UNIVERSIDADE FEDERAL DO CEARÁ

Centro de Tecnologia

Departamento de Engenharia Mecânica

INSTRUMENTAÇÃO COM TINKERCAD - CAPÍTULO 1: INTRODUÇÃO AO TINKERCAD E AO ARDUINO

Thiago Victor Albuquerque de Freitas - 494080

2021

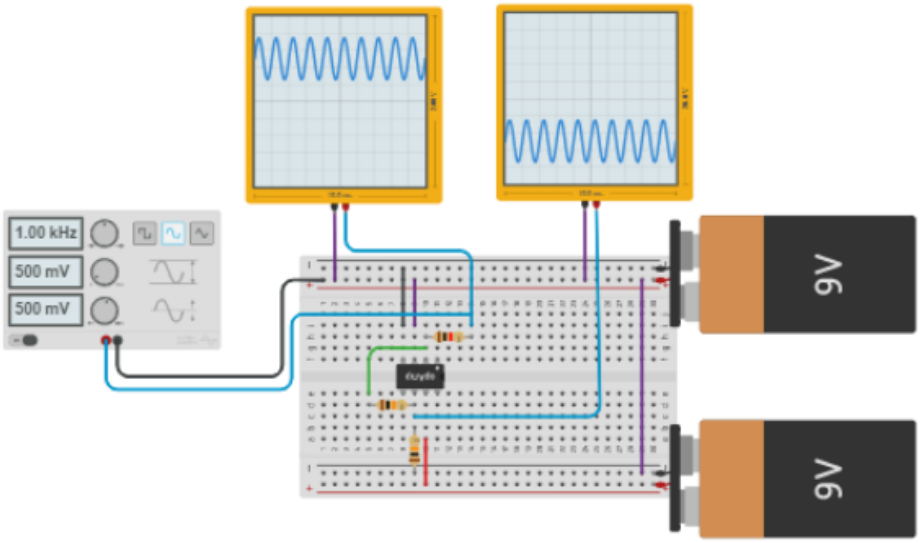
Tinkercad

O que é o Tinkercad?

O Tinkercad é uma plataforma online gratuita que reúne diversas ferramentas de software, como modelagem 3D, simulação de circuitos eletrônicos e programação; os quais ajudam pessoas em todo o mundo a pensar, criar e fazer diversos projetos (Autodesk, 2020). A plataforma é conhecida por sua simplicidade e facilidade de uso.

Para exemplificar, o circuito da Fig. 1 mostra um amplificador eletrônico inversor, o qual é utilizado tanto para amplificar um sinal de entrada quanto para inverter a polarização deste.

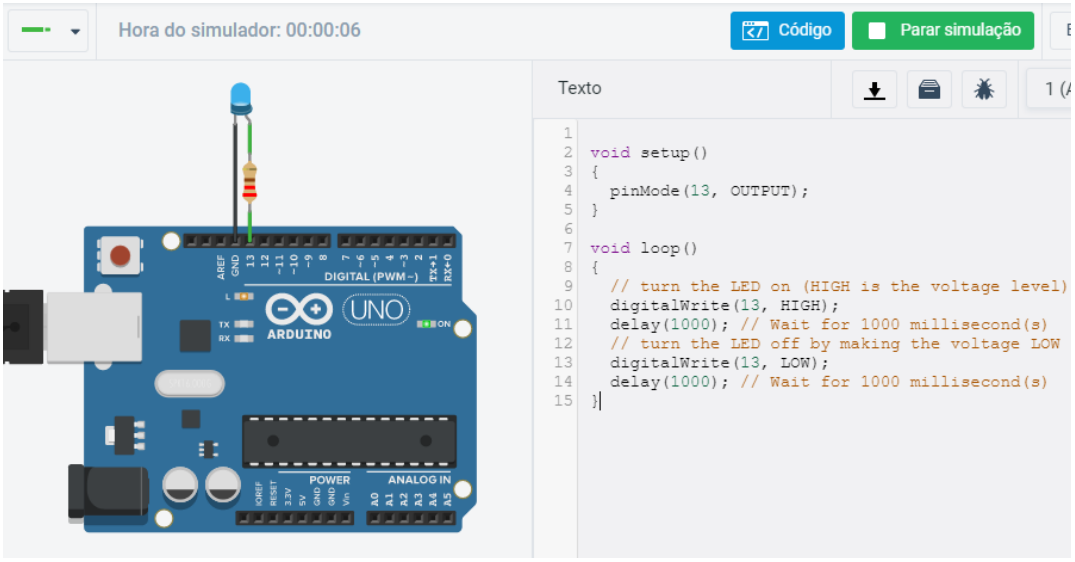
Figura 1 - Circuito formado por um amplificador eletrônico inversor.



Fonte: Autor.

Continuando com os exemplos, também é possível utilizar o microcontrolador arduino e programá-lo dentro da plataforma, como pode ser visualizado na Fig. 2, o qual tem-se um circuito, a esquerda, que liga um simples LED e, a direita, a programação que faz o ele piscar a cada 1 segundo.

Figura 2 - Programação de um circuito com arduino.

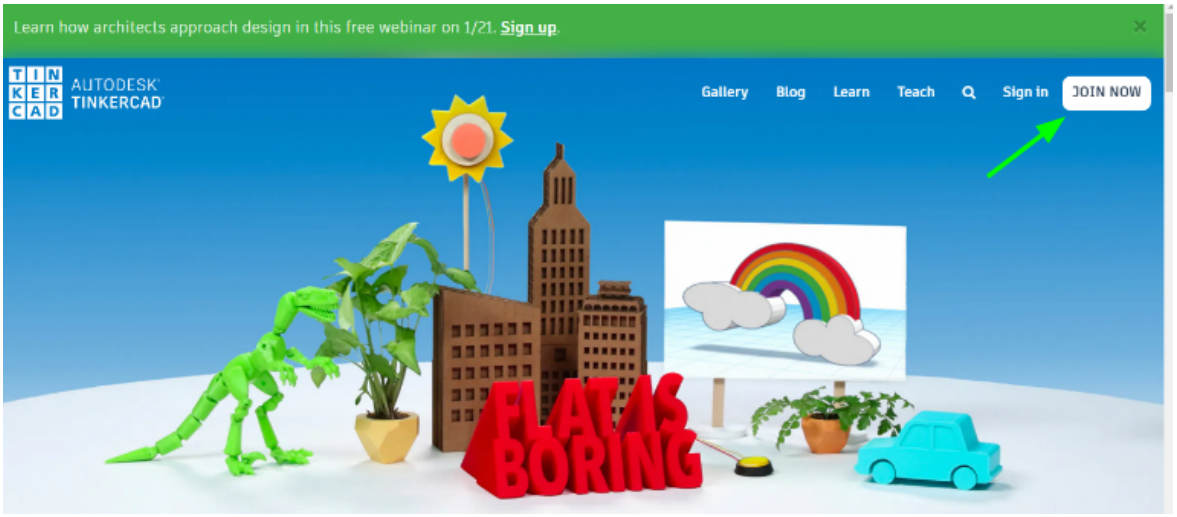


Fonte: Autor.

