



UNIVERSIDADE FEDERAL RURAL DA AMAZÔNIA

INSTITUTO CIBERESPACIAL

CURSO DE LICENCIATURA EM COMPUTAÇÃO

WILLIAM FERREIRA SANCHES

**A UTILIZAÇÃO DO EMULADOR DE ARDUINO TINKERCAD PARA O ENSINO
DE LÓGICA DE PROGRAMAÇÃO E ELETRÔNICA**

BELÉM – PA
2020

WILLIAM FERREIRA SANCHES

**A UTILIZAÇÃO DO EMULADOR DE ARDUINO TINKERCAD PARA O ENSINO
DE LÓGICA DE PROGRAMAÇÃO E ELETRÔNICA**

Trabalho de Conclusão de Curso apresentado ao curso de
Licenciatura em Computação da Universidade Federal
Rural da Amazônia como requisito para obtenção do
grau de Licenciado em Computação.
Orientador: Prof. Dr. João Ferreira de Santanna Filho

BELÉM – PA
2020

Dados Internacionais de Catalogação na Publicação (CIP)
Bibliotecas da Universidade Federal Rural da Amazônia
Gerada automaticamente mediante os dados fornecidos pelo(a) autor(a)

S211u Sanches, William Ferreira
A UTILIZAÇÃO DO EMULADOR DE ARDUINO TINKERCAD PARA O ENSINO DE LÓGICA DE
PROGRAMAÇÃO E ELETRÔNICA / William Ferreira Sanches. - 2020.
54 f. : il. color.

Trabalho de Conclusão de Curso (Graduação) - Curso de Computação (Licenciatura), Campus
Universitário de Belém, Universidade Federal Rural Da Amazônia, Belém, 2020.
Orientador: Prof. Dr. João Ferreira de Santanna Filho

1. Arduino. 2. Tinkercad. 3. Lógica de Programação. 4. Eletrônica.. I. Filho, João Ferreira de Santanna,
orient. II. Título

CDD 004

WILLIAM FERREIRA SANCHES

**A UTILIZAÇÃO DO EMULADOR DE ARDUINO TINKERCAD PARA O ENSINO
DE LÓGICA DE PROGRAMAÇÃO E ELETRÔNICA**

Trabalho de Conclusão de Curso (TCC) apresentado junto à Universidade Federal Rural da Amazônia – UFRA – como requisito parcial para obtenção do título de Licenciado em Computação, em sessão pública realizada no dia 7 de setembro de 2020.

Belém, 07 de setembro de 2020

Banca Avaliadora:

Prof. Dr. João Ferreira de Santanna Filho - Orientador
Universidade Federal Rural da Amazônia - UFRA

Prof. MSc. Alex de Jesus Zissou - Examinador
Universidade Federal Rural da Amazônia - UFRA

Profª. Dra. Deciola Fernandes de Sousa - Examinadora
Universidade Federal Rural da Amazônia - UFRA

AGRADECIMENTOS

Primeiramente gostaria de agradecer a minha pessoa, pois diante de muitas adversidades me fiz forte durante toda a caminhada na graduação.

Agradecer à minha esposa Janaina dos Prazeres, por ser minha companheira e inspiração, seus conselhos e ensinamentos foram fundamentais para o desenvolvimento deste trabalho.

Aos meus pais, Newton Sanches e Nancy Ferreira, por me fornecer as provisões necessárias para que eu pudesse seguir meu objetivo na graduação.

Ao Professor João Santanna, pela orientação e por não medir esforços como coordenador do curso de Licenciatura em Computação, sempre buscando a excelência do curso.

Agradecer a todos os Servidores e Colaboradores da Universidade Federal Rural da Amazônia, em especial os do Instituto Socioambiental e dos Recursos Hídricos - ISARH, no qual tive a oportunidade de fazer parte como estagiário e assim aplicar os conhecimentos adquiridos ao longo do curso, e por fim, agradecer a todos que direta ou indiretamente contribuíram para minha formação.

RESUMO

Este trabalho apresenta o simulador de circuitos *Tinkercad* como tecnologia alternativa à plataforma física do Arduino. O uso do simulador se justifica, pela ausência da placa física do Arduino em Escolas e Universidades, seja por fatores financeiros ou de disponibilidade no mercado, fazendo com que professores deixem de utilizar o Arduino como ferramenta de apoio ao ensino. Assim, o simulador pode suprir essa necessidade, pois oferece os principais componentes eletrônicos e o ambiente de programação totalmente *online* e gratuito. O objetivo da pesquisa foi de identificar os principais componentes disponíveis na plataforma Tinkercad, apresentar experimentos e a utilização da Linguagem Programação C através do Arduino. Foi realizada pesquisa bibliográfica, na qual foram encontradas diversas aplicações do Arduino. Estas provocaram a criação de experimentos para serem projetados dentro e fora do ambiente de simulação. Contudo, o maior desafio é comprovar se realmente o ambiente de simulação pode suprir as necessidades que o Arduino físico e seus componentes propõe solucionar dentro de sala de aula e assim implementar outras formas de aplicação do Arduino e do simulador em conjunto com outras linguagens de programação.

Palavras – Chave: Arduino. Tinkercad. Lógica de Programação.

ABSTRACT

This work presents the Tinkercad circuit simulator as an alternative technology to the Arduino physical platform. The use of the simulator is justified by the absence of the Arduino platform in schools and universities, whether due to financial factors or availability in the market, causing teachers to stop using Arduino as a teaching support tool. Thus the simulator can meet this need, as it offers the main electronic components and the programming environment completely online and free. The objective of the research was to identify the main components available on the Tinkercad platform, to present experiments and the use of the C Programming Language through Arduino. Bibliographic research was carried out, in which several applications of Arduino were found. These caused the creation of experiments to be designed inside and outside the simulation environment. However, the biggest challenge is to verify if the simulation environment can really meet the needs that the physical Arduino and its components proposes to solve within the classroom and thus implement other forms of application of the Arduino and the simulator together with other programming languages.

Keywords: Arduino. Tinkercad. Programming logic. Electronics.

LISTA DE FIGURAS

Figura 1: Lei dos NÓS.....	16
Figura 2: Placa de Arduino Uno.....	17
Figura 3: Ambiente de programação (IDE) Arduino..	18
Figura 4: Arduino UNO do simulador Tinkercad.	20
Figura 5: Diodo Emissor de Luz - LED.	21
Figura 6: Sensor de Temperatura - TMP 36.....	22
Figura 7: Potenciômetro.	22
Figura 8: Micro Servo Motor.	23
Figura 9: Resistor.	23
Figura 10: Tabela dos códigos de cores do Resistor.	24
Figura 11: Micro chave Push Button dip.....	24
Figura 12: Fotorresistor.	25
Figura 13: LED RGB.....	25
Figura 14: Teclado Matricial 4x4.....	26
Figura 15: Tela LCD 16X2.....	26
Figura 16: QR CODE para acessar o site do simulador.	27

SUMÁRIO

1. INTRODUÇÃO	11
1.1. Justificativa	12
1.2. Objetivos	13
1.2.1 Objetivo geral:	13
1.2.2. Objetivo específico:	13
1.3. Metodologia	13
1.4. Organização do trabalho	14
2. REFERENCIAL TEORICO	15
2.1. Eletrônica	15
2.2. Principais leis da Física aplicadas na Eletrônica	15
2.3. Arduino	17
2.4. Linguagem de programação C para Arduino	19
2.5 Componentes eletrônicos	20
2.5.1. Diodo emissor de luz (LED)	21
2.5.2. Sensor de temperatura (tmp36)	22
2.5.3. Potenciômetro	22
2.5.4. Micro servo	23
2.5.5. Resistor	23
2.5.6. Botão	24
2.5.7. Fotorresistor	24
2.5.8. LED RGB	25
2.5.10. Teclado 4X4	26
2.5.11. LCD 16 X 2	26
2.6. Ferramenta de simulação	27
3. EXPERIÊNCIAS DESENVOLVIDAS NO SIMULADOR	28
3.1. Olá mundo	28
3.2. LED com botão	30
3.3. Semáforo	31
3.4 LED RGB	33
3.5. Controlando o servo motor	35
3.6. Fotorresistor com LED	37
3.7. Sensor de temperatura	38
3.8. Controlando o micro servo motor com potenciômetro	41
3.9. Semáforo com cancela	44
3.10. Calculadora	46
4. CONCLUSÃO	50
4.1. Sobre os objetivos do trabalho	50

4.1. Sobre as ferramentas	50
4.2. Sobre a importância do uso de ferramentas nas aulas de Lógica de Programação e Eletrônica.	51
4.3 Limitações	51
4.5 Trabalhos futuros	52
REFERÊNCIAS	53

1. INTRODUÇÃO

A Lógica de Programação é o primeiro passo para o estudante conhecer o processo de criação de um programa de computador. E pode ser definida como o processo de encadear pensamentos, ou seja, a partir desse processo o desenvolvedor definirá o funcionamento do programa e a forma que ele chegará ao seu objetivo (CARVALHO, 2010).

Além de ser utilizada no desenvolvimento de software a Lógica de programação é usada para escrever programas que controlam o funcionamento de componentes eletrônicos em carros, aviões, robôs e é usada também na construção de experimentos com a placa didática do Arduino.

Neste sentido, a Lógica quando aplicada no controle e operação de *hardware*, é importante para entender o funcionamento de um circuito integrado e os conceitos de portas Lógicas, essenciais para verificar a passagem ou não de corrente elétrica no circuito (PUGA; RISSETI, 2004).

Nesse contexto, a plataforma de hardware livre, Arduino, criada em 2005 na Itália, é acessível a estudantes e iniciantes em programação e/ou em Eletrônica possam desenvolver projetos de prototipagem de hardware e software, além de ser uma alternativa para prática da Lógica de programação (ALVES et al., 2013).

Um dos projetos que direciona a utilização da plataforma Arduino em sala de aula, foi publicado em 2018 no livro, “Práticas na Sala de Aula,” se chama “PREPARANDO OS ALUNOS PARA OS AVANÇOS TECNOLÓGICOS UTILIZANDO A FERRAMENTA ARDUÍNO EM SALA DE AULA.” de autoria do Professor Matheus Felipe Trevisan, que diz em seu resumo que o Arduino proporciona a interação direta e indireta do aluno com diversos experimentos da área da Eletrônica e da Programação. Pois, segundo Trevisan (2018) as experiências são feitas em sala de aula com poucos recursos e em pouco espaço.

No entanto, adquirir uma placa de Arduino ou um kit para iniciantes, pode ser um desafio para muitos estudantes e professores, seja pelo fator financeiro ou pela disponibilidade em lojas de produtos eletrônicos em algumas cidades.

Diante disso, este trabalho propõe a professores e estudantes a utilização do simulador de circuitos *Tinkercad* como alternativa a placa física do Arduino, pois além de ser uma ferramenta gratuita que roda diretamente no navegador, este simulador possui o ambiente de programação e os principais componentes eletrônicos, como sensores, resistores e motores.

Além de abordar o conceito de Eletrônica e os componentes eletrônicos disponíveis no simulador, será apresentado, brevemente, neste trabalho as principais leis da física aplicadas na

Eletrônica, pois mesmo sendo um ambiente de simulação, é importante que o estudante e o Professor conheçam esses conceitos básicos para que os projetos realizados na plataforma, sejam seguros e que o mal uso da eletricidade não comprometam os circuitos, a fim de evitar acidentes quando aplicados fora do simulador.

Para demonstrar na prática a utilização do simulador, foram realizados um total de 10 experimentos que podem ser alterados e projetados com a placa física do Arduino, a lista completa com as simulações, códigos e circuitos elétricos, estão todos descritos no capítulo 3.

1.1. Justificativa

A principal motivação para sustentar o presente trabalho, reside em alternativas que possibilitam a aprendizagem da Lógica de Programação e Eletrônica, suas diversas aplicações no cotidiano e nas metodologias empregadas atualmente para desenvolver o processo de ensino.

Neste sentido, professores podem utilizar computadores, placas com microcontrolador e até mesmo papel e caneta como ferramentas de auxílio na construção de sequências Lógicas que irão contribuir no processo de aprendizagem de seus alunos.

Muitos educadores deixam de utilizar o Arduino como base para o ensino da Lógica de Programação e Eletrônica, pois mesmo que o preço de um microcontrolador como o Arduino seja acessível, nem todas as escolas e universidades, dispõem desses equipamentos para oferecer aos seus alunos, visto que sem o microcontrolador e os componentes eletrônicos necessários, fica inviável realizar atividades em sala de aula.

Os kits de Arduino podem ser facilmente encontrados em lojas e sites especializados em produtos de Eletrônica. Os kits Iniciante, Intermediário e Avançado disponíveis no comércio eletrônico¹, apontam um valor médio de R\$ 353,94 (Trezentos e cinquenta e três reais e noventa e quatro centavos).

