

Situação Aprendizagem

Elisson Diego, Gabriel Vinícius, Daniel Mendes Nunes Costa e Luis Gustavo Sardo Furlan

Sumário

Sumário

[**1.** **Portão automático (inclusivo)**. 4](#_Toc167736885)

[**2.** **Código Fonte – Portão Automático** 4](#_Toc167736886)

[**3.** **Sensor de Matéria de Sistemas em Tempo Real.** 7](#_Toc167736887)

[**4.** **Código Fonte – Sensor de Matéria em Tempo Real** 8](#_Toc167736888)

[**5.** **Sensor de Temperatura para monitoramento Industrial** 11](#_Toc167736889)

[**6.** **Código Fonte – Sensor de Matéria de Sistemas em Tempo Real.** 12](#_Toc167736890)

[**7.** **Sensor de Movimento com Alarme.** 14](#_Toc167736891)

[**8.** **Código Fonte – Sensor de Movimento com Alarme.** 15](#_Toc167736892)

[**9.0** **Componentes** 17](#_Toc167736893)

[9.1 Resistor 17](#_Toc167736894)

[9.2 LED 17](#_Toc167736895)

[9.3 Buzzer 18](#_Toc167736896)

[9.4 Sensor Ultrassônico 18](#_Toc167736897)

[9.5 Sensor DHT22 18](#_Toc167736898)

[9.6 Display LCD I2C 19](#_Toc167736899)

[9.7 Placa Arduíno 19](#_Toc167736900)

[9.8 Sensor PIR 20](#_Toc167736901)

[**10.** **Justificativa** 20](#_Toc167736902)

Sumário

[Figura 1 - Portão Automático 4](file:///C:\Users\elisson_silva\Downloads\S-A%20%20DOCUMENTACAO%20ATUALIZADA%2027-05.docx#_Toc167738303)

[Figura 2 - Sensor de Matéria 7](#_Toc167738304)

[Figura 3 - Sensor de temperatura 11](file:///C:\Users\elisson_silva\Downloads\S-A%20%20DOCUMENTACAO%20ATUALIZADA%2027-05.docx#_Toc167738305)

[Figura 4 - Sensor de Movimento com Alarme 14](#_Toc167738306)

[Figura 5 - Foto prática 1 21](#_Toc167738307)

[Figura 6 - Foto prática 2 22](#_Toc167738308)

[Figura 7 - Foto prática 3 23](#_Toc167738309)

1. **Portão automático (inclusivo)**.

Escolhi esse projeto para situação aprendizagem pois andei vendo e pesquisando que muitas pessoas que sofrem com algum tipo de deficiência têm muitas e muitas dificuldades de se adaptarem a tais situações, foi disso que me veio a ideia de fazer um portão automático que possa atender a maioria das pessoas sendo elas portadoras de deficiência ou não.

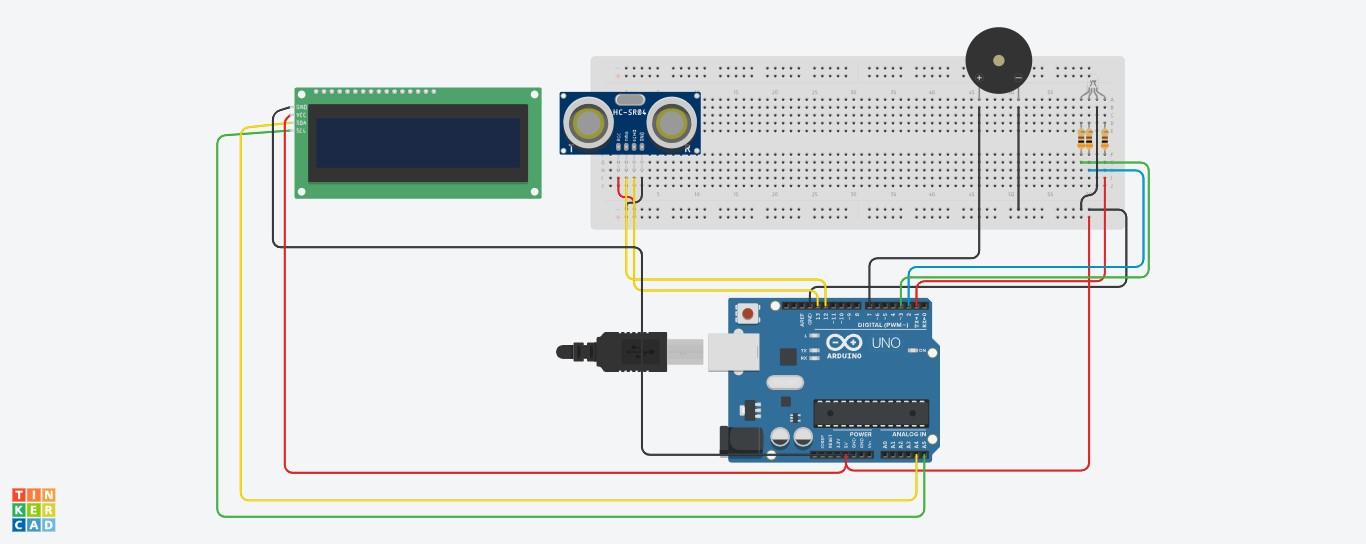
O funcionamento do projeto é bem simples, conforme as pessoas vão chegando perto do portão ele se abre, mas tem vários fatores que indicam que está abrindo por exemplo: Luzes, Som e Uma tela lcd. E com o afastamento do portão ele se fecha automaticamente.

