UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
Programa de Pós-Graduação em Matemática
Mestrado Profissional - PROFMAT/CCT/UFCG

Carlos Gustavo Barreto de Farias Júnior

Produto Educacional

Uma abordagem simultânea das funções e da programação com Arduino

Campina Grande - PB
Agosto/2025

UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
Programa de Pós-Graduação em Matemática
Mestrado Profissional - PROFMAT/CCT/UFCG

Carlos Gustavo Barreto de Farias Júnior

Uma abordagem simultânea das funções e da programação com Arduino

Produto Educacional vinculado ao Trabalho de Conclusão de Curso apresentado ao Corpo Docente do Programa de Pós-Graduação em Matemática - CCT - UFCG, na modalidade Mestrado Profissional, como requisito parcial para obtenção do título de Mestre.

Orientador: Dr. Prof. Marcelo Carvalho Ferreira

Campina Grande - PB
Agosto/2025

Resumo

Na sequência didática aqui apresentada, serão abordados algoritmos de programação com modelagens matemáticas por meio das funções. Tais conceitos foram retirados da dissertação com título: Funções discretas e algoritmos de programação na robótica com Arduino. Nela, buscamos apresentar ao(à) estudante a plataforma de prototipagem Arduino, que é uma ferramenta por intermédio da qual o mesmo pode aplicar conhecimentos de matemática utilizando a criatividade para desenvolver dispositivos robóticos com sensores e saídas para realizar processos automatizados. Desta maneira, almejamos que sejam desenvolvidos nos(as) estudantes o pensamento criativo, a inventividade e a capacidade de resolução de problemas.

Palavras-chave: Funções. Programação. Arduino.

Abstract

The teaching sequence presented here will cover programming algorithms with mathematical modeling using functions. These concepts were taken from the dissertation titled: Discrete Functions and Programming Algorithms in Robotics with Arduino. In this dissertation, we aim to introduce students to the Arduino prototyping platform, a tool that allows them to apply mathematical knowledge and creativity to develop robotic devices with sensors and outputs for automated processes. In this way, we aim to develop students' creative thinking, inventiveness, and problem-solving skills.

Keywords: Functions. Programming. Arduino.

1 Introdução

A sequência começa pelas noções básicas sobre os componentes do kit básico de programação Arduino e o uso do multímetro como ferramenta de medida para verificação das condições do material. Em seguida, tratamos do software de programação Arduino IDE, por meio do qual os dispositivos serão programados e em cuja linguagem as funções matemáticas serão escritas numa representação adaptada.

Posteriormente, são propostas montagens e programações de diversos dispositivos, as quais consistem em exercícios de programação e modelagem matemática, além de preparar o(a) estudante para por em prática suas próprias ideias com os sensores e saídas do kit Arduino.

1.1 Objetivos

A implementação deste trabalho tem por objetivo ensinar ao(à) estudante como utilizar funções para modelar matematicamente padrões de interpretação de medidas de grandezas por meio de variáveis advindas de sensores e controladores e programar comandos de resposta a tais sinais numericamente interpretados. Além disso, pretendemos introduzir um dos muitos meios de criação de tecnologia no qual a matemática é primordial.

Neste trabalho, são propostas as atividades de escrever funções matemáticas em uma representação diferente, aprender noções básicas de uma linguagem de programação, usando a mesma para implementar algoritmos, relacionar progressões aritméticas com funções afins de domínio discreto e resolver problemas envolvendo funções polinomiais do primeiro grau, por exemplo. Tais exercícios são incluídos com o propósito de que as habilidades *EM13MAT501, EM13MAT406 e EM13MAT302* propostas na BNCC sejam desenvolvidas nos(as) estudantes. [1]

1.2 Organização

Inicialmente, descreveremos a sequência didática exibindo informações básicas como título, duração média e público alvo. Em seguida, falaremos das habilidades que serão trabalhadas pelos alunos na sequência e dos recursos necessários.

A sequência é dividida em oito encontros, os quais são sequenciados por ordem de complexidade da programação e da montagem, acompanhados de exercícios de matemática envolvendo funções discretas e progressões, por exemplo.

2 Sequência didática

Descrição

- Título: Funções discretas e algoritmos de programação na robótica com Arduino.
- Professor: Carlos Gustavo Barreto de Farias Júnior.
- Público-alvo: 1°, 2° e 3° anos do Ensino Médio.
- Duração: De 8 a 24 horas-aula.

Habilidades

- Conhecer e empregar os conceitos básicos da linguagem de programação Arduino;
- Manipular funções discretas para obter um resultado determinado;
- Utilizar os modelos matemáticos adaptados à linguagem de programação do Arduino para regular o funcionamento de sensores e saídas;
- Utilizar as funções matemáticas, a montagem e a linguagem de programação Arduino para criar protótipos de facilidades e soluções para problemas reais para melhorar a qualidade de vida das pessoas.

Recursos e materiais

- Computadores;
- Kits básicos Arduino;
- Apostilas, Cadernos e canetas;
- Quadro branco, lápis pilotos e apagador;
- Data show e notebook.
- Multímetros

2.0.1 Primeiro encontro: Introdução ao Arduino e ao multímetro

Na primeira aula, os(as) estudantes devem se organizar em grupos de 4 participantes.Espera-se que esta etapa transcorra em 2 horas aula.

Atividade 1: Distribuir um kit Arduino e um multímetro para cada grupo. A partir daí, apresentar cada componente do kit aos(às) estudantes, utilizando as informações contidas na seção sobre o Arduino. Explicar a importância dos resistores para proteger os circuitos e os componentes oferecendo resistência à corrente elétrica, e etc... Na sequência, o professor deve discorrer brevemente sobre a história da criação do Arduino.(Ver o capítulo 4)

Atividade 2: Propor que os(as) estudantes conectem o cabo vermelho em VΩmA e o cabo preto em COM. (Alguns multímetros tem entradas diferentes para Volts e para Ampères. Assim, deve-se conectar na entrada específica do que se quer medir.) Na sequência, orientar aos(às) estudantes que posicionem o cursor do multímetro em 20V. Assim que todos os grupos tiverem configurado o cursor, propor que cada grupo posicione as pontas dos cabos uma em cada polo da bateria disponível no kit Arduino, e registre no caderno o resultado mostrado no display do multímetro. Apresentar o problema 5.1

Problema 2.1. *Uma bateria pode ser classificada como nova, semi-nova ou usada a partir da tensão medida entre seus pólos. Para classificar, utiliza-se o seguinte critério:*

A bateria está

(i) Nova, se a tensão medida é maior que a informada no rótulo;

(ii) Semi-nova, se a tensão medida é igual à informada no rótulo;

(iii) Usada se a tensão medida é menor que a informada no rótulo.

Utilizando este critério, classifique a bateria do seu kit.

Atividade 3: Após a conclusão do problema 5.1, propor aos(às) estudantes o Problema 5.2

Problema 2.2. *Faça a medição da corrente elétrica entre os pólos da sua bateria e anote o valor em seu caderno. (Para medir a corrente elétrica, basta posicionar as pontas do multímetro uma em cada polo da bateria e apontar o cursor para a faixa de 20 miliAmpères (20 mA))*

2.0.2 Segundo encontro: Medição de resistência elétrica

Neste, e em todos os próximos encontros, a turma deve ficar dividida nos mesmos grupos de quatro integrantes. O tempo estimado para este encontro é de duas horas-aula.