O custo para uma escola ou universidade montar um laboratório mínimo com Arduino contendo 15 kits Iniciante, 15 Kits Intermediário e 15 Kits Avançado, custaria R\$ 15.927,3 (Quinze mil, novecentos e vinte sete reais e três centavos). Isso sem contar o custo de equipamentos como, Multímetros e Osciloscópios.

¹ Plataforma Mercado Livre, pesquisa realizada em 8 de abril de 2020, disponível em:
Kit Iniciante: <https://bitlybr.com/vb5GGd>
Kit Intermediário: <https://bitlybr.com/UmoLqFDy>
Kit Avançado: <https://bitlybr.com/cnioe>

Assim o presente trabalho, busca apresentar como tecnologia alternativa para o ensino de Lógica de Programação e Eletrônica, o emulador da placa de Arduino, *Tinkercad*. Que neste trabalho não será destinado para ensinar uma linguagem de programação específica como Java, Python ou SQL, mas sim a Lógica em conjunto com a Eletrônica, para que assim o estudante tenha mais facilidade em aprender a sintaxe de diversas linguagem de programação (GARLET, 2016).

Para isso, a plataforma *Tinkercad* (<https://www.tinkercad.com/>) da empresa Autodesk inc. Se destaca como alternativa, por possibilitar a criação de experimentos utilizando os mesmos componentes físicos do Arduino e o ambiente de programação direto no navegador.

1.2. Objetivos

1.2.1 Objetivo geral:

Apresentar o emulador de Arduino Tinkercad como tecnologia alternativa no processo de ensino de Eletrônica e Lógica de Programação.

1.2.2. Objetivo específico:

- Identificar os principais componentes disponíveis na plataforma Tinkercad;
- Apresentar experimentos utilizando o emulador de Arduino e seus componentes básicos para o ensino da Eletrônica e Lógica de Programação;
- Apresentar a utilização da Linguagem Programação C através do Arduino para iniciar os estudos com a Lógica de programação.

1.3. Metodologia

O conceito de Método científico, consiste em realizar a aplicação de procedimentos e técnicas que quando empregados, possibilitam a construção do conhecimento, com o propósito de comprovar sua veracidade e utilidade nos diversos âmbitos da sociedade (PRODANOV; FREITAS, 2013).

Dentre as diversas Metodologias de pesquisa, existem, porém, pesquisas científicas que se baseiam exclusivamente na pesquisa bibliográfica, priorizando a procura de referências teóricas publicadas com o objetivo de coletar informações ou conhecimentos prévios sobre o problema a respeito do qual se procura a resposta (FONSECA, 2002).

O capítulo Introdutório e o de fundamentação teórica foi elaborado com base em pesquisa bibliográfica usando livros e artigos pesquisados sobre o assunto.

O primeiro objetivo específico, identificar os principais componentes disponíveis na plataforma Tinkercad, foi realizado usando a metodologia de pesquisa exploratória. A pesquisa exploratória tem como propósito proporcionar maior familiaridade com o objetivo a ser cumprido ou o problema de pesquisa, com vistas a torná-lo mais explícito ou a construir hipóteses (GIL, 2010).

Nesse sentido a ferramenta *Tinkercad* foi utilizada, explorada, e estudada para verificar que componentes estavam disponíveis para uso em simulações de circuitos eletrônicos.

Para realizar o segundo objetivo específico que consiste em apresentar experimentos utilizando o emulador de Arduino e seus componentes básicos foram utilizados em conjunto tanto a metodologia de pesquisa bibliográfica, visando selecionar um grupo de circuitos eletrônicos a partir da bibliografia da área bem como a pesquisa exploratória, que consiste em utilizar o *Tinkercad* e sua biblioteca de componentes de maneira a reproduzir tais experimentos na ferramenta e a partir disso simular o comportamento dos circuitos.

Após a seleção de circuitos eletrônicos para as experiências, alguns foram remodelados conforme a capacidade de simulação da ferramenta utilizada na pesquisa.

Finalmente para o terceiro objetivo específico foi utilizado a pesquisa bibliográfica para estudar o subconjunto da linguagem C utilizada no Arduino, para isso foram utilizadas bibliografias sobre a placa de experimentação e a linguagem que ela utiliza para programação.

1.4. Organização do trabalho

O trabalho está organizado em quatro capítulos, o primeiro capítulo, faz a introdução ao assunto a ser explorado, os objetivos e metodologia utilizada na pesquisa. O segundo capítulo, aborda o referencial teórico sobre os conceitos básicos de Eletrônica, as principais leis da física aplicada na Eletrônica, os componentes eletrônicos utilizados nos experimentos e o ambiente de simulação.

O terceiro capítulo apresenta os experimentos feitos no simulador de Arduino, juntamente com seus respectivos códigos fontes e QR CODE para acessar a simulação. E finalizando, o quarto capítulo apresenta as considerações finais, sobre os objetivos do trabalho, as ferramentas utilizadas e a importância de se utilizar o simulador nas aulas de Lógica de programação bem como as limitações encontradas no decorrer da pesquisa.

2. REFERENCIAL TEORICO

Este capítulo é dedicado aos conceitos básicos de Eletrônica, as leis da Física aplicadas na Eletrônica, plataforma Arduino, os principais componentes eletrônicos utilizados nos experimentos e o simulador *Tinkercad*.

2.1. Eletrônica

Para realizar as simulações no Tinkercad, é necessário conhecer os conceitos básicos de Eletrônica, bem como o funcionamento e aplicação dos principais componentes elétricos utilizados neste trabalho, tais como: Resistores, Motores, Sensores e LEDs. Além do funcionamento da placa didática Arduino.

Ao empregar componentes eletrônicos no desenvolvimento de experimentos, mesmo dentro dos simuladores, a fim de evitar acidentes e o desperdício de componentes na execução do projeto final, é importante compreender como a eletricidade percorre os circuitos elétricos. Para isso, utiliza-se a Eletrônica, pois ela é o campo da ciência e da engenharia que estuda a forma como é controlada a energia elétrica através de dispositivos e meios condutores ou semicondutores (RODRIGUES et al., 2013).

Nesse sentido, os conceitos básicos e essenciais da física aplicados na Eletrônica, abordados em situações como a proposta na pesquisa, utilizam-se das leis Ohm, Kirchhoff e Ampère, estas serão apresentadas no tópico seguinte.

2.2. Principais leis da Física aplicadas na Eletrônica

A partir dos conhecimentos em Física, mais especificamente sobre as leis de Kirchhoff criadas e desenvolvidas pelo físico alemão Gustav Robert Kirchhoff (1824-1887). Lei Ohm, criada pelo o físico alemão Georg Simon Ohm (1789-1854). E a lei de Ampère com André-Marie Ampère, podemos entender o funcionamento da eletricidade dentro e fora dos circuitos.

As leis de Kirchhoff foram criadas para resolver problemas de circuitos elétricos mais complexos e problemas que podem ser encontrados em circuitos em série ou em paralelo (ALMEIDA et al., 2011).

A primeira lei de Kirchhoff, também conhecida como lei dos nós, determina que em qualquer nó a soma das correntes que o deixam é igual a soma das correntes que chegam até ele (ALMEIDA et al., 2011). Portanto, a lei dos nós é válida para qualquer nó e é representada pela fórmula: $\sum I = 0$.

Conforme o exemplo apresentado no esquema abaixo, onde o circuito está assumindo que todas as ramificações desenhadas têm um caminho para voltarem à fonte de alimentação, com a corrente total (I_{TOTAL}) que será igual à soma de I_1 , I_2 , I_3 (TORRES, 2012).

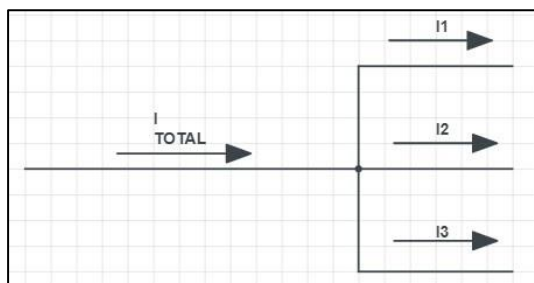


Figura 1: Lei dos Nós.

A segunda lei de Kirchhoff, também conhecida como lei das malhas, ou lei das tensões, determina que a soma das diferenças de potencial de uma malha de um circuito é sempre nula e é representada pela fórmula: $\sum V = 0$.

A lei das malhas é baseada na natureza conservativa das forças eletrostáticas. Suponha que você percorra uma dada malha, medindo sucessivamente todas as diferenças de potencial através dos elementos do circuito. Ao retornar ao ponto de partida, você deve verificar que a soma algébrica de todas as diferenças de potencial é igual a zero; caso contrário, você não poderia associar um potencial definido para o referido ponto. (HUGH D. YOUNG; FREEDMAN, 2009,p.173).

A lei de Ohm determina que a tensão em um resistor é propriamente equivalente à corrente que flui através dele. Deste modo, um componente do circuito cuja característica elétrica é resistiva, é chamado de resistor (SILVA, 2015).

Para descobrir a corrente um circuito utiliza-se a equação descoberta através da lei de Ohm representada pela formula: $V = R \cdot I$.

Onde V é a diferença de potencial elétrico medido em volts (V), I corresponde a intensidade da corrente elétrica medida em ampere (A) e R é a resistência elétrica medida em Ohms (Ω).

A equação relacionada à lei de Ohm também pode ser aplicada sem qualquer problema em sistemas de tensão alternada, mas o valor da tensão deverá ser dado em seu valor eficaz (RMS), ou seja, em uma medida estatística da magnitude de uma quantidade variável e não em seu valor de pico (TORRES, 2012).

Vale lembrar que as letras *V* e *I* que representam respectivamente tensão e corrente, quando grafadas em maiúsculas representam valores de corrente contínua e quando apresentados em minúsculas representam valores extraídos de circuitos de corrente alternada (TORRES, 2012).

A lei de ampere foi desenvolvida para determinar o campo magnético produzido por uma distribuição de correntes com simetria elevada (HUGHES, YOUNG; FREEDMAN, 2009, p. 257).

A equação da lei de ampère, pode além de deduzir casos especiais de fios retilíneos longos e paralelos, a fórmula é válida também para todos os percursos e condutores, independente do percurso escolhido (YOUNG e FREEDMAN, 2009, p.257).

2.3. Arduino

O Arduino é um projeto que surgiu na Itália no ano de 2005, com o intuito de criar um dispositivo para controlar e criar protótipos com baixo custo, tornando assim a plataforma mais acessível e disponível no mercado (SILVA et al., 2014).

O Arduino consiste em uma plataforma *Open-source* baseada em *hardware e software* livres, direcionada para projetos de automação e robótica (SILVA et al., 2014).

A plataforma física do Arduino é composta por uma placa Eletrônica que é constituída por circuitos integrados.

O principal componente é um microprocessador do tipo AVR da Atmel®, com a função de receber e entregar o fluxo de informações de maneira controlada por uso de software. A plataforma e arquivos são licenciados pela *Creative Commons*, uma organização não governamental que permite o uso pessoal, bem como comercial e obras derivadas, desde que seja dado crédito ao Arduino e a liberação de seus projetos sob a mesma licença (SILVA et al., 2014).



Figura 2: Placa de Arduino UNO.

Seu ambiente de desenvolvimento integrado ao hardware, conhecido como (IDE – *Integrated Development Environment*) possui uma interface que tem o papel de gerar programas

(denominados de sketches) que devem ser enviados para placa Eletrônica (ALVES et al., 2013). Para ter acesso ao ambiente de desenvolvimento o usuário pode baixar o arquivo de instalação disponível no site: [site https://www.Arduino.cc/en/Main/Software](https://www.Arduino.cc/en/Main/Software).

Desenvolvido para ser multiplataforma, é um programa escrito em Java e é capaz de depurar, compilar e enviar o binário compilado para o microcontrolador da placa Eletrônica Arduino (FRIZZARIN, 2016).

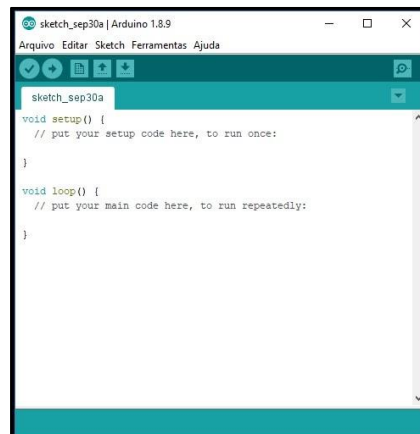


Figura 3: Ambiente de programação (IDE) Arduino.

A linguagem de programação utilizada para controlar o Arduino é uma variação da Linguagem C, baseada na linguagem Wiring (FRIZZARIN, 2016). Os conceitos e o funcionamento da linguagem de programação C, serão abordados no subtópico 2.4.

