REVISTA
DESAFIOS
ISSN: 2359-3652
v. 11 n. 3 (2024): Dossiê Temático: Programa Institucional de Bolsas de Iniciação Científica da Universidade Federal do Tocantins. *DOI:* http://dx.doi.org/10.20873.2024_v3_10

**ARTIGO RECEBIDO: 15/12/2023 – APROVADO: 06/03/2024 - PUBLICADO: 22/04/2024**

# DESENVOLVIMENTO DE UM SISTEMA PARA O MONITORAMENTO DO OXIGÊNIO DISSOLVIDO E DA PRESSÃO EM UM BIORREATOR DE BANCADA

*DEVELOPMENT OF A SYSTEM FOR MONITORING DISSOLVED OXYGEN AND PRESSURE IN A BENCHTOP BIOREACTOR*

*DESARROLLO DE UN SISTEMA PARA EL MONITOREO DE OXÍGENO DISUELTO Y PRESIÓN EN UN BIORREACTOR DE BANCO*

***Wagster Carlos Ataide Tavares,[1]; Leandra Cristina Crema Cruz[1]; Pedro Alexandre da Cruz*,[1]***

[1] Laboratório de Engenharia e Modelagem Matemática - LabEMM, Curso de Graduação, Universidade Federal do Tocantins, Gurupi, Brasil.

*Correspondência: pedrocruz@uft.edu.br

**RESUMO**

O Arduino é uma plataforma aberta e de baixo custo que permite a interação com dispositivos que apresentam diferentes funcionalidades, como sensores, módulos, entre outros. Para o monitoramento de um biorreator de bancada, sensores de oxigênio, pH, pressão, temperatura, entre outros são exemplos de dispositivos que podem ser utilizados em bioprocessos. O microcontrolador Arduino UNO R3 foi utilizado para a criação de um sistema para a obtenção de dados de oxigênio dissolvido e de pressão no biorreator. Foram desenvolvidos um circuito eletrônico e rotinas para a calibração dos sensores, assim como um código de programação que permitisse a obtenção e o armazenamento dos dados provenientes dos sensores. Um esboço foi elaborado pelo Autodesk AutoCAD para desenvolver um aparato experimental para a calibração do sensor transdutor de pressão, que juntamente com um manômetro padrão formarão um único sistema interligados recebendo as mesmas informações de pressão exercida na coluna de líquido. A metodologia empregada para calibração do sensor de oxigênio dissolvido demonstrou ser eficaz. O código de programação desenvolvido para o sensor de pressão será validado a partir de experimentos utilizando o aparato proposto no presente trabalho.

**Palavras-chave:** Arduino UNO R3. Automação. Biorreator. Oxigênio Dissolvido. Pressão.

***ABSTRACT***

Arduino is an open-source, low-cost platform that enables interaction with devices featuring various functionalities such as sensors, modules, among others. For monitoring a benchtop bioreactor, sensors for oxygen, pH, pressure, temperature, among others, are examples of devices that can be used in bioprocesses. The Arduino UNO R3 microcontroller was utilized to create a system for obtaining data on dissolved oxygen and pressure in the bioreactor. Electronic circuits and calibration routines for the sensors were developed, along with programming code to acquire and store data from the sensors. A draft was created using Autodesk AutoCAD to develop an experimental apparatus for calibrating the pressure transducer sensor, which along with a standard manometer, will form a single interconnected

system receiving the same pressure information exerted in the liquid column. The methodology employed for calibrating the dissolved oxygen sensor proved to be effective. The programming code developed for the pressure sensor will be validated through experiments using the proposed apparatus in this study.
***Keywords****:* Arduino UNO R3. Automation. Bioreactor. Dissolved Oxygen. Pressure.

***RESUMEN***
El Arduino es una plataforma abierta y de bajo costo que permite la interacción con dispositivos que tienen diferentes funcionalidades, como sensores, módulos, entre otros. Para el monitoreo de un biorreactor de banco, sensores de oxígeno, pH, presión, temperatura, entre otros, son ejemplos de dispositivos que pueden ser utilizados en bioprocesos. El microcontrolador Arduino UNO R3 se utilizó para crear un sistema para la obtención de datos de oxígeno disuelto y de presión en el biorreactor. Se desarrollaron un circuito electrónico y rutinas para la calibración de los sensores, así como un código de programación que permitiera la obtención y el almacenamiento de los datos provenientes de los sensores. Se elaboró un esbozo mediante Autodesk AutoCAD para desarrollar un aparato experimental para la calibración del sensor transductor de presión, que junto con un manómetro estándar formarán un único sistema interconectado recibiendo la misma información de presión ejercida en la columna de líquido. La metodología empleada para la calibración del sensor de oxígeno disuelto demostró ser eficaz. El código de programación desarrollado para el sensor de presión será validado a partir de experimentos utilizando el aparato propuesto en el presente trabajo.
***Descriptores****:* Arduino Uno R3. Automatización. Biorreactor. Oxígeno Disuelto. Pressión.

## INTRODUÇÃO

Biorreatores são dispositivos essenciais em processos biotecnológicos e bioquímicos, projetados para suportar o crescimento, a proliferação e a atividade de microrganismos, células ou sistemas biológicos. Geralmente, eles são utilizados para a produção de produtos biológicos, como enzimas, vacinas, anticorpos monoclonais, produtos químicos e bioplásticos. Os biorreatores são equipados com sistemas de controle de temperatura, agitação e aeração, bem como sensores para monitorar e ajustar os parâmetros físico-químicos, como pH, concentração de oxigênio dissolvido e concentração de nutrientes, a fim de criar as condições ideais para o crescimento e atividade do organismo hospedeiro (Schmidell, 2001).

O controle das principais variáveis em bioprocesso em biorreatores é essencial para o crescimento microbiano e formação de produto, rendimento industrial e obtenção de produtos homogêneos e de qualidade (Araújo, 2012). Entre essas variáveis, pode-se citar a pressão de operação do biorreator e o oxigênio dissolvido no meio fermentativo. A pressão está diretamente relacionada com o transporte de oxigênio e respiração microbiana e indiretamente, no metabolismo dos microrganismos (Schmidell, 2021). E o oxigênio dissolvido está relacionado com a oxigenação e a aeração do meio fermentativo (Araújo, 2012). Assim, para maior controle e monitoramento do processo fermentativo, faz-se necessário o uso de diferentes sensores para monitorar as principais variáveis e entre eles, um sensor de oxigênio dissolvido e um dispositivo que controle e regule a pressão no interior do biorreator é de suma importância.

Para a realização de bioprocessos em biorreatores de bancada, a utilização da placa Arduino é uma opção viável e de baixo custo que possibilita a instrumentação e o controle das variáveis pela interação entre os dispositivos conectados à placa e seu ambiente de trabalho. Atualmente, existem vários tipos e modelos de dispositivos compatíveis com os diversos modelos da família Arduino, como por exemplo, sensores e módulos externos (temperatura, pH, pressão, nível de oxigênio dissolvido, luz, visor de LCD – *Liquid Crystal Display*, *Data Logger Shield*, *Ethernet*, etc.), criando possibilidades ilimitadas em diversos projetos (Souza, et al., 2011).