Atividade 1: Propor que cada grupo pegue um resistor do kit Arduino e entregar uma folha com o código de cores dos resistores, disponível na figura 19 e o texto explicativo sobre o código de cores. Na sequência, apresentar o texto explicativo e propor o Problema 5.3.

Código de cores dos resistores: Texto explicativo

O código de cores dos resistores consiste numa sequência de faixas coloridas pintadas no resistor que significa o número $d_1d_2.10^p$ variando em v percentuais, onde d_1d_2 é um número inteiro de dois algarismos que correspondem à primeira e segunda cor, respectivamente, o qual é multiplicado pela potência p de 10, correspondente à terceira cor. O número resultante é a resistência oferecida pelo resistor, a qual pode variar na porcentagem v correspondente à quarta cor informada no resistor.

O código de cores a seguir mostra o que significa cada cor. Os estudantes devem anotar as cores e, baseando-se no código, escrever a sequência de dígitos e determinar a resistência correspondente do resistor.

Cor	Dígitos Significativos (1 e 2)	Multiplicador (3)	Tolerância (4)
Preto	0	10^0	
Marrom	1	10^1	± 1%
Vermelho	2	10^2	± 2%
Laranja	3	10^3	
Amarelo	4	10^4	
Verde	5	10^5	
Azul	6	10^6	
Violeta	7	10^7	
Cinza	8	10^8	
Branco	9	10^9	
Ouro	-	10^{-1}	± 5%
Prata	-	10^{-2}	± 10%
Sem Cor	-	-	± 20%

Figura 1 – Fonte: Apostila Robótica Livre com Arduino - p. 14

Exemplo 1. *Considere o resistor na figura 20.*

Figura 2 – Fonte: Apostila Robótica Livre com Arduino - p. 14

A sequência de cores deste resistor é verde, azul, vermelho e ouro. Com isto, ele tem a resistência

$$56.10^2 = 5600\Omega$$

variando em $\pm 5\%$.

Problema 2.3. *Utilizando as instruções do texto explicativo na folha entregue pelo professor, interprete o código de cores do seu resistor e escreva o significado em seu caderno. Na sequência, aponte o cursor do multímetro para 20k*Ω *e meça a resistência de seu resistor. Compare o resultado da medição com o significado do código. Qual foi maior? O código está correto?*

Atividade 2 - Propor que os(as) estudantes peguem a placa protoboard no kit e apontem o cursor do multímetro para o marcador que contém um símbolo de uma pequena buzina. Ao apontar o cursor para este ponto, o multímetro consegue detectar se entre as pontas dos cabos existe condução elétrica e confirma isto com um sinal sonoro. Desta forma, é possível verificar se entre dois pontos de uma placa existe ou não uma ligação. Na sequência, propor para os(as) estudantes o Problema 5.4.

Problema 2.4. *Utilize o multímetro no modo da verificação de circuito (buzina) para verificar como os pontos da placa protoboard se conectam. Com isto, faça um desenho em seu caderno ilustrando como são as ligações internas da placa protoboard, seguindo o modelo na figura 21.*

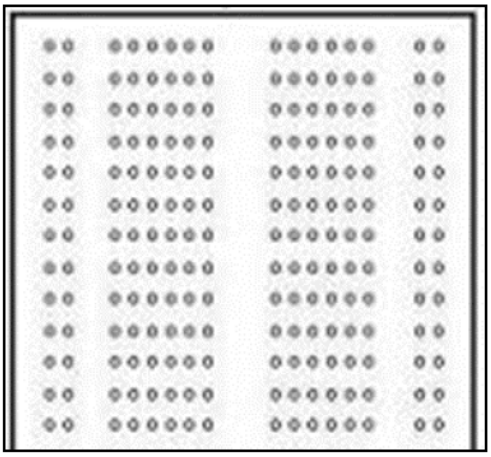

Figura 3 – Fonte: Apostila Robótica livre com Arduino, p. 15

2.0.3 Terceiro encontro: Função da porta digital e o Pisca com LED

Atividade 1: Organizar os grupos nos quais a turma foi dividida desde o início da seguinte maneira: cada integrante deverá ter uma função das listadas a seguir:

1-Montador: É responsável pela montagem dos protótipos e pelos componentes disponibilizados.

2-Programador: É Responsável pela programação, ou seja, escrita, compilação e carregamento dos códigos para a placa Arduino.

3-Líder: É Responsável por garantir que cada um exerça sua função específica.

4- Almoxarifado: Fica responsável por receber e devolver o material que o professor entregar ao grupo.

(Esta divisão deve continuar para todos as atividades até o final desta sequência, porém com revezamento das funções entre os membros do grupo.)

Atividade 2: Propor que os(as) estudantes liguem os computadores, iniciem o Arduino IDE e entregar a cada grupo um material de apoio com as figuras 22 e 23 impressas. Na sequência, propor que montem e programem o dispositivo como nas figuras 22 e 23.

Esquema de Ligações na Protoloard:

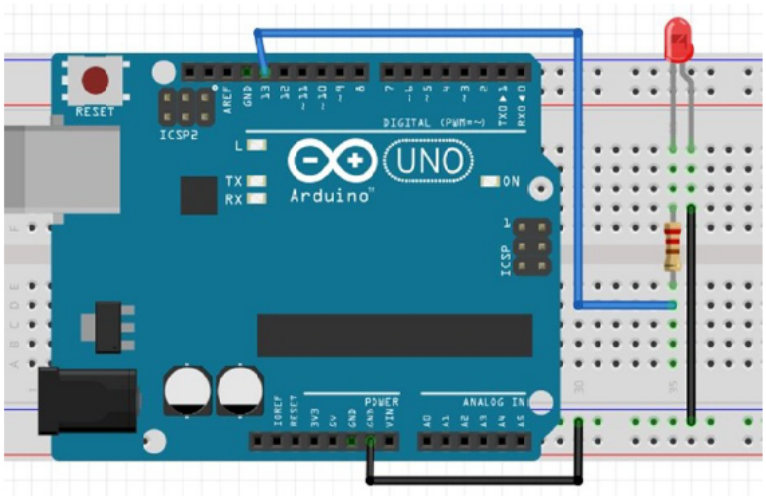

Figura 4 – Fonte: Apostila Robótica livre com Arduino, p. 7.

```
1   int LED = 13; //Declaração de uma variável chamada LED conectada à porta 13
2   void setup() { //Função de ajuste executada uma única vez
3   pinMode (LED, OUTPUT); //Rotula a variável como saída de dados
4   }
5   void loop() {
6   digitalWrite(LED, HIGH);
7   //Torna a variável LED ligada (HIGH coloca a tensão no máximo: 5 V)
8   delay(1000);
9   //Espera um segundo (1000 milisegundos)
10  digitalWrite(LED, LOW);
11  // Torna a variável LED desligada (LOW coloca a tensão no mínimo: 0 V)
12  delay(1000); }
```

Figura 5 – Fonte: Apostila Robótica livre com Arduino, p. 7.

Atividade 3: Explicar aos(às) estudantes o funcionamento da porta digital, onde o LED foi conectado na atividade anterior. Explicar que a porta digital executa apenas dois comandos, que são 1 ou 0, que são interpretados pela porta como sinal alto (5V) ou sinal baixo (0V), respectivamente, e desenhar no quadro o diagrama da figura 24. Na sequência, proponha o problema 5.5.