Primeiro acesso:

- **Etapa 1:** Entre no site do Tinkercad, o qual é mostrado a tela inicial na Fig. 3, e clique em "Join NOW" para se registrar.

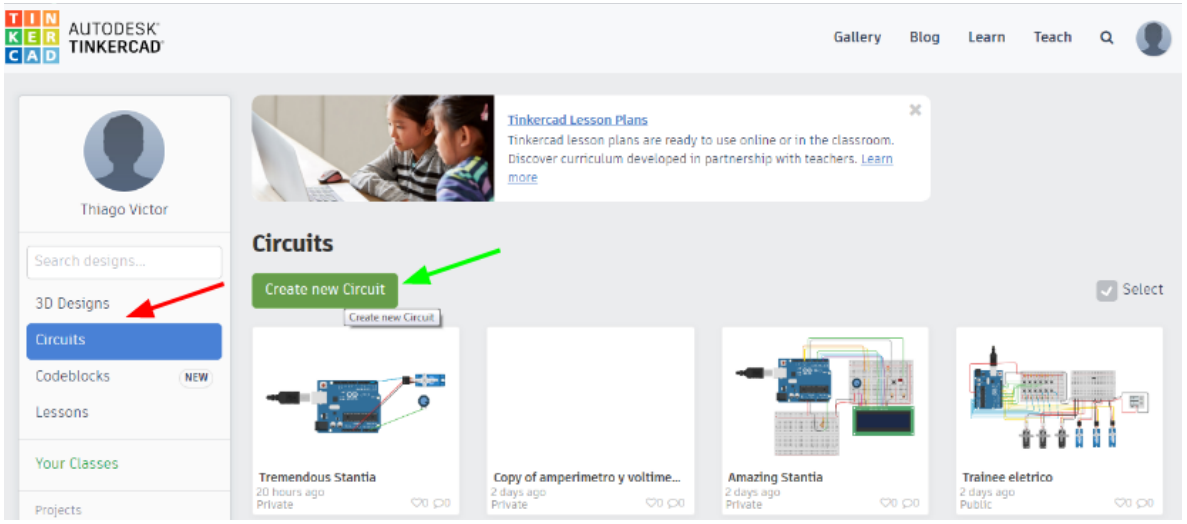
Figura 3 - Site oficial do Tinkercad.



Fonte: Autor.

- **Etapa 2:** Registre-se na plataforma.
- **Etapa 3:** Clique em "Circuits", representado pela seta vermelha na Fig. 4, e então abrirá todos os circuitos que você já utilizou na sua conta. Para criar um novo circuito, clique em "Create new Circuit", como é mostrado pela seta verde.

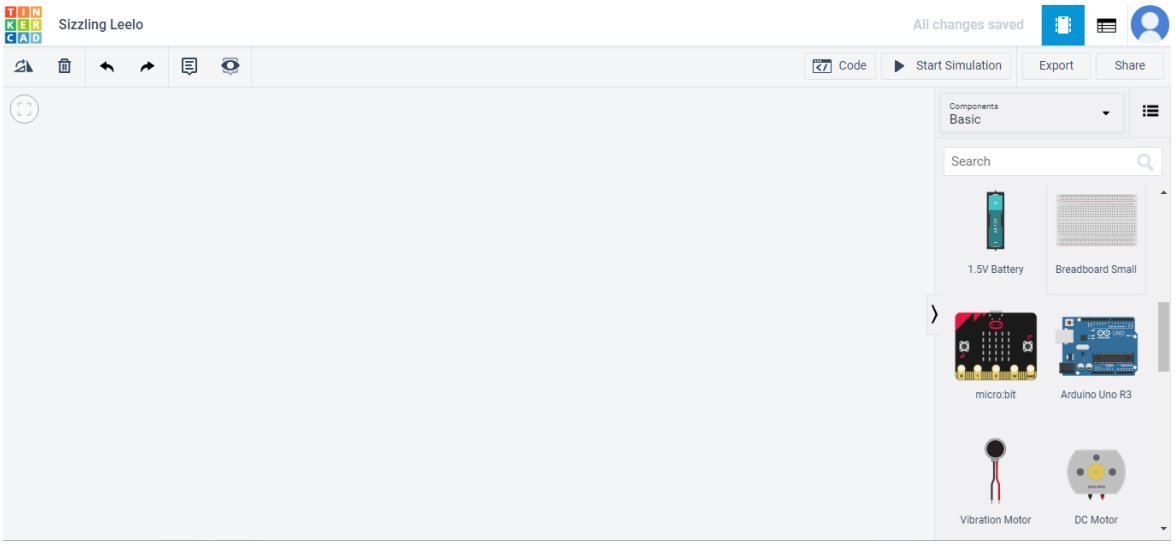
Figura 4 - Tela inicial de todos os circuitos no Tinkercad.



Fonte: Autor.

- **Etapa 4:** Utilize a plataforma de simulações de circuitos eletrônicos mostrada na Fig. 5.

Figura 5 - Tela para desenvolver circuitos eletrônicos no Tinkercad.



Fonte: Autor.

Arduino

O que é o Arduino?

O Arduino é uma plataforma eletrônica de prototipagem de código aberto baseada em hardware e software open source, ou seja, de livre utilização e reprodução (ARDUINO, 2019). Com esse dispositivo, é possível criar diversos projetos eletrônicos, como carrinhos de controle remoto, robôs seguidor de linha e projetos de automação, como pode ser visto no site da USINAINFO.