O Arduino é uma plataforma de prototipagem eletrônica que tem sido amplamente adotada no âmbito da pesquisa científica e do desenvolvimento tecnológico. Este sistema funciona essencialmente como um microcontrolador, geralmente Atmel AVR (*Advanced Virtual RISC*), que incorpora um processador com memória RAM (*Random Access Memory*), EEPROM (*Electrically-Erasable Programmable Read-Only Memory*), memória *flash* e *clock*. Os microcontroladores possuem pinos de entrada/saída que permitem a conexão com outros componentes eletrônicos, e oferece uma estrutura flexível para a implementação de projetos eletrônicos personalizados. No contexto da engenharia e da pesquisa, o Arduino desempenha um papel crucial devido à sua acessibilidade, simplicidade de uso e uma ampla gama de componentes e sensores compatíveis (Monk, 2013).

A flexibilidade do Arduino é ampliada pela sua linguagem de programação baseada em C/C++, tornando possível controlar sensores, atuadores e interfaces, adaptando-se às necessidades de diversas aplicações científicas (Stevan Junior; Silva, 2015). Frequentemente, o Arduino é usado em conjunto com sensores e módulos para medição e coleta de dados em tempo real. Sensores de diversos tipos, como temperatura, umidade, pressão e movimento, podem ser facilmente integrados aos projetos. Essa capacidade de aquisição de dados em tempo real, juntamente com a comunicação serial e interfaces de rede, torna o Arduino uma ferramenta poderosa para aplicações científicas que envolvem controle e monitoramento em tempo real, tornando-o um ativo inestimável na pesquisa interdisciplinar e na experimentação científica (MONK, 2013).

O Arduino IDE (*Integrated Development Environment*) é um *software* de código aberto, proveniente do sistema operacional da plataforma Arduino, descrito em *Java*. Esta ferramenta possui uma biblioteca integrada (*Wiring*), com programação em linguagem C/C++ que permite a criação de operações de entrada e saída com facilidade. O Arduino IDE permite desenvolver rotinas de programação, denominadas de *sketchs*, o qual compila e carrega códigos para a placa Arduino (UNICAMP, 2016). Estes códigos ao serem executados/compilados podem controlar simultaneamente diversos tipos de sensores e outros dispositivos conectados ao Arduino, desde que sejam compatíveis (Monk, 2013).

Tendo-se essas informações como premissa, o presente trabalho teve como objetivo desenvolver um aparato experimental utilizando a plataforma Arduino para o monitoramento do oxigênio dissolvido e pressão em um biorreator de bancada.

## MATERIAIS E MÉTODOS

### Materiais

- Biorreator de bancada de vidro com 4 bocas e capacidade mínima (250 mL) e máxima (1000 mL) por batelada;
- Abraçadeira de aço inox Tipo B de 100 mm;
- Suporte para Biorreator;
- Banho termostatizado digital com capacidade de 2.500 mL e 500 W (DXY, Water Bath);
- Compressor de ar eletromagnético 20 (L) (min-1) (Aco-001, Sunsun);
- Válvula anti-retorno para compressor de ar;
- Mangueiras de silicone de grau alimentício;
- Rolhas de silicone;
- Rotâmetro de ar entre 0 e 5 (L) ($min^{-1}$);
- Filtro de ar sanitário para sistema de aeração;
- Conector de vidro para entrada e saída de gás, junção 24/29;
- Agitador magnético com capacidade de 5 L, 0-3000 rpm e 0,86 W (MS 500, Intllab, Brasil);
- Computador com alto desempenho;
- Placa Microcontroladora Arduino UNO R3;
- Placa conversora de sinal para sensor de oxigênio dissolvido (DFRobot®);
- Sensor de oxigênio dissolvido analógico (DFRobot®) compatível com a placa Arduino;
- Placa conversora de sinal para sensor de oxigênio dissolvido (DFRobot®);
- Sensor transdutor de pressão 1.2 mPa (G1/4, USP-G41, SeedStudio) compatível com a placa Arduino;
- Sensor de temperatura DS18B20 com o adaptador DS18B20-XH (IC, Maxin IC);
- Visor de LCD (*Liquid Crystal Display*) 20X4 com módulo I2C soldado;
- *Data Logger Shield* (SHIELD V1.0, Deek-Robot);
- Cabos jumpers;
- Cabo USB (*Universal Serial Bus*);
- Cabo BNC (*Bayonet Neill Concelman*);
- Protoboard 830 pontos;
- Solução de hidróxido de sódio (NaOH) a 0,5 (mol)( L-1);
- *Software* Autodesk AutoCAD 2022 Institucional

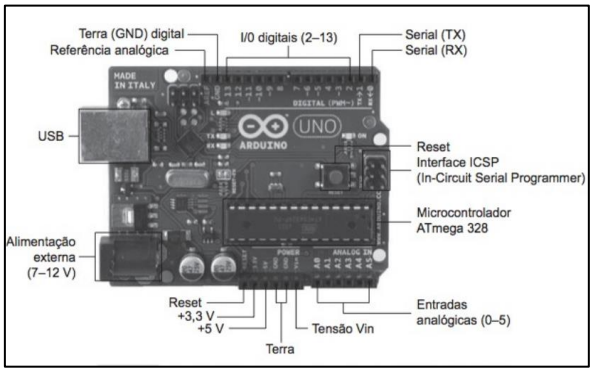
## METODOLOGIA

### Desenvolvimento do circuito eletrônico para aquisição de dados de oxigênio dissolvido e pressão

Para a utilização dos sensores foi necessário o desenvolvimento de um circuito eletrônico para permitir a comunicação entre todos os componentes, a placa microcontroladora e o computador. A placa microcontroladora Arduino UNO R3 foi utilizada no presente trabalho, a qual possui como base o Atmega328P, um microchip com 32 *Kilobytes* (kB) de memória e com funcionamento de 5 V (Volts).

Possui 14 pinos de entradas/saída digitais dos quais 6 podem ser utilizados como saídas PWM (*Pulse Width Modulation*), 6 saídas analógicas, um oscilador de cerâmica com 16 MHz (*Mega Hertz*), conexão USB e barramento ICSP (*In Circuit Serial Programming*) (Figura 1) (Arduino, 2020). A placa Arduino tem como principal função a conversão do sinal elétrico recebido pelos sensores, transmitindo-os ao computador na forma de dados com grandezas conhecidas (Monk, 2013).

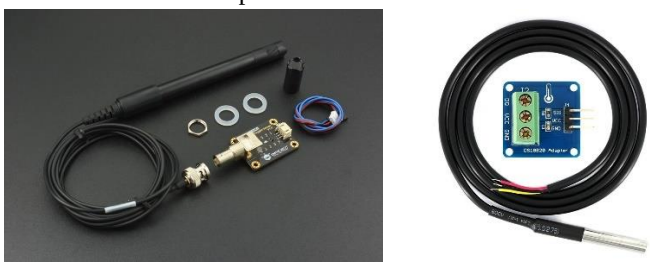
Figura 1 - Placa microcontroladora Arduino UNO R3




Fonte: Toledo, 2022.

Um sensor de oxigênio dissolvido analógico, composto por uma sonda galvânica e uma placa conversora de sinal (DFRobot®) foi utilizado para compor o circuito eletrônico. Este modelo de sensor é compatível com o Arduino, possui uma faixa de detecção entre 0 e 20 (mg) ($L^{-1}$) de oxigênio dissolvido no meio e faixa de temperatura entre 0 e 40 ºC. A placa conversora de sinal foi utilizada em conjunto com o sensor, a qual converte os dados recebidos em milivolts (mV), enviando-os para serem processados pelo Arduino. O sensor é conectado à placa de conversão por um cabo BNC (*Baioneta Neill-Concelman*) e sucessivamente conectada ao Arduino através de cabos *jumpers* nas entradas 5 V (alimentação), A1 (porta analógica 1) e GND (*Graduated Neutral Density Filter*, acesso ao terra da placa) (Figura 2ª). Um sensor de temperatura DS18B20 tipo sonda com o adaptador DS18B20-XH foi utilizado em conjunto com o sensor de oxigênio dissolvido, pois a dissolução do oxigênio no meio líquido é dependente da temperatura de operação (Figura 2b) (Marinho et al., 2023). O adaptador DS18B20-XH tem a função de facilitar a conexão entre o sensor de temperatura e a placa microcontroladora (Toledo, 2022).