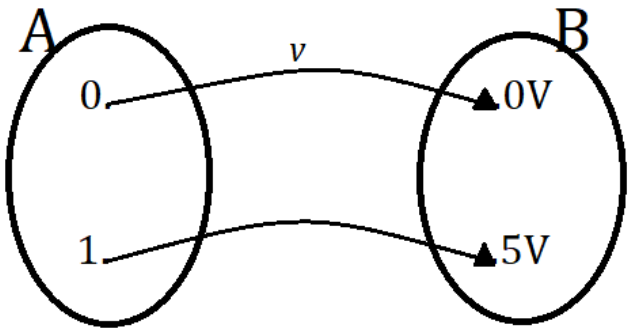

Figura 6 – Fonte: autor

Problema 2.5. *Determine uma relação matemática que defina o funcionamento da porta digital, que transforma os sinais 0 e 1 nas tensões 0V e 5V, respectivamente.*

Atividade 4: Mostrar aos(às) estudantes a seguinte resolução: Define-se uma função $v : \{0, 1\} \to \mathbb{R}$ na qual $v(n) = 5n$, onde o valor $v(n)$, dado em volts (V), é a tensão resultante do valor n atribuído à porta. Após isto, propor o problema 5.6.

Problema 2.6. *Faça as seguintes alterações no protótipo do pisca com LED:*

(a)LED com sinal alto por 1 segundo e baixo por 1 segundo;

(b)LED com sinal alto por 3 segundos e baixo por 1 segundo;

(c)LED com sinal alto por 1,5 segundo e baixo por 3,2 segundos;

(d)Utilize outra porta digital para o LED e faça-o funcionar nela.

Atividade 5: Propor o problema 5.7

Problema 2.7. *Segundo a biologia, o olho humano consegue perceber o piscar de uma luz se ele ocorrer num tempo maior que ou igual a 40 milissegundos. Faça esse teste usando o LED Arduino.*

O professor deve verificar, de grupo em grupo, como os(as) estudantes estão desenvolvendo as atividades e dar o suporte necessário.

Caso as aulas não sejam geminadas, é importante orientar os(as) estudantes a salvar os códigos escritos numa pasta com o nome do grupo.

2.0.4 Quarto encontro: Controlar LED com sinal PWM

Nesta atividade, os(as) estudantes utilizarão portas PWM, que são de 8 bits, para ativar o LED. Esta atividade tem como objetivo mostrar como uma porta PWM pode modular a tensão indo da mais baixa para a mais alta passando por vários níveis.

Função de uma porta PWM

Atividade 1:Explicar aos(às) estudantes o funcionamento básico da modulação do sinal por uma porta PWM, que o sinal modulado pela PWM vai de $00000000 = 0$ **a** $11111111 = 255$**, os quais correspondem a tensões de 0V a 5V, respectivamente. Na sequência, propor o problema 5.8**

Problema 2.8. *Sabendo que a tensão emitida no LED é proporcional ao sinal obtido pela porta PWM, de maneira que os sinais* 0 *e* 255 *(maior e menor sinal) correspondem às tensões 0V e 5V (tensão mínima e tensão máxima), respectivamente, determine a função crescente que transforma o sinal PWM* n *na voltagem* $v(n)$ *da porta Arduino. Ilustre esta função num diagrama.*

Solução: O domínio da função v é composto pelos valores dos sinais recebidos pela porta PWM, ou seja, se trata do conjunto $\{0, 1, 2, ..., 255\}$. Como a tensão $v(n)$ é proporcional a n e $v(0) = 0$, a função v é do tipo $v(n) = an$, em que a é a constante de proporcionalidade, a qual pode ser obtida a partir de um sinal $n \neq 0$, pela fórmula

$$a = \frac{v(n)}{n}.$$

Utilizando valores dados na questão, obtemos

$$a = \frac{5}{255} = \frac{1}{51}.$$

Então, a função desejada é $v : \{0, 1, 2, ..., 255\} \to \mathbb{R}$, na qual $v(n) = \frac{1}{51} \cdot n$, onde $v(n)$ é a voltagem emitida pela porta para a saída acoplada, em função do valor n atribuído à porta. O diagrama a seguir ilustra tal função.

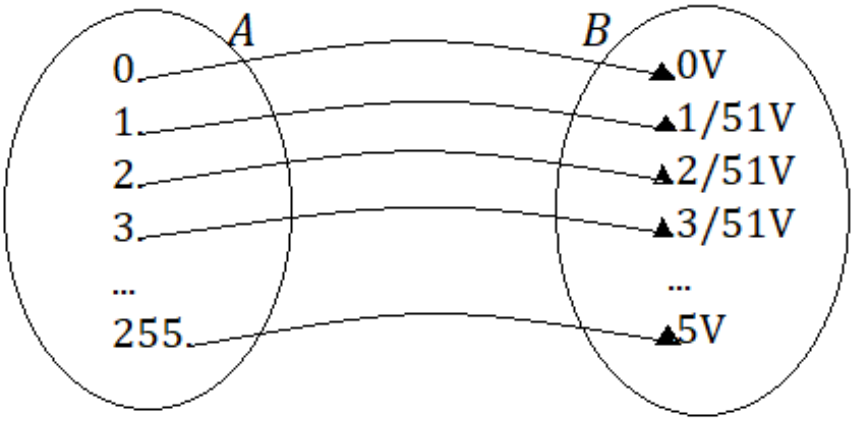

Figura 7 – Fonte: Autor.

Atividade 2: Propor a montagem e programação do dispositivo de Pisca com led da figura 26 com o código da figura 27.

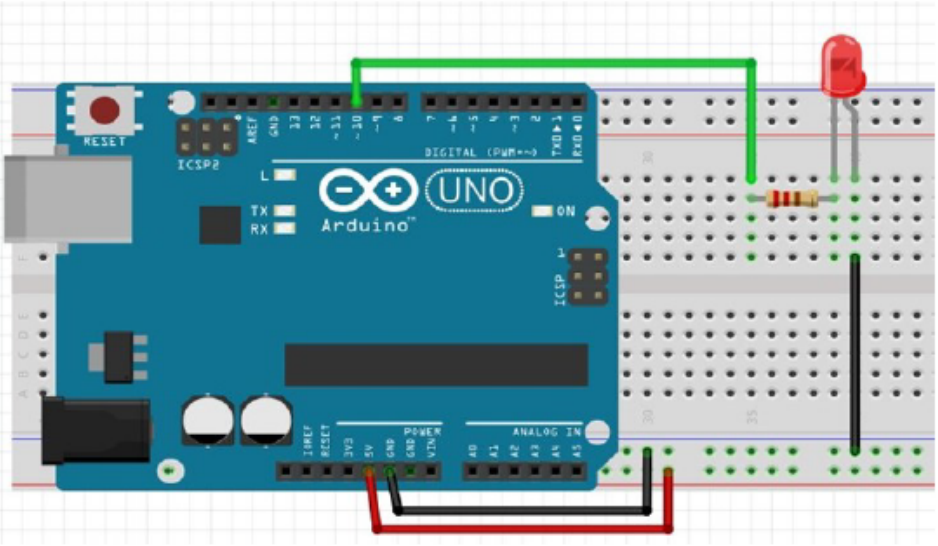

Figura 8 – Fonte: Apostila Robótica livre com Arduino, p.13.

```
1   int led=10; //Declara a variável led na porta 10;
2   int brilho=0; // Fixa o valor inicial da variável brilho = 0;
3   int pulo=5; // Fixa a constante pulo com o valor 5.
4   void setup () {
5       pinMode (led, OUTPUT);} //Configura led como uma saída.
6   void loop () {
7   analogWrite (led, brilho); // Atribui à variável led o valor da variável brilho;
8   brilho = brilho + pulo;
9   // Define que o valor de brilho será dado por seu valor atual mais pulo;
10  if (brilho==0 || brilho==255){pulo= - pulo;}
11  //Se brilho = 0 ou brilho 0 255, o valor de pulo muda de sinal. (||="ou")
12  delay(30);} //Espera 30 milissegundos.
```

Figura 9 – Fonte: Autor.