Por se tratar de uma plataforma acessível, é comumente utilizada para pequenos projetos de automação residencial e de robótica, quando utilizada em ambiente educacional o Arduino oferece inúmeras possibilidades de aprendizagem, uma delas é a utilização da plataforma para o ensino da Lógica de programação.

Conforme MARTINAZZO et al., (2014) o sistema Arduino em conjunto com outros componentes eletrônicos, possibilita a aplicação para fins didáticos em escolas e universidades para favorecer o aprendizado do aluno.

Embora seja uma ferramenta de baixo custo, é pouca utilizada em sala de aula por professores, um dos fatores que contribuem para esse cenário é a dificuldade de se ter todos os componentes necessários para construir projetos complexos que exijam a disponibilidade de peças específicas.

Como forma alternativa a placa física do Arduino, existem simuladores online que emulam a placa Eletrônica, seus componentes e o ambiente de programação (IDE – *Integrated Development Environment*).

Para este trabalho, utilizaremos o *Tinkercad*, simulador desenvolvido pela empresa Autodesk, que tem por finalidade a criação e a alteração de projetos de circuitos elétricos. Suas principais características serão apresentadas no subtópico 2.5 ferramentas de simulação.

2.4. Linguagem de programação C para Arduino

A linguagem de programação utilizada para escrever códigos que irão controlar o funcionamento da plataforma Arduino é denominada *Wiring*, e para entender a sintaxe dessa linguagem, é importante estudar os conceitos da linguagem de programação C/C++ (RIBEIRO et al., 2017).

A linguagem C é a base para a programação para Arduino. Apesar de ter alguns comandos específicos para o microcontrolador, é basicamente a mesma utilizada em programação para computadores. (FRIZZARIN, 2016, p.54).

Neste sentido, é fundamental entender a sintaxe e a estrutura básica da linguagem C para programação do Arduino. Como toda linguagem de programação possui suas particularidades, veremos brevemente os detalhes e estrutura básica da linguagem C, aplicada no Arduino (FRIZZARIN, 2016).

Os detalhes básicos são: O ponto e vírgula (;) empregados para delimitar as linhas de comandos, é importante se atentar para suas exceções, pois no Arduino não se utiliza ponto e vírgula para declarar funções e nem estruturas de controle ou repetição; A chave { }, indica o início e o término de blocos de comandos, além de, delimitar a porção de código para uma função de estrutura de seleção ou repetição; Os comentários são implementados a partir de duas barras / /, os comentários são utilizados quando o programador deseja manter seu código organizado e de fácil entendimento conforme o exemplo abaixo (FRIZZARIN, 2016).

```
void setup() { //Exemplo de comentário
}
void loop() { //Exemplo de comentário
}
```

A estrutura básica compreende a função setup() e loop(), a função setup() determina o que será executado ao ligar o Arduino, portanto é executada uma única vez, após executar a função setup(), a função loop() é acionada em seguida, para executar o bloco de código inseridos nela, do início ao fim, esta função se repete enquanto o Arduino estiver ligado (FRIZZARIN, 2016). No exemplo abaixo a função setup () inicia o pino digital 1 como saída e dentro da função loop () o ciclo de ligar e desligar o pino digital 1 se repete dentro do intervalo de 1 segundo enquanto o Arduino estiver ligado.

```

void setup()
{
  pinMode(1, OUTPUT);
}

void loop() {
  digitalWrite(1, HIGH);
  delay(1000);
  digitalWrite(1, LOW);
  delay(1000);
}

```

Para o desenvolvimento dos experimentos apresentados neste trabalho, foi utilizada a Linguagem de Programação C, baseada no Arduino, e não a Programação em Blocos, portanto, conhecer a estrutura básica desta linguagem é fundamental para iniciar os estudos e entender a estrutura Lógica não só dos projetos feitos no simulador *Tinkercad*, mas também para a criação de jogos e aplicativos desenvolvidos em outras Linguagens de Programação.

2.5 Componentes eletrônicos

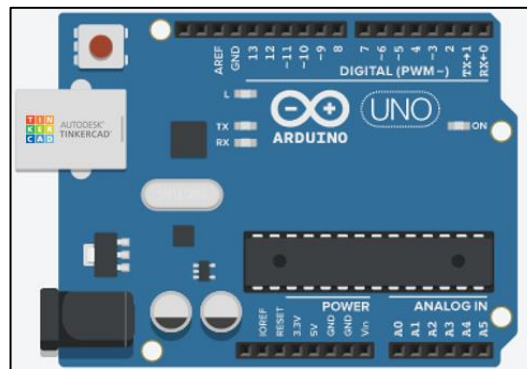


Figura 4: Arduino UNO do simulador Tinkercad.

Todos os experimentos deste trabalho, utilizam especificamente a placa didática Arduino Uno R3, como base para o seu funcionamento, físico, lógico e seu ambiente de desenvolvimento emulado na plataforma *Tinkercad*.

O Arduino Uno R3 conta com um microcontrolador ATmega328, possui 14 pinos de entrada/ saída digital, sendo que 6 dessas podem ser utilizadas como saídas (PWM – *Pulse Width Modulation*) ou Modulação de Largura de Pulso.

Ainda conta com 6 entradas analógicas, um cristal oscilador de 16MHz, uma conexão USB para alimentação de 5V que também é utilizada para o carregamento dos programas na placa.

2.5.1. Diodo emissor de luz (LED)

O diodo emissor de luz, comumente conhecido como LED (*Light Emitting Diode*), possui a função de emitir luz visível nas cores amarela, verde, vermelha, laranja ou azul. Além de luz infravermelho (RODRIGUES et al., 2013).

Os LEDs possuem polaridade, ou seja, dispõem de terminais positivo e negativo, no qual o polo negativo é chamado de Cátodo e polo positivo é conhecido como Ânodo, para facilitar a identificação entre Ânodo (+) e Cátodo (-). A estrutura física dos terminais possui peculiaridades, onde o terminal positivo é mais longo e o negativo mais curto.

Por se tratar de diodos, para realizar a passagem de corrente elétrica, deve-se atentar a polaridade positiva (+) e negativa (-) do LED, conforme a figura abaixo, em que devem ser conectadas aos respectivos terminais da fonte de energia, caso isso não esteja de acordo o LED não acenderá.

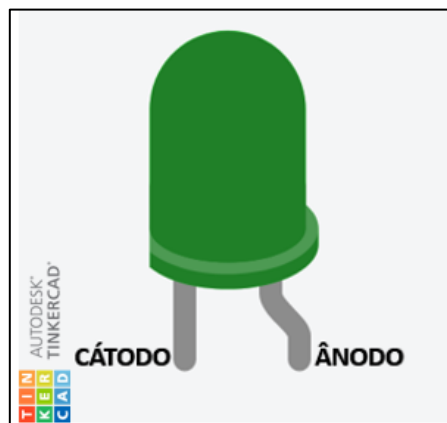


Figura 5: Diodo Emissor de Luz - LED.

2.5.2. Sensor de temperatura (tmp36)

Este componente tem a finalidade de medir a temperatura do ambiente utilizando o Arduino, o sensor TMP36, tem a aparência de um transistor de 3 terminais conforme a figura abaixo, sua alimentação funciona na faixa de 2.7V a 5.5VDC, não necessita de ajustes externos para seu funcionamento, ele também fornece uma saída de tensão diretamente proporcional a temperatura em graus celsius, com precisão de $\pm 1^\circ\text{C}$ na faixa 1° a 25°C e $\pm 2^\circ$ para a faixa de -40° a 125°C (OOMLOUT.COM, 2020).

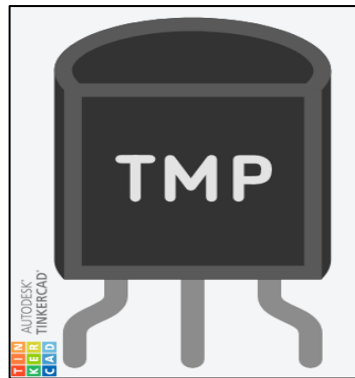


Figura 6: Sensor de Temperatura - TMP 36.

2.5.3. Potenciômetro

Por ter sua resistência variada, semelhante aos resistores e suportar valores de resistência e potência máxima que podem dissipar, os potenciômetros são comumente utilizados para controlar volumes em aparelhos de som e a velocidade em ventiladores. Combinado a plataforma Arduino, o potenciômetro pode ser utilizado em diversos projetos de robótica e automação residencial, controle de movimentos e velocidade em motores (FREIRE; FILHO, 2004)

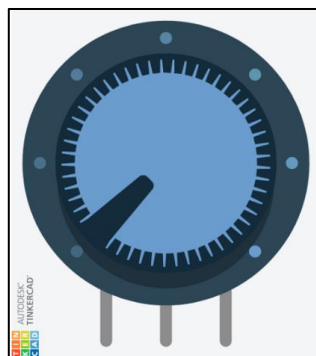


Figura 7: Potenciômetro.

2.5.4. Micro servo

O Micro Servo-Motor é um tipo de motor CC que possui três terminais, conforme a figura abaixo, sendo dois deles para alimentação Positivo (+) e Negativo (-) e mais um para o controle de posição do motor, sua principal característica é um conjunto de engrenagens, limitadores de fim de curso, um potenciômetro para realimentar a posição do motor e um circuito integrado para o controle da posição do eixo do motor (FREIRE; FILHO, 2004).

O motor micro servo é utilizado em projetos de robótica, no qual exerce a função de movimentar braços, pernas e mãos dos robôs. Além de, ser empregado em experimentos de automodelismo, com objetivo de realizar os movimentos das rodas dianteiras dos carrinhos (FREIRE; FILHO, 2004).

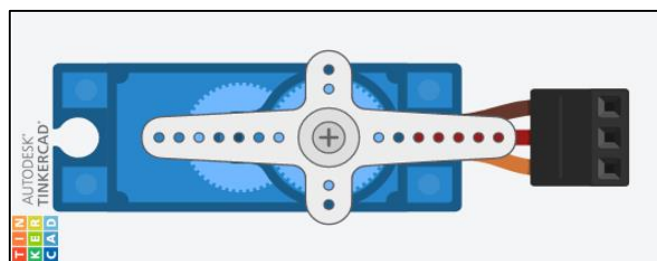


Figura 8: Micro Servo Motor.

2.5.5. Resistor

Conhecido como resistência, o resistor é um dispositivo eletrônico utilizado em quase todos os projetos que envolve eletrônica, tem a finalidade de transfigurar energia elétrica em energia térmica, assim reduzindo a corrente elétrica em um circuito.

Por se tratar de um dispositivo não polarizado, os resistores podem ser instalados em qualquer sentido no circuito elétrico, sem necessidade de se preocupar com os polos negativo e positivo (OLIVEIRA, 2018).

Para determinar sua resistência, os resistores possuem 4 faixas de cores que determinam o valor da resistência, dispensando a necessidade de medição com aparelhos medidores. Esse código de cores está disposto em uma tabela conforme a imagem.

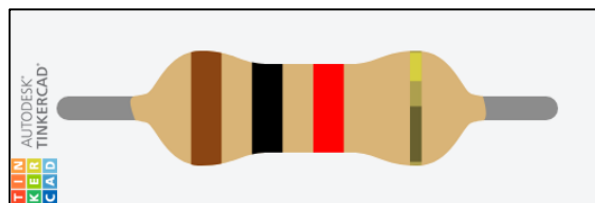
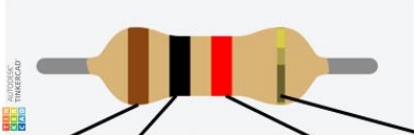


Figura 9: Resistor.



COR	1ª FAIXA	2ª FAIXA	3ª FAIXA	MULTIPLICADOR	TOLERÂNCIA
PRETO	0	0	0	x1	
MARRON	1	1	1	x10	1%
VERMELHO	2	2	2	x100	2%
LARANJA	3	3	3	x1k	
AMARELO	4	4	4	x10k	
VERDE	5	5	5	x100k	.5%
AZUL	6	6	6	x1M	.25%
LILAS	7	7	7	x10m	.1%
CINZA	8	8	8		.05%
BRANCO	9	9	9		
DOURADO				x.1	.5%
PRATA				x0.1	.10%

Figura 10: Tabela dos códigos de cores do Resistor.

2.5.6. Botão

Este botão é do tipo micro chave *Push Button dip*, onde o interruptor momentâneo é um componente que conecta dois pontos de um circuito ao pressioná-lo (ATOMIC; PIMENTA, 2015). Aplicado em projetos de prototipagem eletrônica, a chave tátil ou simplesmente botão, representado pela figura abaixo é uma chave que exerce o papel de um interruptor no circuito elétrico, ou seja, só conduz energia elétrica quando pressionado.

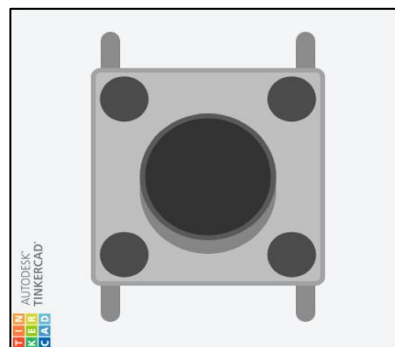


Figura 11: Micro chave *Push Button dip*.