Figura 2 - (a) Sensor de oxigênio dissolvido analógico e (b) sensor de temperatura DS18B20 tipo sonda com o adaptador DS18B20-XH



(a) (b)

Fonte: DFRobot, 2023; SeedStudio, 2023.

Um sensor transdutor de pressão compatível com a plataforma Arduino também foi utilizado para compor o circuito eletrônico, o qual permite uma leitura entre 0 e 1,2 Mpa (Mega Pascal) em uma faixa de temperatura de 0 e 85 ºC. Este sensor possui um conector *Delphi* de 3 pinos que foram conectados nas entradas A2, GND e 5 V da placa Arduino para transferência de dados e alimentação de energia. Seu dispositivo é produzido a partir de uma liga de aço carbono, podendo ser integrada em diversas estruturas de equipamentos industriais (Figura 3ª) (Usainfo, 2022).

Figura 3 - (a) Sensor transdutor de pressão, (b) *Data Logger Shield* e (c) visor de LCD (*Liquid Crystal Display*)


(a) (b) (c)

Fonte: SeedStudio, 2023; Eletrogate, 2023; Gross, 2022

Os sensores de pressão desempenham a função de converter a pressão exercida pelo ambiente, seja atmosférica, de um gás ou de um líquido, em um sinal elétrico mensurável. Esses sensores englobam três categorias distintas de pressão passíveis de medição, sendo (i) pressão manométrica, (iii) pressão absoluta e (iii) pressão diferencial. A pressão manométrica se refere à pressão medida em relação à pressão atmosférica circundante, podendo ser positiva em situações de pressões superiores à atmosférica ou negativa em pressões inferiores. Os sensores de pressão absoluta oferecem resultados em relação a um ponto de referência de zero, que se equipara a um vácuo perfeito. E os sensores de pressão diferencial são capazes de medir a diferença de pressão entre duas amostras, funcionando de maneira análoga a um manômetro (MARTINS, 2012).

Um *Data Logger Shield* (V1.0, *Deek-Robot*) foi utilizado para o registro de dados em tempo real (data e hora) durante a experimentação. O *Data Logger Shield* possui um módulo RTC (*Real Time Clock*) e um cartão SD (*Secure Digital*) que possibilita o armazenamento dos dados automaticamente.

Essa *Shield* é encaixada diretamente na parte de cima do Arduino, mantendo a possibilidade de uso das portas de entrada/saída da placa microcontroladora (Figura 3b) (Toledo, 2022).

O Arduino também foi conectado a um visor de LCD com módulo I2C integrado, programado para exibir em tempo real as leituras provenientes dos sensores de oxigênio dissolvido e pressão (Figura 3c). Foi utilizado uma *protoboard* e cabos *jumpers* para facilitar a integração entre todos os componentes utilizados. Um cabo USB conectando a placa Arduino e um computador proporcionou o carregamento dos códigos de programação, transferência de dados e fonte de energia.

**Calibração do sensor de oxigênio dissolvido**

O sensor de oxigênio dissolvido deverá ser calibrado toda vez que for utilizado. Assim, a rotina de calibração seguiu os mesmos procedimentos descritos na literatura, que consiste no emprego de um ponto único de referência, utilizando água de qualidade 100% saturada com ar. Inicialmente foi preparada uma solução de hidróxido de sódio (NaOH) a 0,5 (mol) ($L^{-1}$) e com o auxílio de um conta gotas, a membrana do sensor, localizada na ponta do eletrodo (Figura 2ª) foi completamente umedecida com esta solução. Para a execução da calibração foi necessário fazer a transferência do código de programação base, disponibilizado pelo fabricante, ao computador. Ao final da transferência do código base, a tela inicial do sensor é aberta na tela do computador para posteriormente armazenar as leituras da concentração de oxigênio dissolvido (Marinho, et al., 2023).

Na sequência, a água destilada supersaturada de ar em diferentes temperaturas foi utilizada. Um banho Maria termostatizado (DXY, Water Bath) foi utilizado para o controle da temperatura, além de uma bomba de ar (Aco-001, Sunsun) e um agitador magnético (MS500, Intllab TM®) para a saturação de ar e homogeneização da água, respectivamente. A temperatura do líquido foi monitorada através de um sensor de temperatura DS18B20 (IC, Maxin IC). A membrana do sensor foi imersa em água sob agitação branda e constante para sua umidificação antes da aquisição de dados. Posteriormente, o sensor foi imerso em um recipiente com água saturada de ar e três distintas temperaturas (24, 28 e 32 ºC ±1 ºC) foram avaliadas. Após a estabilização das leituras em função da temperatura avaliada, os dados foram registrados, inseridos no código de programação, seguido de compilação e posteriormente comparados com os valores de oxigênio dissolvido na água em função de cada temperatura avaliada (Anexo 1).

**Desenvolvimento do código de programação para calibração do sensor transdutor de pressão**

Para obter resultados livres de erros é necessário calibrar o sensor transdutor de pressão. Uma maneira é a eliminação do *offset* que é o valor de tensão de saída do sensor quando não está sendo aplicado nenhuma pressão. Este valor pode alterar significativamente o resultado da leitura. Outro

parâmetro que deve ser corrigido é a sensibilidade ou escala do sensor expressa em quantos volts de saída por unidade de pressão [(V) ($kPa^{-1}$)]. Durante a rotina de calibração, estes parâmetros podem ser facilmente identificados e corrigidos diretamente no *sketch* para obtenção correta da leitura de pressão (Balavalad; Sheeparamatti, 2015).

A lógica do código é projetada para realizar uma calibração em pontos de referências pré-determinados, onde as leituras analógicas são mapeadas para valores conhecidos de pressão em Mega Pascal (Mpa). Esta calibração permite posteriormente a conversão das leituras de tensão (V) do sensor em unidades conhecidas de pressão. O código deve iniciar criando dois vetores principais para a calibração, um responsável para o armazenamento dos dados das leituras analógicas fornecidas pelo sensor de pressão, e outro vetor, armazena os valores de pressão de referência que serão utilizados durante a calibração (Martins; Farinha; Cardoso, 2020).

Neste contexto, um código de programação para calibração do sensor transdutor de pressão foi desenvolvido no presente trabalho. Nesta etapa foi necessário inserir valores de pressões pré-definidas no *sketch*, executando-o para posteriormente visualizar as constantes de calibração no monitor serial do Arduino IDE. Estas constantes deverão ser anotadas e posteriormente inclusas no *sketch* do sensor de pressão para sua validação (Martins; Farinha; Cardoso, 2020).

Após a calibração do sensor transdutor de pressão é necessário validar o sistema desenvolvido para aquisição de dados entre o sensor de pressão e a placa Arduino UNO R3 e, principalmente, adaptar o código de programação fornecido pelo fabricante do sensor ao sistema para aquisição de dados. Para tal validação é necessário comparar as medidas do sensor de pressão por um outro elemento mecânico de medição direta de pressão, como por exemplo, um manômetro padrão (instrumento de alta precisão) (Bega, 2006).