Atividade 3: Propor os problemas 5.9, 5.10 e 5.11

Problema 2.9. *Altere o código no Arduino IDE para que o led oscile*

(a) mais rápido.

(b) mais lento.

Problema 2.10. *Escreva de maneira abreviada a progressão aritmética finita com os valores atribuídos à variável brilho no primeiro momento do funcionamento, ou seja, quando a tensão está aumentando de 0 a 255.*

Problema 2.11. *Considere a P.A do item anterior. Seja a_n seu termo geral.*

(a) Determine a_{10}.

(b) Determine, se possível, a_{25} e a_{40}.

(c) Escreva uma fórmula para encontrar um termo qualquer a_n em função de n.

(d) Quantos termos tem a P.A. em questão?

(e) Calcule a soma de todos os termos desta P.A..

2.0.5 Quinto encontro: Controlar um pisca de LED com potenciômetro

Nos dois encontros anteriores, controlamos a saída LED utilizando sinal digital e sinal PWM. Nesta atividade, controlaremos a saída LED utilizando sinal analógico. Para controlar o sinal analógico, utilizaremos um potenciômetro, que é um botão giratório que servirá como um controle de intervalo do pisca de LED.

Atividade 1: Explicar aos estudantes o funcionamento básico da modulação do sinal por uma porta analógica, que vai de $0000000000 = 0$ **a**

1111111111 = 1023**, os quais correspondem a tensões de 0V a 5V, respectivamente. Na sequência, propor o problema 5.12**

Problema 2.12. *Se os sinais 0 e 1023 da porta analógica correspondem a 0V e a 5V, respectivamente, determine a função que transforma o sinal analógico n na voltagem $v(n)$ da porta Arduino. Ilustre esta função num diagrama.*

Solução: É a função $v : A \to \mathbb{R}$, na qual $A = \{0, 1, 2, ..., 1023\}$ e $v(n) = \frac{5}{1023}n$, onde $v(n)$ é a voltagem emitida pela porta para a saída acoplada, em função do valor n atribuído à porta.

Atividade 2: Entregar a cada grupo os componentes listados a seguir e propor a montagem e programação do protótipo nas figuras 28 e 29.

Componentes

1 led;
1 resistor de 1000 Ohms;
3 jumpers pretos;
4 jumpers coloridos de cores distintas;
1 placa protoboard;
1 potenciômetro;
1 placa Arduino Uno.

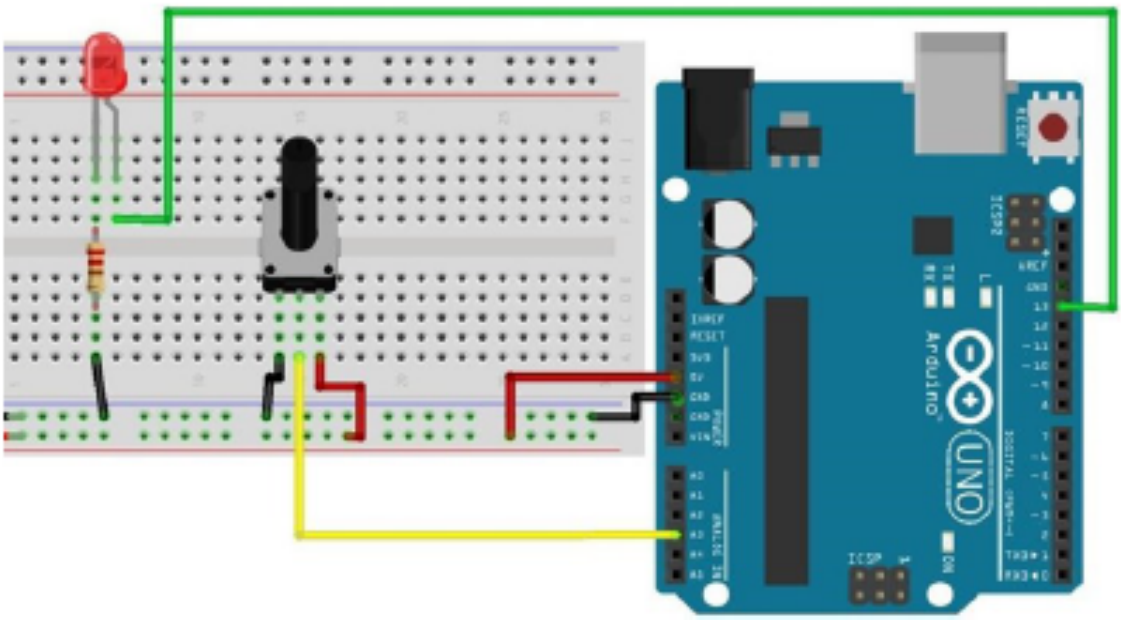

Figura 10 – Fonte: Apostila Robótica livre com Arduino, p. 20

```
1   int sensorPin = A3; // Seleciona o pino de entrada do potenciômetro;
2   int ledPin = 13; // Seleciona o pino do LED ;
3   int sensorValue =0;
4   //Configura a variável sensorValue, que carrega o valor do potenciômetro.
5   void setup() {
6   Serial.begin(9600);
7   // Inicia comunicação com o monitor serial;
8   pinMode(ledPin, OUTPUT); }
9   // Declara ledPin como uma saída}
10  void loop() {
11  sensorValue = analogRead(sensorPin);
12  //Lê o valor do sensor e o atribui a sensorValue;
13  Serial.println(sensorValue); // Imprime <sensorValue> na serial
14  digitalWrite(ledPin, HIGH); // Liga o LED
15  delay(sensorValue); // mantém o comando por <sensorValue> milissegundos
16  digitalWrite(ledPin, LOW); // Desliga o LED
17  delay(sensorValue); // Mantém o comando por <sensorValue>
18  }
```

Figura 11 – Fonte: Autor. Feito com auxílio do software Arduino IDE.

A porta A3 é a porta analógica escolhida para ser a porta de entrada onde será conectado o potenciômetro, declarado no código como sensorPin (Pino de sensor). O Led, assim como foi feito no terceiro encontro, é declarado numa porta digital, pois vai apenas ligar ou desligar e, a última declaração é da variável *sensorValue*, que assumirá o valor dado pelo Potenciômetro, de 0 a 1023. O código novamente declara ledPin como uma saída em pinMode(ledPin, OUTPUT). Em "sensorValue = analogRead(sensorPin)", é dado o comando de ler o valor do sensor potenciômetro, que será representado por "sensorValue". Na linha seguinte, "Serialprintln(sensorValue)", o valor do potenciômetro é impresso no monitor serial. Então o tempo que o Led fica apagado e aceso é mostrado em milissegundos no monitor. As quatro últimas linhas são análogas às da programação do pisca com Led feita no terceiro encontro, só que com o tempo de delay sendo dado pelo valor do sensor.