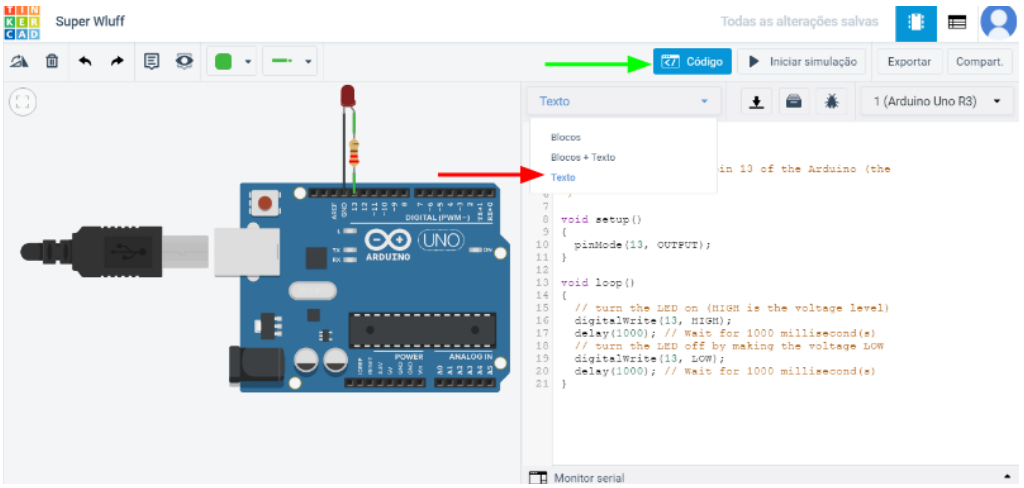
As placas de Arduino contém um conector USB para ligá-lo em um computador, o que servirá para transferir dados. Além disso, elas possuem diversas portas que permitem sua conexão a algum circuito eletrônico externo, como servo motores, sensores, alto falante, etc. Na Fig. 2, é possível visualizar um exemplo simples de circuito que utiliza o Arduino para piscar um LED no site Tinkercad.

Como utilizar o Arduino

Para a execução de alguma tarefa com a placa Arduino, é preciso, inicialmente, criar um programa de controle utilizando uma linguagem de programação baseada em C/C++ e enviá-lo para a placa através de software que pode ser adquirido no site oficial do Arduino.

No Tinkercad, é possível programar o arduino ao selecionar a placa nos dispositivos, clicar em CODE, como é indicado pela seta verde na Fig. 6; e depois em Texto, sinalizado pela seta vermelha.

Figura 6 - Programação do Arduino no Tinkercad.

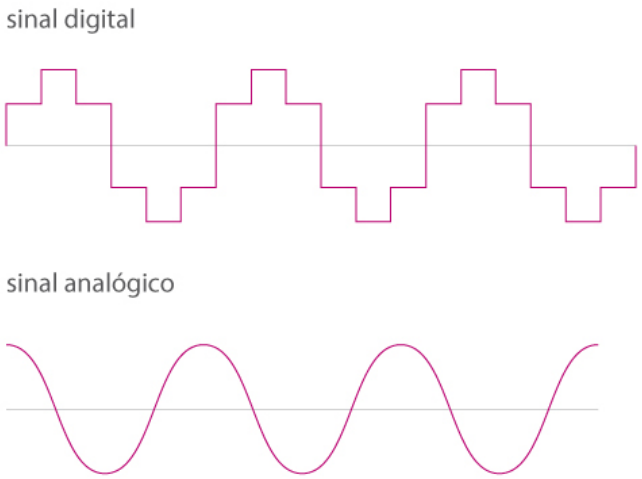


Fonte: Autor.

Portas do Arduino

Inicialmente, podemos citar dois tipos de sinais: analógico e digital. O primeiro é uma função contínua associada ao processo que se mede, um potênciometro, por exemplo, fornece um sinal analógico, pois é possível variar a resistência continuamente em um determinado intervalo. Por outro lado, os sinais digitais são representados por valores inteiros, ou seja, valores discretos. A Fig. 7 mostra como esses dois sinais se comportam em um determinado tempo.

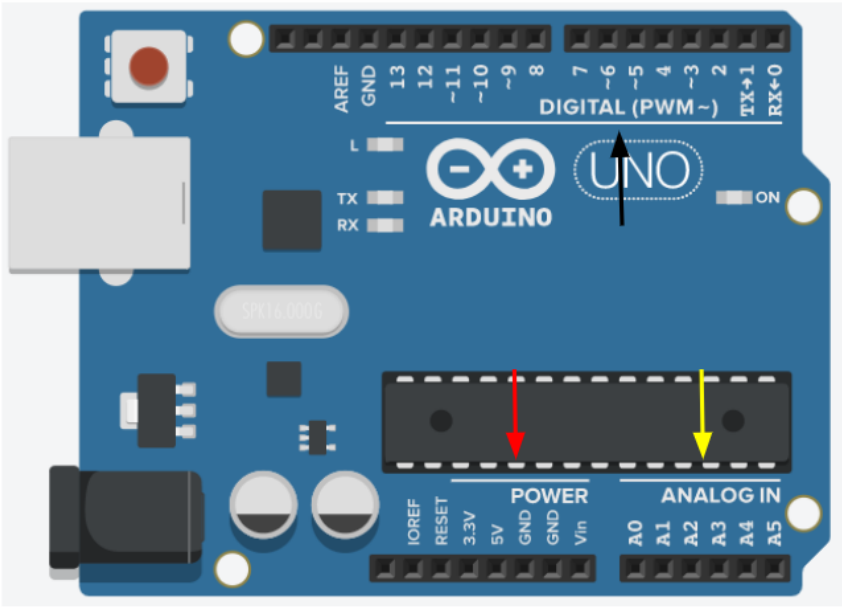
Figura 7 - Sinal digital e analógico.



Fonte: HomeIT (2021).

Com base nisso, o Arduino possui portas que tanto recebem quanto transmitem sinal analógico ou digital, o que é essencial para projetos de eletrônica. A Fig 8. mostra onde se encontram os pinos digitais, sinalizado pela seta preta, os pinos analogicos, sinalizados pela seta amarela. Além disso, também é indicado, pela seta vermelha, os pinos de Power, os quais são usados para alimentar o circuito que está sendo criado. Aqui é importante destacar que o Arduino é o cérebro do sistema e não a fonte de alimentação. Dessa forma, em projetos complexos que envolvem vários componentes, é necessário que o sistema seja alimentado por uma bateria ou qualquer outra fonte senão o Arduino pode ser danificado a ponto de não funcionar mais. Porém, em projetos pequenos, o Arduino pode ser utilizado como fonte.