Neste contexto, um segundo sistema foi elaborado para proporcionar a coleta de dados de pressão com precisão no biorreator. Um aparato para a calibração foi construído de forma que ambos os instrumentos (sensor transdutor de pressão e o manômetro padrão) estejam em um único sistema interligados e recebendo as mesmas informações da variável a ser medida, ou seja, dados de pressão iguais (Alves, 2012; Bega, 2006). Assim, os resultados obtidos entre os dois dispositivos poderão posteriormente serem comparados. Uma pequena coluna de líquido estática foi elaborada utilizando o *software* Autodesk AutoCAD 2022 Institucional para realizar o esboço do projeto e posteriormente sua construção.

## Desenvolvimento do código de programação (*sketch*) para aquisição de dados de oxigênio dissolvido e pressão em um biorreator de bancada

O Arduino IDE foi o ambiente de programação utilizado neste trabalho e juntamente com a placa microcontroladora Arduino UNO R3 foi desenvolvido um código de programação para aquisição de dados de oxigênio dissolvido e pressão em um biorreator de bancada (Figura 4). Para seu desenvolvimento foi utilizado como base os mesmos códigos disponibilizados pelos fabricantes de ambos os sensores (oxigênio dissolvido e pressão) (Dfrobot, 2023; Seedstudio, 2023), porém, durante a execução do projeto foram necessárias alterações adicionais para melhor funcionamento do sensor e coleta de dados experimentais.

Figura 4 - (a) Biorreator de bancada montado (b) componentes do biorreator

(a) (b)

Fonte: Toledo (2022).

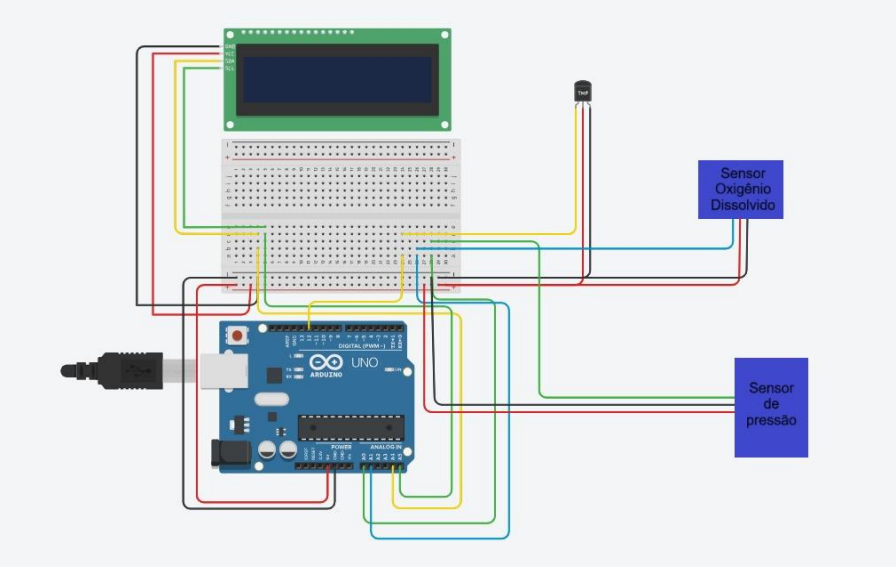
O Arduino IDE compila toda a programação redigida em linguagem C/C++, convertendo-a para a linguagem *Assembly* (sintaxe distinta da linguagem humana) e posteriormente, em um código binário, gravando-o na placa Arduino (Stevan Junior; Silva, 2015; Cavalcante, et al., 2014). Os projetos que envolvem o Arduino são compostos por declarações de variáveis, funções de configuração (*setup*) e funções de repetição (*loop*). Declarações de variáveis permitem a alocação e manipulação de dados, enquanto as funções de configuração estabelecem as configurações iniciais do sistema. A função de repetição é executada continuamente, permitindo a execução de tarefas específicas (Stevan Junior; Silva, 2015).

## RESULTADOS E DISCUSSÃO

### Desenvolvimento do circuito eletrônico para aquisição de dados de oxigênio dissolvido e pressão

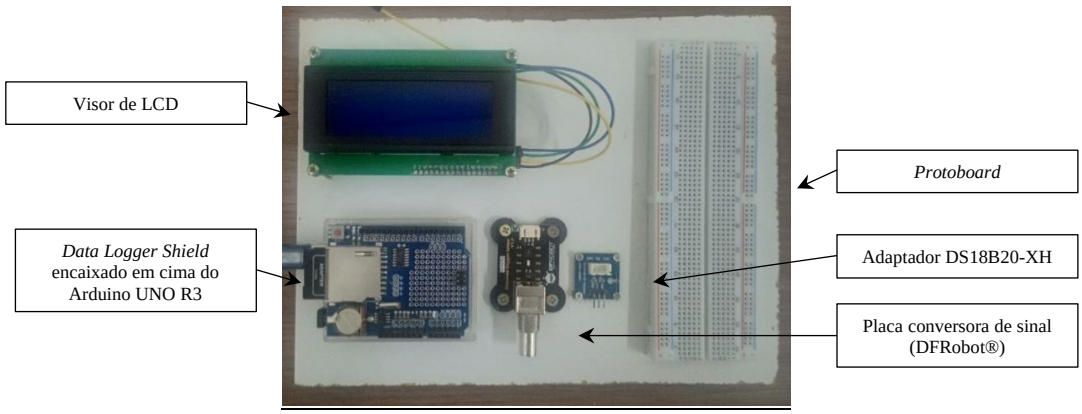
As Figuras 5 e 6 apresentam o resultado final da montagem do circuito eletrônico para aquisição de dados de oxigênio dissolvido e pressão. Para a transferência de dados obtidos através dos sensores conectado à placa Arduino UNO R3 e posteriormente ao computador, foram utilizadas as entradas analógicas A0, A4, A5, a entrada digital 10 e as entradas de energia de 5 V e a GND. A *protoboard* e cabos *jumpers* facilitaram a integração dos componentes utilizados.

Figura 5 - Esquematização do circuito eletrônico entre os sensores de oxigênio dissolvido e pressão, placa Arduino UNO R3, *Data Logger Shield*, visor de LCD e *protoboard*




Fonte: Autor, 2023.

Figura 6 - Imagem do circuito eletrônico entre os sensores de oxigênio dissolvido e pressão, placa Arduino UNO R3, *Data Logger Shield*, visor de LCD e *protoboard*




Fonte: Autor, 2023.

## Calibração do sensor de oxigênio dissolvido

Foi realizado a calibração do sensor de oxigênio utilizando o *sketch* fornecido pelo desenvolvedor do próprio sensor, porém com a necessidade de algumas adaptações. A Tabela 1 apresenta os dados recebidos e convertidos pelo sensor em milivolts (mV) em água destilada 100% saturada com ar, sob agitação e temperatura constantes. Os valores obtidos nesta etapa foram comparados com os dados calculados pelo fabricante do sensor de oxigênio dissolvido (Anexo 1).

Tabela 1 - Valores obtidos durante a calibração do sensor de oxigênio dissolvido

| T (ºC) | Voltagem (mV) | OD [(g) ($L^{-1}$)] | OD Calculado* ([(g) ($L^{-1}$)] |
|---|---|---|---|
| 24±1 | 1660 | 8,40±0,01 | 8,41 |
| 28±1 | 1750 | 7,83±0,02 | 7,83 |
| 32±1 | 2034 | 7,28±0,02 | 7,30 |

*Dados calculados e disponibilizados no Anexo 1.

Fonte: Autor, 2023.