Atividade 2: Após a montagem e programação, escrever no quadro o algoritmo 1 e orientar os estudantes a seguir os passos do algoritmo para abrir o monitor serial.

Algoritmo 1

1-Abra o Arduino IDE;

2-Clique em Tools, ou Ferramentas;

3- Clique em Serial monitor, ou monitor serial.

4- O monitor serial será aberto e, por fim, você verá as informações nele impressas.

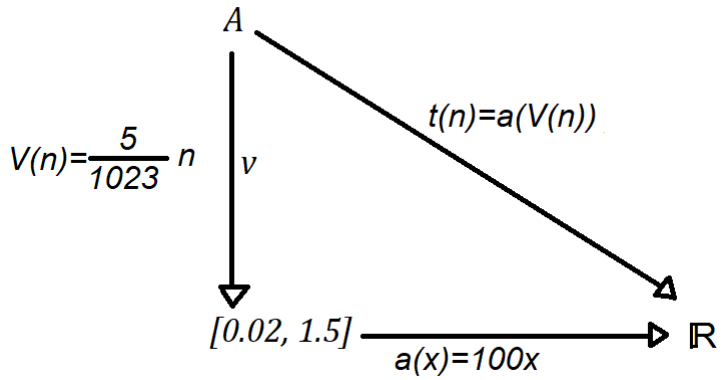

Figura 12 – Fonte: Autor. Feito com auxílio do software Paint.

Função discreta da porta analógica

A porta analógica do Arduino Uno serve apenas como entrada, e os valores captados por esta porta vão de 0000000000 = 0 a 1111111111 = 1023. No caso do potenciômetro utilizado nesta atividade, as funções definidas nas linhas 11 e 15 transformam o valor do sensor em um intervalo em milissegundos entre um ligar e um desligar do led. Desta forma, controlamos o intervalo do pisca utilizando a entrada analógica.

O tempo t em milissegundos de intervalo entre o ligar e o desligar do led é dado em função do sinal n analógico, em que

$$t(n) = n$$

.

2.0.6 Sexto encontro: Sensor de temperatura

Atividade 1: Explicar aos alunos como a função de conversão do sinal analógico em temperatura do sensor LM35 funciona, como descrito no texto explicativo a seguir:

Função de conversão do sinal analógico em temperatura do sensor LM35

O sensor LM35 pode medir temperaturas de -55°C a 150 °C. Para tal, existem dois modos de ligar o sensor: A *ligação básica* e a *ligação de faixa completa.* A montagem que será feita neste encontro seguirá a ligação básica. Esta ligação mede temperaturas de 2°C a 150°C e, portanto, não pode medir as temperaturas abaixo de 2°C. Tais medidas só podem ser efetuadas na ligação de faixa completa, que mede de -55°C a 150°C.

A fórmula que transforma o sinal analógico n, dado pelo Arduino em função da tensão emitida pelo sensor, na temperatura t correspondente, em graus Celsius, é $t(n) = \frac{500n}{1023}$.

Para obter esta fórmula, utilizamos a função $a : [0.02, 1.5] \to \mathbb{R}$ na qual $a(x) = 100x$ que dá a temperatura a em graus Celsius em função da tensão x emitida pelo sensor, pois cada $0.01V$ vindo do sensor corresponde a 1°C, e esta tensão varia dentro do intervalo $[0.02, 1.5]$. Como o sensor é ligado a uma porta analógica, então as tensões que vão de 0.02V a 1.5V, o que é apenas uma parte da tensão admitida pelas portas do Arduino (0V a 5V), são interpretadas como sinais analógicos pertencentes ao conjunto {5,6,7,...,306}, que também corresponde a uma parte equivalente dos sinais analógicos possíveis descritos anteriormente. Para obter a tensão correspondente ao sinal analógico dado, basta multiplicar este sinal pela constante de proporcionalidade $\frac{5}{1023}$. Daí, a função que dá a temperatura t em função do sinal analógico n é $t : A' \to \mathbb{R}$, onde $A' = \{5, 6, 7, ..., 306\}$ e $t(n) = \frac{5}{1023} \cdot n \cdot 100 \Rightarrow C(n) = \frac{500n}{1023}$.

A equação utilizada na programação que será feita na atividade 1 está na forma padrão, como é orientado no manual e na apostila Robótica Livre com Arduino.

Atividade 1: Entregar a cada grupo os componentes do esquema de montagem na figura 30 e propor que os estudantes façam a montagem e a programação contidas nas figuras 30 e 31.

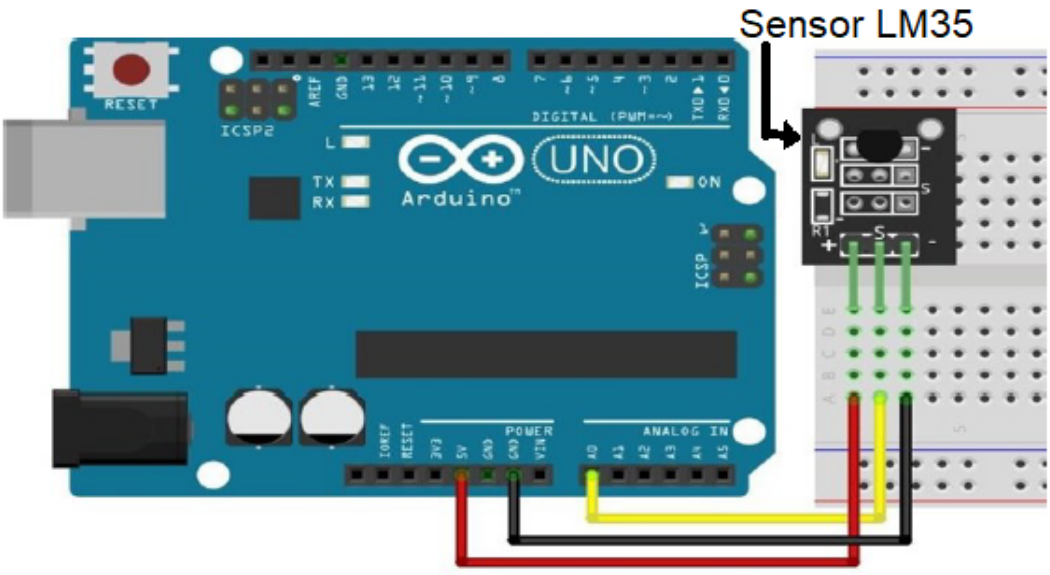

Figura 13 – Fonte: Autor. Adaptada da apostila robótica livre com Arduino

```
1   const int LM35 = A0; // Define o pino que vai ler a saída do LM35
2   float temperatura; // Variável que armazenará a temperatura medida
3   void setup() {
4   Serial.begin(9600);}
5   void loop() {
6   temperatura = (float(analogRead(LM35))*5/(1023))/0.01;
7   //Fórmula que transforma tensão em temperatura em ºC
8   Serial.print("Temperatura (Celsius): ");
9   //Imprime "Temperatura (Celsius):" no monitor serial
10  Serial.println(temperatura);
11  //Imprime o valor de temperatura no monitor serial
12  delay(2000);}
```

Figura 14 – Fonte: Autor. Adaptação da apostila robótica livre com Arduino

Na parte de declaração de variáveis, o código declara que A0 é a porta que vai receber os valores do sensor LM35 e a variável que armazenará esses valores será "temperatura". Na configuração, apenas o monitor serial é ativado com o comando "Serial.begin(9600)". Na lógica, a variável temperatura é dada pela equação

$$temperatura = \frac{float(analogRead(LM35)).5}{1023} : 0,01$$

Esta equação transforma o valor do sensor LM35 lido pela porta A0 em uma temperatura na escala celsius. E, atribui esse valor à variável temperatura, que será impressa no monitor serial pelos comandos "serial.print". A leitura é feita a cada 2 segundos, intervalo que pode ser alterado na linha delay(2000).

