03 de junho de 2019

CURSO ARDUÍNO PRÁTICO

Instrutores: Rafael e Ruben

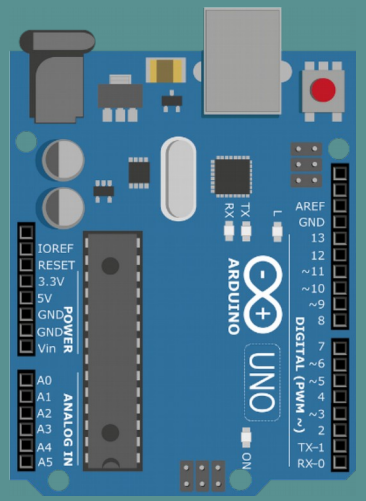
Simplificando o processo de um computador...

Entrada

Processo

Saída

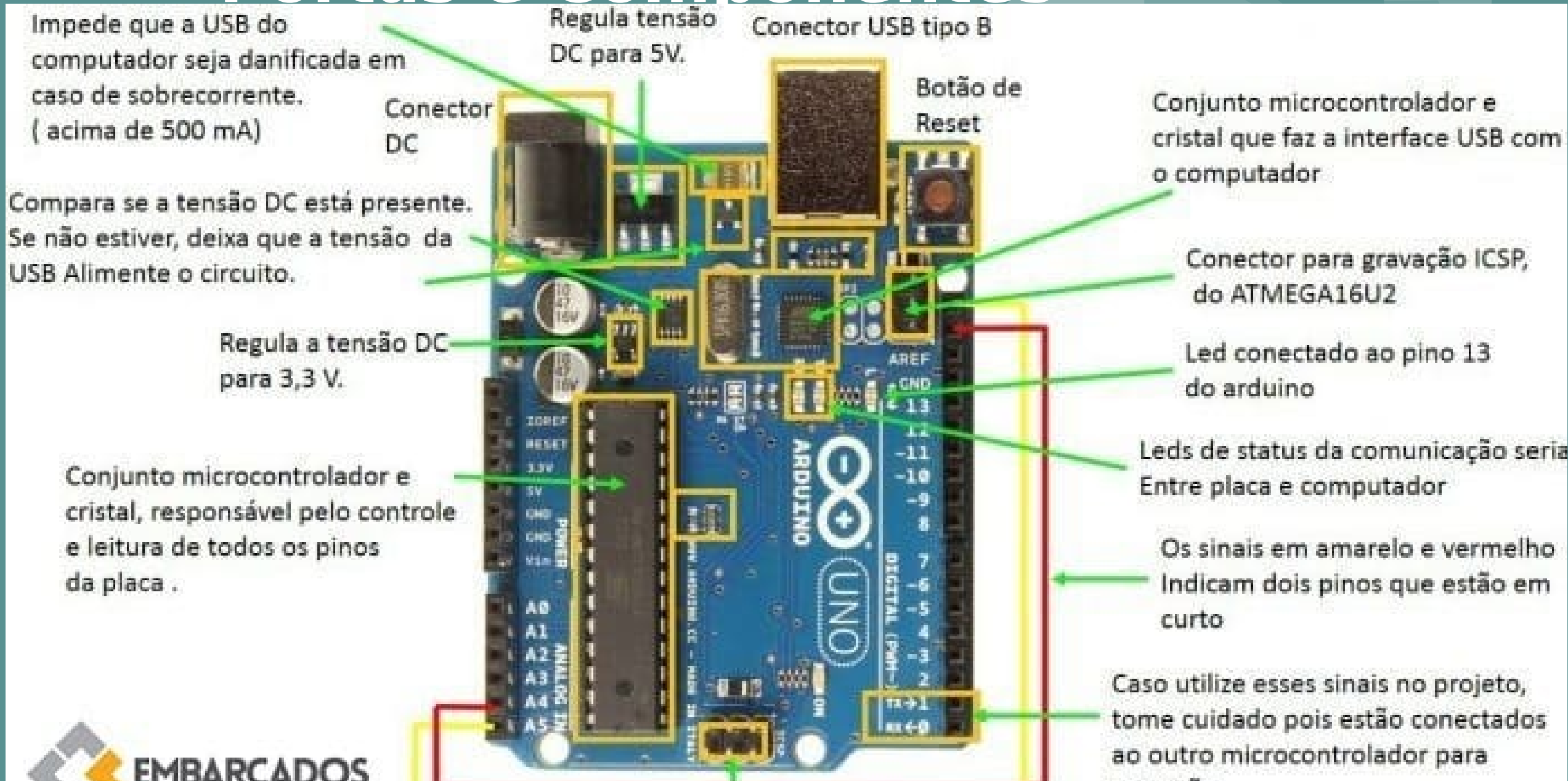
... para o arduino.