Os resultados obtidos na fase experimental de calibração se mostraram dentro dos valores disponibilizados pelo fabricante do sensor (Anexo 1), havendo pequenas variações, possivelmente causadas pela turbulência da água sob agitação. Outro fator que deverá ser considerado é possíveis bolhas presas à membrana durante a medição, ocasionando erros de leituras.

**Desenvolvimento do código de programação para calibração do sensor transdutor de pressão**

No Quadro 1 apresenta o código de programação desenvolvido para a calibração do sensor transdutor de pressão. Os valores de *offset* e o fator de conversão são fundamentais para a calibração deste sensor e são determinados pelas leituras analógicas do sensor quando não há pressão aplicada ao sensor e pelas relações entre as leituras analógicas e os valores de conhecidos de pressão, respectivamente. O *offset* foi obtido durante a fase de inicialização do código, onde as leituras analógicas do sensor são definidas como zero (Linhas 12 a 14). Este valor representa a tensão de saída do sensor quando não há nenhuma pressão sendo aplicada. O fator de conversão é determinado durante o *loop* de calibração (Linhas 20 a 40). Ele reflete a relação entre as leituras analógicas (em Volts) e os valores conhecidos de pressão (em MPa). O fator de conversão é calculado dividindo a leitura analógica pelo valor de pressão correspondente (Martins; Farinha; Cardoso, 2020).

Quadro 1 - Código para calibração do sensor transdutor de pressão

```
1.  const int pressureSensorPin = A0;
2.  const float VCC = 5.0;
3.  const int numCalibrationPoints = 5;
4.  float pressureReadings[numCalibrationPoints];
5.  float pressureValues[numCalibrationPoints];
6.  void setup()
7.  {
8.  Serial.begin(9600);
9.  Serial.println("Calibracao sensor");
10. for (int i = 0; i < numCalibrationPoints; i++)
11. {
12. pressureReadings[i] = 0.0;
13. }
14. pressureValues[0] = 0.0;
15. pressureValues[1] = 0.2;
16. pressureValues[2] = 0.4;
17. pressureValues[3] = 0.6;
18. pressureValues[4] = 0.8;
19. }
20. void loop()
21. {
22. for (int i = 0; i < numCalibrationPoints; i++)
23. {
24. Serial.print("Aplicar pressao: ");
25. Serial.print(pressureValues[i]);
26. Serial.println(" MPa e pressione Enter");
27. while (!Serial.available())
28. {
29. }
30. while (Serial.available())
31. {
32. Serial.read();
33. }
34. float sensorValue = analogRead(pressureSensorPin);
35. float Vout = sensorValue * VCC / 1023.0;
36. pressureReadings[i] = Vout;
37. Serial.print("Lendo pressao: ");
38. Serial.print(pressureReadings[i], 4);
39. Serial.println(" V");
40. }
41. Serial.println("Calibracao completa");
42. Serial.println("Leituras realizadas:");
43. for (int i = 0; i < numCalibrationPoints; i++)
44. {
45. Serial.print("Pressao: ");
46. Serial.print(pressureValues[i]);
47. Serial.print(" MPa - Voltage: ");
48. Serial.print(pressureReadings[i], 4);
49. Serial.println(" V");
50. }
51. while (true) {
52. }
```

Fonte: Autor, 2023.

Para a leitura da pressão, o código precisa de algumas informações declaradas previamente, como o pino de entrada que o sensor estará ligado, a definição das variáveis para o armazenamento das leituras do sensor e as constantes relacionadas à alimentação do sensor, a pressão mínima e máxima que o sensor é capaz de ler, o fator de conversão de unidade, o *offset* previamente definido (Linhas 12 a 14 do Quadro 1). Todos esses dados são obtidos durante a calibração ou pelo fornecedor do sensor. Definido as variáveis, parte-se para a etapa da execução da leitura (Albuquerque; Thomanzini, 2005).

A voltagem de saída (Volt) se dá pela leitura obtida na porta analógica em que o sensor está conectado, multiplicando pela voltagem de entrada (VCC) dividido por 1023 que é o valor máximo possível da leitura analógica (Linha 52 do Quadro 2). Este cálculo permite a obtenção da voltagem real medida pelo sensor de pressão, que será utilizada posteriormente para calcular a pressão baseada em outros parâmetros definidos (Mcroberts, 2018).

Quadro 2 - Código para leitura da pressão utilizando o sensor transdutor de pressão

```
1.  #include <Wire.h>
2.  #include <Arduino.h>
3.  #include <LiquidCrystal_I2C.h>
4.  #include <RTClib.h>
5.  #include <SPI.h>
6.  #include <SD.h>
7.  LiquidCrystal_I2C lcd(0x3F, 20, 4);
8.  const int pressureSensorPin = A0;
9.  float Vout, pressure, pressure_psi, pressure_bar, pressure_mmH2O;
10. const float VCC = 5.0;
11. const float pressureMin = 0.0;
12. const float pressureMax = 1.2;
13. const float pressureFactor = 0.75;  fornecedor/calibração
14. const float pressureOffset = 0.1;
15. const float PSI_TO_PA = 6894.76;
16. const float BAR_TO_PA = 100000.0
17. const float MMH2O_TO_PA = 9.80665
18. const int chipSelect = 4;
19. RTC_DS1307 rtc;
20. File file;
21. void setup()
22. {
23. Serial.begin(9600);
24. lcd.init();
25. lcd.backlight();
26. lcd.setCursor(0, 0);
27. lcd.print("Pressure Sensor");
28. delay(2000);
29. lcd.clear();
30. Serial.println("Iniciando cartao SD...");
31. pinMode(SS, OUTPUT);
32. if (!SD.begin(chipSelect))
33. {
34. Serial.println("Falha na inicializacao do SD!");
35. while (1) ;
36. }
37. Serial.println("card initialized.");
38. if (! rtc.begin())
39. {
40. Serial.println("RTC nao encontrado!");
41. while (1);
42. }
43. if (! rtc.isrunning())
44. {
45. Serial.println("RTC nao operante!");
46. rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
47. }
48. }
49. void loop()
50. {
51. int sensorValue = analogRead(pressureSensorPin);
52. Vout = sensorValue * VCC / 1023.0;
53. pressure = (Vout - VCC * pressureOffset) / (VCC * pressureFactor);
54. if (pressure < pressureMin)
55. {
56. pressure = pressureMin;
57. } else if (pressure > pressureMax)
58. {
59. pressure = pressureMax;
60. }
61. pressure_psi = pressure * PSI_TO_PA / 1e6;
62. pressure_bar = pressure * BAR_TO_PA / 1e6;
63. pressure_mmH2O = pressure * MMH2O_TO_PA / 1e6;
64. lcd.clear();
65. lcd.setCursor(0, 0);
66. lcd.print("Pressure: ");
67. lcd.print(pressure_psi, 2);
68. lcd.print(" PSI");
69. lcd.setCursor(0, 1);
70. lcd.print("Bar: ");
71. lcd.print(pressure_bar, 2);
72. lcd.setCursor(0, 2);
73. lcd.print("mmH2O: ");
74. lcd.print(pressure_mmH2O, 2);
75. lcd.setCursor(0, 3);
76. lcd.print("Voltage: ");
77. lcd.print(Vout, 2);
78. lcd.print(" V");
79. DateTime now = rtc.now();
80. file = SD.open("pressure.txt", FILE_WRITE);
81. file.print("Data/hora: ");
82. file.print(now.day() < 10 ? "0" : "");
83. file.print(now.day(), DEC);
84. file.print('/');
85. file.print(now.month() < 10 ? "0" : "");
86. file.print(now.month(), DEC);
87. file.print('/');
88. file.print(now.year(), DEC);
89. file.print(' ');
90. file.print(now.hour() < 10 ? "0" : "");
91. file.print(now.hour(), DEC);
92. file.print(':');
93. file.print(now.minute() < 10 ? "0" : "");
94. file.print(now.minute(), DEC);
95. file.print(':');
96. file.print(now.second() < 10 ? "0" : "");
97. file.print(now.second(), DEC);
98. file.print(" Pressure_PSI:\t");
99. file.println(pressure_psi, 2);
100. file.print(" Pressure_Bar:\t");
101. file.println(pressure_bar, 2);
102. file.print(" Pressure_mmH2O:\t");
103. file.println(pressure_mmH2O, 2);
104. file.print(" Voltage:\t");
105. file.println(Vout, 2);
106. file.close();
107. delay(1000);
108. }
```