Atividade 2: Exibir a lei de correspondência da função t da atividade 1 e propor aos estudantes o problema 5.13 e 5.14

Problema 2.13. *Substitua a equação da linha 6 do código pela lei de correspondência da função t apresentada pelo professor e carregue o código para o Arduino. A temperatura continuou sendo medida corretamente? Explique este fato.*

Problema 2.14. *Pesquise as equações de conversão das escalas Kelvin e Fahrenheit para a escala Celsius e vice versa. Escreva um código para que a temperatura seja dada em Kelvin e depois faça o mesmo para a escala Fahrenheit.*

Nesta atividade, observamos o emprego das funções na resolução de problemas, que faz parte das habilidades EM13MAT302 e EM13MAT501, da BNCC, dispostas no início deste capítulo.

2.0.7 Sétimo encontro: Sensor de luminosidade e sensor ultrassônico

O sensor de luminosidade disponível no kit Arduino é um resistor dependente de luz ou, simplesmente, LDR que é um resistor que modula sua resistência em função da luz nele emitida. Quanto mais forte a luz, maior é a resistência que ele aplica ao circuito. Desta forma, ele se torna eficiente para dispositivos de relé e automações que sejam ativadas ou desativadas de acordo com a luminosidade ambiente.

Atividade 1: Entregar a cada grupo os componentes dispostos na lista a seguir e propor a montagem e a programação demonstradas nas figuras 32 e 33.

Componentes

1- Um sensor LDR;
2- Uma protoboard;
3- Uma placa Arduino Uno;
4- Um cabo USB de compilação do kit Arduino;
5- Três jumpers de cores distintas.
A montagem deve seguir o padrão na figura

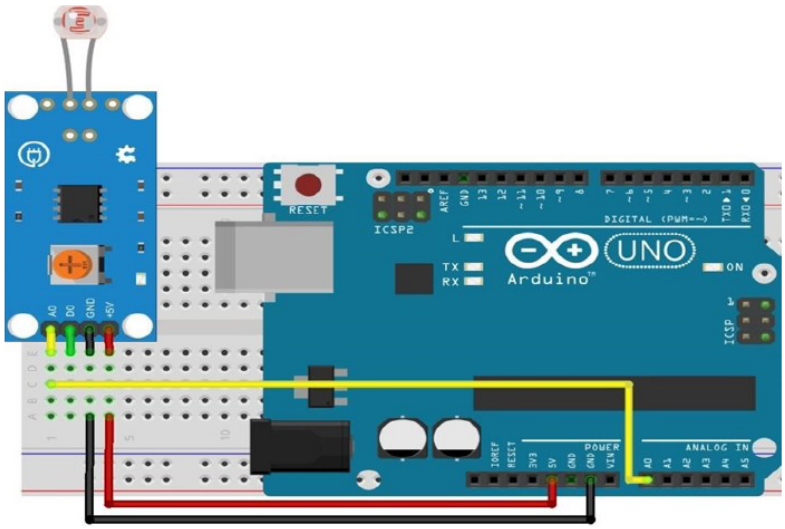

Figura 15 – Fonte: Apostila Robótica livre com Arduino, p. 16.

Observação: Os sensores LDR podem variar, dependendo do fabricante. Alguns mais atuais são menores, contendo apenas o disco espelhado e dois conectores, então a montagem pode precisar alterações.

Programação do LDR

O código sugerido é o seguinte:

```
int LDR = A0; //Declara a variável LDR cujos valores são recebidos
                pela porta analógica A0.
int val = 0; //Declara a variável val, que será o valor analógico recebido do LDR.
void setup() {
  Serial.begin (9600); //Inicia o monitor serial.
}
void loop() {
  val = analogRead (LDR); //Adquire o valor na porta analogica e atribui a "val".
Serial.print (val); //Exibe no monitor serial o valor da variável "val".
delay (1000); //Espera 1000 mS = 1 segundo.
}
```

Figura 16 – Fonte: Autor. Adaptado da apostila Robótica livre com Arduino.

Atividade 2: Após a montagem e a programação, propor que os estudantes abram o monitor serial e, com as luzes da sala apagadas, anotem o valor impresso. Em seguida, acender as luzes da sala e pedir que os estudantes anotem o valor dado no monitor serial. Assim que cada grupo tiver os valores registrados, o professor deve propor que comparem os valores e discutam o porquê, caso haja diferença nos valores de um grupo e de outro.

Atividade 3: Distribuir a cada grupo os componentes listados a seguir e, na sequência, propor que os estudantes montem e programem o Sensor ultrassônico HC SR04, cujos esquemas estão dispostos nas figuras 34 e 35.

Componentes

1- Uma protoboard;
2- Uma placa Arduino Uno
3- Um cabo USB de compilação do kit Arduino;
4- Três pares de jumpers, sendo cada par com dois jumpers de cores iguais;
5- Um sensor ultrassônico HC SR04.

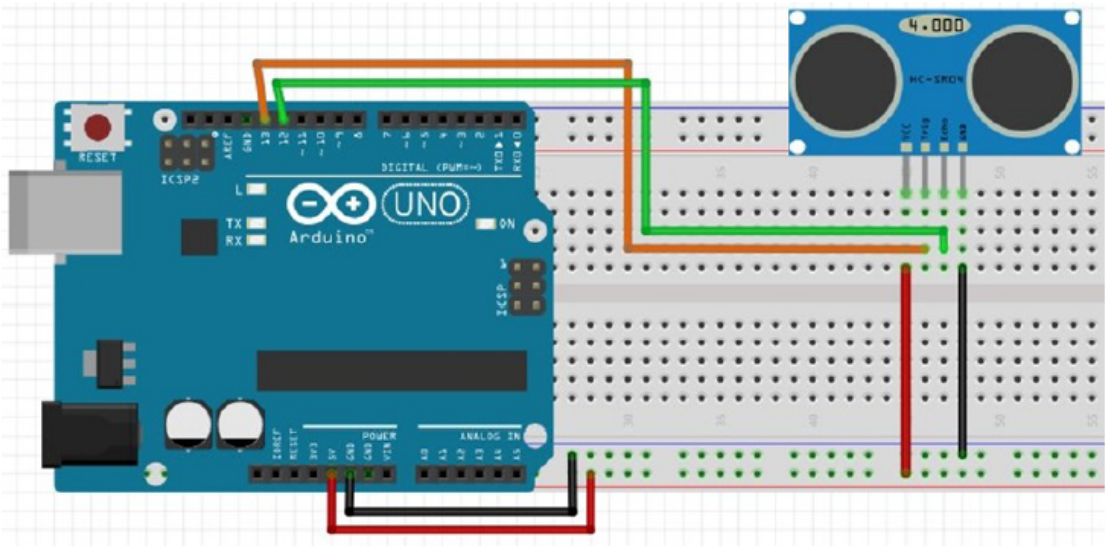

Figura 17 – Fonte: apostila Robótica livre com Arduino, p. 17.