Figura 1 - Portão Automático

*Figura 1 - Portão automático*

1. **Código Fonte – Portão Automático**

#include <Wire.h>  
#include <LiquidCrystal\_I2C.h>  
  
#define col 16  
#define lin 2  
  
#define ende 0x20  
LiquidCrystal\_I2C led(ende, col, lin);  
int buz = 7;  
int som = 261;  
int pinoLEDR = 1;  
int pinoLEDB = 2;  
int pinoLEDG = 3;  
int ECHO = 13;  
int TRIG = 12;  
long duracao = 0;  
long distancia = 0;    
void setup()  
{  
 Serial.begin(9600);  
  pinMode(ECHO, INPUT);  
  pinMode(TRIG, OUTPUT);  
  led.init();  
  led.backlight();  
  led.clear();  
}  
  
void loop()  
{  
   digitalWrite(TRIG, LOW);  
  delay(10);  
  digitalWrite(TRIG, HIGH);  
  delay(10);  
  digitalWrite(TRIG, LOW);  
  duracao = pulseIn(ECHO, HIGH);  
  distancia = (duracao / 58);  
  Serial.print("Distancia em cm:");  
  Serial.println(distancia);  
   
   if (distancia >= 300) {  
     analogWrite(pinoLEDR, 255);  
     analogWrite(pinoLEDB, 0);  
     analogWrite(pinoLEDG, 0);  
     led.setCursor(0,0);  
  led.print("Fechado");  
  led.setCursor(0,1);  
     noTone(buz);  
  } else if (distancia >= 200 && distancia <= 299) {  
    analogWrite(pinoLEDR, 0);  
    analogWrite(pinoLEDG, 0);    
    analogWrite(pinoLEDB, 255);  
    led.setCursor(0,0);  
  led.print("Fechado");  
  led.setCursor(0,1);  
      noTone(buz);  
  } else if (distancia >= 1 && distancia <= 100) {  
    analogWrite(pinoLEDR, 0);    
    analogWrite(pinoLEDB, 0);  
    analogWrite(pinoLEDG, 255);  
    led.setCursor(0,0);  
  led.print("Abrindo...");  
  led.setCursor(0,1);  
    tone(buz, som);  
  }  
}

1. **Sensor de Matéria de Sistemas em Tempo Real.**

Eu escolhi este projeto pois o desenvolvimento de sistemas embarcados é uma área de crescente importância na engenharia moderna, especialmente com o avanço da Internet das Coisas (IoT) e automação. O Arduino, uma plataforma de prototipagem eletrônica de código aberto, tem sido amplamente utilizado devido à sua facilidade de uso e vasta comunidade de suporte. Neste projeto, exploramos a criação de um sistema de monitoramento ambiental utilizando um Arduino. O sistema é capaz de medir e exibir níveis de luz, presença de pessoas, temperatura e níveis de gás, fornecendo informações em tempo real através de um display LCD.

O objetivo principal deste projeto é desenvolver um sistema que possa monitorar várias condições ambientais em um determinado espaço e exibir esses dados em um formato facilmente compreensível.