2.5.7. Fotorresistor

Conhecido pela sigla, LDR – *Light Dependent Resistor*, o fotorresistor é um tipo de resistor sensível a luz, que é capaz de variar sua resistência conforme a intensidade da luz captada por ele, ou seja, quando o ambiente está iluminado, V_{out} é igual a aproximadamente 5 V; no caso de pouca iluminação, V_{out} é igual a aproximadamente 0 V (FREIRE; FILHO, 2004).

O LDR representado na figura abaixo, possui um grande valor de resistência (alguns $M\Omega$) na ausência de luz e baixo valor de resistência na presença de luz (alguns $k\Omega$) (FREIRE; FILHO, 2004).

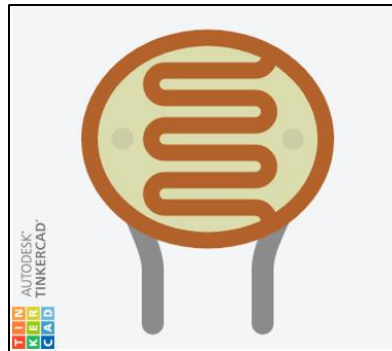


Figura 11: Fotoresistor.

2.5.8. LED RGB

Os LEDs RGB conforme figura abaixo, são chamados assim, por possuírem a capacidade de emitir uma infinidade de cores, graças à junção de três LEDs; Vermelho (R de *Red*); Verde (G de *Green*) e azul (B de *Blue*) (FÉLIX et al., [s.d.]).

Os LEDs RGB possuem duas variações, que são denominados como, Anodo Comum, onde os terminais das cores Vermelha, Verde e Azul são conectados ao terminal negativo da fonte de energia, já o Catodo Comum os terminais das cores Vermelha, Verde e Azul são conectado no terminal positivo da fonte de energia (FÉLIX et al., [s.d.]).

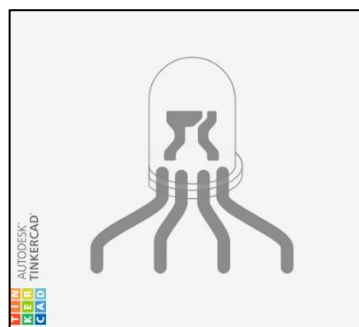


Figura 13: LED RGB.

2.5.10. Teclado 4X4

Conhecido como teclado de membrana C, representado pela figura abaixo, o teclado 4x4 é utilizado quando o projeto exige uma interação entre o homem e a máquina para inserção de comandos e informações. Seu circuito possui Linhas e Colunas que assim formam uma matriz (CUSTÓDIO, 2015).

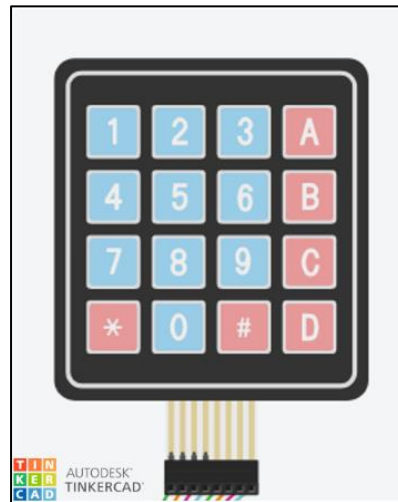


Figura 14: Teclado Matricial 4x4.

2.5.11. LCD 16 X 2

Aplicado quando o experimento exige um acompanhamento dos dados inseridos, o display 16x2 representado pela figura abaixo, é como todo LCD (*liquid 26rystal display*) consiste em um líquido polarizador de luz, eletricamente controlado, que se encontra comprimido dentro de celas entre duas lâminas transparentes polarizadoras. Os eixos polarizadores das duas lâminas estão alinhados perpendicularmente entre si. Cada cela é provida de contatos elétricos que possibilitam um campo elétrico ser aplicado ao líquido no interior (ATOMIC; PIMENTA, 2015).



Figura 15: Tela LCD 16X2

2.6. Ferramenta de simulação

No contexto educacional, as simulações computacionais são concebidas para serem estimulantes e interativas, para conectar conteúdos em estudo com o mundo real, para fornecer múltiplas representações, permitindo experimentações e verificações (CERCONI; MARTINS, 2014).

Neste sentido, será utilizado como alternativa a placa física do Arduino, o simulador de circuitos *Tinkercad*, é uma ferramenta online e gratuita, desenvolvida pela empresa *Autodesk Inc.* Onde o objetivo é emular os componentes eletrônicos e o ambiente de programação diretamente no computador. Além disso, a plataforma possui o emulador do Arduino que permite criar e alterar projetos de prototipagem de *hardware e software*.

Essa alternativa se faz necessária, pois diante das possíveis dificuldades de se ter um kit de Arduino contendo os mais diversos componentes, os simuladores computacionais oferecem recursos tecnológicos que podem auxiliar em diversas áreas.

Uma delas é a utilização no ensino da Lógica de programação e Eletrônica, o qual é abordada nesta pesquisa. Os simuladores podem auxiliar os professores a provocar, favorecer e orientar situações controladas de ensino e aprendizagem, pedagogicamente interessantes para o nível de ensino que desejar aplicar (TORRES et al., 2012).

Até o momento da publicação deste trabalho a plataforma fornece o simulador de circuitos elétricos, além de, emular a placa de Arduino modelo Uno R3 e o seu ambiente de desenvolvimento. Para acessar a ferramenta, o usuário pode inserir o endereço: <https://www.tinkercad.com/>, diretamente no navegador ou acessar pelo *QR CODE* abaixo.



Figura 16: *QR CODE* para acessar o site do simulador.

3. EXPERIÊNCIAS DESENVOLVIDAS NO SIMULADOR

Os experimentos foram realizados no 2º semestre do ano de 2019, diretamente no simulador *Tinkercad* e tem por objetivo demonstrar como a Lógica de Programação aplicada em conjunto a plataforma Arduino pode ser mais atrativa e significativa para o educando.

E para auxiliar professores, alunos ou quem tenha interesse em iniciar seus estudos em Arduino, os experimentos aqui apresentados possuem o desenho do circuito elétrico, lista de componentes utilizados, código fonte, endereço URL e QR CODE da simulação que possibilita o acesso direto à plataforma do simulador online *Tinkercad*.

3.1. Olá mundo

APRESENTAÇÃO DA EXPERIÊNCIA:

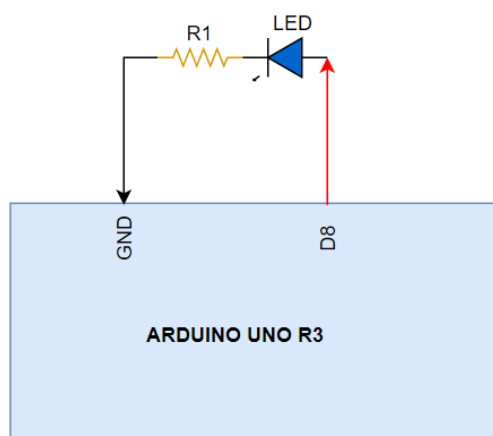
Geralmente o primeiro código na maioria das linguagens de programação é o “Olá Mundo”, já no Arduino o primeiro passo é fazer um LED piscar, portanto neste experimento o objetivo é utilizar o simulador de Arduino para piscar um LED.

MATERIAL UTILIZADO:

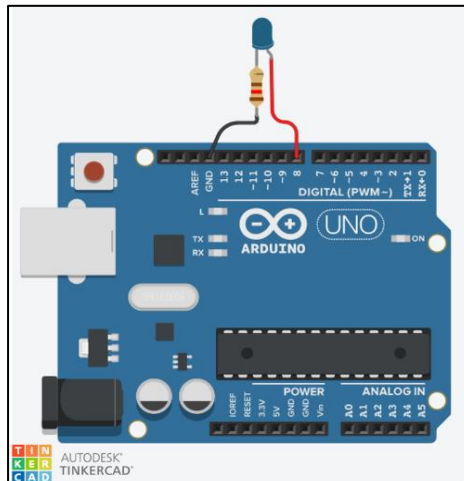
- 1 - Arduino Uno R3
- 1 - LED azul
- 1 - Resistor de 115 Ω

CIRCUITO ELÉTRICO DA EXPERIÊNCIA:

Primeiramente o código através da função setup configura o pino D8 como saída para que seja usado no experimento. Quando o código é executado no Arduino o pino D8 assume a tensão de 5 volts , uma corrente passa a fluir no circuito entrando pelo catodo do LED e saindo pelo anodo , essa passagem de corrente faz o LED acender , a seguir a corrente passa pelo resistor 115 Ω até chegar ao pino terra (GND) fechando o circuito.



SIMULAÇÃO:



URL: <https://www.tinkercad.com/embed/5zSe57HjeE8?editbtn=1>

CÓDIGO:

```
void setup() // Define a função a ser executada
{
    pinMode(8, OUTPUT); // Função, definir o pino 8 como saída
}

void loop() // Executa o loop enquanto o Arduino
            // estiver ligado.
{
    digitalWrite(8, HIGH); //define o Led do pino 8 como ligado
    delay(1000); // Aguarda um segundo (1s = 1000ms)
    digitalWrite(8, LOW); //define o Led do pino 8 como
    desligado
    delay(1000); // Aguarda um segundo (1s = 1000ms)
}
}
```

3.2. LED com botão

APRESENTAÇÃO DA EXPERIÊNCIA:

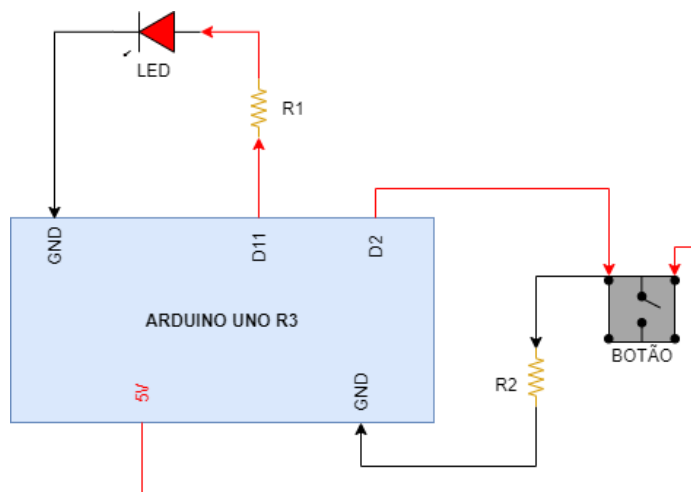
O experimento consiste em utilizar o Arduino para acender um LED quando o botão for pressionado.

MATERIAL UTILIZADO:

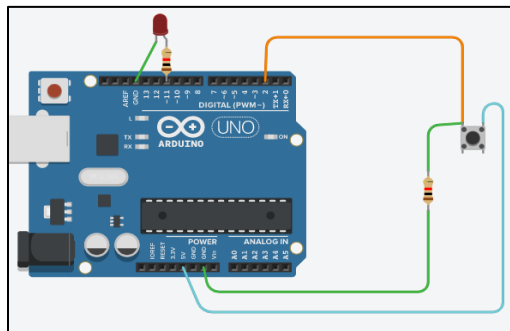
- 1 - Arduino Uno R3
- 1 - LED Vermelho
- 2 - Resistor de 1 k Ω
- 1 - Botão

CIRCUITO ELÉTRICO DA EXPERIÊNCIA:

De início a função setup está programada para designar o pino digital D11 como saída de tensão 5v para o LED, pino digital D2 como entrada de sinal (comando) para o botão e o pino de 5v para alimentação do botão. Quando o código é executado no Arduino, o pino de 5v alimenta o circuito do botão e quando pressionado pelo usuário, o botão envia o comando através do pino digital D2 para que seja acionado o LED e após isso a tensão flui pelo resistor de 1 k Ω conectado à terra do botão, fechando o circuito do mesmo. Em seguida uma corrente de 5v flui através do pino digital D11 passando pelo resistor de 1 k Ω e saindo pelo pino GND, este processo se repete enquanto o Arduino estiver ligado e se o botão for pressionado.