Entrada **Processo** **Saída**

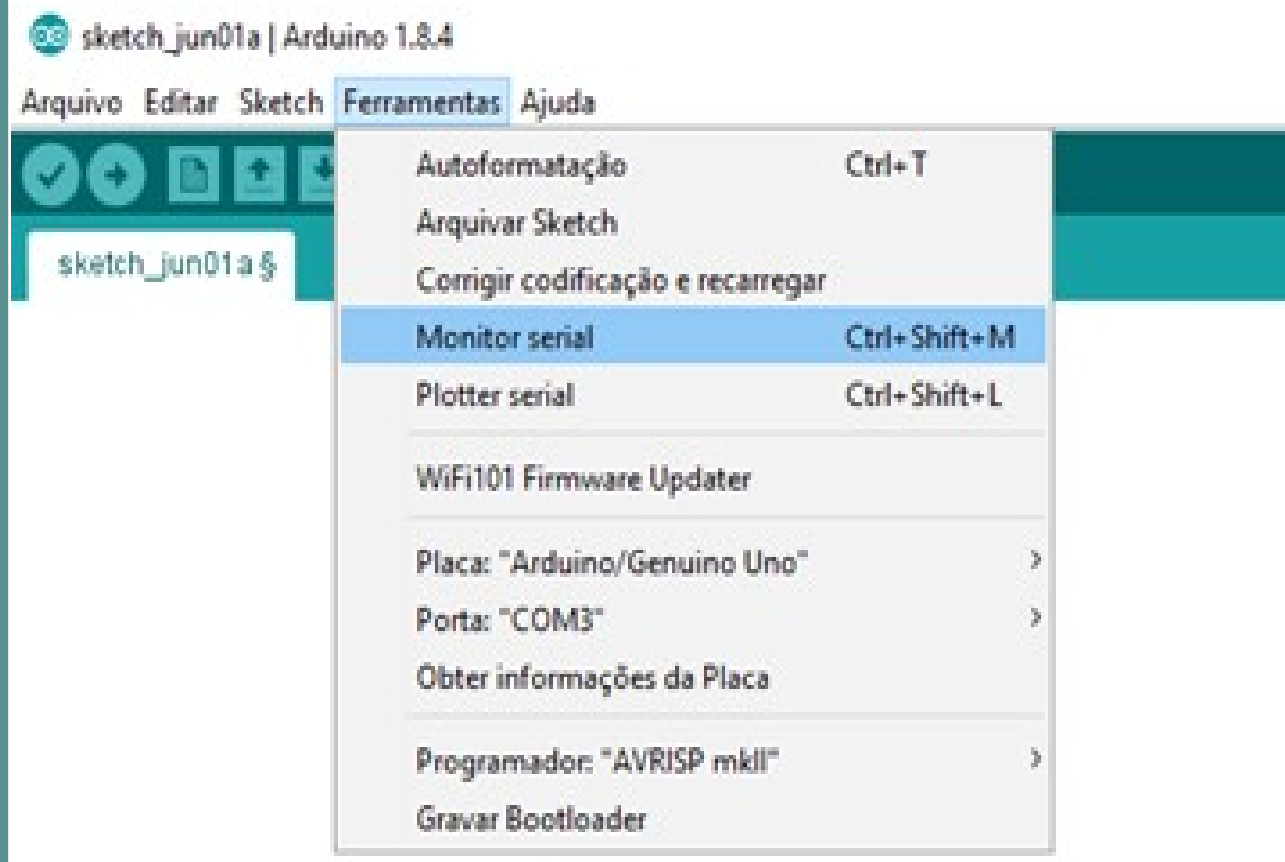
Sensor > Processo > Atuador

Portas e Componentes



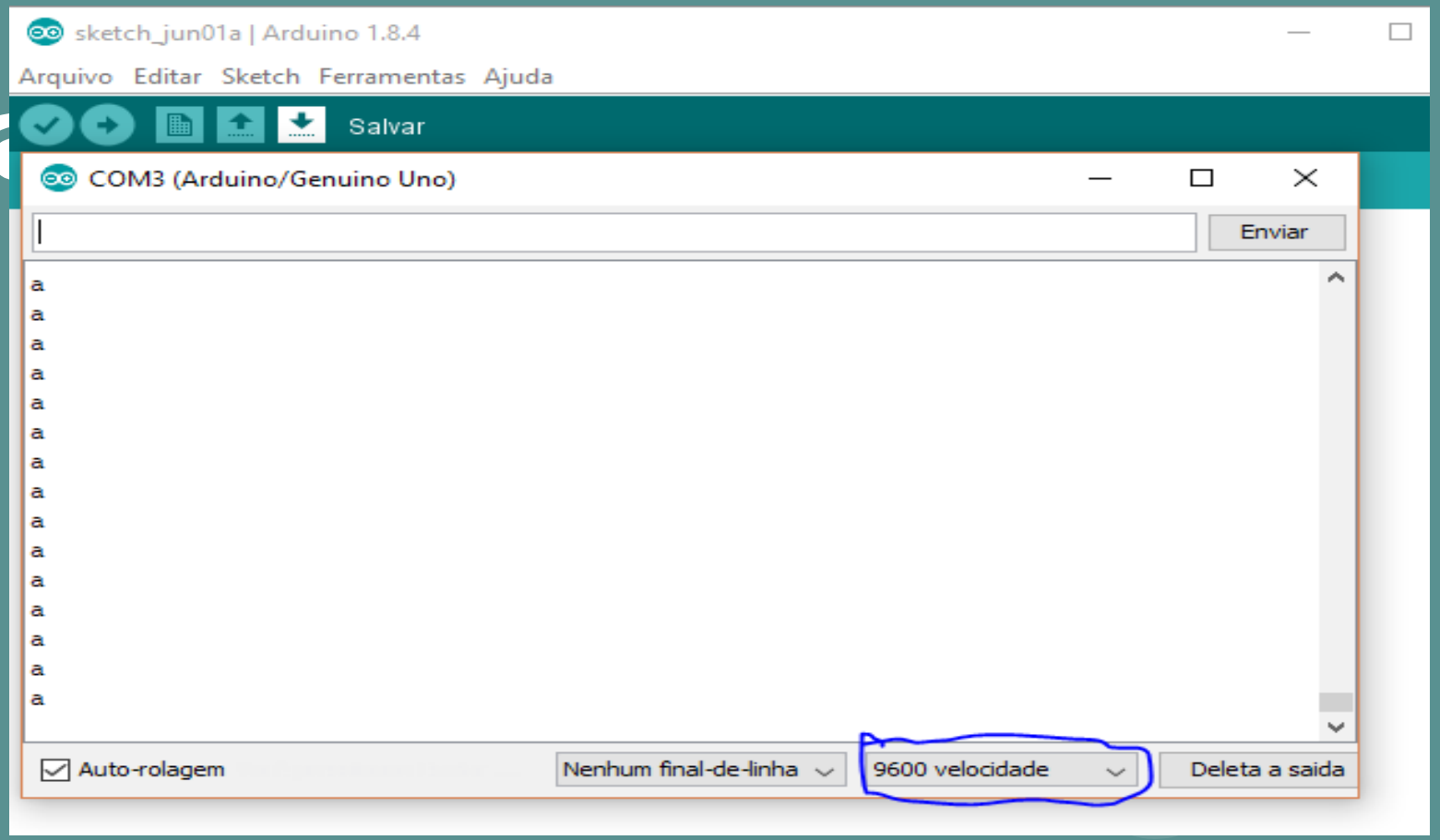
EMBARCADOS

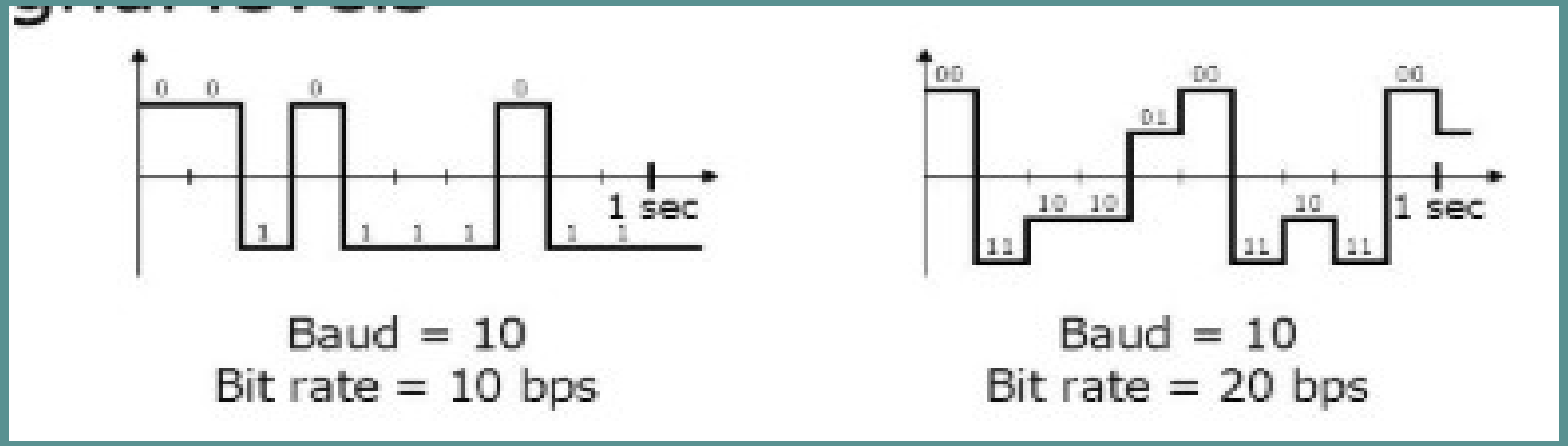
Monitor Serial



Monitor Seria

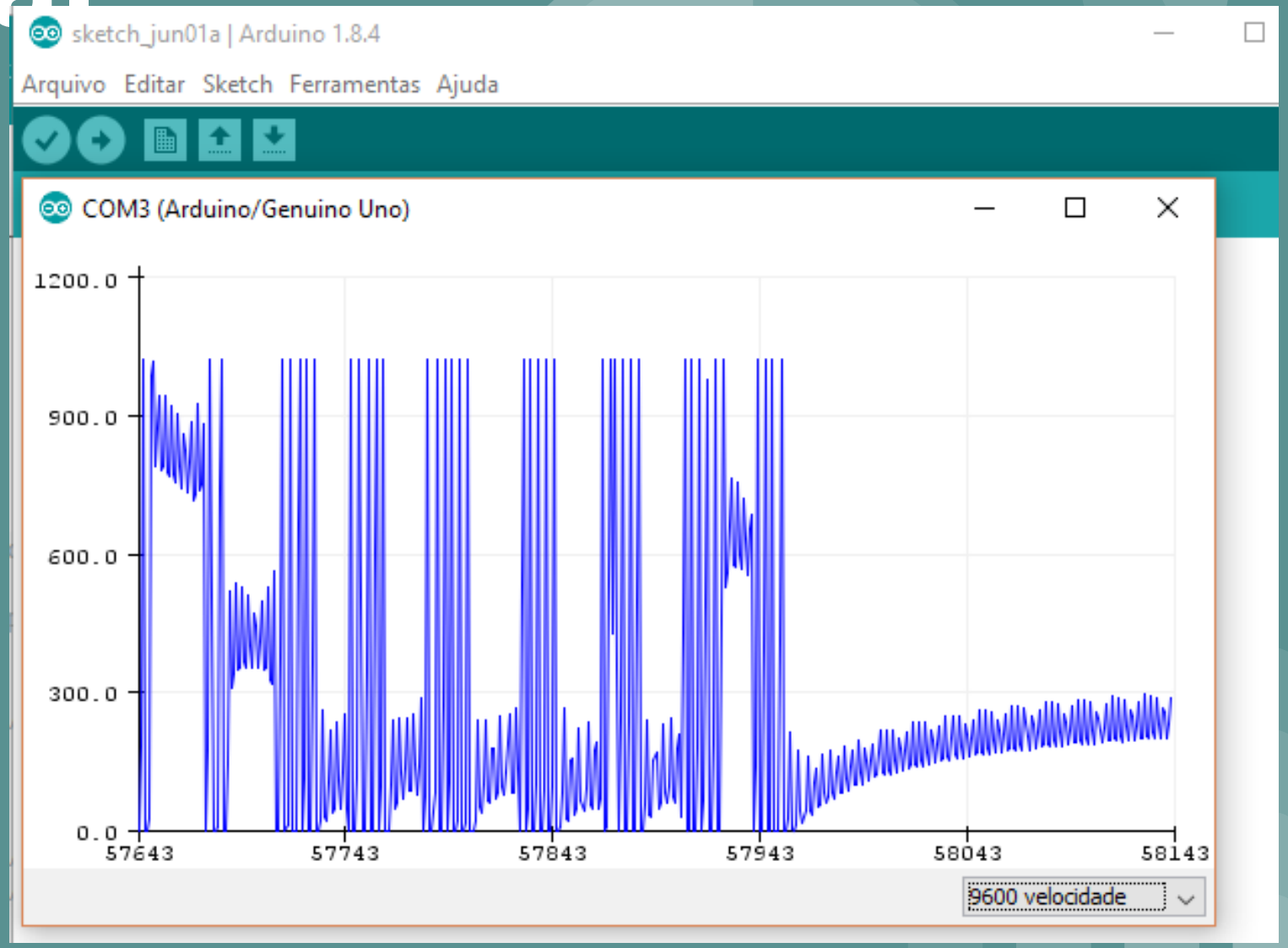
Baud rate
vs
Bit rate





Monitor Serial

Plotter Serial



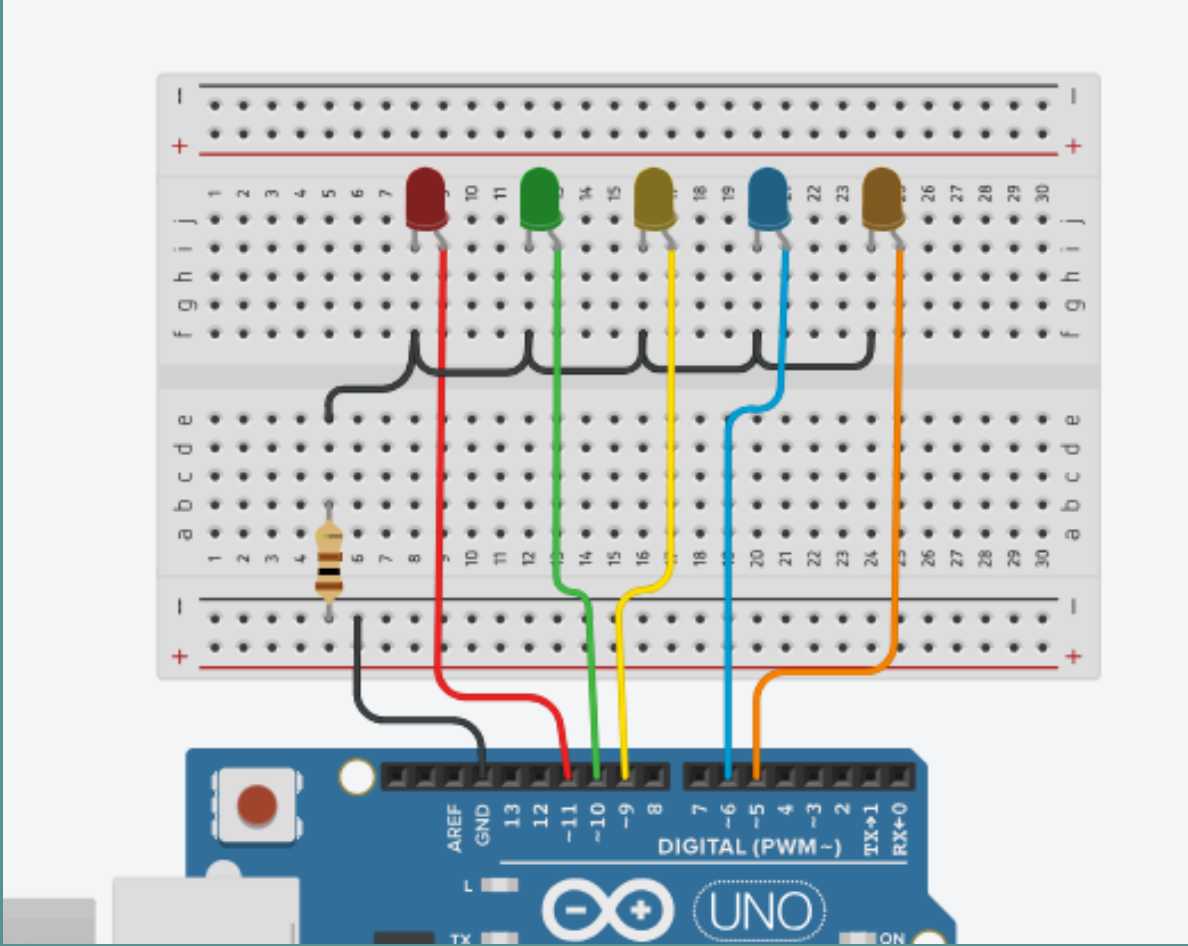
Monitor Serial

Vejamos na prática!

Deixando os monitores Serial em “stand by”, vamo para um Show de Luzes!

Projeto 01 – Show de luzes

Monte o esquema:
Componentes:



- 1 – resistor 100 Ohms
- 5 - LEDS

DIGITE O CÓDIGO NA IDE

```
int arrayLed[] = {11, 10, 9, 6, 5};
int tamanhoArray = sizeof(arrayLed) / sizeof(arrayLed[0]);
```

```
void apaga()
{
  for (int i =0; i<tamanhoArray; i++)
  {
    digitalWrite(arrayLed[i], LOW);
  }
}
```

```
void acende()
{
  for (int i =0; i<tamanhoArray; i++)
  {
    digitalWrite(arrayLed[i], HIGH);
  }
}
```

```
void trocarEstadoLed(char estado)
{
  switch (estado)
  {
    case '0':
      apaga();
      break;
    case '1':
      acende();
      break;
    case '2':
      sequencial();
      break;
  }
}
```

```
void sequencial()
{
  for (int i=0; i < tamanhoArray; i++)
  {
    digitalWrite(arrayLed[i], HIGH);
    delay (150);
    digitalWrite(arrayLed[i], LOW);
    delay (150);
  }
}
```

```
void setup()
{
  for (int i =0; i<tamanhoArray; i++)
  {
    pinMode(arrayLed[i], OUTPUT);
  }
}
```

```
void loop()
{
    trocarEstadoLed(1);
}
```

setup() – Parte do código dedicado para iniciar o programa. Dizer para o arduino com quais portas ele deve trabalhar, além de iniciar algumas funções e modos de trabalho. Esta parte é lida uma única vez.

loop() – Parte do código dedicado ao trabalho repetitivo. Esta parte roda infinitas vezes.

Função pinMode()

- **Sintaxe : pinMode(PORTA, FORMATO);**
- **As portas podem ser as portas analógicas A0,A1,A2..A7, ou digital 0,1,2,3...13.**
- **O formato pode ser INPUT para sinais que chegam ao arduino, e OUTPUT para sinais enviados pelo arduino.**