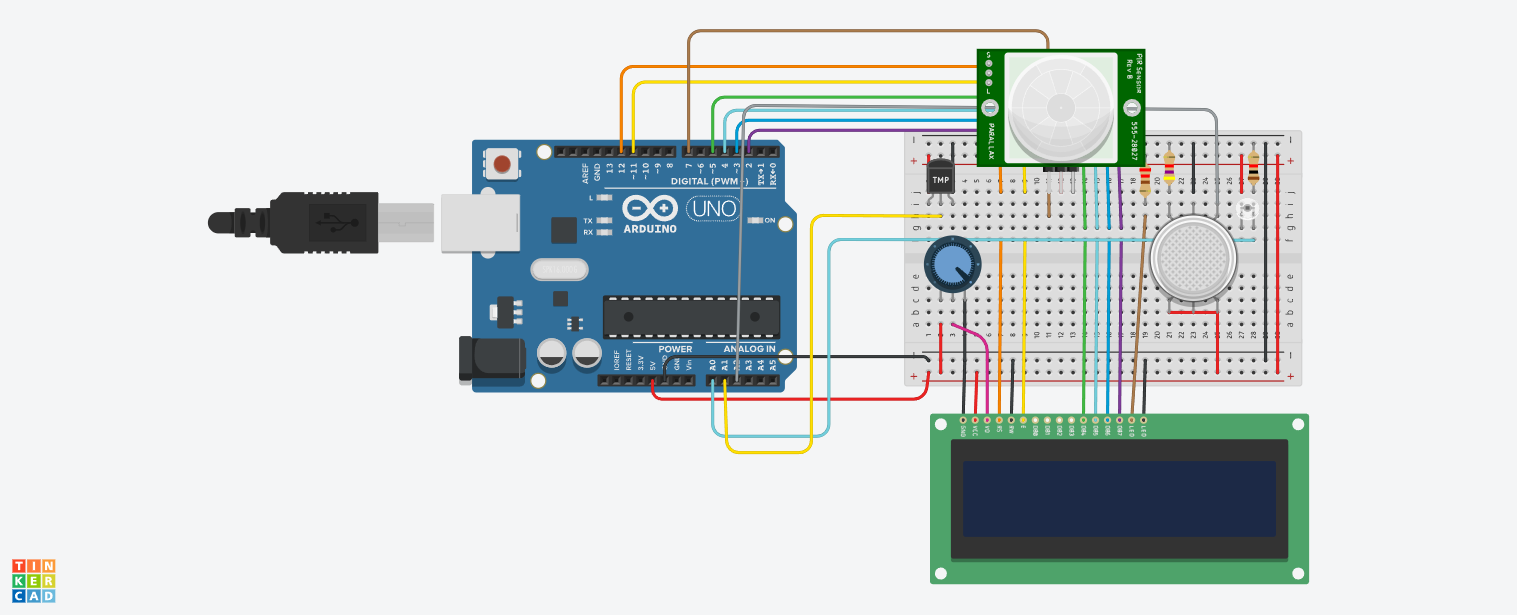


Figura 2 - Sensor de Matéria

1. **Código Fonte – Sensor de Matéria em Tempo Real**

#include <LiquidCrystal.h>  
  
// Define sensor pins  
#define sLuz A0  
#define sPresenca 7  
#define sTemp A1  
#define sGas A2  
  
// Initialize the LCD with the appropriate pins  
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);  
  
// Function prototypes  
void limparTela();  
void nivelLuz();  
void presenca();  
void temperatura();  
void gas();  
  
void setup() {  
  // Initialize the LCD  
  lcd.begin(16, 2);  
   
  // Set the presence sensor pin as an input  
  pinMode(sPresenca, INPUT);  
}  
  
void loop() {  
  // Measure and display light level  
  nivelLuz();  
  limparTela();  
   
  // Measure and display presence  
  presenca();  
  limparTela();  
   
  // Measure and display temperature  
  temperatura();  
  limparTela();  
   
  // Measure and display gas level  
  gas();  
  limparTela();  
}  
  
void limparTela() {  
  // Wait for 2 seconds  
  delay(2000);  
   
  // Clear the first line of the LCD  
  lcd.setCursor(0, 0);  
  lcd.print("                ");  
   
  // Clear the second line of the LCD  
  lcd.setCursor(0, 1);  
  lcd.print("                ");  
   
  // Wait for 1 second  
  delay(1000);  
}  
  
void nivelLuz() {  
  // Read the light sensor value  
  int luz = analogRead(sLuz);  
   
  // Map the light sensor value to a 0-10 range  
  luz = map(luz, 0, 471, 0, 10);  
   
  // Display the light level on the LCD  
  lcd.setCursor(0, 0);  
  lcd.print("Nivel de luz:");  
  delay(500);  
  lcd.setCursor(0, 1);  
  lcd.print(luz);  
  delay(200);  
  lcd.print("   ");  
}  
  
void presenca() {  
  // Read the presence sensor value  
  int presenca = digitalRead(sPresenca);  
   
  // Display the presence status on the LCD  
  lcd.setCursor(0, 0);  
  lcd.print("Sensor presenca:");  
  lcd.setCursor(0, 1);  
  if (presenca == 1) {  
    lcd.print("Estado: presente");  
  } else {  
    lcd.print("Estado: Ausente");  
  }  
}  
  
void temperatura() {  
  // Read the temperature sensor value  
  float temp = analogRead(sTemp) \* 5.0;  
  float tempC = ((temp / 1023.0) - 0.5) \* 100.0;  
   