SIMULAÇÃO:



URL: <https://www.tinkercad.com/embed/1Vh6ogjjY0t?editbtn=1>

CÓDIGO:

```
void setup() // Define a função a ser executada
{
  pinMode(2, INPUT); // Função, definir o pino D2 como entrada
  pinMode(11, OUTPUT); // Função, definir o pino 11 como saída
}
void loop() // método Loop que será executado enquanto o
Arduino estiver ligado
{
  if (digitalRead(2) == 1) { // lê o valor do pino digital 2
e se for igual a 1(ligado) o led acende
    digitalWrite(11, HIGH); // liga o pino digital 11
  } else { // se não
    digitalWrite(11, LOW); // desliga o pino digital 11
  }
  delay(10); //aguardar 10s
}
```

3.3. Semáforo**APRESENTAÇÃO DA EXPERIÊNCIA:**

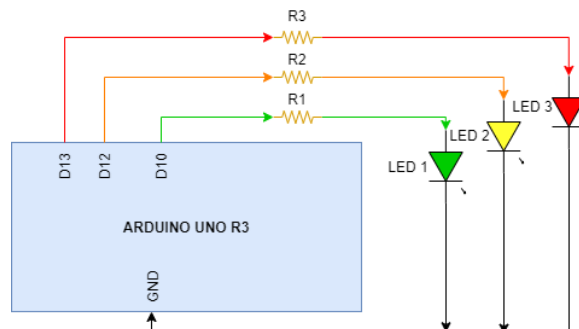
O experimento consiste em utilizar o Arduino e LEDs nas cores amarelo, verde e vermelho para simular os semáforos de trânsito.

MATERIAL UTILIZADO:

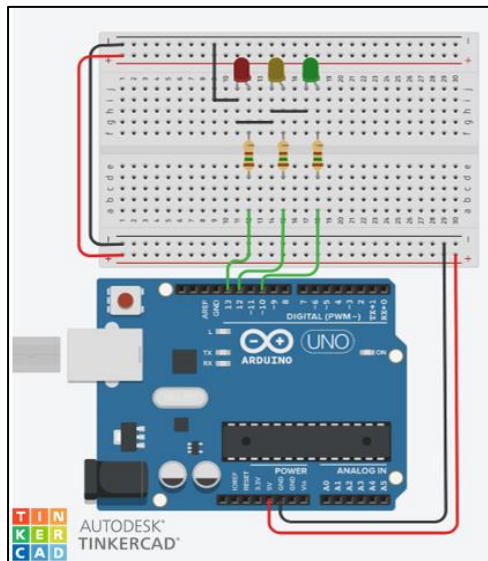
- 1 - Arduino Uno R3
- 1 - LED Vermelho
- 1 - LED Amarelo
- 1 - LED Verde
- 3 - Resistores de 150 Ω

CIRCUITO ELÉTRICO DA EXPERIÊNCIA:

Primeiramente os pinos digitais D10, D12 e D13 estão configurados para controlar os LEDs respectivamente, LED verde, LED amarelo e o vermelho. Após executar a função setup o programa inicia carregando uma tensão de 5v que passa pelo resistor de 150 Ω até chegar ao Ânodo do LED vermelho, após 2 segundos a corrente flui pelo Cátodo do LED até chegar ao pino GND e o mesmo se apaga, este processo se repete pelos circuitos dos LEDs amarelo e verde, enquanto o Arduino estiver ligado.



SIMULAÇÃO:



URL: <https://www.tinkercad.com/embed/llvCaQRDT6x?editbtn=1>

CÓDIGO:

```
void setup() // Define a função a ser executada
{
  pinMode(13, OUTPUT); //pino digital 13 como saída de dados
  pinMode(10, OUTPUT); // pino digital 10 como saída de dados
  pinMode(12, OUTPUT); // pino digital 12 como saída de dados
}

void loop() // método Loop que será executado enquanto o
Arduíno estiver ligado

{
  digitalWrite(13, HIGH); // acende led vermelho (pino 13)
  delay(2000); // aguarda 2 segundos
  digitalWrite(13, LOW); // apaga o led vermelho (pino 13)

  digitalWrite(10, HIGH); // acende o led verde (pino 10)
  delay(3000); // aguarda 3 segundos
  digitalWrite(10, LOW); // apaga led verde (pino 10)

  digitalWrite(12, HIGH); // acende led amarelo (pino 12)
  delay(1000); // aguarda 1 segundo
  digitalWrite(12, LOW); //apaga led amarelo (pino 12)

  digitalWrite(13, HIGH); // acende led vermelho (pino 13)
  delay(3000); // aguarda 3 segundos
  digitalWrite(13, LOW); // apaga o led vermelho (pino 13)
}
```


3.4 LED RGB

APRESENTAÇÃO DA EXPERIÊNCIA:

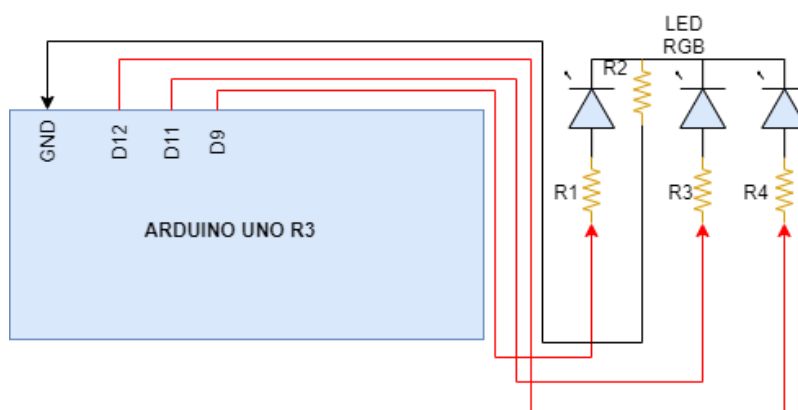
o experimento consiste em demonstrar o funcionamento do Led RGB no Arduino, para além do Vermelho, Verde e Azul, emitir outras cores através da mistura dos tons nativos.

MATERIAL UTILIZADO:

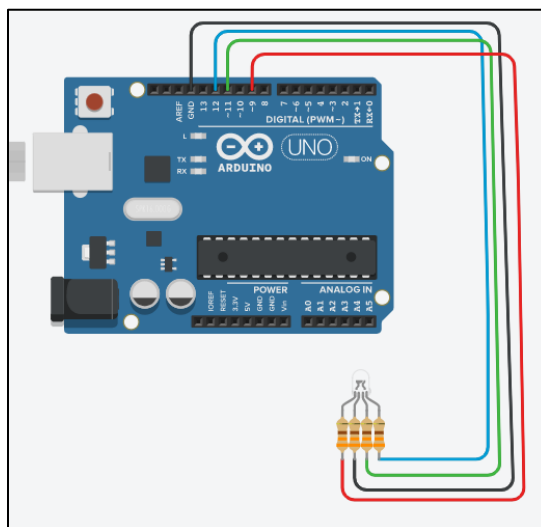
- 1 - Arduino Uno R3.
- 1 – LED RGB.
- 4 - Resistores de 330 Ω .

CIRCUITO ELÉTRICO DA EXPERIÊNCIA:

Os pinos digitais D9, D11 e D12 estão configurados como saída de dados para realizar a alternância de cores além do vermelho, verde e do azul. Ao executar a função Setup, o circuito é carregado com uma tensão de 5v que flui pelos pinos D9, D11 e D12, chegando nos resistores de 330 Ω conectados nos Ânodos do LED RGB, seguindo pelo resistor de 330 Ω conectado ao pino GND, assim, fechando o circuito.



SIMULAÇÃO:



URL: <https://www.tinkercad.com/embed/kyntvF6TfhQ?editbtn=1>

CÓDIGO:

```
//Declaração das constantes
const int ledAzul = 12; //ativa o pino digital 12.
const int ledVerde = 11; //ativa o pino digital 11.
const int ledVermelho = 9; //ativa o pino digital 9.

//Método setup, executado uma vez ao ligar o Arduino.
void setup() {
    //Definindo os pinos digitais (12, 11, 9) como de saída.
    pinMode(ledAzul,OUTPUT); // define pino 12 como saída
    pinMode(ledVerde,OUTPUT); // define pino 11 como saída
    pinMode(ledVermelho,OUTPUT); // define pino 9 como
saída
}
//Método loop, executado enquanto o Arduino estiver ligado.
void loop() {
    //Acendendo cada cor individualmente.
    digitalWrite(ledAzul,HIGH); // pino 12 ligado
    delay(500); //aguarda 0,5 segundos
    digitalWrite(ledAzul,LOW); // pino 12 desligado

    digitalWrite(ledVerde,HIGH); // pino 11 ligado
    delay(500); //aguarda 0,5 segundos
    digitalWrite(ledVerde,LOW); // pino 11 desligado

    digitalWrite(ledVermelho,HIGH); // pino 9 ligado
    delay(500); //aguarda 0,5 segundos
    digitalWrite(ledVermelho,LOW); // pino 9 desligado
    //Misturando as cores do led para obter cores diferentes.
    digitalWrite(ledAzul,HIGH); // pino 12 ligado
    digitalWrite(ledVerde,HIGH); // pino 11 ligado
    digitalWrite(ledVermelho,HIGH); // pino 9 ligado
    delay(1000); // aguarda 1 segundo

    digitalWrite(ledAzul,HIGH); // pino 12 ligado
    digitalWrite(ledVerde,HIGH); // pino 11 ligado
    digitalWrite(ledVermelho,LOW); // pino 9 ligado
    delay(1000); // aguarda 1 segundo
    digitalWrite(ledAzul,LOW); // pino 12 ligado
    digitalWrite(ledVerde,HIGH); // pino 11 ligado
    digitalWrite(ledVermelho,HIGH); // pino 9 ligado
    delay(1000); // aguarda 1 segundo

    digitalWrite(ledAzul,HIGH); // pino 12 ligado
    digitalWrite(ledVerde,LOW); // pino 11 ligado
    digitalWrite(ledVermelho,HIGH); // pino 9 ligado
    delay(1000); // aguarda 1 segundo
}
```

3.5. Controlando o servo motor

APRESENTAÇÃO DA EXPERIÊNCIA:

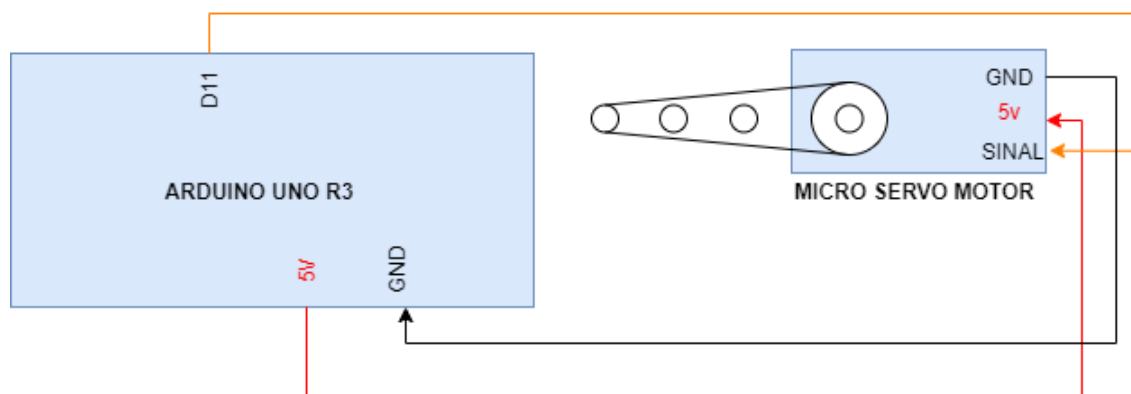
O experimento consiste em demonstrar o funcionamento do Micro servo motor, no Arduino, utilizando um código com os movimentos pré-determinados, podendo ser alterado para outros valores (posições).

MATERIAL UTILIZADO:

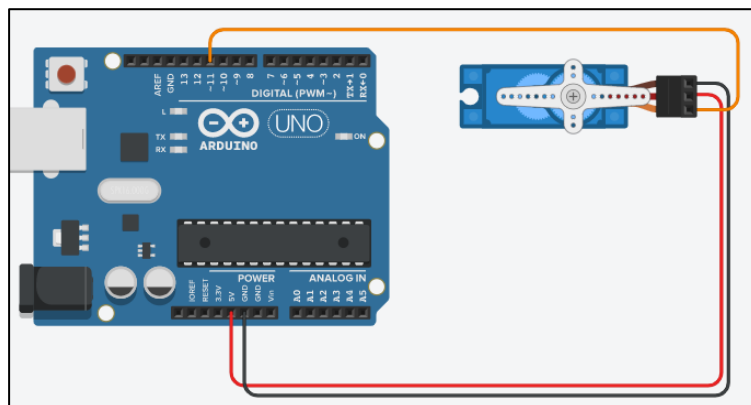
- 1 - Arduino Uno R3.
- 1 – Micro servo motor.

CIRCUITO ELÉTRICO DA EXPERIÊNCIA:

Primeiramente o pino digital 11 assume como saída de dados para comandar o servo motor, o pino de 5v irá alimentar o micro servo. Ao ligar a placa o programa irá acionar os comandos através do pino D11 e o pino 5v irá fornecer a corrente para o circuito que vai fluir até o pino terra (GND), fechando o circuito.