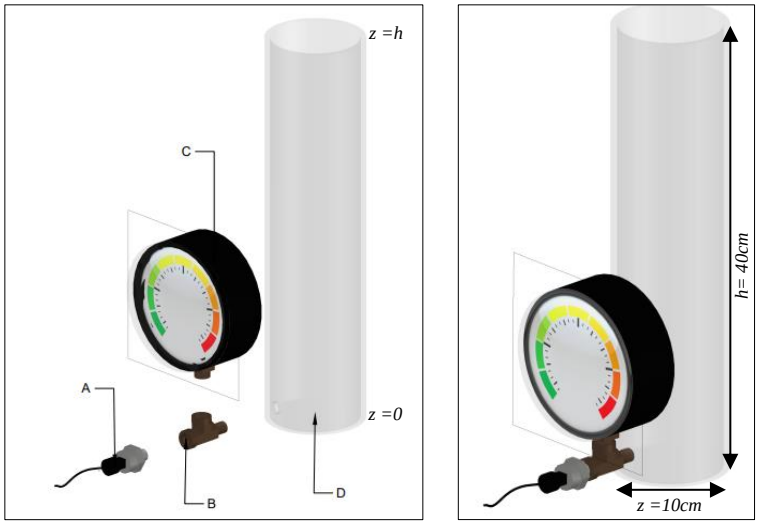
Fonte: Autor, 2023.

Após a obtenção desta voltagem, é realizado o cálculo que converte este valor em pressão. O cálculo se dá pela subtração da compensação (VCC*offset) da voltagem lida (Volt) (Linha 53 do Quadro 2). Isto remove qualquer desvio ou *offset* indesejado da leitura, para garantir uma maior acuidade. Em seguida, o resultado desta subtração é então dividido pelo fator de conversão (VCC*fator de conversão), que irá dimensionar a voltagem corrigida para uma unidade de pressão específica, levando em consideração a calibração (Linha 53 do Quadro 2) (Martins; Farinha; Cardoso, 2020).

O Quadro 2 apresenta o código de programação para obtenção de dados de pressão seguindo as informações obtidas após a calibração do sensor (Quadro 1).

Após a calibração do sensor transdutor de pressão é necessário validar o sistema desenvolvido para aquisição de dados entre o sensor de pressão. A Figura 7 apresenta o esboço do aparato para validar o sensor transdutor de pressão.

Figura 7 - Aparato para validação do circuito eletrônico e código de programação utilizando o transdutor de pressão. **(a)** elementos para conecção na coluna e **(b)** coluna montada, em detalhes, sua altura (*h*) e diâmetro (*z*), com o manômetro acoplado.




Nota: (A) Sensor transdutor de pressão; (B) Conector hidráulico em T; (C) Manômetro de precisão e (D) Tubo de acrílico.

Fonte: Autor, 2023.

O esboço da Figura 7 mostra com detalhes uma pequena coluna de líquido estática com uma das extremidades fechada ($z = 0$) e a outra aberta ($z = h$) sob pressão atmosférica com altura/profundidade conhecida utilizando um líquido com densidade aparente também conhecida. Os sensores transdutor de pressão e o manômetro padrão formam um único sistema interligados e recebendo as mesmas informações de pressão exercida na coluna de líquido. A coleta de dados e posteriormente a comparação dos dados obtidos, validarão o circuito eletrônico e o código de programação desenvolvido para aquisição de dados de pressão utilizando o sensor transdutor de pressão no biorreator de bancada.

O presente trabalho apresenta um aparato experimental para tal validação utilizando um manômetro padrão, o qual deverá ser projetado para medir a pressão exercida na coluna de líquido cuja extremidade está aberta para a atmosfera. Assim, é necessário utilizar um instrumento adequado para fornecer dados de pressão preciso no fundo da coluna de acrílico, comparando seu resultado com o resultado obtido pelo sensor transdutor de pressão (Figura 7).

**Desenvolvimento do código de programação (*sketch*) para aquisição de dados de oxigênio dissolvido e pressão em um biorreator de bancada**

O Quadro 3 apresenta o código de programação para aquisição de dados de oxigênio dissolvido e de pressão do biorreator de bancada. Para realizar o armazenamento dos dados no cartão SD do *Data Logger Shield*, foi necessário a instalação de duas bibliotecas no Arduino IDE e inclusão no código. As bibliotecas foram a RTCLib, que permite a utilização do módulo RTC (*Real-Time Clock* presente no *Data Logger Shield* e a SD, que permite o armazenamento de dados no cartão de memória (Linha 6). Após a instalação destas bibliotecas, foi necessário fazer uma pequena alteração no código base para que os dados armazenados no cartão SD fossem salvos no formato adequado (Toledo, 2022.).

Quadro 3 - Código para aquisição de dados de oxigênio dissolvido e pressão em um biorreator de bancada

```
1.  #include <Wire.h>
2.  #include <LiquidCrystal_I2C.h>
3.  #include <OneWire.h>
4.  #include <DallasTemperature.h>
5.  #include <SD.h>
6.  #include <RTClib.h>
7.  #include <SPI.h>
8.  RTC_DS1307 rtc;
9.  LiquidCrystal_I2C lcd(0x3F, 20, 4);
10. #define DO_PIN A1
11. #define VREF 5000
12. #define ADC_RES 1024
13. #define TWO_POINT_CALIBRATION 0
14. #define READ_TEMP (24)
15. #define CAL1_V (1860)
16. #define CAL1_T (28)
17. #define CAL2_V (1640)
18. #define CAL2_T (25)
19. const uint16_t DO_Table[41] = {
20. 14460, 14220, 13820, 13440, 13090, 12740, 12420, 12110,
    11810, 11530,
21. 11260, 11010, 10770, 10530, 10300, 10080, 9860, 9660,
    9460, 9270,
22. 9080, 8900, 8730, 8570, 8410, 8250, 8110, 7960, 7820, 7690,
23. 7560, 7430, 7300, 7180, 7070, 6950, 6840, 6730, 6630, 6530,
    6410};
24. uint8_t Temperaturet;
25. uint16_t ADC_Raw;
26. uint16_t ADC_Voltage;
27. uint16_t DO;
28. const int PINO_ONEWIRE = 12;
29. OneWire oneWire(PINO_ONEWIRE);
30. DallasTemperature sensor(&oneWire);
31. DeviceAddress insideThermometer = {0x28, 0xAA, 0x2E,
    0x32, 0x48, 0x14, 0x01, 0x05};
32. DeviceAddress outsideThermometer = {0x28, 0xAA, 0x2E,
    0x32, 0x48, 0x14, 0x01, 0x05};
33. File dataFile;
34. const int pressureSensorPin = A0;
35. float Vout, pressure, pressure_psi, pressure_bar, pres-
    sure_mmH2O;
36. const float VCC = 5.0;
37. const float pressureMin = 0.0;
38. const float pressureMax = 1.2;
39. const float pressureFactor = 0.75;
40. const float pressureOffset = 0.1;
41. const float PSI_TO_PA = 6894.76;
42. const float BAR_TO_PA = 100000.0;
43. const float MMH2O_TO_PA = 9.80665;
44. const int Pino_CS = 10;
45. void setup()
46. {
47. Serial.begin(9600);
48. lcd.init();
49. lcd.backlight();
50. rtc.begin();
51. sensor.begin();
52. if (!SD.begin(Pino_CS))
53. {
54. lcd.clear();
55. lcd.print("SD Card Error");
56. while (true);
57. }
58. }
59. void displaytemperatura(int indicesensor)
60. {
```