  // Display the temperature on the LCD  
  lcd.setCursor(0, 0);  
  lcd.print("Temperatura: ");  
  lcd.setCursor(0, 1);  
  lcd.print("No momento ");  
  lcd.print(tempC);  
}  
  
void gas() {  
  // Read the gas sensor value  
  int gas = analogRead(sGas);  
   
  // Map the gas sensor value to a 0-100 range  
  gas = map(gas, 300, 750, 0, 100);  
   
  // Display the gas level on the LCD  
  lcd.setCursor(0, 0);  
  lcd.print("Nivel do Gas:");  
  lcd.setCursor(0, 1);  
  lcd.print("Atual: ");  
  lcd.print(gas);  
  delay(200);  
}

1. **Sensor de Temperatura para monitoramento Industrial**

Este projeto foi escolhido tendo em mente as necessidades das indústrias em monitorar a temperatura de suas máquinas e processos. Essa informação é necessária para a eficiência e bom desempenho das máquinas, além da importância para a segurança de seus funcionários.

Neste projeto eu utilizei o Arduíno, juntamente com um sensor de temperatura DHT22, um display LCD I2C 16x2, três LEDs, verde, vermelho, e azul, e um Buzzer.

O objetivo é desenvolver um projeto que faça o sensoriamento da temperatura utilizando o sensor DHT22, exibindo a temperatura no display LCD I2C, e realizando um alerta sonoro e visual com o Buzzer e os LEDs, respectivamente.

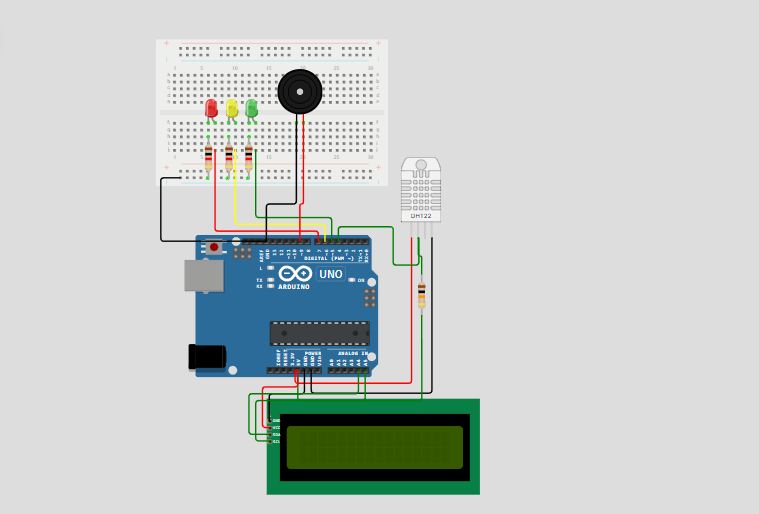


Figura 3 - Sensor de temperatura

*Figura SEQ Figura \\* ARABIC 3 - Sensor de Temperatura DHT22 e display LCD I2C*

1. **Código Fonte – Sensor de Matéria de Sistemas em Tempo Real.**

/\*Programa que mede temperatura e umidade

com sensor DHT22 e Exibe no Display 16x2

\*/

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include <DHT.h>

DHT dht(4,DHT22);     // Declara objeto do tipo DHT

float temp;     // Declara variáveis

#define col 16

#define lin 2

#define ende 0x27

int pinoVerde = 5;

int pinoAmarelo = 6;

int pinoVermelho = 7;

int pinoBuzzer = 9;

LiquidCrystal\_I2C led(ende, col, lin); // Declara objeto do tipo LiquidCrystal

void setup(){

**Serial**.begin(9600);

  led.init();

  led.backlight();

  led.clear();

  dht.begin();

  pinMode(pinoVermelho, OUTPUT);

  pinMode(pinoAmarelo, OUTPUT);

  pinMode(pinoVerde, OUTPUT);

  pinMode(pinoBuzzer, OUTPUT);

}

void loop(){

  temp = dht.readTemperature(); // Lê temperatura

  led.setCursor(0, 0);

  led.print("Temperatura: ");

  led.setCursor(0, 1);

  led.print(temp);

  led.print(" C");

  led.setCursor(9, 1);

if (temp <= -20){

    led.print(" PERIGO!!");

    digitalWrite(pinoVermelho, 1);

    digitalWrite(pinoAmarelo, 0);

    digitalWrite(pinoVerde, 0);

    tone(pinoBuzzer, 390);

  } else if (temp <= 10){

    led.print(" ALERTA!!");

    digitalWrite(pinoVermelho, 0);

    digitalWrite(pinoAmarelo, 1);

    digitalWrite(pinoVerde, 0);

    tone(pinoBuzzer, 390);

    delay(200);

    noTone(pinoBuzzer);

    delay(1000);