SIMULAÇÃO:



URL: <https://www.tinkercad.com/embed/lob7msAdsQ2?editbtn=1>

CÓDIGO:

```
#include <Servo.h> // cria um objeto do tipo Servo
para controlar um servomotor
#define pinServo 11 // define pino digital 11
como controlador do
motor

Servo servol; // Cria um objeto servo e a Variável para
armazenar a posição do servol
int grau = 0; // define o grau como valor inteiro para
posição do servo

void setup() { // Método setup, executado uma vez ao ligar o
Arduino.
    servol.attach(pinServo); // associa o motor no pino 11 ao
objeto servol
    Serial.begin(9600); // inicia a porta serial, configura a
taxa de dados para 9600 bps
    servol.write(0); // definir o ângulo de 0° para o giro do
micro servo
    delay(2000); // aguarda 2 segundos
}

void loop() { //Método loop, executado enquanto o Arduino
estiver ligado.
    servol.write(180); // definir o ângulo de 180° para o giro do
micro servo
    delay(500); // aguarda 5 segundos
    servol.write(0); // definir o ângulo de 0° para o giro do
micro servo
    delay(500); // aguarda 5 segundos
}
```

3.6. Fotorresistor com LED

APRESENTAÇÃO DA EXPERIÊNCIA:

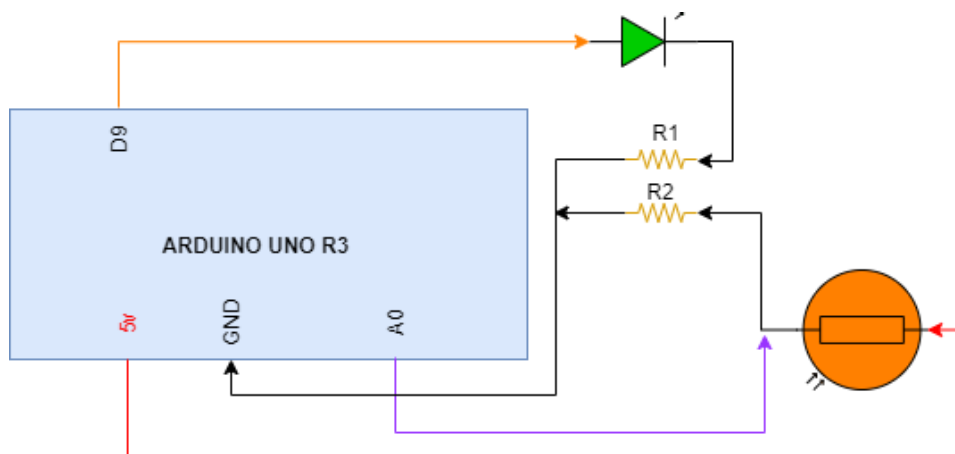
O experimento com Fotorresistor tem o objetivo de indicar a presença ou ausência de luz no ambiente onde o sensor estiver instalado, sua lógica consiste em demonstrar através do Led a intensidade do feixe de luz captado pelo Fotorresistor.

MATERIAL UTILIZADO:

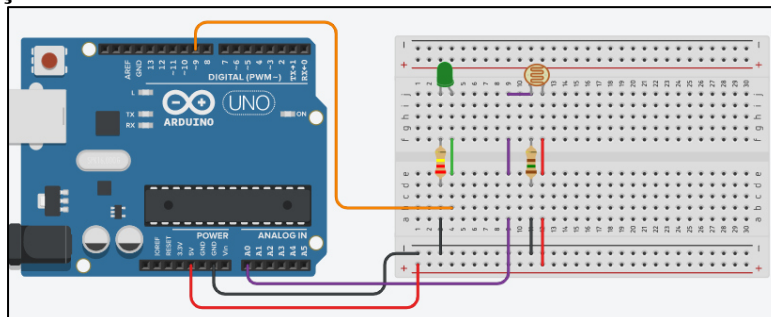
- 1 - Arduino Uno R3.
- 1 – LED verde
- 1 – Resistor 2 de 220 Ω .
- 1 – Resistor 1 de 150 Ω .
- 1 - Fotorresistor

CIRCUITO ELÉTRICO DA EXPERIÊNCIA:

De início o pino digital 9 é designado para acionar o LED verde e o pino analógico 0 recebe os dados do Fotorresistor. Ao executar o programa o Fotorresistor perceberá ausência ou presença de luz, enquanto isso essas informações são repassadas através do pino GND, passando pelo resistor de 220 Ω do Fotorresistor conectado ao pino A0, onde a partir daí acionará o LED através da tensão de 5v que flui pelo ânodo do LED, chegando ao Cátodo, seguindo pelo resistor de 150 Ω e chegando ao terra da placa (GND).



SIMULAÇÃO:



URL: <https://www.tinkercad.com/embed/fuLybRWAr1k?editbtn=1>

CÓDIGO:

```
int sensorValue = 0; // Define os valores do sensor como
inteiros

void setup() // Método setup, executado uma vez ao ligar o
Arduino
{
    pinMode(A0, INPUT); // define o pino analógico A0 como
    entrada de dados do sensor
    Serial.begin(9600); // inicia a porta serial, configura a
    taxa de dados para 9600 bps

    pinMode(9, OUTPUT); // Define o pino digital 9, como saída
    de dados
}

void loop() //Método loop, executado enquanto o Arduino
estiver ligado.
{
    sensorValue = analogRead(A0); // variável para armazenar os
    valores lidos no sensor.
    Serial.println(sensorValue); //imprimi as informações
    coletadas no sensor direto no monitor serial
    analogWrite(9, map(sensorValue, 0, 1023, 0, 255));
    //converte a faixa de tensão de entrada, de 0 a 5 volts, em
    um valor digital entre 0 e 1023.
    delay(100); // Aguarda 100 milissegundos
}
```

3.7. Sensor de temperatura

APRESENTAÇÃO DA EXPERIÊNCIA:

O experimento possui como base o sensor de temperatura e LEDs nas cores verde e vermelho, respectivamente indicando a intensidade da menor para maior temperatura captada pelo dispositivo, com base nos dados inseridos na programação.

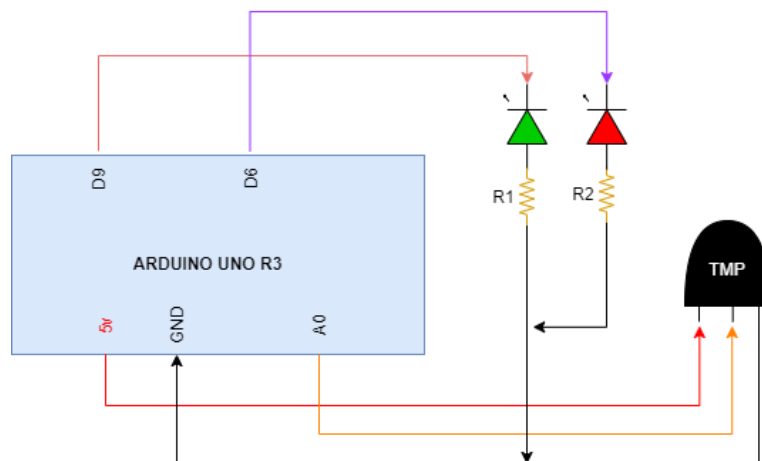
A Lógica presente no projeto possibilita o acompanhamento de resultados de temperatura na escala Celsius captada pelo sensor, o papel dos LEDs é de indicar a temperatura máxima e a mínima desejada para o experimento, que neste caso será de 20°C para o LED verde acender e de 30°C para o LED vermelho acender.

MATERIAL UTILIZADO:

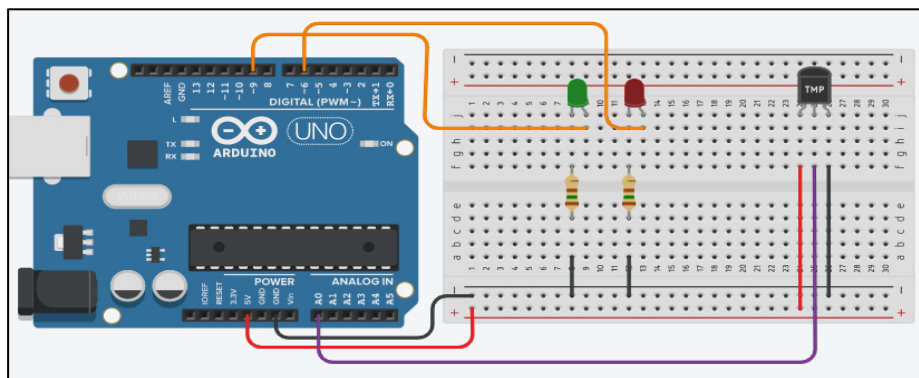
- 1 - Arduino Uno R3.
- 1 - Sensor de temperatura [TMP36].
- 2 - Resistores de 150 Ω .
- 1 - LED vermelho.
- 1 - LED verde.

CIRCUITO ELÉTRICO DA EXPERIÊNCIA:

Primeiramente o pino digital 9 e 6 estão designados para acionar os LEDs verde e vermelho, pino analógico 0 para receber as informações do sensor e o pino 5v para alimentar o sensor de temperatura. Ao ligar a placa o pino de 5v alimenta até a saída da tensão pelo pino GND, enquanto isso o sensor irá realizar a leitura da temperatura e enviará as informações através do pino A0, acionando a tensão de 5v para os LEDs conectados ao pino D9 e D6, onde irá fluir até o pino GND.



SIMULAÇÃO:



URL: <https://www.tinkercad.com/embed/INvkaqkMipj?editbtn=1>

CÓDIGO:

```
int temperatura = 0; // determina o tipo de dados da
temperatura como inteiro(números)

void setup() //método Loop que será executado enquanto o
Arduino estiver ligado
{
    pinMode(A0, INPUT); // entrada de dados do sensor TMP
    pinMode(6, OUTPUT); // Saída de dados para o Led Vermelho
    pinMode(9, OUTPUT); // Saída de dados para o Led Verde
}

void loop()// método loop que será executado enquanto o
Arduino estiver ligado
{
    if (-40 + 0.488155 * (analogRead(A0) - 20) < 20) { // Lê o
valor do sensor conectado no pino analógico A0 e cria a
condição: se a temperatura for de 20°C ou menor que 20°C o
led vermelho apaga e o verde acende.
        digitalWrite(6, LOW); // Define o pino digital 6
Desligado (Led Vermelho)
        digitalWrite(9, HIGH); // Define o pino digital 9 Ligado
(Led Verde)
    }
    if (-40 + 0.488155 * (analogRead(A0) - 20) >= 30) { // Lê o
valor do sensor conectado no pino analógico A0 e cria a
condição: se a temperatura for de 20°C ou maior ou igual a
30°C o led vermelho se acende e o verde apaga.
        digitalWrite(6, HIGH); // Define o pino digital 6 ligado
(Led Vermelho)
        digitalWrite(9, LOW); // Define o pino digital 9 Desligado
(Led Verde)
    }
    delay(10); //Atraso um pouco para melhorar o desempenho da
simulação
}
```


3.8. Controlando o micro servo motor com potenciômetro

APRESENTAÇÃO DA EXPERIÊNCIA:

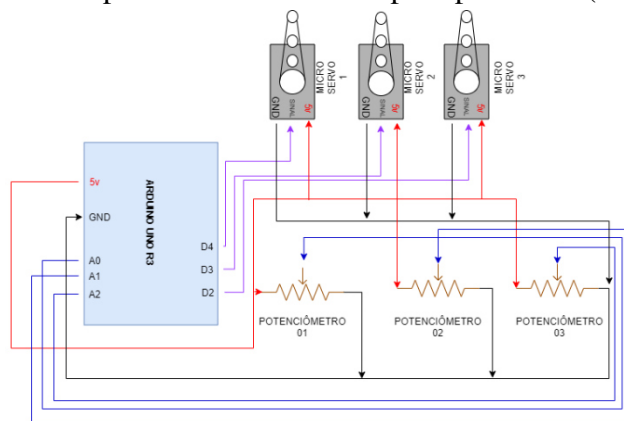
O experimento tem por objetivo demonstrar o funcionamento do micro servo utilizando o potenciômetro para controlar os movimentos do motor.

MATERIAL UTILIZADO:

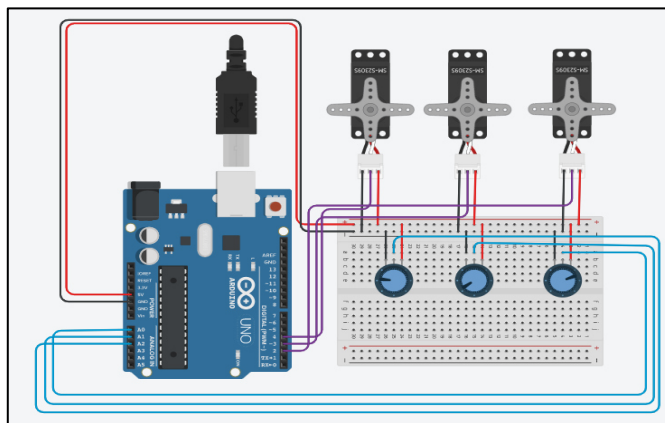
- 1 - Arduino Uno R3.
- 3 – Micro servo motor.
- 3 – Potenciômetros de 250 k Ω .