Programação

O código para programação do sensor é

```
1   int trigPin = 13; //Declaração da porta de saída de emissão do pulso
2   int echoPin = 12; //Declaração da porta de recepção do pulso
3   void setup() {
4   Serial.begin (9600); //Ativa o monitor serial
5   pinMode (trigPin, OUTPUT); //Rotula a variável trigPin como saida de dados (emissão do pulso)
6   pinMode (echoPin, INPUT); //Rotula a variável echoPin como entrada de dados (recepção do pulso)
7   }
8   void loop() {
9   long duration, distance; //Declara as variáveis duration e distance
10  digitalWrite (echoPin, LOW); //Inicia o receptor de sinal como desligado
11  delayMicroseconds (2); //Aguarda 2 microssegundos
12  digitalWrite (trigPin, HIGH); //Aciona o emissor de sinal
13  delayMicroseconds (10); //Aguarda 10 microssegundos
14  digitalWrite (trigPin, LOW); //Desliga o emissor de sinal
15  duration = pulseIn (echoPin, HIGH); //contabiliza o instante entre a emissão e recepção do sinal sonoro
16  distance = (duration / 2) / 29.4; //Distância dada em função do tempo entre a emissão e a recepçõ do sinal
17  }
```

Figura 18 – Fonte: Descrições: Autor. Código: Autor, adaptado da apostila Robótica livre com Arduino, p. 18.

$$\textit{Explicação: } d = vt = \frac{340m}{s} \cdot \frac{duration}{2} = 340 \cdot \frac{100cm}{10^6 \mu s} \cdot \frac{duration}{2} \cdot 0.034 \approx \frac{1}{29{,}4}$$

Nesta atividade, será utilizado um sensor ultrassônico HC SR04 e, utilizando a propriedade física da reflexão das ondas, este sensor servirá para medir a distância entre ele e um objeto à frente. O HC SR04 emite um pulso ultrassônico e também o recebe. A diferença entre o instante da emissão e o recebimento é um intervalo de tempo que

será transformado em distância através da função horária da posição.

Atividade 4: Resolver o problema 5.16 no quadro junto com os estudantes

Problema 2.15. *A velocidade do pulso ultrassônico da HC SR04 é a mesma do som no ar, que é de, aproximadamente, 340 m/s. Utilizando o cálculo da distância d de um movimento, em função da velocidade v, em metros por segundo e do tempo t em segundos, que é*

$$d = v \cdot t,$$

determine a lei de correspondência da função d, que dá a distância do sensor ao objeto imediatamente à sua frente, a partir do tempo t, em segundos.

Solução

O sensor calcula o tempo em microssegundos que o pulso ultrassônico leva do momento em que sai do sensor até o momento em que retorna. Desta forma, o tempo t em microssegundos calculado pelo sensor deve ser dividido por dois, pois é o tempo de ida e volta do pulso, já que ele reflete no objeto. O tempo obtido na função *pulseIn* é medido em microssegundos, que equivalem a $10^{-}6$ vezes um segundo.(0,000001 s) [2].

Assim, a distância d, em metros, em função do tempo obtido pelo sensor pode ser escrita como

$$d(t) = \frac{t}{2} \cdot 10^{-6} \cdot 340 \iff d(t) = 17 \cdot 10^{-5}t$$

Como a linguagem matemática utilizada no Arduino se baseia apenas nas quatro operações básicas, a função encontrada acima pode ser escrita como

$$d(t) = 0,00017t$$

Atividade 5: Propor que os(as) estudantes resolvam os problemas 5.17 e 5.18

Problema 2.16. *Realize medições de distância com seu sensor na sala de aula e conjecture qual a unidade de medida na qual a distância dada pelo HC SR04 é mostrada no monitor serial.*

Problema 2.17. *2- Substitua a equação da linha 16 do código pela que o professor mostrou na resolução do problema 6.16. A unidade na qual a distância é mostrada é realmente metros?*

2.0.8 Oitavo encontro: Buzzer e sensor de campo magnético

Esta atividade consistirá em programar uma saída sonora, que é o buzzer. Para isto, utilizaremos a função tone, a qual pode reproduzir músicas e melodias por meio do buzzer a partir das frequências das notas musicais.

Atividade 1: Entregar os componentes listados a seguir para cada grupo e propor que montem e programem a saída sonora buzzer, cujos esquemas de montagem e programação constam nas figuras 36 e 37.

Componentes

1- Uma placa protoboard;
2- Uma placa Arduino Uno;
3- Um cabo USB de compilação do kit Arduino;
4- Um buzzer;
5- Dois jumpers de cores distintas.

Dependendo do modelo do buzzer, a montagem abaixo pode necessitar de adaptações. Existe uma variação de montagem com a qual é possível fazer o som do buzzer ser emitido em um volume mais alto. Não é difícil, por experimentação, encontrar esta montagem. Para isto, o professor pode disponibilizar um jumper a mais para os grupos.

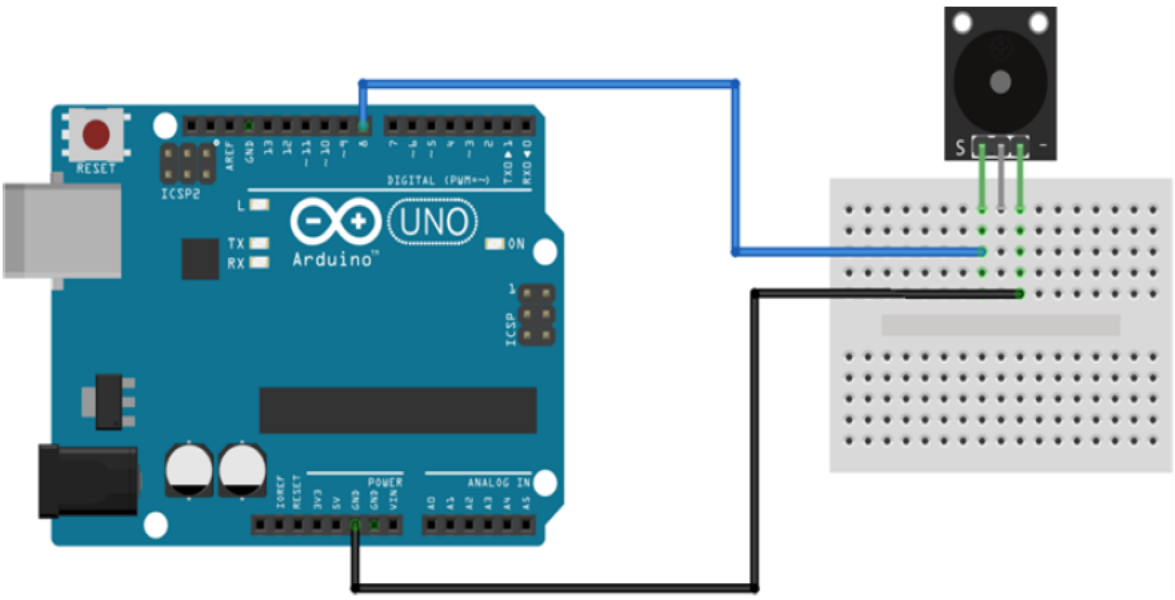

Figura 19 – Fonte: apostila Robótica livre com Arduino, p. 23.