  } else if (temp <= 50){

    led.print(" SEGURO!!");

    digitalWrite(pinoVermelho, 0);

    digitalWrite(pinoAmarelo, 0);

    digitalWrite(pinoVerde, 1);

    noTone(pinoBuzzer);

  } else if (temp <= 80){

    led.print(" ALERTA!!");

    digitalWrite(pinoVermelho, 0);

    digitalWrite(pinoAmarelo, 1);

    digitalWrite(pinoVerde, 0);

    tone(pinoBuzzer, 390);

    delay(200);

    noTone(pinoBuzzer);

    delay(1000);

  }

}

1. **Sensor de Movimento com Alarme.**

Este projeto de Arduino integra um sensor PIR, um LED, um buzzer e um display LCD I2C para detectar movimento e informar o status visual e sonoramente. Inicialmente, os componentes são configurados e preparados para operação no bloco de setup. No loop principal, o código monitora continuamente o sensor PIR: quando movimento é detectado, o LED acende, o buzzer emite um som e o LCD exibe a mensagem "Movimento Detectado!", enquanto quando não há movimento, o LED apaga, o buzzer silencia e a mensagem "Nenhum Movimento" é mostrada no LCD. A função playTone() controla a frequência e duração do som do buzzer. Esse sistema é eficaz para aplicações de segurança ou automação residencial, fornecendo feedback imediato sobre a presença de movimento no ambiente monitorado.

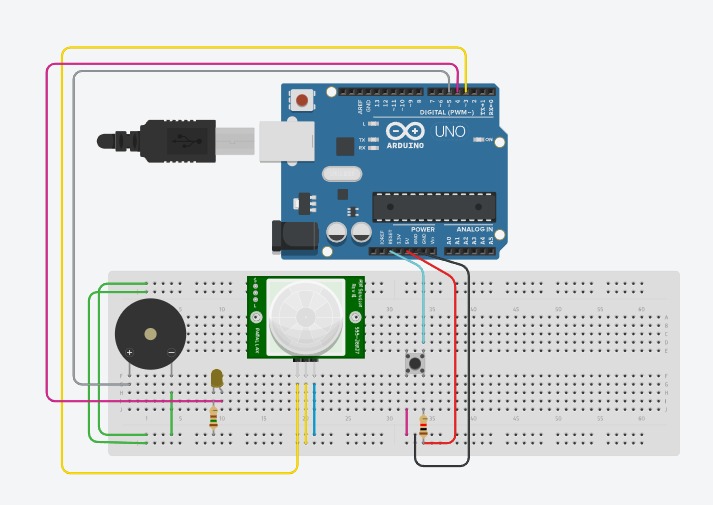
****

Figura 4 - Sensor de Movimento com Alarme

1. **Código Fonte – Sensor de Movimento com Alarme.**

int Sensor;

int Alarme;

int Led;

int Power;

int IV;

int X;

void setup()

{

Power = 2;

Sensor = 3;

Led = 4;

Alarme = 5;

pinMode(Power, INPUT);

pinMode(Sensor, INPUT);

pinMode(Led, OUTPUT);

pinMode(Alarme, OUTPUT);

}

void loop()

{

X = digitalRead(Power);

IV = digitalRead(Sensor);

if (X == LOW) {

if (IV == HIGH) {

TocarAlarme(1);

} else {

noTone(Alarme);

}

}

}

void TocarAlarme(int Y) {

tone(Alarme, 220);

digitalWrite(Led, HIGH);

delay(400);

digitalWrite(Led, LOW);

delay(400);

TocarAlarme(2);

}

1. **Componentes**

Componentes são dispositivos eletrônicos que servem para dar controle ao fluxo do circuito elétrico.

Têm como função o acionamento, barramento, direcionamento ou criação de fluxo de corrente elétrica.

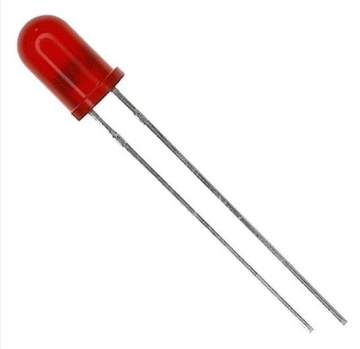
## 9.1 Resistor

Estes componentes tem o objetivo de barrar a corrente elétrica que passa através destes. É extremamente útil em circuitos elétricos, pois assim, garante a segurança e o bom funcionamento de outros componentes.



## 9.2 LED

Estes componentes convertem a energia elétrica em energia luminosa, e possui a função de iluminação.



## 9.3 Buzzer

Estes dispositivos são capazes de gerar frequências sonoras, e consequentemente, emitir sons.