CIRCUITO ELÉTRICO DA EXPERIÊNCIA:

Primeiramente os pinos digitais D2, D3 e D4 estão configurados como saída de dados para os motores micro servo, os pinos analógicos A0, A1 e A2 estão configurados como saída de dados para os potenciômetros de 250 k Ω e os pinos 5v forneceram tensão para o experimento. Primeiramente ao ligar a placa do Arduino o programa irá executar os comandos e o usuário deve realizar o giro no potenciômetro, em seguida os comandos serão enviados através dos pinos analógicos e recebidas pelos pinos digitais que irão efetuar o giro nos motores, por fim a tensão dos motores e dos potenciômetros fluem pelo pino terra (GND).



SIMULAÇÃO:



URL: <https://www.tinkercad.com/embed/jWIBaKYhKse?editbtn=1>

CÓDIGO:

```
#include <Servo.h> //inclui a biblioteca "Servo" para
controlar um servomotor.

Servo servo1; // cria um objeto do tipo Servo para controlar
o servomotor 1
Servo servo2; // cria um objeto do tipo Servo para controlar
o servomotor 2
Servo servo3; // cria um objeto do tipo Servo para controlar o
servomotor 3

const int servo1Pin = 2; // define a variável do motor 1 no
pino digital 2 como uma constante do tipo inteiro.
const int servo2Pin = 3; // define a variável do motor 2 no
pino digital 3 como uma constante do tipo inteiro.
const int servo3Pin = 4; // define a variável do motor 3 no
pino digital 4 como uma constante do tipo inteiro.

const int pot1 = 0; // define a variável do potenciômetro 1
no pino analógico A0 como uma constante do tipo inteiro
const int pot2 = 1; // define a variável do potenciômetro 2
no pino analógico A1 como uma constante do tipo inteiro
const int pot3 = 2; // define a variável do potenciômetro 3
no pino analógico A2 como uma constante do tipo inteiro

int potValue1; // declara os valores do potenciômetro 1, como
inteiros
int potValue2; // declara os valores do potenciômetro 2, como
inteiros
int potValue3; // declara os valores do potenciômetro 3, como
inteiros

void setup() { // Define a função a ser executada

servo1.attach(servo1Pin); // anexa a variável do servo 1 ao
pino 2.
servo2.attach(servo2Pin); // anexa a variável do servo 2 ao
pino 3.
servo3.attach(servo3Pin); // anexa a variável do servo 3 ao
pino 4.

}
void loop() { // Executa o loop enquanto o Arduino estiver
ligado.

    potValue1 = analogRead (pot1); // leia a entrada no pino
analógico A0
    potValue1 = map(potValue1, 0, 1023, 0, 180); // Mapear os
valores do potenciômetro conectado ao pino analógico A0.
```

```
servo1.write(potValue1); // Grava os dados no servo para
controlar os movimentos.

potValue2 = analogRead (pot2); // leia a entrada no pino
analógico A1
potValue2 = map(potValue2, 0, 1023, 0, 180); // Mapear os
valores do potenciômetro conectado ao pino analógico A1.
servo2.write(potValue2); // Grava os dados no servo para
controlar os movimentos.

potValue3 = analogRead (pot3); // leia a entrada no pino
analógico A2
potValue3 = map(potValue3, 0, 1023, 0, 180); // Mapear os
valores do potenciômetro conectado ao pino analógico A2.
servo3.write(potValue3); // Grava os dados no servo para
controlar os movimentos.

}
```

3.9. Semáforo com cancela

APRESENTAÇÃO DA EXPERIÊNCIA:

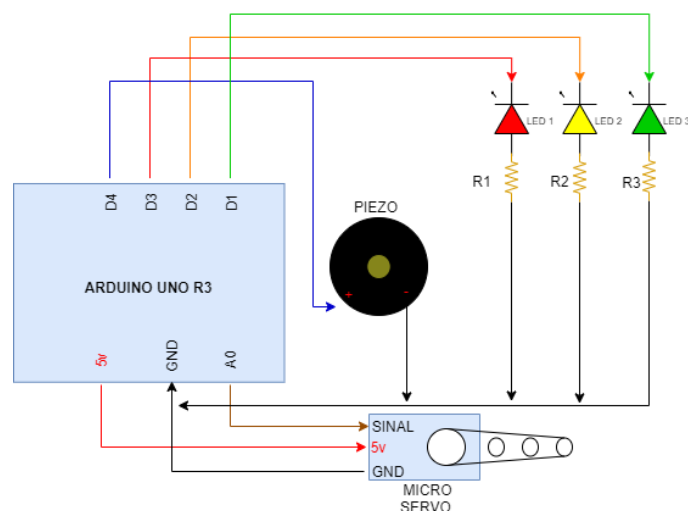
O experimento consiste em simular um semáforo de trânsito que emite um alerta sonoro quando o sinal estiver vermelho. Além de, uma cancela fechada para assegurar a interrupção do fluxo. Estando o semáforo verde o alerta sonoro é desativado e a cancela é liberada.

MATERIAL UTILIZADO:

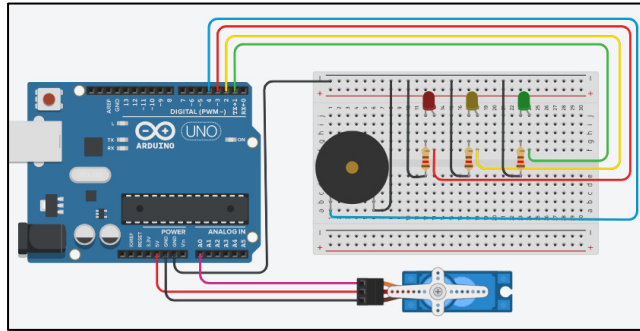
- 1 - Arduino Uno R3.
- 1 – Piezo.
- 1 – LED vermelho.
- 1 – LED amarelo.
- 1 – LED verde.
- 1 – Micro servo motor.
- 1 – Resistores de 115Ω .

CIRCUITO ELÉTRICO DA EXPERIÊNCIA:

De início os pinos digitais D1, D2, D3 alimentaram os LEDs respectivamente, verde, amarelo e vermelho e o pino digital D4 alimenta o PIEZO, a porta analógica A0 está configurada como saída de dados para o micro servo motor e o pino 5v fornece tensão para o mesmo. Ao ligar a placa do Arduino, o programa irá acender o LED vermelho através da tensão de 5v fornecida pelo pino D3 e fluir pelo cátodo do LED vermelho até o resistor de 115Ω . Na sequência o micro servo motor realiza o movimento de 180° que simula o fechamento da cancela, 1 segundo o LED verde acende e o PIEZO é acionado emitindo sinal sonoro em seguida o LED amarelo é acionado e o sinal sonoro é desativado quando o LED amarelo se apaga e o LED vermelho se acende e a cancela é acionada a tensão flui pelos resistores 115Ω dos LED até chegar ao pino GND.



SIMULAÇÃO:



URL: <https://www.tinkercad.com/embed/86XC22I1BGi?editbtn=1>

CÓDIGO:

```
#include <Servo.h> //inclui a biblioteca "Servo" para
controlar um servomotor.

Servo servo_A0; // cria objeto do tipo Servo

void setup() // Define a função a ser executada
{
    pinMode(3, OUTPUT); // Saída de dados para o led
(vermelho)
    servo_A0.attach(A0); // anexa o objeto do "servo_A0" ao
pino analógico A0.

    pinMode(4, OUTPUT); // piezzo
    pinMode(2, OUTPUT); // led amarelo
    pinMode(1, OUTPUT); // led verde
}

void loop() // Executa o loop enquanto o Arduino estiver
ligado.
{
    digitalWrite(3, HIGH); // define o led (vermelho)conectado
no pino digital 3 como ligado
    servo_A0.write(90); //Move o servo para o ângulo de 90°
    tone(4, 523, 2000); //tom de reprodução 60 (C5 = 523 Hz)
    delay(1000); // Aguarda 1000 milissegundos
    digitalWrite(3, LOW); // define o led (vermelho)conectado
no pino digital 3 como desligado
    delay(1000); // Aguarda 1000 milissegundos
    digitalWrite(2, HIGH); // define o led (amarelo)conectado
no pino digital 2 como ligado
    tone(4, 523, 1000); // tom de reprodução 60 (C5 = 523 Hz)
    delay(1000); // Aguarda 1000 milissegundos
    digitalWrite(2, LOW); // define o led (amarelo)conectado no
pino digital 2 como desligado
    noTone(4); //tom de reprodução
```

```

delay(1000); // Aguarda 1000 milissegundos
digitalWrite(1, HIGH); // define o led (verde) conectado no
pino digital 1 como ligado
servo_A0.write(180); // Move o servo para o ângulo de 180°
delay(1000); // Aguarda 1000 milissegundos
digitalWrite(1, LOW); // define o led (verde) conectado no
pino digital 1 como desligado
}

```

3.10. Calculadora

APRESENTAÇÃO DA EXPERIÊNCIA:

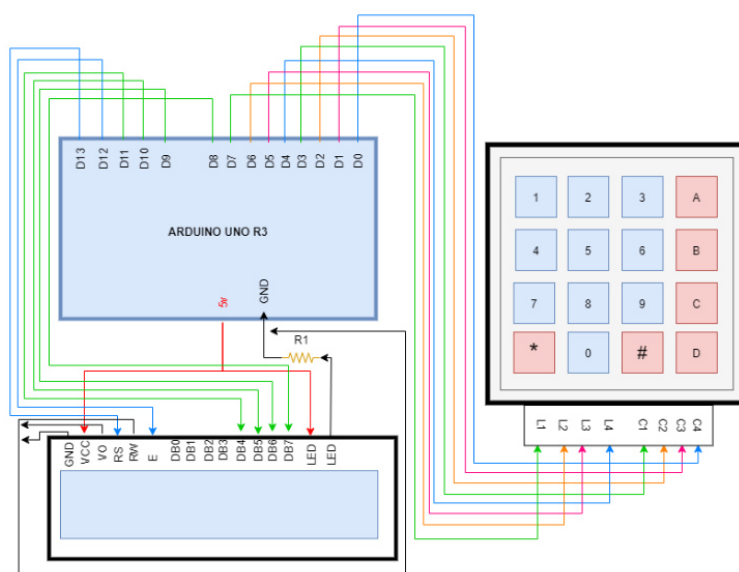
O experimento consiste em simular uma calculadora que realiza as quatro operações básicas da matemática: adição, subtração, multiplicação e divisão.

MATERIAL UTILIZADO:

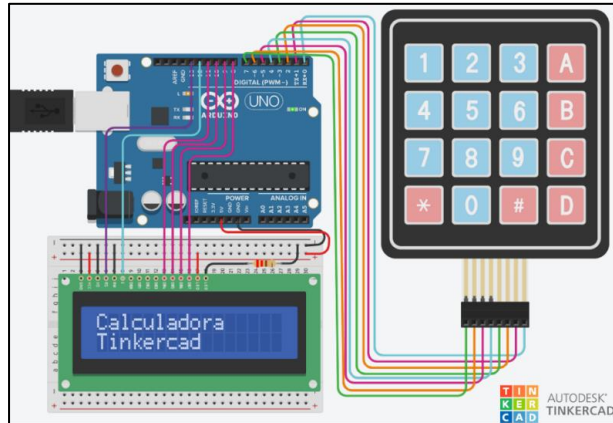
- 1 - Arduino Uno R3.
- 1 - Teclado 4x4.
- 1 – LCD 16X2.
- 1 – Resistor de 220Ω.

CIRCUITO ELÉTRICO DA EXPERIÊNCIA:

Primeiramente os pinos digitais D7, D6, D5, D4, D3, D2, D1 e D0 estão configurados como saída de dados para linhas e colunas do teclado 4x4, respectivamente, L1, L2, L3, L4, C1, C2, C3 e C4. Os pinos digitais D13, D12, D11, D10, D9 e D8 estão configurados como saída de dados para a tela LCD 16x2 o pino 5v irá alimentar a tela e o Led interno da mesma, a tensão chega ao resistor de 220Ω conectado ao GND, assim fechando o circuito da tela.