Figura 8 - Sinal digital e analógico.



Fonte: Autor.

Para finalizar este capítulo, serão apresentados como criar alguns circuitos no Tinkercad utilizando o arduino. É importante citar que o próximo capítulo irá apresentar vários circuitos para mostrar conceitos básicos de eletrônica.

Projeto Piscando LED

Descrição:

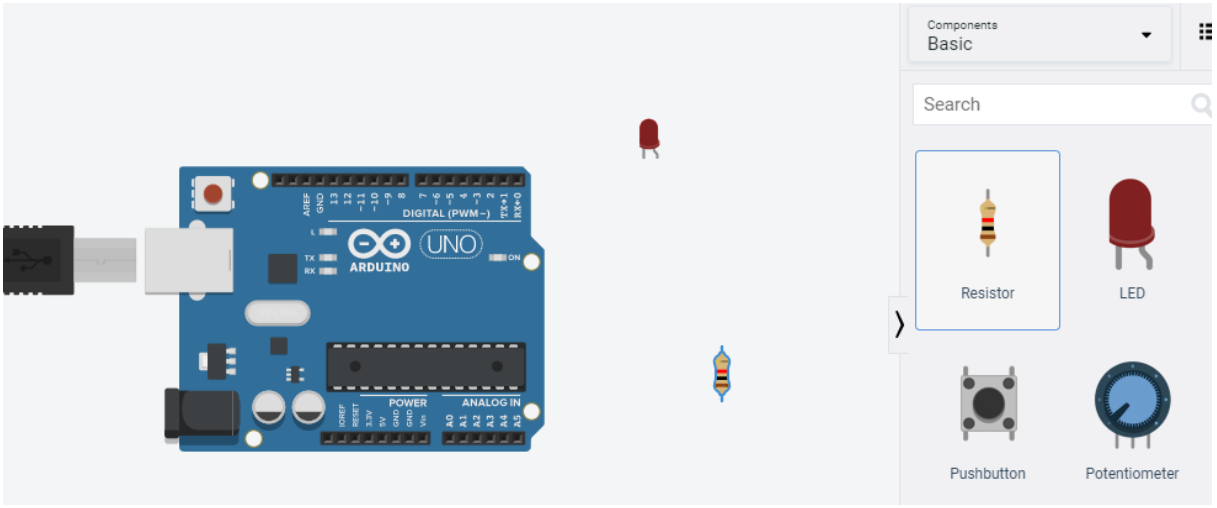
O presente projeto irá utilizar a plataforma Tinkercad para simular um circuito que irá fazer um LED piscar a cada 1 segundos com o Arduino.

Montagem do circuito:

- O circuito é composto:
  - 1 LED;
  - 1 Resistor de 100 Ω;
  - 1 Arduino;
- Montagem do circuito:

Etapa 1: Adicione o LED, o Arduino e o resistor na tela buscando “Led”, "Arduino" e “Resistor”, respectivamente, nos componentes, como é mostrado na Fig. 9.

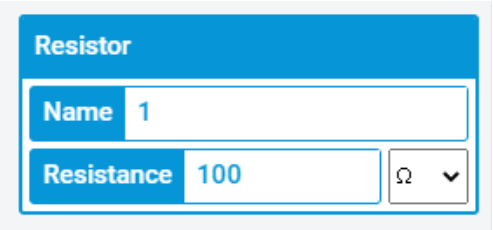
Figura 9 - Sinal digital e analógico.



Fonte: Autor.

Etapa 2: Configure o resistor para ter resistência de 100 Ω ao clicar nele, como é mostrado na Fig. 10.

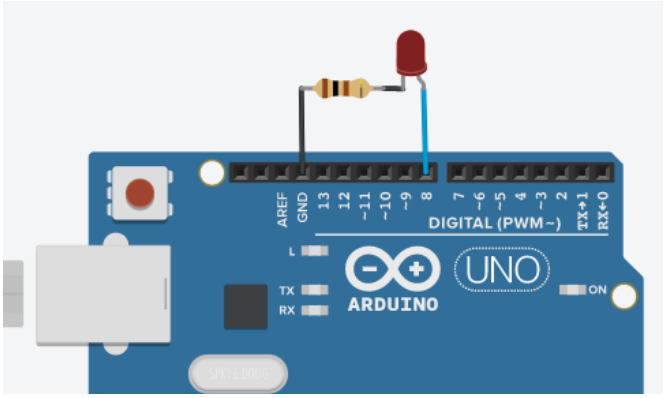
Figura 10 - Configuração do resistor.



Fonte: Autor.

Etapa 3: Conecte a porta GND do Arduino, o qual representa o polo negativo, em um dos pinos do resistor. Já a outra porta deste, conecte na pino menor do LED, pois o pino maior representa o polo positivo do LED. Para fechar o circuito, conecte o pino positivo do LED na porta digital 8 do Arduino, como é mostrado na Fig. 11.

Figura 11 - Ligações do circuito.



Fonte: Autor.

Programação:

A programação do arduino é dividida em 3 partes. Inicialmente tem-se as definições de variáveis, bibliotecas, etc. Após isso, existe o void setup(), onde se faz as configurações do arduino como qual porta vai ser de INPUT ou de OUTPUT, por exemplo. Após isso, tem-se o void loop(), onde coloca-se os comandos que serão executados e repetidos enquanto o arduino estiver ligado. Essa parte é essencial para colocar as ações de controle.

Com base nisso, a Fig. 12 mostra o código utilizado neste projeto, o qual precisa apenas estabelecer a o pino utilizado, se é de entrada ou saída e, no void loop, configurar o arduino para ligar e desligar o LED.