Quadro 3 - Código para aquisição de dados de oxigênio dissolvido e pressão em um biorreator de bancada (Continuação)

```
sensor.requestTemperatures();
lcd.clear();
float temp_C2 = sensor.getTempC(outsideThermometer);
lcd.print("Temp.: ");
lcd.print(temp_C2, 1);
lcd.write(B11011111);
lcd.print("C");
}
int16_t readDO(uint32_t voltage_mv, uint8_t temperature_c)
{
#if TWO_POINT_CALIBRATION == 0
uint16_t V_saturation = (uint32_t)CAL1_V + (uint32_t)35 * temperature_c - (uint32_t)CAL1_T * 35;
return (voltage_mv * DO_Table[temperature_c] / V_saturation);
#else
uint16_t V_saturation = (int16_t)((int8_t)temperature_c - CAL2_T) * ((uint16_t)CAL1_V - CAL2_V) / ((uint8_t)CAL1_T - CAL2_T) + CAL2_V;
return (voltage_mv * DO_Table[temperature_c] / V_saturation);
#endif
}
void logData(float temp, float doValue, float voltage, DateTime currentTime)
{
dataFile = SD.open("datalog.txt", FILE_WRITE);
if (dataFile)
{
dataFile.print(currentTime.year(), DEC);
dataFile.print('/');
dataFile.print(currentTime.month(), DEC);
dataFile.print('/');
dataFile.print(currentTime.day(), DEC);
dataFile.print(' ');
dataFile.print(currentTime.hour(), DEC);
dataFile.print(':');
dataFile.print(currentTime.minute(), DEC);
dataFile.print(':');
dataFile.print(currentTime.second(), DEC);
dataFile.print(',');
dataFile.print(temp, 1);
dataFile.print(',');
dataFile.print(doValue, 2);
dataFile.print(',');
dataFile.print(voltage, 2);
dataFile.println();
dataFile.close();
}
}
void loop()
{
Temperaturet = (uint8_t)READ_TEMP;
ADC_Raw = analogRead(DO_PIN);
ADC_Voltage = uint32_t(VREF) * ADC_Raw / ADC_RES;
float temp_C2 = sensor.getTempC(outsideThermometer);
lcd.setCursor(0, 1);
sensor.requestTemperatures();
displaytemperatura(0);
lcd.setCursor(0, 1);
lcd.print("ADC Voltage: " + String(ADC_Voltage) + " mV ");
lcd.setCursor(0, 2);
lcd.println("DO: " + String(readDO(ADC_Voltage, Temperaturet)) + " ug/l ");
Serial.println("-----------------------------------------------");
Serial.println("Oxigênio Dissolvido em Função da Temperatura");
Serial.println("-----------------------------------------------");
Serial.println("Temp. Ref.:\t" + String(Temperaturet) + "\t");
Serial.print("Temp.:\t       ");
Serial.println(temp_C2, 1);
Serial.println("ADC RAW:\t" + String(ADC_Raw) + "\t");
Serial.println("ADC Voltage:\t" + String(ADC_Voltage) + "\t");
Serial.println("DO:\t\t" + String(readDO(ADC_Voltage, Temperaturet)) + "\t");
Serial.println("-----------------------------------------------");
int sensorValue = analogRead(pressureSensorPin);
Vout = sensorValue * VCC / 1023.0;
pressure = (Vout - VCC * pressureOffset) / (VCC * pressureFactor);
if (pressure < pressureMin)
{
pressure = pressureMin;
}
else if (pressure > pressureMax)
{
pressure = pressureMax;
}
pressure_psi = pressure * PSI_TO_PA / 1e6;
pressure_bar = pressure * BAR_TO_PA / 1e6;
pressure_mmH2O = pressure * MMH2O_TO_PA / 1e6;

lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Pressure: ");
lcd.print(pressure_psi, 2);
lcd.print(" PSI");
lcd.setCursor(0, 1);
lcd.print("Bar: ");
lcd.print(pressure_bar, 2);
lcd.setCursor(0, 2);
lcd.print("mmH2O: ");
lcd.print(pressure_mmH2O, 2);
lcd.setCursor(0, 3);
lcd.print("Voltage: ");
lcd.print(Vout, 2);
lcd.print(" V");
DateTime now = rtc.now();
file = SD.open("pressure_data.txt", FILE_WRITE);
if (file)
{
file.print("Data/hora: ");
file.print(now.day() < 10 ? "0" : "");
file.print(now.day(), DEC);
file.print('/');
file.print(now.month() < 10 ? "0" : "");
file.print(now.month(), DEC);
file.print('/');
file.print(now.year(), DEC);
file.print(' ');
file.print(now.hour() < 10 ? "0" : "");
file.print(now.hour(), DEC);
file.print(':');
file.print(now.minute() < 10 ? "0" : "");
file.print(now.minute(), DEC);
file.print(':');
file.print(now.second() < 10 ? "0" : "");
file.print(now.second(), DEC);
file.print(" Pressure_PSI: \t");
file.print(pressure_psi, 2);
file.print(" Pressure_Bar: \t");
file.print(pressure_bar, 2);
file.print(" Pressure_mmH2O: \t");
file.print(pressure_mmH2O, 2);
file.print(" Voltage: \t ");
file.println(Vout, 2);
dataFile.print('temperatura: \t,');
dataFile.print(temp, 1);
dataFile.print('Oxigenio Dissolvido: \t,');
dataFile.print(doValue, 2);
dataFile.print('Voltagem-DO: \t,');
dataFile.print(voltage, 2);
file.close();
}
delay(3000);
}
```

Fonte: Autor, 2023

O código base ao ser executado, salva os dados no cartão SD no formato de texto corrido (“.txt”), o qual dificulta a transferência do arquivo para uma planilha e posteriormente tratamento e análise dos dados. Assim, para adequar o formato do arquivo foi utilizado do comando “\t” que é lido pelo código de programação como o comando “tab”, proporcionando dados adequados para serem salvos em uma planilha (Linhas 183 a 195).

Um visor de LCD com módulo I2C integrado foi utilizado para a visualização dos dados em tempo real. Para sua utilização, foi necessário instalar a biblioteca *Liquid Crystal I2C*, onde também foi implementada no código de programação inicial desenvolvido neste trabalho. Para seu funcionamento foi necessário a adição de alguns comandos, como a definição do endereço da placa Arduino UNO R3 e sua respectiva quantidade de linhas e colunas, inicializar o visor de LCD e ligar sua luz de fundo (Linhas 9, 49 e 50).

Ao final do desenvolvimento, o código de programação foi compilado com sucesso e sem erros. Os dados das leituras dos sensores (oxigênio dissolvido e pressão) foram visualizados no visor de LCD em tempo real e armazenados com sucesso no cartão SD contido no *Data Logger Shield*. Para acessar os dados obtidos e verificar seu formato adequado para uma planilha, o cartão de memória SD foi removido e inserido no computador.