SIMULAÇÃO:



URL: <https://www.tinkercad.com/embed/hRDXsKzzDFg?editbtn=1>

CÓDIGO:

```
#include <Keypad.h> // inclui a biblioteca de teclados do Arduino
#include <Wire.h> // inclui biblioteca de comunicação
#include <LiquidCrystal.h> // inclui biblioteca do LCD

LiquidCrystal lcd(13, 12, 11, 10, 9, 8); // Configura os pinos do Arduino para se
comunicar com o LCD
long first = 0; // cria a variável do primeiro número
long second = 0; // cria a variável do segundo número
double total = 0; // cria a variável do resultado

char customKey;
const byte ROWS = 4; // quatro linhas
const byte COLS = 4; // quatro colunas

char keys[ROWS][COLS] = { // define os símbolos nos botões do teclado
  {'1','2','3','+'},
  {'4','5','6','-'},
  {'7','8','9','*'},
  {'C','0','=','/'}}
};
byte rowPins[ROWS] = {7,6,5,4}; // conecta-se às pinagens das linhas do teclado
byte colPins[COLS] = {3,2,1,0}; // conecta-se às pinagens das colunas do teclado

// inicializa uma instância da classe New Keypad
Keypad customKeypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS,
COLS);
void setup() // Define a função a ser executada
{
  lcd.begin(16, 2); // inicia o lcd
  for(int i=0;i<=3;i++);
  lcd.setCursor(0,0); // exibir no canto superior esquerdo
```

```

    lcd.print("Calculadora"); // imprime a mensagem "Calculadora" na tela de boas
vinda
    lcd.setCursor(0,1); // exibir no canto inferior esquerdo
    lcd.print("Tinkercad"); // imprime a mensagem "Tinkercad" na tela de boas vindas

delay(4000); //aguardar 4 segundos
lcd.clear(); // limpa a mensagem de boas vindas a tela LCD
lcd.setCursor(0, 0); // exibir mensagem no canto superior esquerdo
}

void loop() // Executa o loop enquanto o Arduino estiver ligado.
{
    customKey = customKeypad.getKey();
    switch(customKey)
    {
        case '0' ... '9': // permite que a calculadora continue coletando o primeiro valor até
que um operador seja pressionado "+ - * /"
            lcd.setCursor(0,0); // exibir no canto superior esquerdo
            first = first * 10 + (customKey - '0');
            lcd.print(first); //exibe no LCD O primeiro número pressionado
            break; // pausa

        case '+': // cria a função soma
            first = (total != 0 ? total : first);
            lcd.setCursor(0,1); // exibir mensagem no canto inferior esquerdo.
            lcd.print("+"); // exibe na tela o sinal de soma.
            second = SecondNumber(); // obtém o segundo número coletado.
            total = first + second; // Realiza a soma dos valores.
            lcd.setCursor(0,3); // posição do resultado a ser exibida no LCD.
            lcd.print(total); // exibe o valor total no LCD.
            first = 0, second = 0; // redefine os valores para zero para o próximo uso
            break; // pausa

        case '-': // cria a função subtrair
            first = (total != 0 ? total : first);
            lcd.setCursor(0,1); // exibir mensagem no canto inferior esquerdo
            lcd.print("-"); // exibe na tela o sinal de subtração
            second = SecondNumber(); // obtém o segundo número coletado.
            total = first - second; // Realiza a subtração dos valores.
            lcd.setCursor(0,3); // posição do resultado a ser exibida no LCD.
            lcd.print(total); // exibe o valor total no LCD.
            first = 0, second = 0; // redefine os valores para zero para o próximo uso
            break; // pausa

        case '*': // cria a função multiplicar
            first = (total != 0 ? total : first);
            lcd.setCursor(0,1); // exibir mensagem no canto inferior esquerdo
            lcd.print("*"); // exibe na tela o sinal de multiplicação
            second = SecondNumber(); // obtém o segundo número coletado.
            total = first * second; // Realiza a multiplicação dos valores.

```



```

lcd.setCursor(0,3);// posição do resultado a ser exibida no LCD.
lcd.print(total); // exibe o valor total no LCD.
first = 0, second = 0;// redefine os valores para zero para o próximo uso
break;// pausa

case '/': // cria a função divisão
first = (total != 0 ? total : first);
lcd.setCursor(0,1); // exibir mensagem no canto inferior esquerdo
lcd.print("/"); // exibe na tela o sinal de divisão
second = SecondNumber(); // Realiza a divisão dos valores.
lcd.setCursor(0,3);// posição do resultado a ser exibida no LCD.

second == 0 ? lcd.print("Valor Inválido") : total = (float)first / (float)second;

lcd.print(total);// Exibe total no LCD
first = 0, second = 0; // redefine os valores para zero para o próximo uso
break;// pausa

case 'C': // cria a função limpar
total = 0; // total igual a zero
lcd.clear(); // limpa o LCD
break; // pausa
}
}

long SeconNumber()
{d
while( 1 )
{
customKey = customKeypad.getKey();
if(customKey >= '0' && customKey <= '9')
{
second = second * 10 + (customKey - '0');
lcd.setCursor(0,2); // posição do resultado a ser exibida no LCD.
lcd.print(second);
}

if(customKey == '=') break; // retorna segundo;
}
return second;
}
}

```

4. CONCLUSÃO

Este capítulo traz uma reflexão sobre os objetivos do trabalho, apresenta as ferramentas utilizadas na pesquisa e sua importância nas aulas de Lógica de Programação e Eletrônica e também aborda as limitações no decorrer da pesquisa e trabalhos futuros.

4.1. Sobre os objetivos do trabalho

Para identificar os principais componentes disponíveis no simulador, foi elaborado um tópico que apresenta cada componente e suas respectivas figuras para evidenciar as características físicas.

Para apresentar os experimentos projetados no simulador, foi disposto um capítulo, onde encontra-se os objetivos do experimento, QR CODE da Simulação, endereço URL, Circuito Elétrico, material utilizado– e código fonte de cada projeto.

E para a linguagem de programação C através do Arduino no ensino da Lógica de programação, foi utilizado um tópico para apresentar os conceitos, sintaxe, estrutura básica, principais comandos utilizados para controlar o Arduino e a importância de se aprender ou apenas conhecer a linguagem de programação C.

4.1. Sobre as ferramentas

Para o desenvolvimento deste trabalho, foi utilizado o simulador de circuitos *Tinkercad* da empresa *Autodesk Inc.* A plataforma é acessível a estudantes, profissionais ou para aqueles que estão iniciando os estudos com Arduino e Eletrônica.

A plataforma possui uma base de dados com as seções de Disparadores, Lições e Projetos. Os Disparadores são exemplos de circuitos simples para iniciar os conhecimentos em Eletrônica, as Lições são exercícios práticos com o objetivo de aperfeiçoar os conhecimentos em Eletrônica e os Projetos são experimentos simples que auxilia os usuários no início dos estudos em codificação e prototipagem utilizando o Arduino.

Por ser necessário somente um computador com acesso à internet para acessar a ferramenta, o simulador de circuitos *Tinkercad*, é uma excelente alternativa para suprir as necessidades da placa física do Arduino e seus componentes, além do seu ambiente de programação. Sendo essencial para o desenvolvimento dos experimentos apresentados neste trabalho.

4.2. Sobre a importância do uso de ferramentas nas aulas de Lógica de Programação e Eletrônica.

Diante da infinidade de tecnologias presente em nosso cotidiano, cabe destacar o computador como sendo a mais utilizada atualmente. Com ele podemos realizar inúmeras tarefas, conectado ou não à internet, e para que a sociedade usufrua das melhores ferramentas, existem estudantes e profissionais que dedicam seu tempo desenvolvendo soluções para os problemas atuais.

Nesse contexto, o desenvolvimento de novas ferramentas se faz necessário para que o avanço tecnológico esteja alinhado às demandas da sociedade, e o primeiro passo para que se construa aplicações, seja ela para sites, computadores, celulares, projetos de robótica, automação residencial, carros e aviões é estudar a Lógica de programação para desenvolver o pensamento lógico, seja através de conceitos matemáticos, pseudocódigos, descrições narrativas e algoritmos de programação.

Assim, a lógica pode ser compreendida não somente através de programas virtuais no computador, pois será ela que dará sentido ao programa e a vida ao *hardware* e por fim se chegará a solução para o problema no qual o programador deseja solucionar.

Portanto, explorar as inúmeras formas de se aplicar a Lógica de programação é fundamental para o desenvolvimento do estudante ou do profissional que trabalha ou tem como *hobby* o desenvolvimento de software.

4.3 Limitações

Após o desenvolvimento dos experimentos somente no ambiente virtual, notei a necessidade de aplicação em sala de aula, pois o desafio é comprovar se realmente o ambiente de simulação pode suprir as necessidades que o Arduino físico e seus componentes propõe solucionar.

Durante a procura por projetos fora da galeria de simulações do *Tinkercad*, muitos foram descartados pois o simulador não permitiu a execução de alguns experimentos, por exemplo: pressionar dois botões ao mesmo tempo para acender um LED, ou seja, a combinação dos botões não foi possível, pois os botões são acionados através do click do *mouse*.

4.5 Trabalhos futuros

A partir dos experimentos apresentados neste trabalho, pretendo implementar outras formas de aplicação do simulador e o Arduino para avançar nos estudos com programação, uma delas é utilizar a plataforma física do Arduino para aprender ou ensinar outras linguagens de programação.

Além disso, tendo em vista a imensidão de áreas que o Arduino permite explorar, pretendo após este trabalho, realizar estudos mais aprofundados sobre o tema e construir artigos científicos e a aplicação dos experimentos em sala de aula.

REFERÊNCIAS

- ALMEIDA, M. et al. **SISTEMAS LINEARES E AS LEIS DE KIRCHHOFF**. p. 2011, 2011.
- ALVES, R. M. et al. **Uso do Hardware Livre Arduino em Ambientes de Ensino-aprendizagem**. Jornada de Atualização em Informática na Educação, v. 1, n. 1, p. 162–187, 2013.
- GIL, A. **Como elaborar projetos de pesquisa**, 5ª Edição, editora Atlas. São Paulo, p. 184, 2010.
- ATOMIC, H.; PIMENTA, A. **Hack.ATOMIC**. v. 1, 2015.
- CARVALHO, V. A. DE. **Lógica de Programação**. Instituto Federal do Espírito Santo, v. 1, p. 106, 2010.
- CERCONI, F. DO B. M.; MARTINS, M. A. **Recursos Tecnológicos No Ensino De Matemática**: Considerações Sobre Três Modalidades. IV Simposio Nacional de Ensino de Ciência e Tecnologia, 2014.
- CUSTÓDIO, L. P. **Teclado Matricial**. Microcontrolador, 2015.
- FÉLIX, G. B. et al. **ARDUINO NAS ESCOLAS**: a introdução do pensamento computacional para o ingresso na área de tecnologia da informação. [s.d.].
- FONSECA, J. J. S. DA. **Metodologia da Pesquisa Científica**. [s.l: s.n.].
- FREIRE, T.; FILHO, B. **Apostila de Eletrônica básica II**. 2004.
- FRIZZARIN, F. B. **Arduino**: Guia para colocar suas ideias em prática. p. 198, 2016.
- GARLET, D. **Uma proposta para o ensino de programação na educação básica**. 2016.
- HUGH D. YOUNG; FREEDMAN, R. A. **Física 3 Eletromagnetismo**. 12. ed. SÃO PAULO: [s.n.].
- MARTINAZZO, C. A. et al. **ARDUINO: UMA TECNOLOGIA NO ENSINO DE FÍSICA** Arduino: a technology in teaching physics. p. 21–30, 2014.
- OLIVEIRA, C. L. V. **Aprenda Arduino**. [s.l: s.n.].
- OOMLOUT.COM. **TMP36 Temperature Sensor source code**. n. 1, p. 1–4, 2020.
- PRODANOV, C. C.; FREITAS, E. C. DE. **Metodologia do trabalho científico**: métodos e técnicas da pesquisa e do trabalho acadêmico. [s.l: s.n.].
- PUGA, S.; RISSETI, G. **Logica de Programacao e Estruturas de Dados** (com aplicacoes em Java) - Sandra Puga e Gerson Rossetti. Pearson Ed ed. SÃO PAULO: [s.n.].

RIBEIRO, I. S. et al. **A PLATAFORMA ARDUINO: PRINCÍPIOS DE FUNCIONAMENTO E DEMONSTRAÇÃO PRÁTICA COM UM CONTROLADOR DE VENTILADORES**. VII CONGRESSO BRASILEIRO DE ENGENHARIA DE PRODUÇÃO, p. 9, 2017.
RODRIGUES, C. et al. **Eletrônica**. 2013.

SILVA, J. B. **Guia de aplicação: Lei de Ohm**. Revista de Ciência Elementar, v. 3, n. 2, p. 1–12, 2015.

SILVA, J. L. DE S. et al. **Plataforma Arduino integrado ao PLX-DAQ: Análise e aprimoramento de sensores com ênfase no LM35**. Erbase, n. January, 2014.

TORRES, A. L. L. et al. **Ferramenta de auxílio no ensino de Organização e Arquitetura de Computadores** : extensão Ptolemy para fins educacionais. n. December 2012, p. 21–29, 2012.

TORRES, G. **Eletrônica para Autodidatas , Estudantes e Técnicos**. Rio de Janeiro: [s.n.].