Função digitalWrite()

- Serve para enviar um sinal digital, verdadeiro ou falso...5v ou 0v.
- Sintaxe : digitalWrite(PORTA, FORMATO);
- As portas podem ser as portas analógicas A0,A1,A2..A7, ou digital 0,1,2,3...13.
- O formato pode ser HIGH (ou 1) para enviar 5V, e LOW (ou 0) para enviar 0V.

Função delay()

- Serve para espera um tempo em milisegundos.
- Sintaxe : delay(tempo);
- O tempo é dado em milissegundos.

Vamos voltar para o Monitor Serial?

Projeto 1.1 – Controlando o Show de luzes pela Serial.

Adicionando alguns códigos ao anterior

►Variável global:

```
char entrada;
```

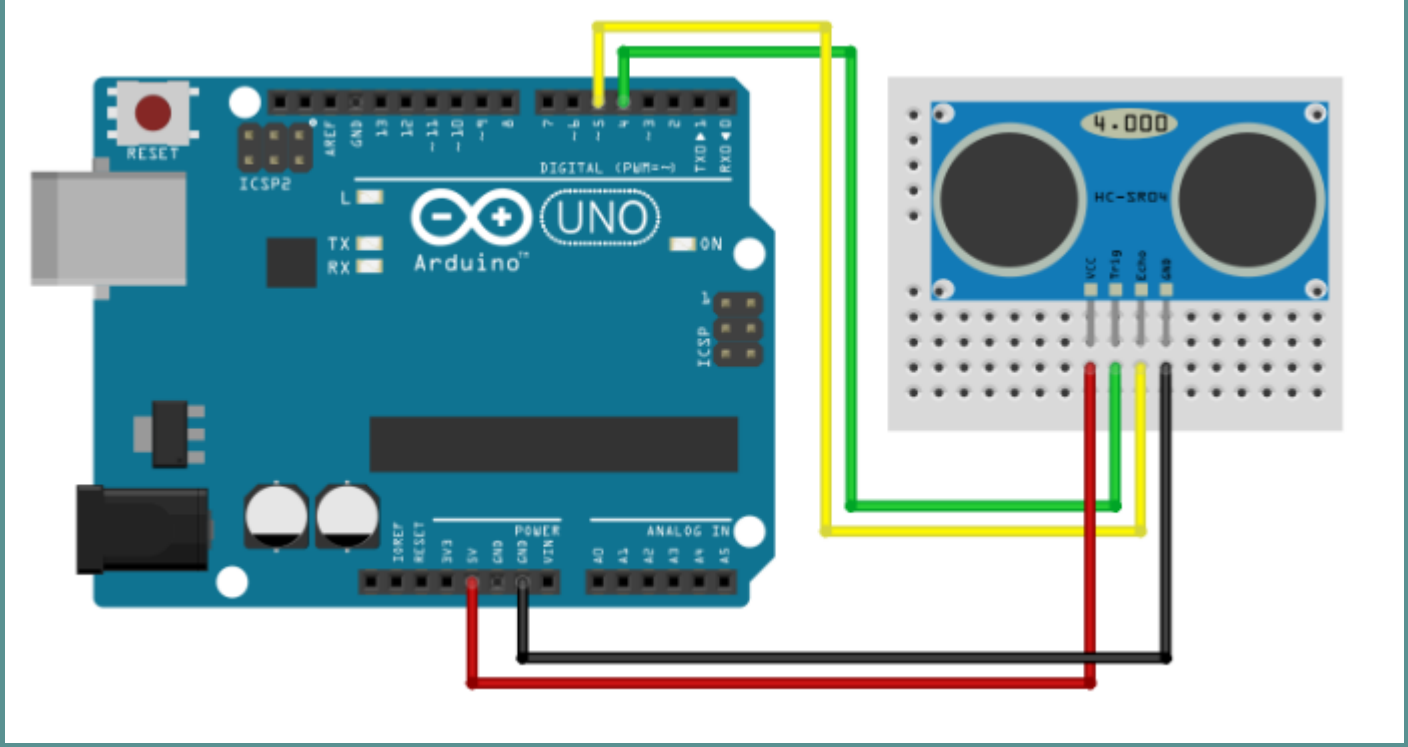
Atualizando a loop():

```
void loop()
{

  if (Serial.available() > 0)
  {
    // Lê byte da serial
    entrada = Serial.read();
    Serial.println(entrada);
  }
    trocarEstadoLed(entrada);
}
```

MELHOR FORMA DE APRENDER É PRATICANDO.

Projeto 02 – Sensor ultrassônico

Monte o esquema:
Componentes:



- **1 – Sensor Ultrassônico**
- **Library:**

  https://github.com/ErickSimoes/Ultrasonic

DIGITE O CÓDIGO NA IDE

```
/**
 * Leitura de distância com o sensor HC - SR04
 * Library used: https://github.com/ErickSimoes/Ultrasonic
 */
#include <Ultrasonic.h>
  /*
   * Definindo o nome do sensor (ultrassom) - um objeto e
   * especificando as portas de conexão trig(8) e echo(7)
   * respectivamente.
   */
  Ultrasonic ultrassom(8,7);

//Função setup() roda apenas 1 vez quando a placa é ligada ou resetada
void setup()
{
  //Habilitando a comunicação serial a uma taxa de 9600 bauds.
  Serial.begin(9600);
}
//Definindo a variável (global) que receberá a distância
long distancia;

//Função loop(): executa um loop infinitamente enquanto a placa estiver ligada.
void loop()
{
  distancia = ultrassom.read(CM); //ultrassom.read(CM) retorna a distância em
                                  //centímetros (CM)
  Serial.print(distancia); //imprime o valor da variável distancia
  Serial.println(" cm");

  delay(100); //Aguarda 100ms antes de executar o loop novamente
}
```

Lembra do Plotter?

