5.2 GRÁFICOS EM TEMPO REAL

5.2.1 MATLAB

Durante o processo, em que iniciamos a comunicação entre o Arduino e o Matlab, para fazer a transmissão das leituras captadas pelos sensores para o Matlab, que se encarregaria de gerar os gráficos em tempo real, das temperaturas de ambos os reservatórios. Houve a percepção que o Matlab não aceitava transmitidos pelos pinos digitais do Arduino, o Matlab aceita a transmissão de dados pelos pinos analógicos.

5.2.2 OCTAVE

Também foram encontrados problemas para fazer a transmissão das leituras captadas pelos sensores no Octave, os mesmos problemas vieram a surgir, durante as tentativas de comunicação entre o Arduino e o Octave.

5.2.2.1 Código desenvolvido no Matlab:

```

fx >> a = serial('COM3','BaudRate',9600);
value = readDigitalPin(a,4) % define the Arduino Communication port
plotTitle = 'Arduino Data Log'; % plot title
xlabel = 'Elapsed Time (s)'; % x-axis label
ylabel = 'Temperature (C)'; % y-axis label
legend1 = 'Temperature Sensor 1'
legend2 = 'Temperature Sensor 2'
legend3 = 'Temperature Sensor 3'
yMax = 40 %y Maximum Value
yMin = 0 %y minimum Value
plotGrid = 'on'; % 'off' to turn off grid
min = 0; % set y-min
max = 40; % set y-max
delay = .01; % make sure sample faster than resolution
%Define Function Variables
time = 0;
data = 0;
data1 = 0;
data2 = 0;
count = 0
%Set up Plot
plotGraph = plot(time,data,'-r' ) % every AnalogRead needs to be on its own Plotgraph
hold on %hold on makes sure all of the channels are plotted

```

Figura 16 – Código para gerar gráficos com o MATLAB
Fonte: Autoria própria

```

plotGraph1 = plot(time,data1,'-b')
plotGraph2 = plot(time, data2,'-g' )
title(plotTitle,'FontSize',15);
xlabel(xlabel,'FontSize',15);
ylabel(ylabel,'FontSize',15);
legend(legend1,legend2,legend3)
axis([yMin yMax min max]);
grid(plotGrid);
tic
while ishandle(plotGraph) %Loop when Plot is Active will run until plot is closed
    dat = value* 0.48875855327; %Data from the arduino
    dat1 = a.digitalRead(5)* 0.48875855327;
    %%dat2 = a.analogRead(4)* 0.48875855327;
    count = count + 1;
    time(count) = toc;
    data(count) = dat(1);
    data1(count) = dat1(1)
    data2(count) = dat2(1)
    %This is the magic code
    %Using plot will slow down the sampling time.. At times to over 20
    %seconds per sample!
    set(plotGraph,'XData',time,'YData',data);
    set(plotGraph1,'XData',time,'YData',data1);
    set(plotGraph2,'XData',time,'YData',data2);
    axis([0 time(count) min max]);
    %Update the graph
    pause(delay);
end
delete(a);
disp('Plot Closed and arduino object has been deleted');

```

Figura 17 – Código para gerar gráficos com o MATLAB
Fonte: Autoria própria

5.3 CONSIDERAÇÕES FINAIS

Mesmo diante das dificuldades encontradas durante o desenvolvimento, esse projeto possibilitou o aprendizado sobre o funcionamento do arduíno integrado com o shield relé, montagem do layout de placa de circuito impresso e a solda de seus componentes, organização de projeto, envolvendo desde o cronograma até montagem de relatório.

REFERÊNCIAS

1. WEBER, Guilherme H.;LONGO, Jean P. N.;MURAKAMI, Pedro H. W. - **Aplicação de sensores capacitivos para monitoramento da formação de parafina em oleodutos. 2013.**