Figura 12 - Programação para piscar o LED.

```
void setup(){
  /*
Comando pinMode(porta,MODO): Este função permite configurar um pino específico
para se comportar como um pino de entrada ou de saída. Modos:
    INPUT: Pino de entrada.
    Output: Pino de saída.
*/
  pinMode(8, OUTPUT);//Configura verd_pedestre como porta de saída.
}

void loop(){
  /*
Comando digitalWrite(porta,valor): Caso a porta seja OUTPUT,
esta função permite configurar a tensão de saída do pino. Valores:
    HIGH: 5 V.
    LOW: 0 v.
*/
  digitalWrite(8, HIGH);//Faz a porta 8 ficar com 5 V, ligando o LED.
/*
Comando delay(tempo): pausa o programa por uma quantidade especifica de tempo
em milissegundos( 1000 = 1 s)
*/
  delay(1000); //Espera 1000 millisegundos
  digitalWrite(8, LOW);//Faz o Led desligar
  delay(1000); //Espera 1000 millisegundos
}
```

Fonte: Autor.

Projeto Potenciômetro

Descrição:

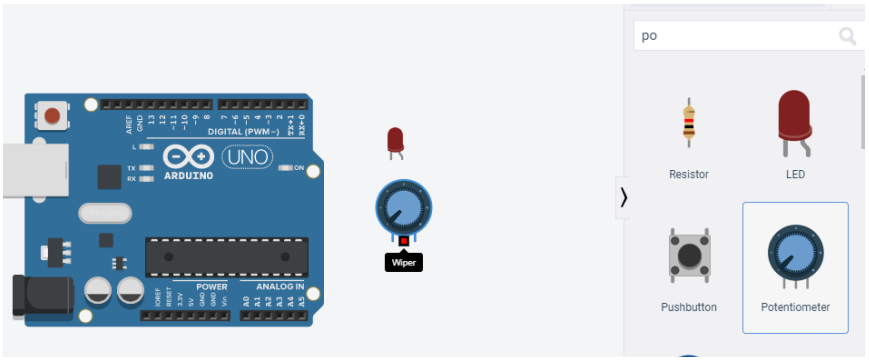
O presente projeto irá utilizar um potenciômetro para variar a intensidade da luz de um led e, também, mostrar, em um gráfico no monitor serial do arduino, os valores lidos dentro de um intervalo de 0 a 255, que, para o Arduino representam 0 e 5 V, respectivamente.

Montagem do circuito:

- O circuito é composto:
  - 1 LED;
  - 1 Potenciômetro (Características);
  - 1 Arduino;
  - 1 Protoboard;
- Montagem do circuito:

Etapa 1: Adicione o LED, o Arduino e o potenciômetro na tela buscando “Led”, "Arduino" e “Potentiometer”, respectivamente, nos componentes, como é mostrado na Fig. 13.

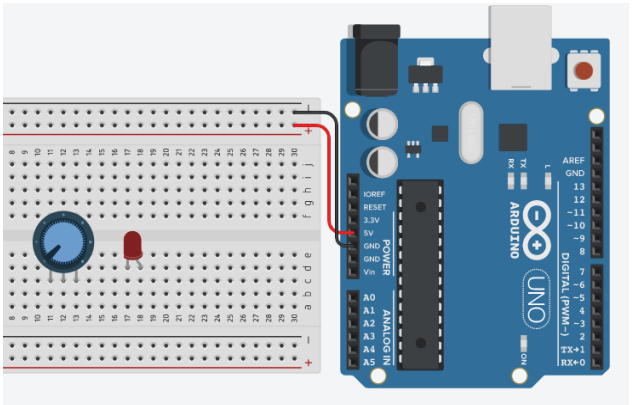
Figura 13 - Sinal digital e analógico.



Fonte: Autor.

Etapa 2: Para simplificar as ligações, adicione uma protoboard ao sistema buscando “potentiometer” nos componentes. Além disso, conecte o GND do arduino na protoboard bem como o 5V, como mostrado na Fig. 14.

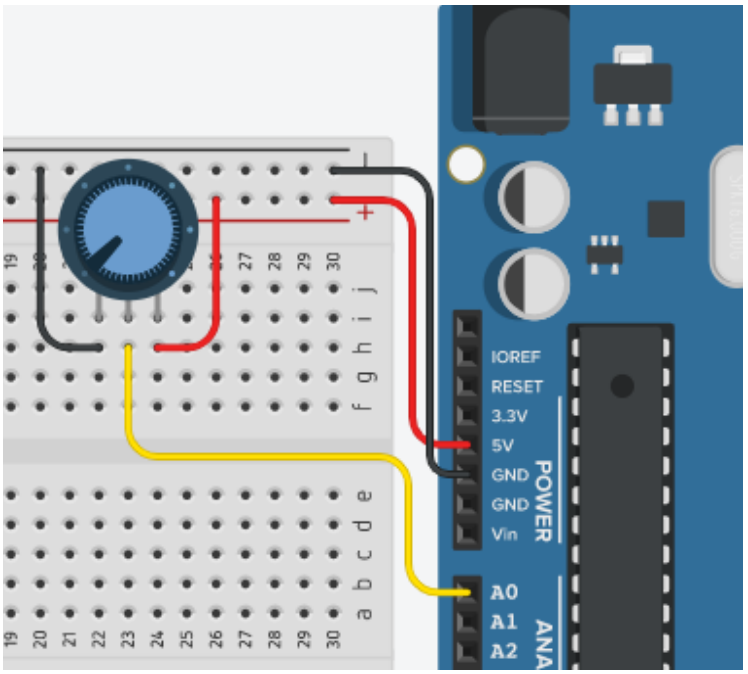
Figura 13 - Alimentação da protoboard.



Fonte: Autor.

Etapa 3: Alimente o potenciômetro conectado o polo positivo da protoboard no pino mais a direita e o polo negativo no pino mais a esquerda. O pino do meio do potenciômetro é usado para transmitir a resistência que ele gera, já que esse dispositivo varia esse valor ao rotacionar seu pino. Dessa forma, o potenciómetro transmite sinal analógico e esse pino do meio deve ser conectado na porta A0 do Arduino, como é mostrado na Fig. 14.