Filtros são usados para reduzir ruídos vindos de sinais.

Vamos praticar?!

PROJETO 3 – FILTRO LINEAR

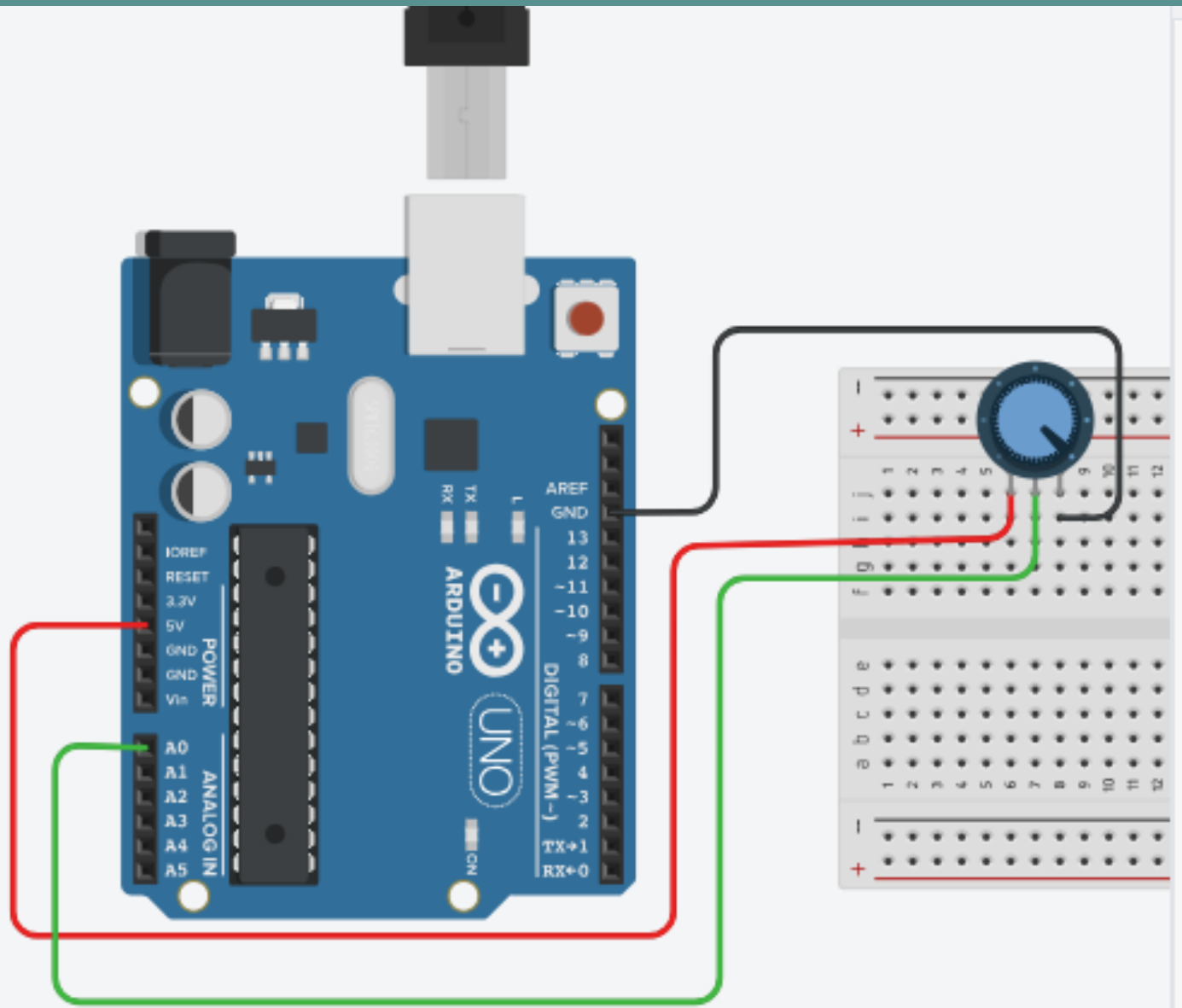
MODELO DE CÓDIGO

```
#define N_INTERACAO 700
int filtroLinear(int porta)
{
  unsigned long sinalFiltrado = 0;

  for (int i = 0; i < N_INTERACAO; i++)
  {
    sinalFiltrado += analogRead(porta);
  }
  return sinalFiltrado / N_INTERACAO;
}
```

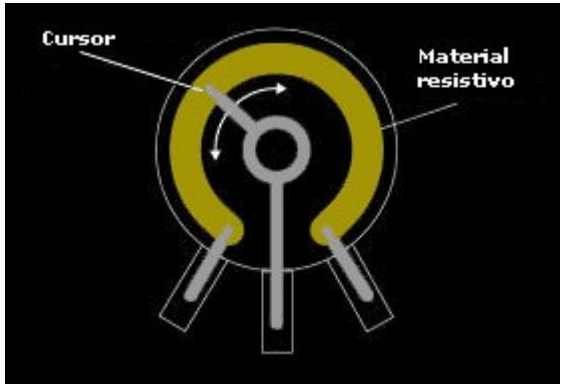
Experimente passar o filtro linear no sinal lido por um potenciômetro

PROJETO 3.1 – FILTRO LINEAR COM POTENCIÔMETRO

SIGA O ESQUEMA:



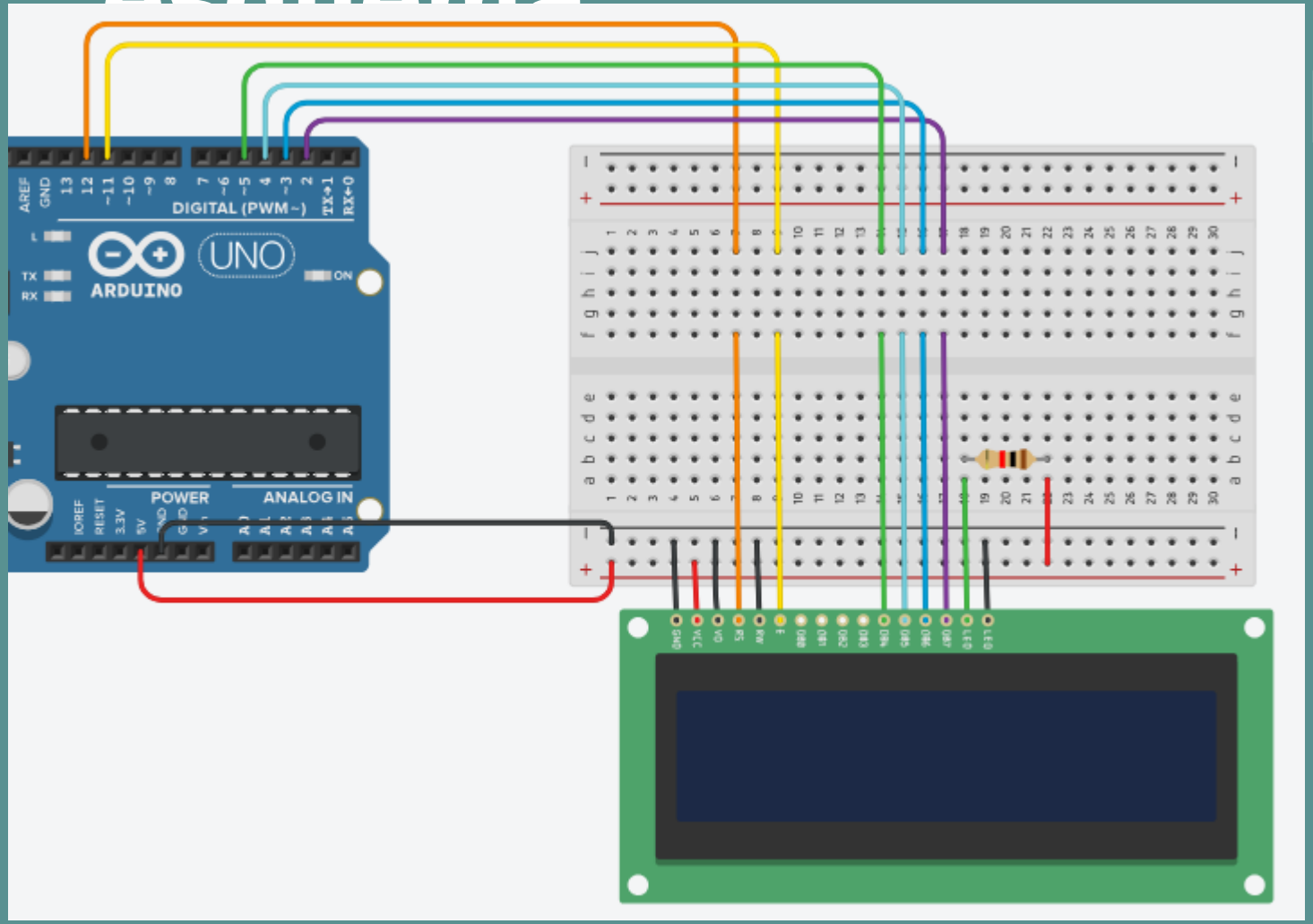

```arduino
#define POT A0

void setup()
{
  Serial.begin(9600);
  pinMode(POT, INPUT);

}

void loop()
{
    int valor = 0;
    valor = analogRead(POT);
    Serial.print("valor : ");
    Serial.print(valor);
}
```



Vamos aprender a utilizar o LCD (Light Cristal Display)

- PROJETO 4 – LCD

Monte o esquema:



Descrição dos pinos

Pino	Nome	Função
1	Vss	Terra
2	Vdd	Positivo (normalmente 5V)
3	Vo	Contraste do LCD. Às vezes também é chamado de Vee
4	RS	Register Select
5	R/W	Read/Write
6	E	Enable
7	D0	Bit 0 do dado a ser escrito no LCD (ou lido dele).
8	D1	Bit 1 do dado a ser escrito no LCD (ou lido dele).
9	D2	Bit 2 do dado a ser escrito no LCD (ou lido dele).
10	D3	Bit 3 do dado a ser escrito no LCD (ou lido dele).
11	D4	Bit 4 do dado a ser escrito no LCD (ou lido dele).
12	D5	Bit 5 do dado a ser escrito no LCD (ou lido dele).
13	D6	Bit 6 do dado a ser escrito no LCD (ou lido dele).
14	D7	Bit 7 do dado a ser escrito no LCD (ou lido dele).
15	A	Anodo do back-light (se existir back-light).
16	K	Catodo do back-light (se existir back-light).