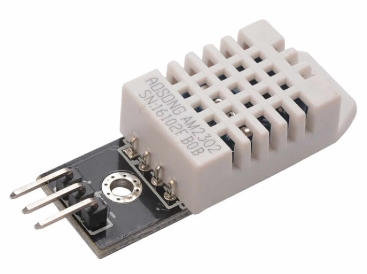
## 9.4 Sensor Ultrassônico

Um sensor ultrassônico é utilizado para a detecção de objetos. Estes dispositivos são capazes de calcular a distância até um objeto.



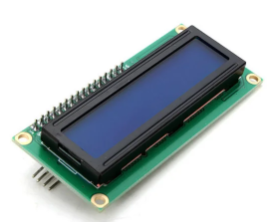
## 9.5 Sensor DHT22

Estes sensores são capazes de medir a temperatura, permitindo realizar medições de -40°C até 80°C.



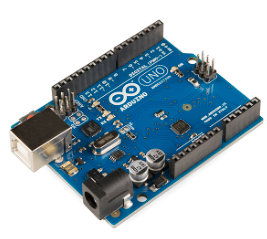
## 9.6 Display LCD I2C

Estes componentes funcionam criando uma interface para o usuário. Sendo utilizado para diversas finalidades, como exibição de dados de outros componentes, como por exemplo sensores.



## 9.7 Placa Arduíno

É um microcontrolador, o coração do circuito, sendo utilizado para o controle de fluxo das informações. Possui interface de programação, utilizada para integração entre componentes e dispositivos no circuito.



## 9.8 Sensor PIR

É uma espécie de sensor de proximidade, assim como o sensor ultrassônico, mas seu funcionamento, enquanto o sensor ultrassônico utilizado ondas para enviar e receber, e assim calcular a distância, o sensor PIR utilizada as ondas infravermelhas (calor) emitidas pelos objetos para calcular a distância.

1. **Justificativa**

Escolhemos o projeto de sensor de temperatura com Arduino para monitoramento industrial entre as quatro opções porque ele combina praticidade e inovação, é tecnicamente viável para nosso nível de conhecimento, e oferece um alto potencial de aprendizado prático. Além disso, ele tem aplicações reais e relevantes no campo industrial, o que pode ser um diferencial no nosso desenvolvimento acadêmico e profissional. Em resumo, é a opção que mais nos motiva e que consideramos mais útil e interessante.