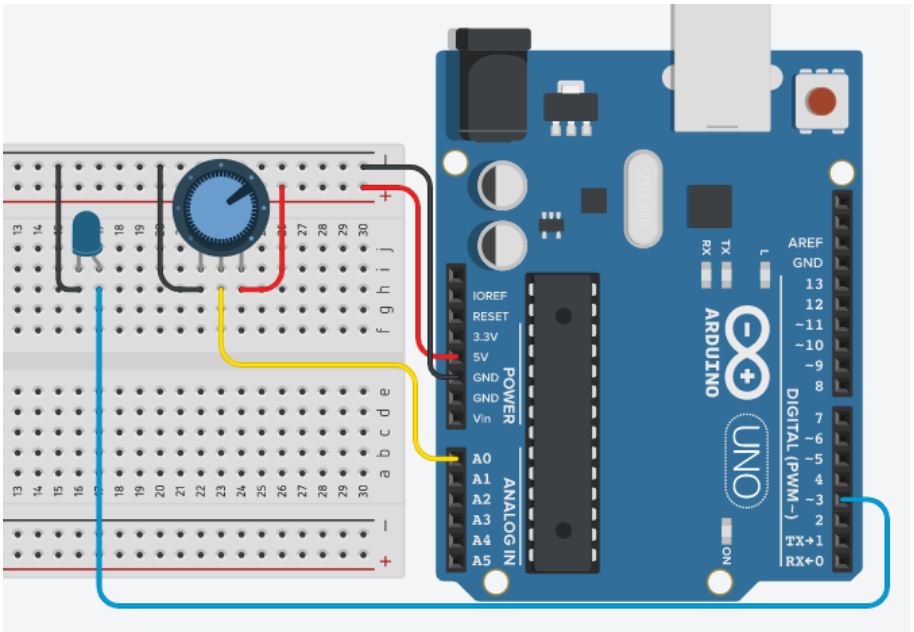
Figura 14 - Alimentação do potenciômetro e conexão no arduino.



Fonte: Autor.

Etapa 4: Conecte o polo negativo da protoboard no pino menor do LED e o pino maior na porta digital 3 do Arduino, como é mostrado na Fig. 15.

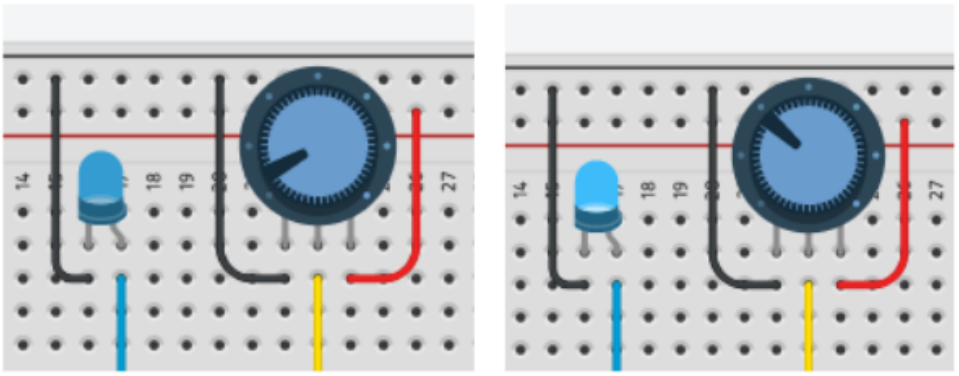
Figura 14 - Conexão do LED no Arduino.



Fonte: Autor.

Por fim, é importante ressaltar que, para modificar o valor da resistência transmitida pelo potenciômetro, basta clicar nele e rotacionar seu pino, como mostrado na Fig. 15.

Figura 15 - Variação da resistência do Potenciômetro.



Fonte: Autor.

Programação:

Inicialmente, define-se as variáveis que representam as portas do arduino e, também, a variável que irá armazenar o valor que será transmitido para o LED, como é mostrado na Fig. 16. Após isso, no setup, são configuradas as portas de entrada e saída e, por último, é utilizado o comando Serial.begin() para iniciar a comunicação serial e poder gerar o gráfico dos valores lidos.

Figura 16 - Configurações iniciais do código.

```
//Autor: Thiago Victor A. de Freitas - Estudante de Engenharia Mecânica (UFC)

//Inicialmente, devemos nomear quem vai está conectado a cada porta
int potenciometro = A0;//O potenciometro esta conectado na porta A0
int led = 3;//O LED esta conectado na porta 2
float valor_lido = 0;//variável que irá armazenar o valor
                     //que será transmitido para o LED

void setup(){
/*
Comando pinMode(porta,MODO): Este função permite configurar um pino específico
para se comportar como um pino de entrada ou de saída. Modos:
    INPUT: Pino de entrada.
    Output: Pino de saída.
*/
  pinMode(potenciometro, INPUT);//Configura a porta A0 como porta de entrada
  pinMode(led, OUTPUT);//Configura a porta 2 como porta de saisa
  Serial.begin(9600);//Iniciando a comunicação serial do arduino
//Este ultimo comando é necessário para criar o gráfico de valores lidos
}
```

Fonte: Autor.

Em relação ao void loop, primeiro é utilizado o comando analogRead para ler o valor da porta analógica, que está entre 0 e 1023. Na mesma linha, é utilizado o comando map para deixar esse valor entre 0 e 255, os quais representam 0 e 5 V para o arduino e, assim, será possível variar o brilho do LED com base na tensão. Após isso, utiliza-se o comando analogWrite para transmitir a tensão na porta a qual está conectado o LED, como é mostrado na Fig. 17.

Figura 17 - Void loop do código do potenciômetro.

```
22 void loop(){
23  //Comando analogRead(porta): Ler a porta analogica
24  /*
25  Comando map(valor, deMenor, deMaior, paraMenor, paraMaior):Remapeia um número
26  de um intervalo para outro. Isto é, um valor de deMenor seria mapeado para
27  paraMenor, um valor de deMaior para  paraMaior, valores dentro de uma faixa
28  para volores dentro da outra faixa, etc.
29 */
30     valor_lido = map(analogRead(potenciometro),0,1023,0,255);
31     //Pega o valor lido e remapeia para o sinal está entre 0 e 5 V.
32 /*
33 Comando analogWrite(porta,valor): Aciona uma onda PWM em um pino. Pode ser usada
34 para variar o brilho de um LED ou acionar um motor a diversas velocidades.
35 */
36     analogWrite(led,valor_lido);//Faz o LED brilhar na intensidade lida
37 /*
38 Comando Serial.println: Escreve e pula a linha no monitor serial do arduino
39 */
40     Serial.println(valor_lido);//Escreve no monitor serial
41 /*Comando delay(tempo): pausa o programa por uma quantidade especifica de tempo
42 em milissegundos( 1000 = 1 s)
43 */
44     delay(10);//Espera 10 millisegundos
45 }
```

Fonte: Autor.