Tabela 1: Descrição das funções dos pinos do LCD

DIGITE O CÓDIGO :

```
// Inclue a biblioteca:
#include <LiquidCrystal.h>

// Inicializa o LCD
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  // Inicia o LCD com suas dimensões
  lcd.begin(16, 2);
  // Imprime uma mensagem no LCD
  lcd.print("hello, world!");
}

void loop() {
  //Coloca o cursor na coluna 1 e linha 0
  lcd.setCursor(0, 1);
  // Imprime o tempo de operação
  lcd.print(millis() / 1000);
}

```

Algo de uso mais prático...

- PROJETO 4.1 – LETREIRO ELETRÔNICO

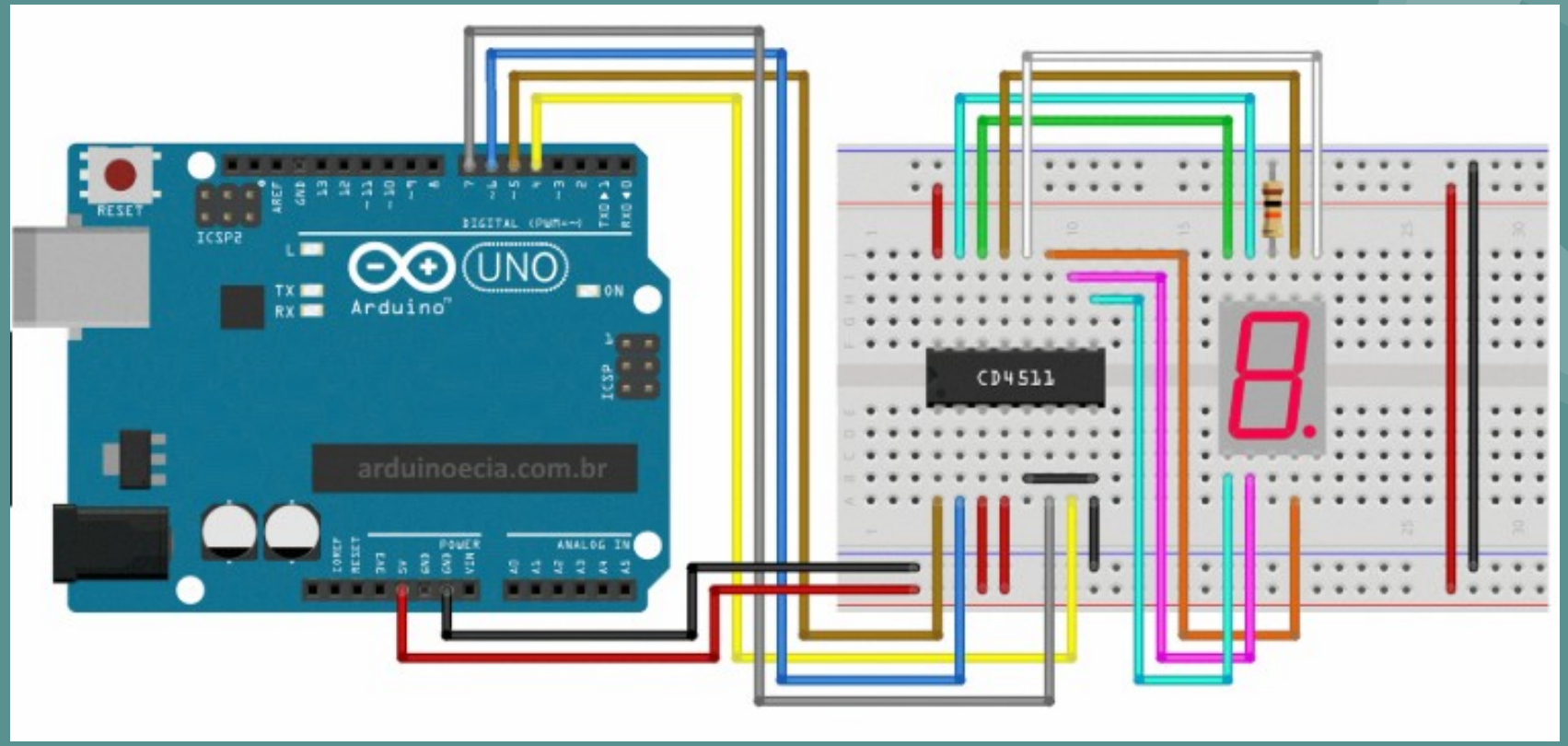
NO CÓDIGO ANTERIOR, INCLUA E FAÇA USO DESTA FUNÇÃO:

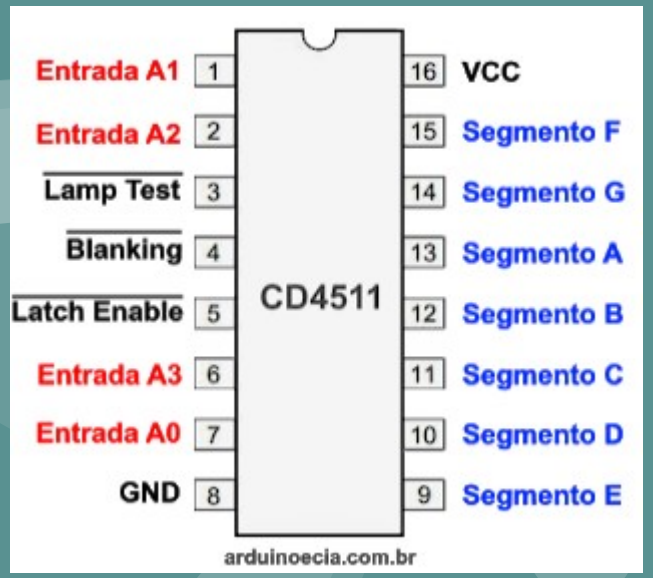
```
String lerSerial ()
{
  String conteudo1 = "";
  char entrada1;

  // Se receber algo pela serial
  while (Serial.available() > 0) {
    //Segue fazendo a leitora de cada byte enviado
    entrada1 = Serial.read();

    //Ignora sinal/caractere de ENTER
    if (entrada1 != '\n') {
      conteudo1.concat(entrada1);
    }
    // Aguarda buffer serial ler próximo caractere
    delay(10);
  }
  //Quando não houver mais bytes a ler...
  if (Serial.available() <= 0) {
    return conteudo1;
  }
}
```