Programação

```
#define tempo 10
int frequencia = 0;
int Pinofalante = 8;
void setup() {
  Serial.begin(115200);
pinMode(Pinofalante,OUTPUT); //Pino do buzzer
}
void loop()
{
  //Varia frequência entre 150 e 1800 somando 5
for (frequencia = 150; frequencia < 1800; frequencia += 5)
{
tone(Pinofalante, frequencia, tempo);
Serial.println(frequencia);
delay(1);
}
//Varia frequencia entre 1800 e 150 subtraindo 5
for (frequencia=1800; frequencia >150; frequencia-=5)
{
  tone(Pinofalante, frequencia, tempo);
  Serial.println(frequencia);
  delay(1);
}
}
```

Figura 20 – Fonte: Autor. Adaptado a partir da Apostila Robótica livre com Arduino.

Uma pergunta que pode surgir é: Como é possível oscilar a frequência em mais de 2 tons diferentes se o buzzer está ligado em uma porta digital, que só recebe dois bytes?

A resposta a essa pergunta é que o tom do som é controlado por uma frequência de pulsos de onda, e não por tensões específicas. Assim, o Arduino pode usar a porta digital para emitir pulsos com a mesma tensão e mudar apenas a frequência dos pulsos, o que alterará o tom emitido pelo buzzer.

Atividade 2: Propor o problema 5.19

Problema 2.18. *Seja $t \geq 0, t \in \mathbb{N} \cup \{0\}$, o instante em milissegundos em que a frequência $f(t)$ é emitida pelo buzzer. Pela programação, sabemos que no instante 0, a frequência é 150 Hz e aumenta 5Hz a cada 1 milissegundo até chegar a 1800 Hz, que é quando esta frequência começa a diminuir 5 Hz a cada segundo. A partir disso, responda:*

(a) Quanto vale $f(10)$? E $f(25)$?

(b) Em que momento t a frequência começa a diminuir?

(c) Sendo $A = \{t; 0 \leq t \leq 330\}$ *o domínio da função* f*, determine a lei de correspondência de* f*.*

(d) Determine a lei de correspondência de f *se o domínio é* $D = \{t \in \mathbb{N}; 331 \leq t \leq 660\}$*.*

Atividade 3: Entregar a cada grupo os componentes listados a seguir e pedir que montem e programem o sensor de campo magnético cujo esquema de montagem e programação consta nas figuras 38 e 39.

Componentes

Para esta atividade, cada grupo precisará de

1- Uma placa protoboard;

2- Uma placa Arduino Uno;

3- Um cabo USB de compilação do kit Arduino;

4- Um sensor de efeito hall 49e;

5 - Três jumpers de cores distintas;

6- Um imã qualquer.

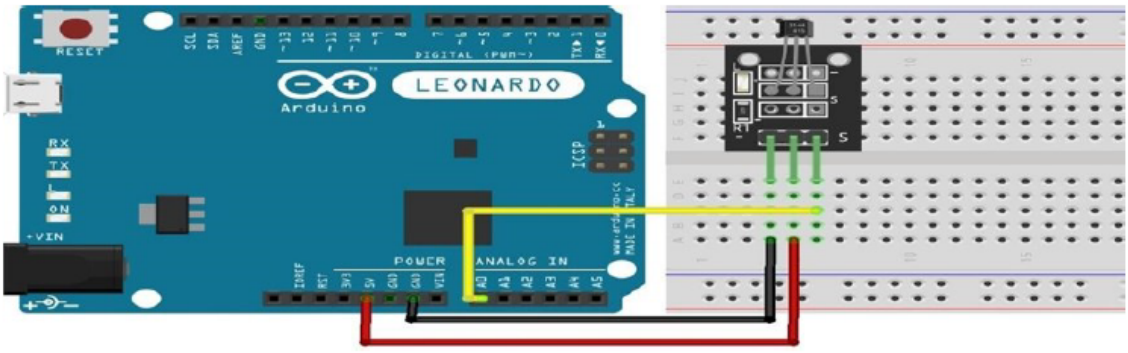

Figura 21 – Fonte: apostila Robótica livre com Arduino, p. 20.

Programação

```
1   float refVoltage = 5.0/1023; //Fixa a constante referência
2   float sensorVolts;          //Cria a variável da tensão vinda do sensor
3   int val1;                  //Cria a variável do valor analógico de A0
4   void setup() {
5   Serial.begin(9600);      //Inicia comunicação com o monitor serial
6   }
7   void loop() {
8   val1=analogRead(0); //Lê o valor da porta A0 e o atribui à variável val1
9   sensorVolts=refVoltage*(val1);
10  // Multiplica o sinal de A0 por 5/1023 calculando assim a tensão
11  Serial.print("B=");
12  //Imprime "B´´ no serial, como a variável do campo magnético
13  Serial.print((sensorVolts*895.52773449)-2238.8193362 );
14  // Multiplica a tensão por uma constante e subtrai outra, obtendo o valor de B
15  Serial.println("Gauss");//Imprime a palavra "Gauss´´ no serial
16  if (((sensorVolts*667)-1667) > 2) { Serial.println (" Norte");};
17  if (((sensorVolts*667)-1667) < -2 ) {Serial.println (" Sul");};}
18  //Define se o campo magnético é norte ou sul pela tensão tensão vinda do sensor.
```

Figura 22 – Fonte: Autor. Adaptado a partir da apostila Robótica livre com Arduino, ps. 20 e 21.

Nesta atividade, é feita a programação e montagem de um sensor de campo magnético que identifica a polaridade de um imã. Porém, o sensor hall49e pode ser utilizado para programar diversos dispositivos diferentes envolvendo campos magnéticos, como detectores de metal, por exemplo. Quando o hall 49e é atravessado por um campo magnético, ele produz um corrente elétrica proporcional ao campo, que vai de 0,86V a 4,21 V e mede campos elétricos de -1500 Gauss a +1500 Gauss.

Atividade 4: Propor os problemas 5.18 e 5.19

Problema 2.19. *Na equação da linha 9, sensorVolts é a tensão obtida por multiplicar refVoltage por val*1*, onde a variável val*1 *é o valor analógico recebido pela porta A0 e refVoltage é a constante* $\frac{5}{1023}$*.Baseando-se nesta equação, responda às questões a seguir:*

(a) Seja t a tensão, e n o valor analógico, escreva a lei de formação da função $t(n)$*.*

(b) Determine os valores máximo e mínimo da função t

Problema 2.20. *A expressão na linha 13 do código calcula o valor do campo magnético captado pelo sensor em função da tensão advinda dele, representada por sensorVolts.*

Seja $c(t)$ *a função definida pela expressão da linha 13 do código, que determina o campo magnético a partir da tensão t vinda do sensor.*

(a) Usando o item (c) da questão 1, determine o domínio da função c.

(b) Escreva a lei de formação da função c usando a expressão da linha 13 do código.

(c) Calcule $c(2)$, $c(3)$ *e* $c(5)$.

(d) Calcule os valores máximo e mínimo da função c.

(e)Qual o tipo da função (c)?

Referências

1 BRASIL. *Base Nacional Comum Curricular*. 2017. BNCC. Disponível em: <https://www.gov.br/mec/pt-br/cne/base-nacional-comum-curricular-bncc>. (Citado na página 4).

2 Arduino Documentation. *pulseIn() — Referência de Linguagem Arduino*. 2024. Acessado em 27 de maio de 2025. Disponível em: <https://docs.arduino.cc/language-reference/pt/fun%C3%A7%C3%B5es/advanced-io/pulseIn/>. (Citado na página 23).