Para finalizar, é utilizado o comando Serial.println() para escrever o valor lido no monitor serial. Dessa forma, é possível visualizar o esse no monitor serial de forma numérica clicando na opção “MONITOR SERIAL”, indicado pela seta verde na Fig. 18; e, também, é possível visualizar na forma de gráfico clicando na opção sinalizada pela seta vermelha.

Figura 18 - Setas indicando as opções do monitor serial .

Fonte: Autor.

Projeto Semáforo

Descrição:

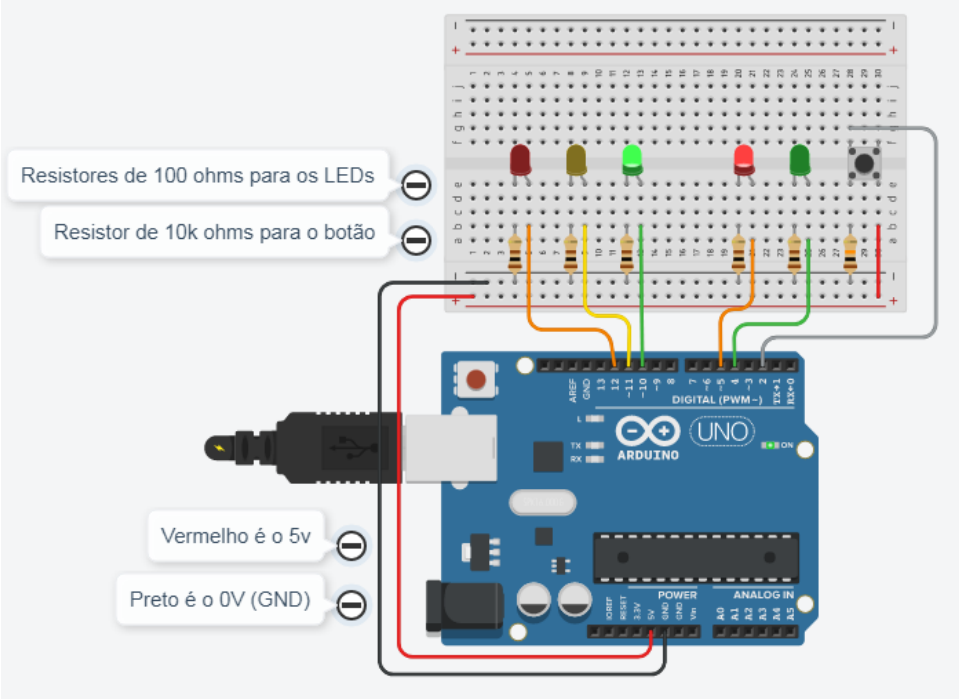
O presente projeto irá simular um semáforo, o qual tem-se o sinal dos carros e o sinal dos pedestres e, caso um pedestre queira passar, este aperta um determinado botão e espera o sinal ficar vermelho para os carros e verde para ele.

Montagem do circuito:

- O circuito é composto:
  - 2 LED verdes;
  - 2 LED vermelhos;
  - 1 LED amarelo;
  - 1 botão;
  - 5 resistores de 100 ohms;
  - 1 resistor de 10k ohms;
  - 1 Arduino UNO;
  - 1 protoboard;
- Montagem do circuito:

A Fig. 19 mostra o circuito, onde os LED à esquerda representam o semáforo para os carros enquanto os LED à direita representam o sinal para os pedestres. É importante ressaltar que os dispositivos estão conectados em portas digitais, o qual podem ser configuradas para fornecer 5V ou 0V. Além disso, a figura também mostra o circuito antes de ser apertado o botão.

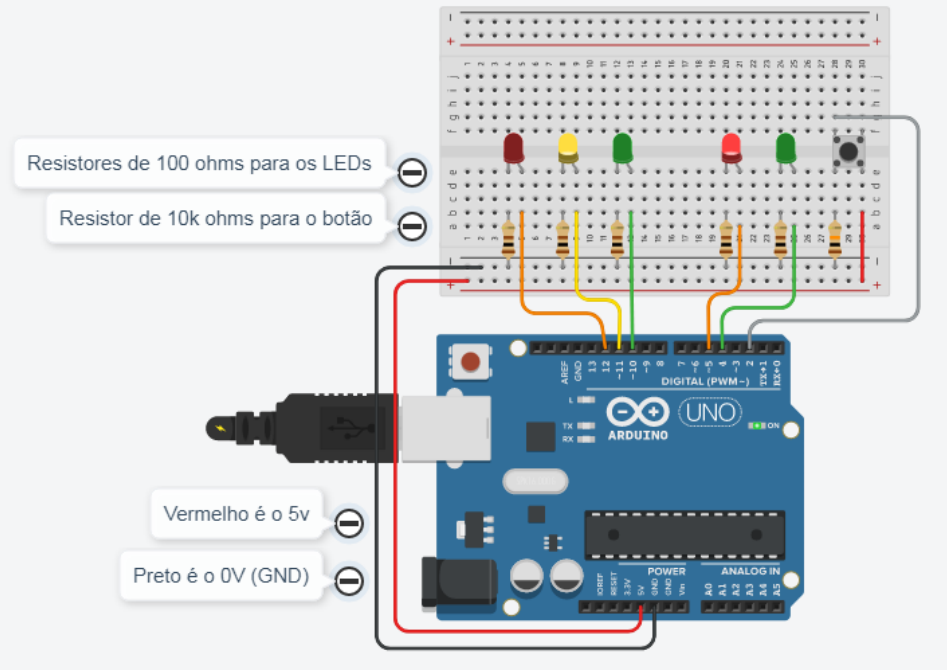
Figura 19 - Circuito do projeto semáforo.



Fonte: Autor.

Após pressionar o botão, a luz amarela acende e a luz verde se apaga para os carros, que podem ser vistos na Fig. 20.