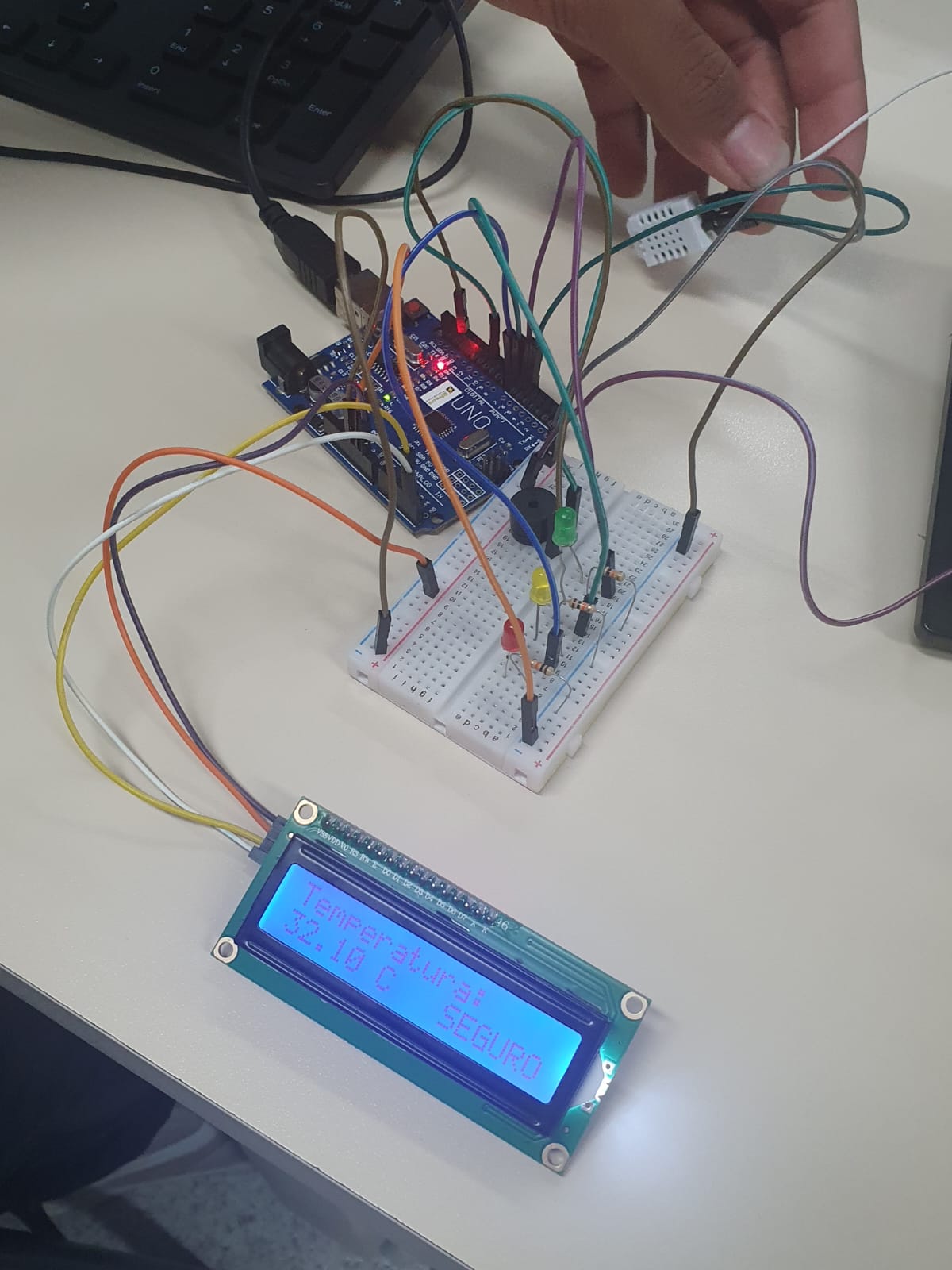


Figura 5 - Foto prática 1

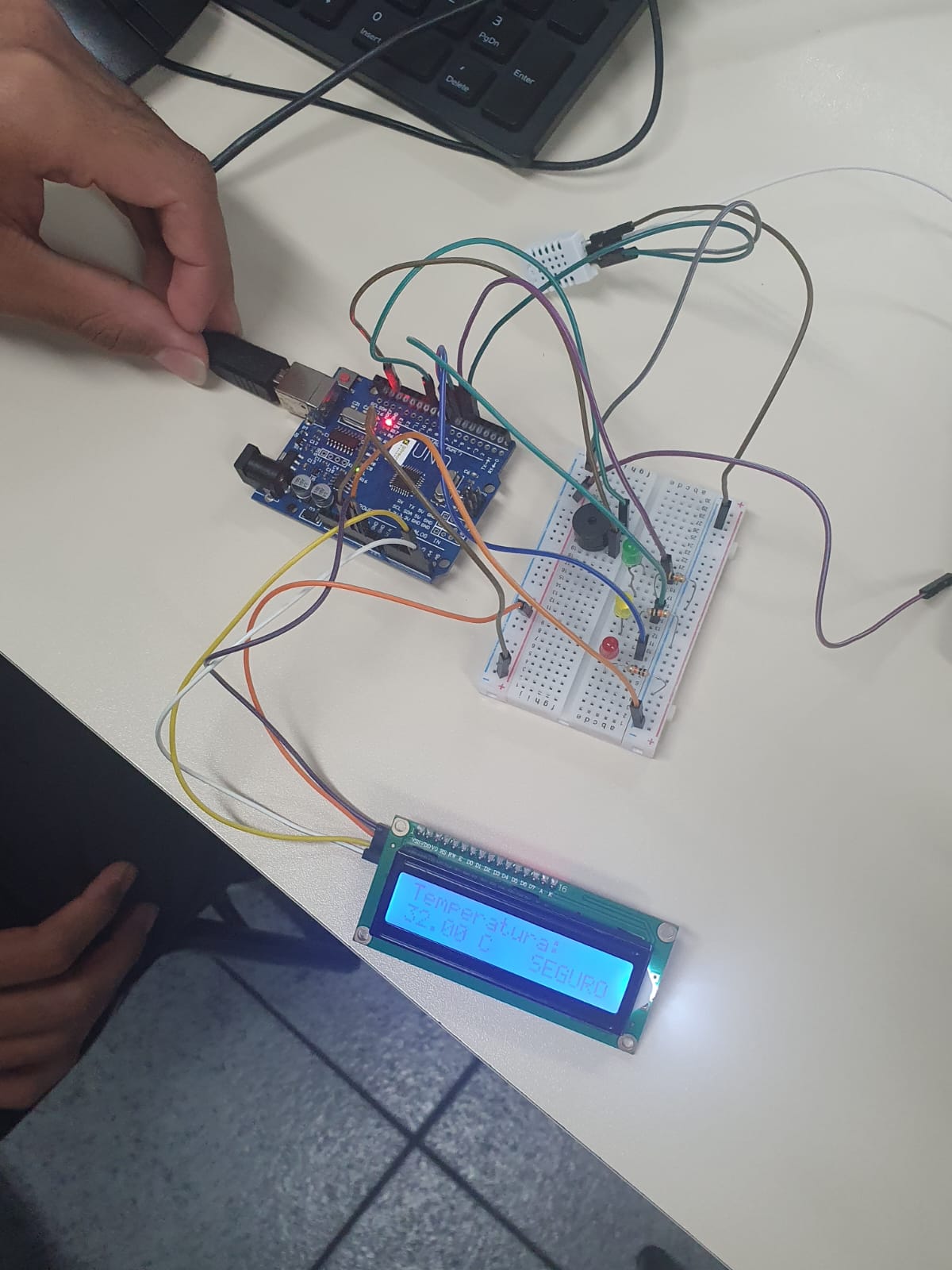