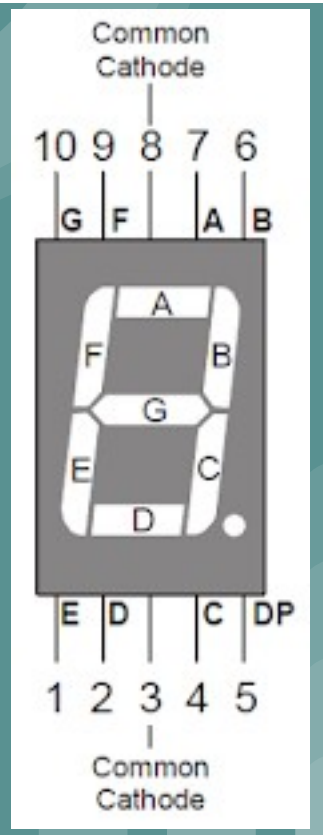
USANDO CIRCUITO INTEGRADO

- PROJETO 5 – Led de 7 segmentos com CI Decodificador

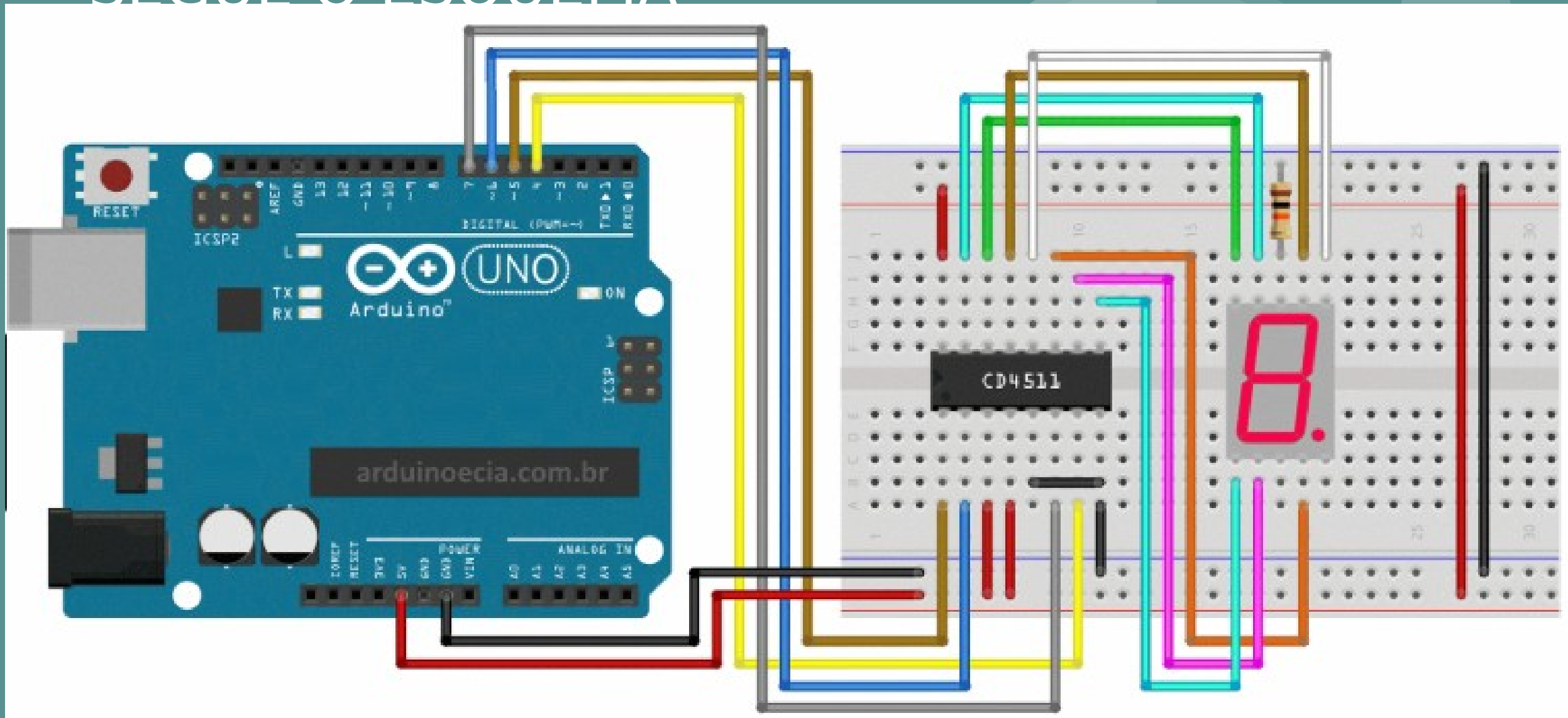
SEGUE O ESQUEMA







SEGUE O ESQUEMA



O CÓDIGO

```
//Definicao dos pinos de entrada me relação ao CI
#define PinoA0 4
#define PinoA1 5
#define PinoA2 6
#define PinoA3 7
void setup()
{
  Serial.begin(9600);
  //Define os pinos como saida
  pinMode(PinoA0, OUTPUT);
  pinMode(PinoA1, OUTPUT);
  pinMode(PinoA2, OUTPUT);
  pinMode(PinoA3, OUTPUT);
}
void loop()
{
  Serial.print("Numero: 0 ");
  digitalWrite(PinoA0, LOW);
  digitalWrite(PinoA1, LOW);
  digitalWrite(PinoA2, LOW);
  digitalWrite(PinoA3, LOW);
  delay(1000);

  Serial.print("1 ");
  digitalWrite(PinoA0, HIGH);
  digitalWrite(PinoA1, LOW);
  digitalWrite(PinoA2, LOW);
  digitalWrite(PinoA3, LOW);
  delay(1000);
```

UMA CURIOSIDADE...

- “RESETANDO” A PROGRAMAÇÃO VIA CÓDIGO

O CÓDIGO

```
void (*reset)(void) = 0;
void setup() {
  Serial.begin(9600);
  Serial.println("Comando 1 Executado!");

  delay(1000);
  reset();
  Serial.println("Comando 2 Executado!");
}
void loop() {
}
```

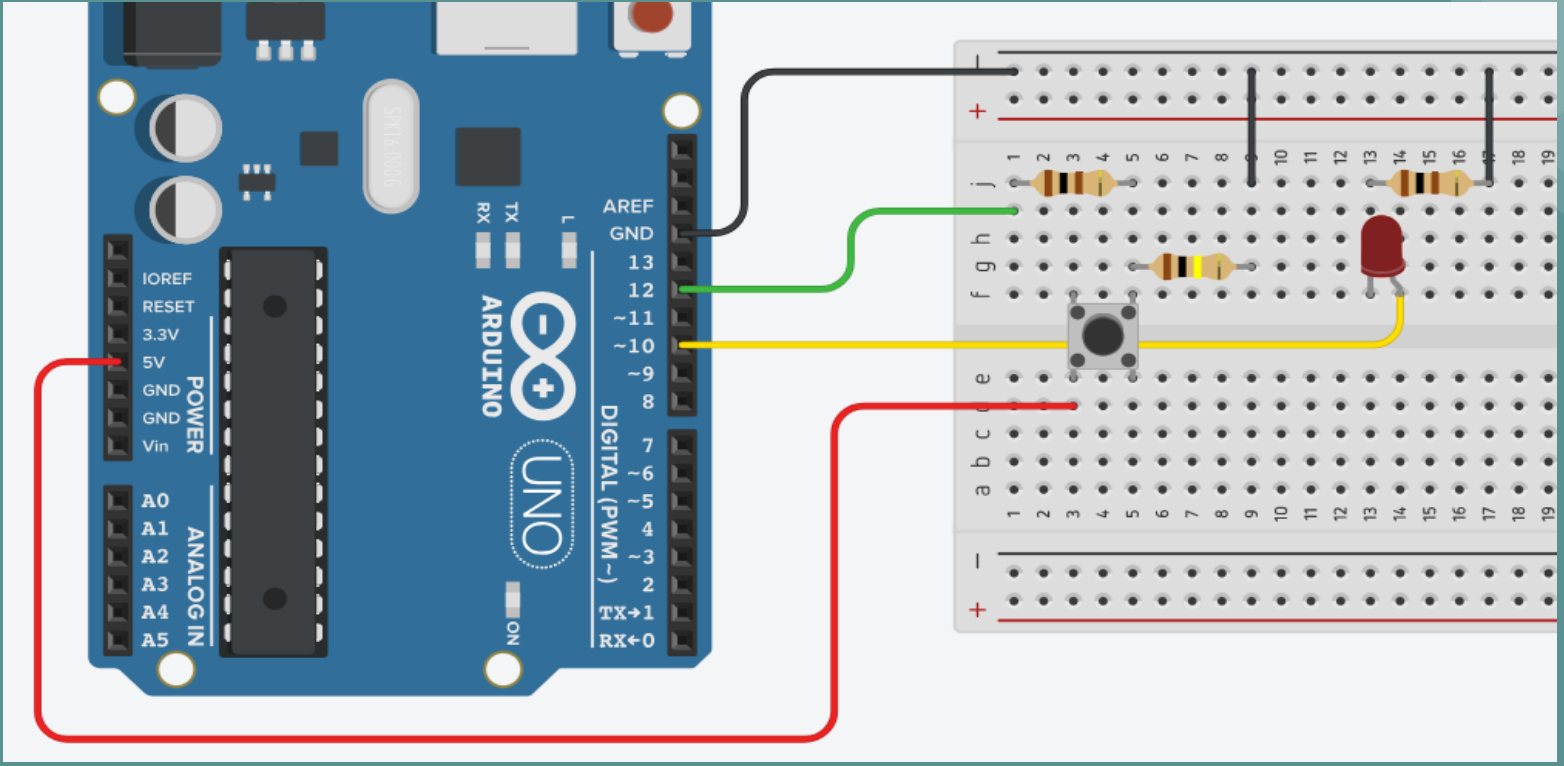
Instrução que levará o controlador a ler a posição inicial da memória onde está o código a ser executado.

Para projeto que não permitam acesso direto à placa.

Vamos montar um esquema pull-down de um botão, e ele vai determinar se o nosso LED vai está ligado ou não.

► PROJETO 5 – BOTÃO

ESQUEMA COMPONENTES



1 – BOTÃO (PUSHBUTTON)

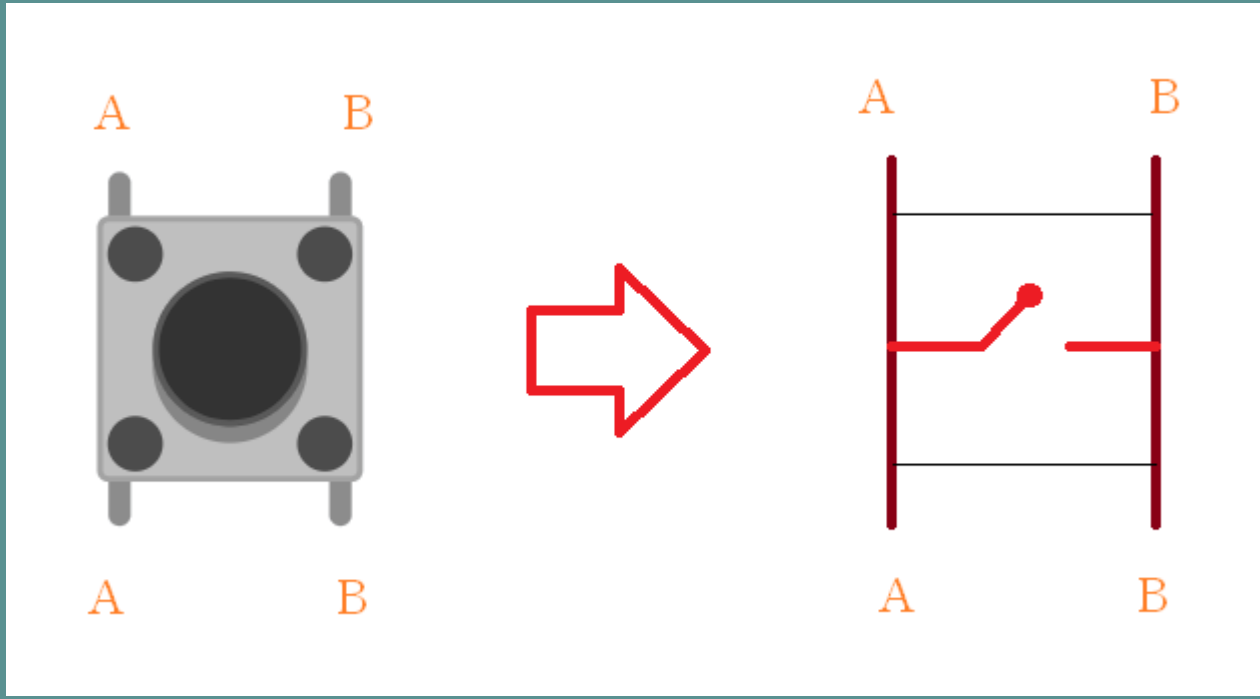
2 – RESISTOR 100 OHM

1- RESISTOR >10KOHMS

DIGITE O CÓDIGO:

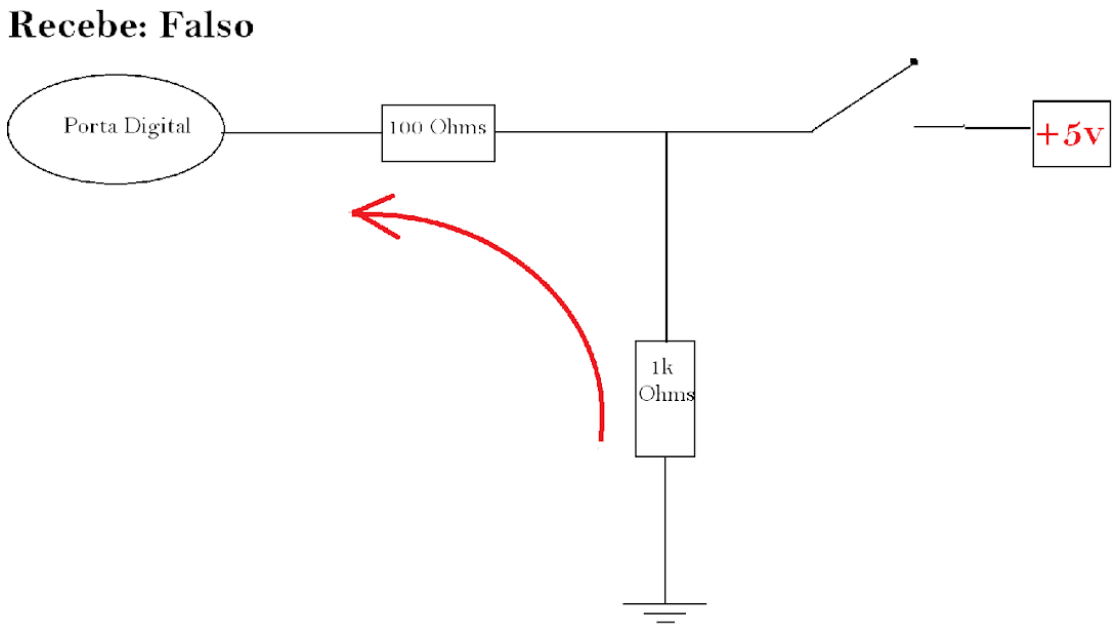
```
int botao = 12;
int led = 10;

void setup()
{
  pinMode(botao, INPUT);
  pinMode(led, OUTPUT);
}

void loop()
{
    if(digitalRead(botao) == 1)
    {
        digitalWrite(led,HIGH);
    }
    else
    {
        digitalWrite(led,LOW);
    }
}
```

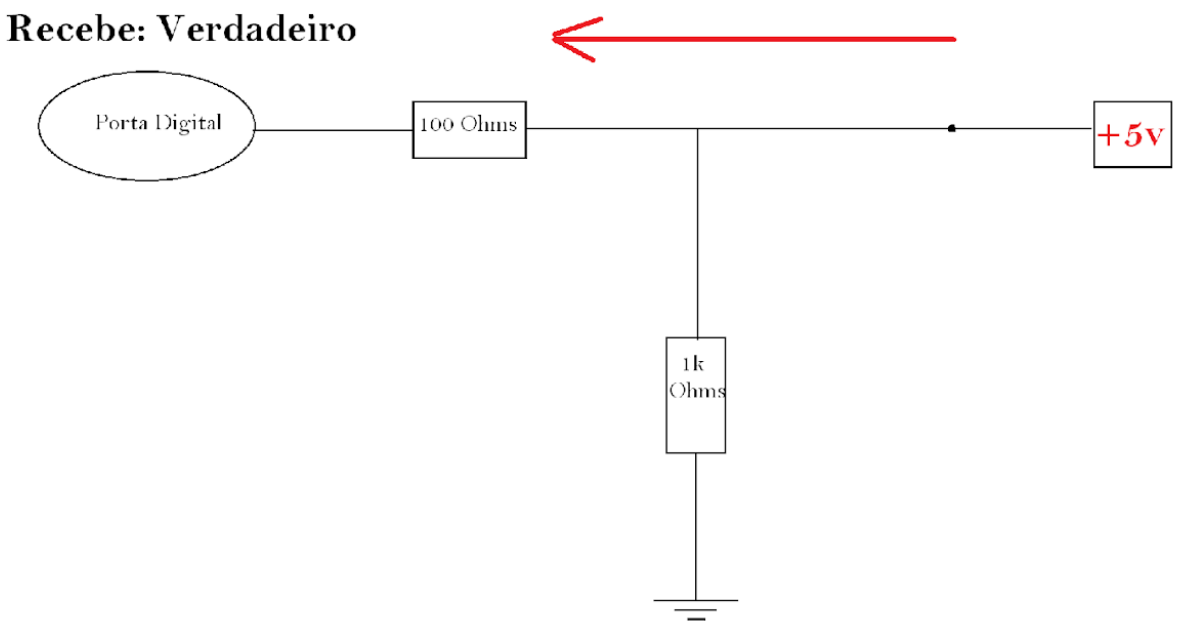
DETALHES DO BOTÃO (PUSHBUTTON)



O botão é uma chave na qual quando é pressionado liga os pontos A e B.

ESQUEMA PULL-DOWN





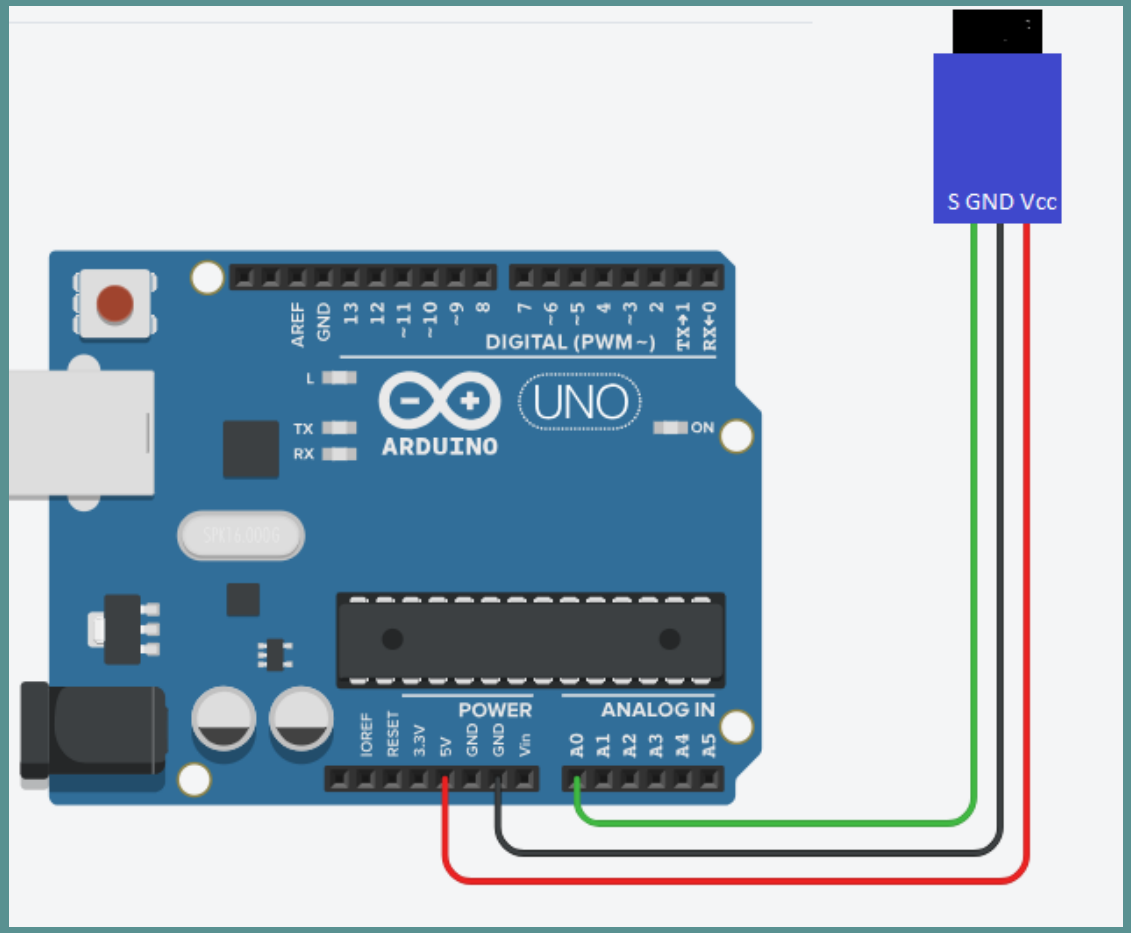
Função digitalRead()

- Sintaxe : digitalRead(PORTA);
- Retorno: 1 caso tenha sinal e 0 caso não tenha sinal
- As portas podem ser as portas analógicas A0,A1,A2..A7, ou digital 0,1,2,3...13.
- O formato pode ser INPUT para sinais que chegam ao arduino, e OUTPUT para sinais enviados pelo arduino.