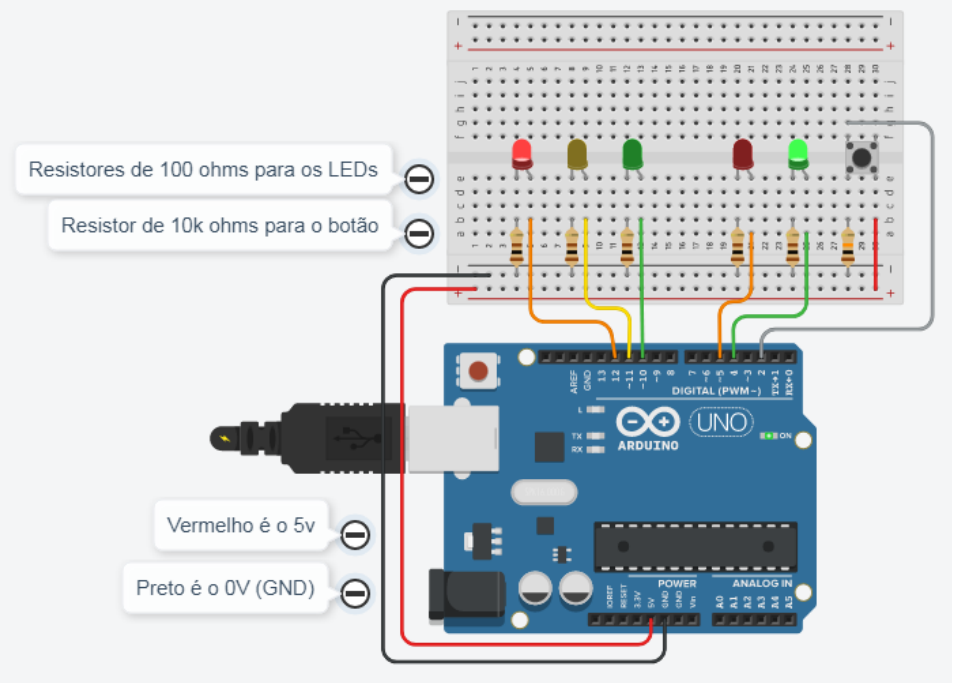
Figura 20 - Circuito do projeto semáforo após apertar o botão.



Fonte: Autor.

Por fim, após 2 segundos que o botão foi pressionado, o LED verde acendeu para os pedestres e o LED vermelho acendeu para os carros, como pode ser visto na Fig. 21. Após um determinado tempo, o circuito volta para a configuração da Fig. 19.

Figura 21 - Circuito do projeto semáforo 2 segundos após apertar o botão.



Fonte: Autor.

Programação:

Inicialmente, são definidas variáveis para representar cada Led e o botão, após isso é definido quais são as portas de entrada e saída com base nessas variáveis, como pode ser visto na Fig. 22.

Figura 22 - Primeira parte do código do projeto semáforo.

```
//Autor: Thiago Victor A. de Freitas - Estudante de Engenharia Mecânica (UFC)

//Inicialmente, devemos nomear quem vai está conectado a cada porta
int verd_pedestre = 4;//O led verde dos pedestre é a porta 4
int verm_pedestre = 5;//O led vermelho dos pedestre é a porta 5
int verd_carro = 10;//O led verde dos carros é a porta 10
int amare_carro = 11;//O led amarelo dos carros é a porta 11
int verm_carro = 12;//O led vermelho dos carros é a porta 12
int botao = 2;// O botão que os pedestres irão aperta está conectado a porta 4
void setup()
{
/*
Comando pinMode(porta,MODO): Este função permite configurar um pino específico
para se comportar como um pino de entrada ou de saída. Modos:
    INPUT: Pino de entrada.
    Output: Pino de saída.
*/
  pinMode(verd_pedestre, OUTPUT);//Configura verd_pedestre como porta de saída.
  pinMode(verm_pedestre, OUTPUT);
  pinMode(verd_carro, OUTPUT);
  pinMode(amare_carro, OUTPUT);
  pinMode(verm_carro, OUTPUT);
  pinMode(botao, INPUT);//Configura botao como porta de entrada.
}
```

Fonte: Autor.

No void loop, inicialmente, são ativados o Led verde para o carro e o Led vermelho para os pedestres. Com isso, é utilizado o comando IF para colocar as ações de controle. Para exemplificar, se o botão for apertado, faça isso, senão for apertado faça outra coisa, como pode ser visto na Fig. 23.

Figura 23 - Segunda parte do código do projeto semáforo.

```
void loop()
{
/*
Comando digitalWrite(porta,valor): Caso a porta seja OUTPUT,
esta função permite configurar a tensão de saída do pino. Valores:
    HIGH: 5 V.
    LOW: 0 v.
*/
  digitalWrite(verm_pedestre, HIGH);//Acende o led vermelho para os carros
  digitalWrite(verd_carro, HIGH);//Acende o led vermelho para os carros
  if (digitalRead(botao) == HIGH) {
/* Se o botão for apertado, ou seja, se a porta em que o botao estiver ligado
receber uma tensão, então ele irá fazer tudo que está entre as chaves
*/
    digitalWrite(verd_carro, LOW);//Apaga o led verde para os carro
    digitalWrite(amare_carro, HIGH);//Acende o led amarelho para os carros
/*Comando delay(tempo): pausa o programa por uma quantidade especifica de tempo
em milissegundos( 1000 = 1 s)
*/
    delay(1000); //Espera 1000 millisegundos
    digitalWrite(amare_carro, LOW);//Apaga o led amarelo para os carro
    digitalWrite(verm_carro, HIGH);//Acende o led vermelho para os carros
    digitalWrite(verm_pedestre, LOW);//Apaga o led vermelho para os pedestre
    digitalWrite(verd_pedestre, HIGH);//Acende o led verde para os pedestre
    delay(3000); //Espera 3000 millisegundos
    digitalWrite(verd_pedestre, LOW);//Apaga o led verde para os pedestres
    digitalWrite(verm_carro, LOW);//Apaga o led vermelho para os carro
  }
}
```

Fonte: Autor.

Referências:

- Site Tinkercad da Autodesk
- Site do Arduino
- AUTOMAÇÃO RESIDENCIAL ARDUINO: IDEIAS PARA DEIXAR SUA CASA IGUAL A DO HOMEM DE FERRO
- Livro do autor França sobre instrumentação
- Vídeo explicativo do Manual do mundo sobre o Arduino
- Canal Brincando com idéias que tem vários conteúdos sobre arduino
- **Circuito do projeto Piscando Led**
- **Circuito do projeto Potênciometro**
- **Circuito do projeto semáforo criado pelo autor**