Figura 6 - Foto prática 2

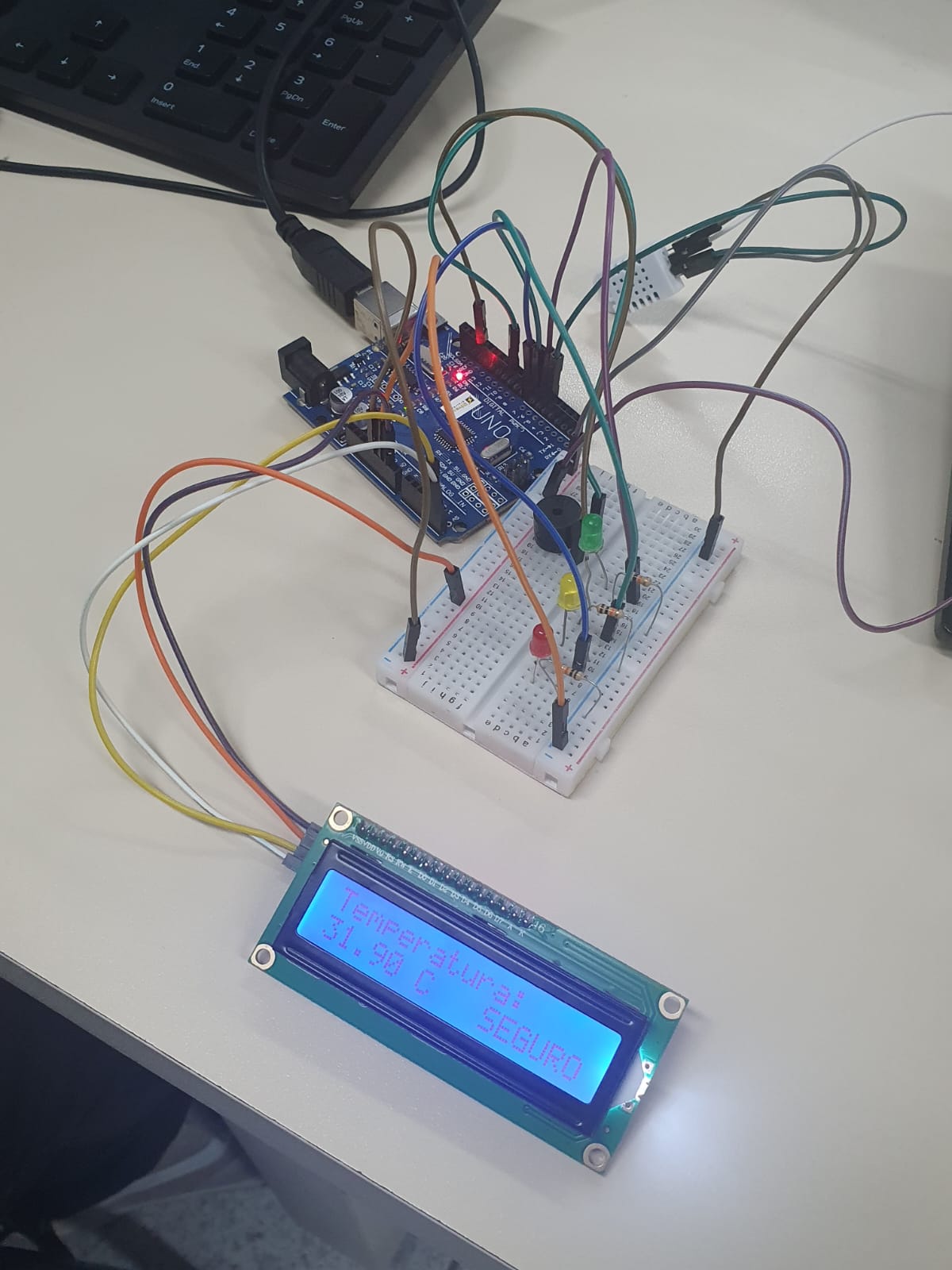


Figura 7 - Foto prática 3