Vamos aprender a trabalhar com um poderoso sensor de som.

- PROJETO 6 – SENSOR DE SOM

ESQUEMA COMPONENTES



1 – SENSOR DE SOM

DIGITE O CÓDIGO:

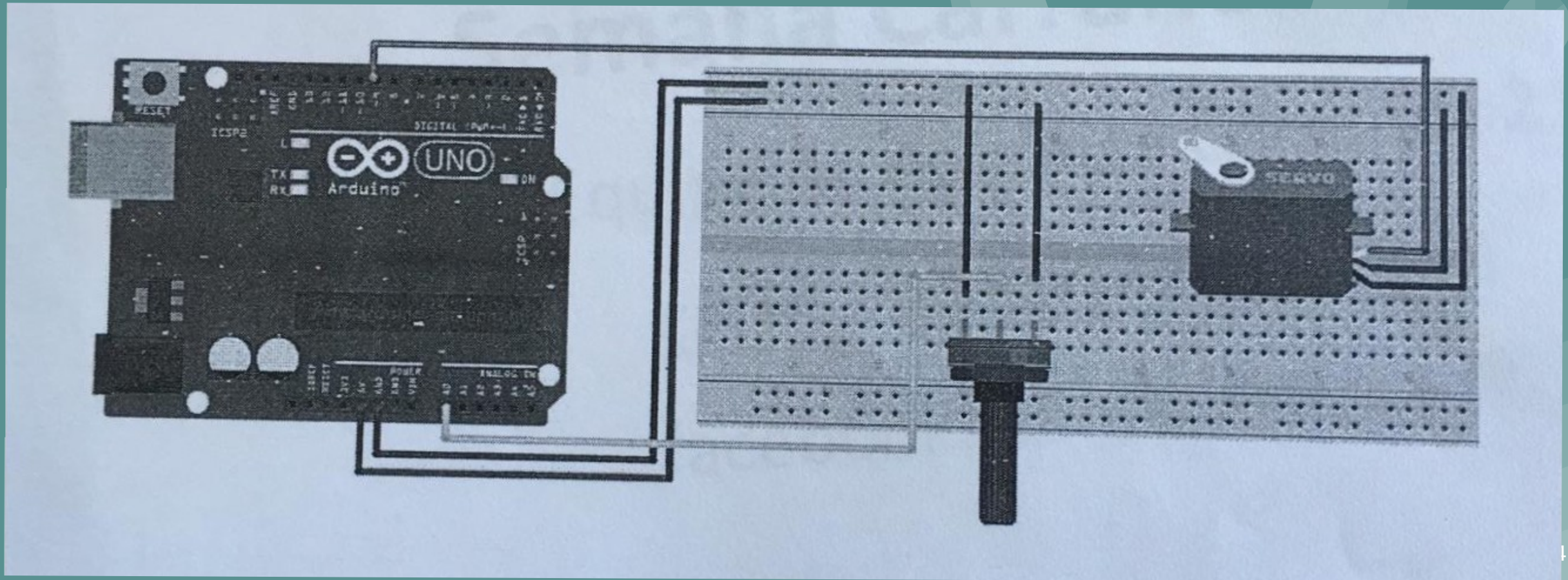
```
#define N_INTERACOES 400

void setup() {
  pinMode(A0, INPUT);
  Serial.begin(9600);
}

void loop()
{
  int valor = 0;

  valor = filtro(A0);
  Serial.print("Valor =");
  Serial.println(valor);
}

int filtro(int porta)
{
  unsigned long valorfiltrado = 0;
  for(int i =0; i< N_INTERACOES; i++)
  {
     valorfiltrado += analogRead(A0);
  }
  return valorfiltrado/N_INTERACOES;
}
```

Vamos aprender a trabalhar e como funciona um servo motor.

▶ PROJETO 7 – SERVO MOTOR

ESQUEMA COMPONENTES

1 – POTENCIOMETRO
2 – SERVO MOTOR

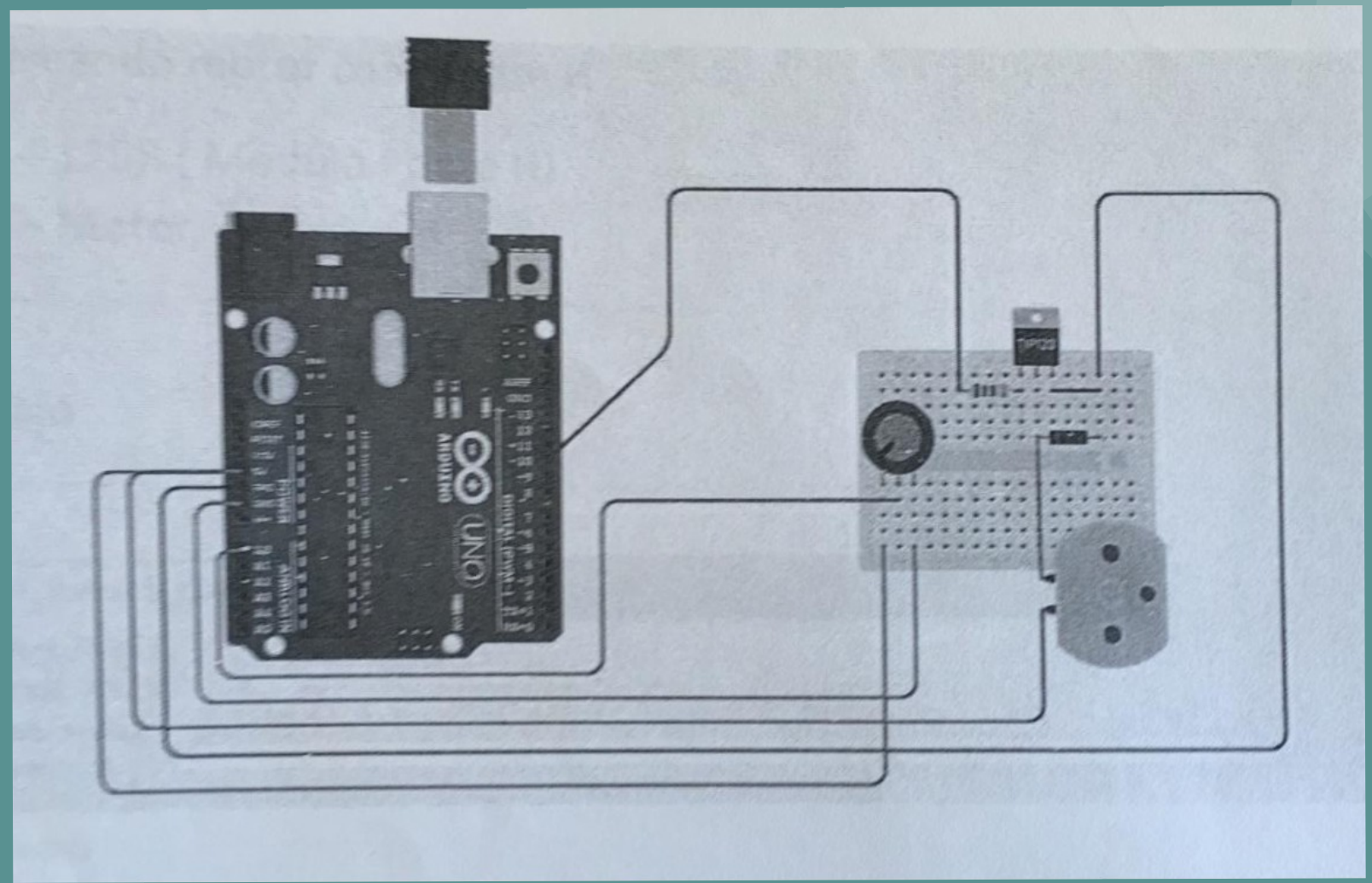


```
#include <Servo.h>
Servo meuservo; //create servo object to control a servo

// Pino analogico do potenciometro
int potpin = 0;
int servopin = 9;
// Variavel que armazena o valor lido do potenciometro
int val;

void setup() {
  // Define que o servo esta ligado a porta 9
  meuservo.attach(servopin);

}


void loop() {
  // Le o valor do potenciometro (valores entre 0 e 1023)
  val = analogRead(potpin);

  // Converte o valor pra ser usado(valores entre 0 e 180)
  val = map(val, 0, 1023, 0, 179);

  // Move o eixo do servo, de acordo com o angulo
  meuservo.write(val);

  // Aguarda o servo atingir a posiçao
  delay(15);
}
```

Vamos aprender a trabalhar e como funciona um motor DC.

▶ PROJETO 8 – MOTOR DC

ESQUEMA COMPONENTES



1 – POTENCIOMETRO
2 – MOTOR DC
3 – TIP 122
4 – DIODO
5 - RESISTORES

```
int motor = 10;
int poten = A0;
int valor = 0;

void setup() {
  pinMode(poten, INPUT);
  Serial.begin(9600);

}

void loop() {
  valor = analogRead(poten)/4;
  analogWrite(motor,valor);
  Serial.println(valor);

}
```