## CONCLUSÃO

O desenvolvimento dos códigos de programação para o monitoramento do oxigênio dissolvido e pressão, assim como as rotinas de calibrações necessárias para o funcionamento de ambos os sensores demonstrou ser adequado e eficiente, cumprindo o objetivo proposto. O circuito eletrônico montado para a observação dos parâmetros das variáveis no visor de LCD e armazenamento no cartão SD contido no *Data Logger Shield* demonstrou funcional, assim como as adaptações realizadas nos códigos de programação disponibilizados pelos fabricantes dos sensores. A construção do aparato experimental é essencial para validação do circuito eletrônico e do código de programação utilizando sensor o transdutor de pressão. Esta validação proporcionará a obtenção de dados mais precisos e confiáveis de pressão no biorreator de bancada. O código para aquisição de dados de oxigênio dissolvido e pressão no biorreator de bancada não demostrou erros ao ser compilado, porém deverá ser avaliado novamente após a validação do sensor o transdutor de pressão.

## AGRADECIMENTO

Agradeço ao grupo de pesquisa do Laboratório de Engenharia e Modelagem Matemática (LabBEMM) pelo apoio e ensinamentos e também à Universidade Federal do Tocantins pela infraestrutura.

O presente trabalho foi realizado com o apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq – Brasil.

Todos os autores declararam não haver qualquer potencial conflito de interesses referente a este artigo.

## DADOS COMPLEMENTARES

*Anexo 1*

Tabela 1 - Valores de oxigênio dissolvido (OD) em diferentes temperaturas (T)

| T (ºC) | OD (mg)($L^{-1}$) | T (ºC) | OD (mg)($L^{-1}$) | T (ºC) | OD (mg)($L^{-1}$) |
|---|---|---|---|---|---|
| 0 | 14,60 | 16 | 9,86 | 32 | 7,30 |
| 1 | 14,22 | 17 | 9,64 | 33 | 7,17 |
| 2 | 13,80 | 18 | 9,47 | 34 | 7,06 |
| 3 | 13,44 | 19 | 9,27 | 35 | 6,94 |
| 4 | 13,08 | 20 | 9,09 | 36 | 6,84 |
| 5 | 12,76 | 21 | 8,91 | 37 | 6,72 |
| 6 | 12,44 | 22 | 8,74 | 38 | 6,60 |
| 7 | 12,11 | 23 | 8,57 | 39 | 6,52 |
| 8 | 11,83 | 24 | 8,41 | 40 | 6,40 |
| 9 | 11,56 | 25 | 8,25 | 41 | 6,33 |
| 10 | 11,29 | 26 | 8,11 | 42 | 6,23 |
| 11 | 11,04 | 27 | 7,96 | 43 | 6,13 |
| 12 | 10,76 | 28 | 7,83 | 44 | 6,06 |
| 13 | 10,54 | 29 | 7,68 | 45 | 5,97 |
| 14 | 10,31 | 30 | 7,56 | 46 | 5,88 |
| 15 | 10,06 | 31 | 7,43 | 47 | 5,79 |

Fonte: Marinho et al. (2023).

## REFERÊNCIAS

ALBUQUERQUE, Pedro Urbano Braga de; THOMAZINI, Daniel. **Sensores industriais**: fundamentos e aplicações. Saraiva: São Paulo, 2005.

ALVES, José Luiz Loureiro. **Instrumentação, controle e automação de processos**. Rio de Janeiro: LTC, 2012.

ARAÚJO, Lucynei Tenório de. **Estudo da Produção e do Envelhecimento do Vinagre de Laranja Lima**. Dissertação em Engenharia Química - Universidade Federal de Alagoas. 123 f. 2012.

ARDUINO. **Arduino UNO Rev3**. Store. Disponível em: <https://store.arduino.cc/usa/arduino-uno-rev3>. Acesso em: 20 abr. 2020.

BALAVALAD, Kirankumar; SHEEPARAMATTI, Basavaprabhu. A critical review of MEMS capacitive pressure sensors. **Sensors & Transducers**, v. 187, n. 4, p. 120, 2015.

BEGA, Egídio Alberto. **Instrumentação industrial**. Interciência: Rio de Janeiro, 2006.

DFROBOT. **Gravity**: Analog Dissolved Oxygen Sensor - Meter Kit For Arduino. Store. Disponível em: <https://www.dfrobot.com/product-1628.html>. Acesso em: 11 nov. 2023.

ELETROGATE. **Data Logger Shield para Arduino com RTC DS1307.** Store. Disponível em: <https://www.eletrogate.com/data-logger-shield-para-arduino-com-rtc-ds1307>. Acesso em: 11 nov. 2023.

GROSS, Camila Luz. **Desenvolvimento de um sistema para o monitoramento do pH em um biorreator de bancada durante o processo fermentativo do leite por grão de Kefir**. 2022. Trabalho de Conclusão de Curso-TCC (Graduação em Engenharia de Bioprocessos e Biotecnologia)−Universidade Federal do Tocantins, Gurupi, 2022.

MARINHO, Natanael da Costa.; GROSS, Camila Luz; TOLEDO, Gustavo Pereira; CREMA-CRUZ, Leandra Cristina; CRUZ, Pedro Alexandre da. Aquisição e controle de oxigênio dissolvido em um biorreator de bancada para fermentação de leite por grãos de Kefir utilizando a plataforma Arduino. **Revista Campo da História**, v. 8, n. 2, p. 716-728, 2023.

MARTINS, Alexandre Batista; FARINHA, José Torres; CARDOSO, António Marques. Calibration and certification of industrial sensors—a global review. WSEAS Trans. **Syst. Control**, p. 394-416, 2020.

MONK, Simon. Programação com Arduino: começando com Sketches. Porto Alegre: Bookman, 2013. 173 p.

MCROBERTS, Michael. **Arduino básico**. Novatec: São Paulo, 2018.

SEEDSTUDIO. **Water Pressure Sensor G1/4 1.2Mpa.** Store. Disponível em: <https://www.seeedstudio.com/Water-Pressure-Sensor-G1-4-1-2MPa-p-2887.html>. Acesso em: 10 nov. 2023.

SOUZA, Anderson R. de et al. **A placa Arduino**: uma opção de baixo custo para experiências de física assistidas pelo PC. Revista Brasileira de Ensino de Física, v. 33, n. 1, p. 01-05, 2011.

SCHMIDELL, Willibaldo; LIMA, Urgel de Almeida; AQUARONE, Eugênio; BORZANI, WALTER. **Biotecnologia Industrial**: engenharia bioquímica. São Paulo: Blücher, 2001. v. 2. 541 p.

TOLEDO, Gustavo Pereira. **Desenvolvimento de um sistema para o monitoramento da temperatura em um biorreator de bancada durante o processo fermentativo do leite por grão de Kefir**. 2022.

Trabalho de Conclusão de Curso-TCC (Graduação em Engenharia de Bioprocessos e Biotecnologia)−Universidade Federal do Tocantins, Gurupi, 2022.

UNICAMP - UNIVERSIDADE ESTADUAL DE CAMPINAS. **Estudo do Ambiente de Programação Arduino Software (IDE) com Intel Galileo Gen2**. Disponível em: <https://wordpress.ft.unicamp.br/lapet/wp-content/uploads/sites/23/2016/02/Estudo-do-Ambiente-de-Programação-Arduino-Software-IDE-com-Intel-Galileo-Gen2.pdf>. Acesso em: 20 abr. 2020.