



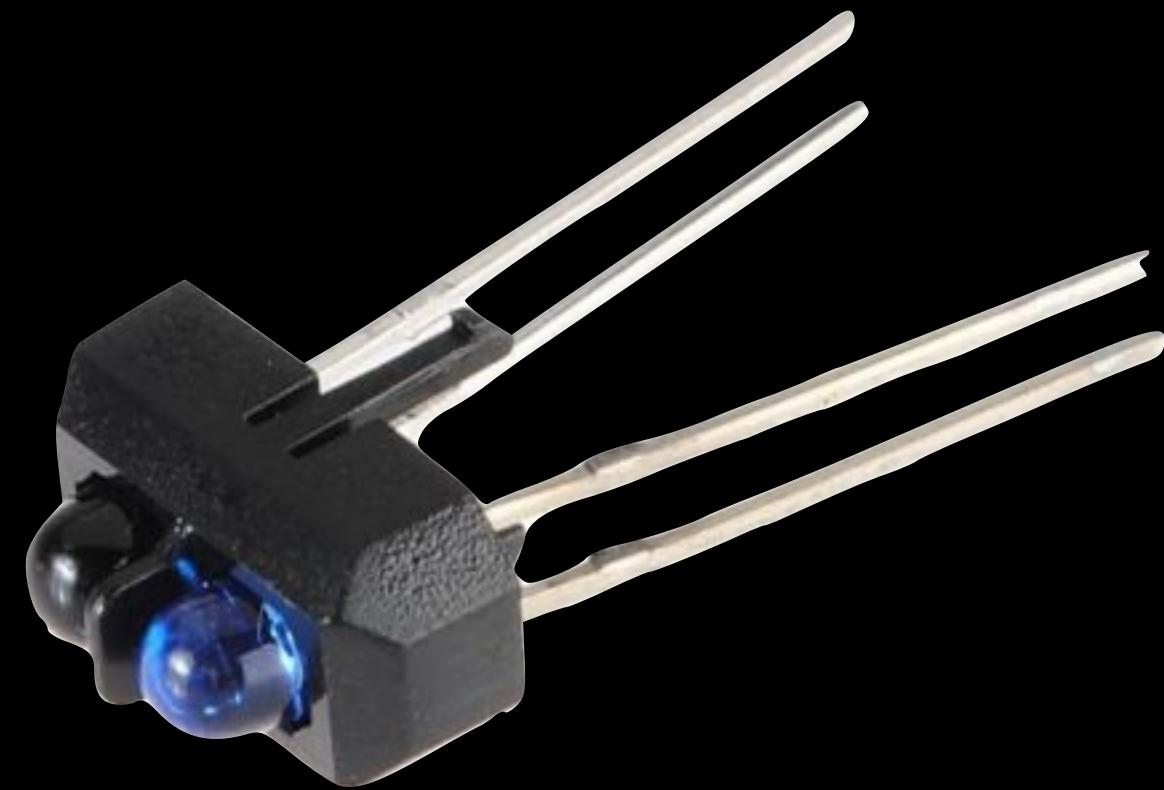
# Projeto 09

## Controle Serial – Teoria

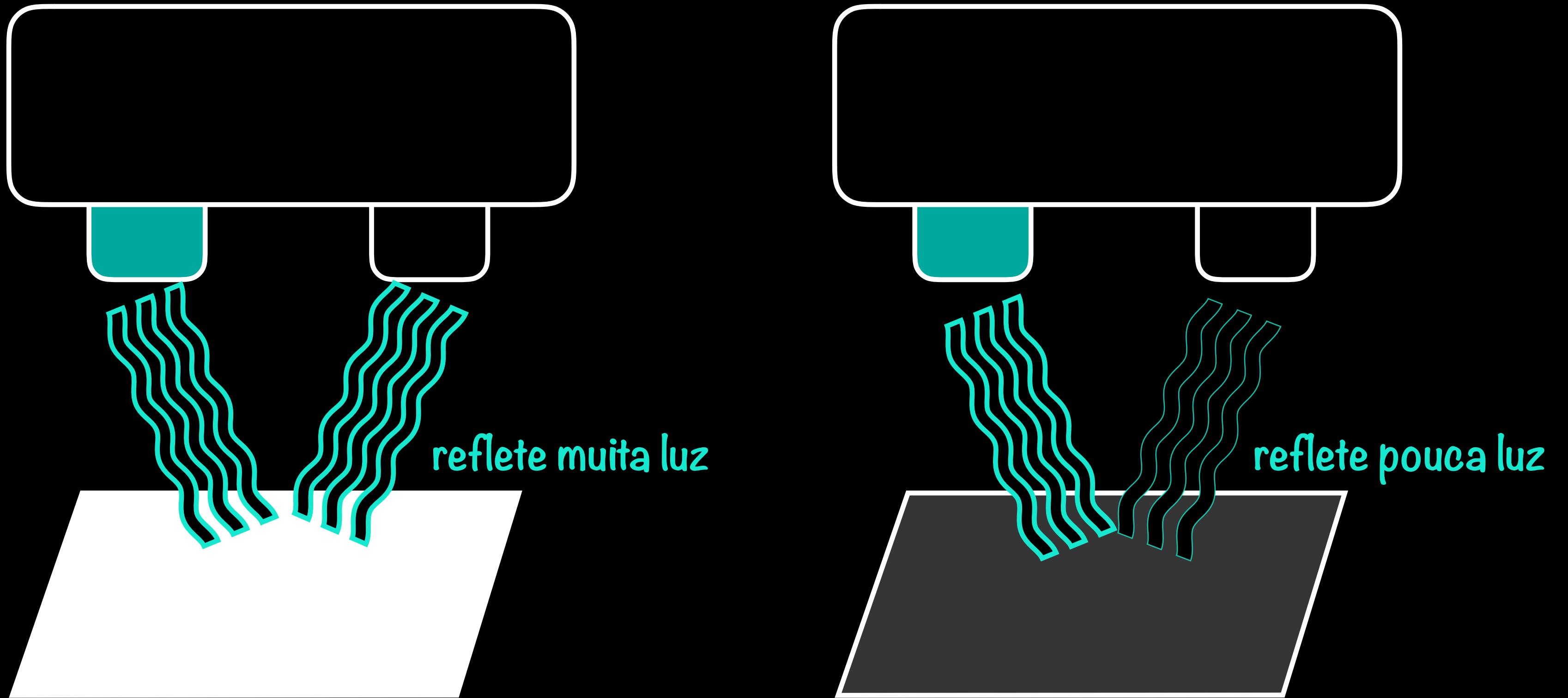
Jan K. S. – janks@puc-rio.br

ENG1419 – Programação de Microcontroladores

# Hardware



Sensor Ótico (TCRT5000)



Funcionamento do Sensor Ótico



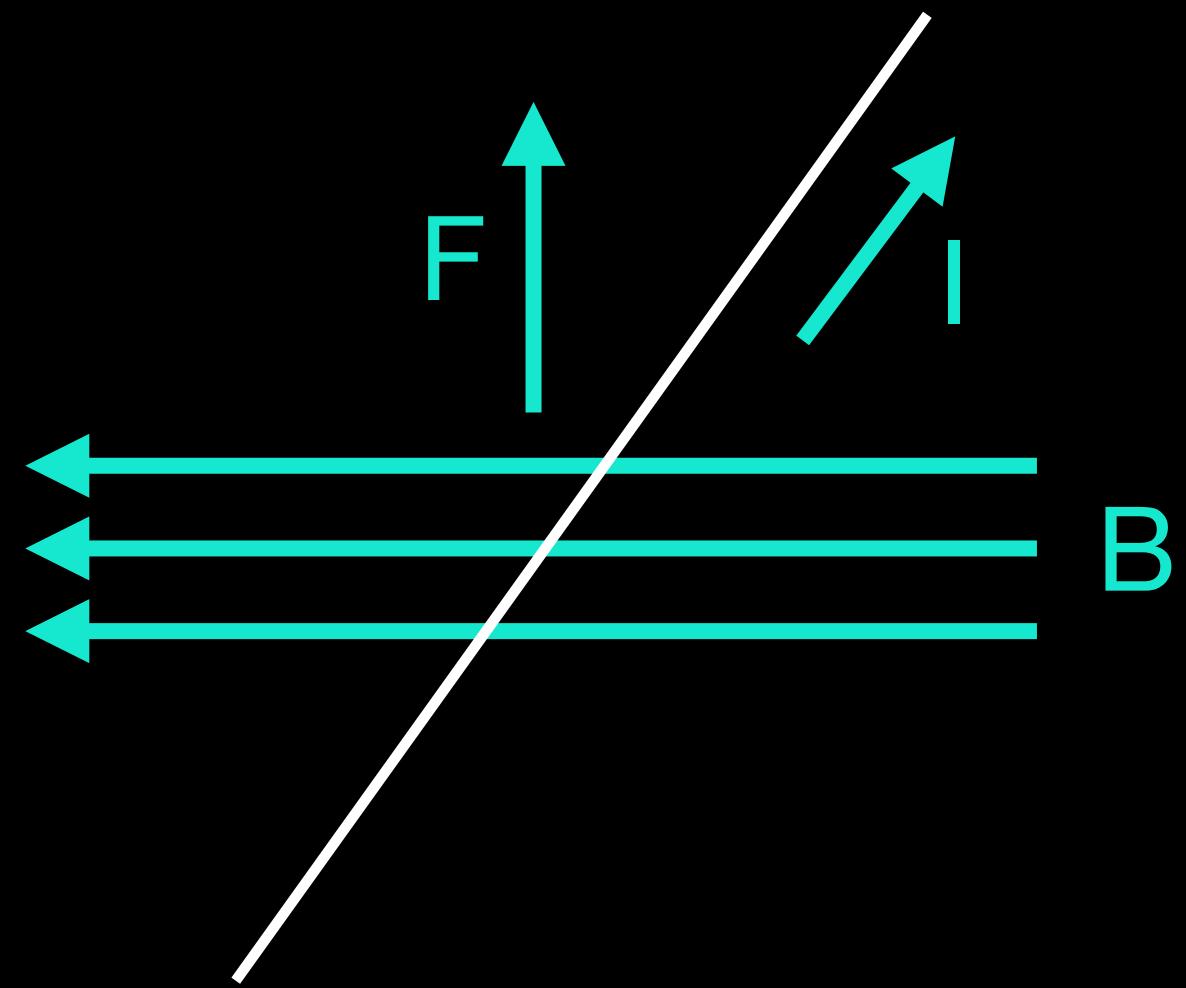
Limitação de Distância do Sensor Ótico

```
int sensorOtico = A11;
void setup () {
    pinMode(sensorOtico, INPUT);
    Serial.begin(9600);
}

void loop () {
    int valorAnalogico = analogRead(sensorOtico);
    Serial.println(valorAnalogico);
    int valorDigital = digitalRead(sensorOtico);
    if (valorDigital == LOW) {
        Serial.println("Superfície clara");
    }
    else {
        Serial.println("Superfície escura");
    }
    delay(200);
}
```

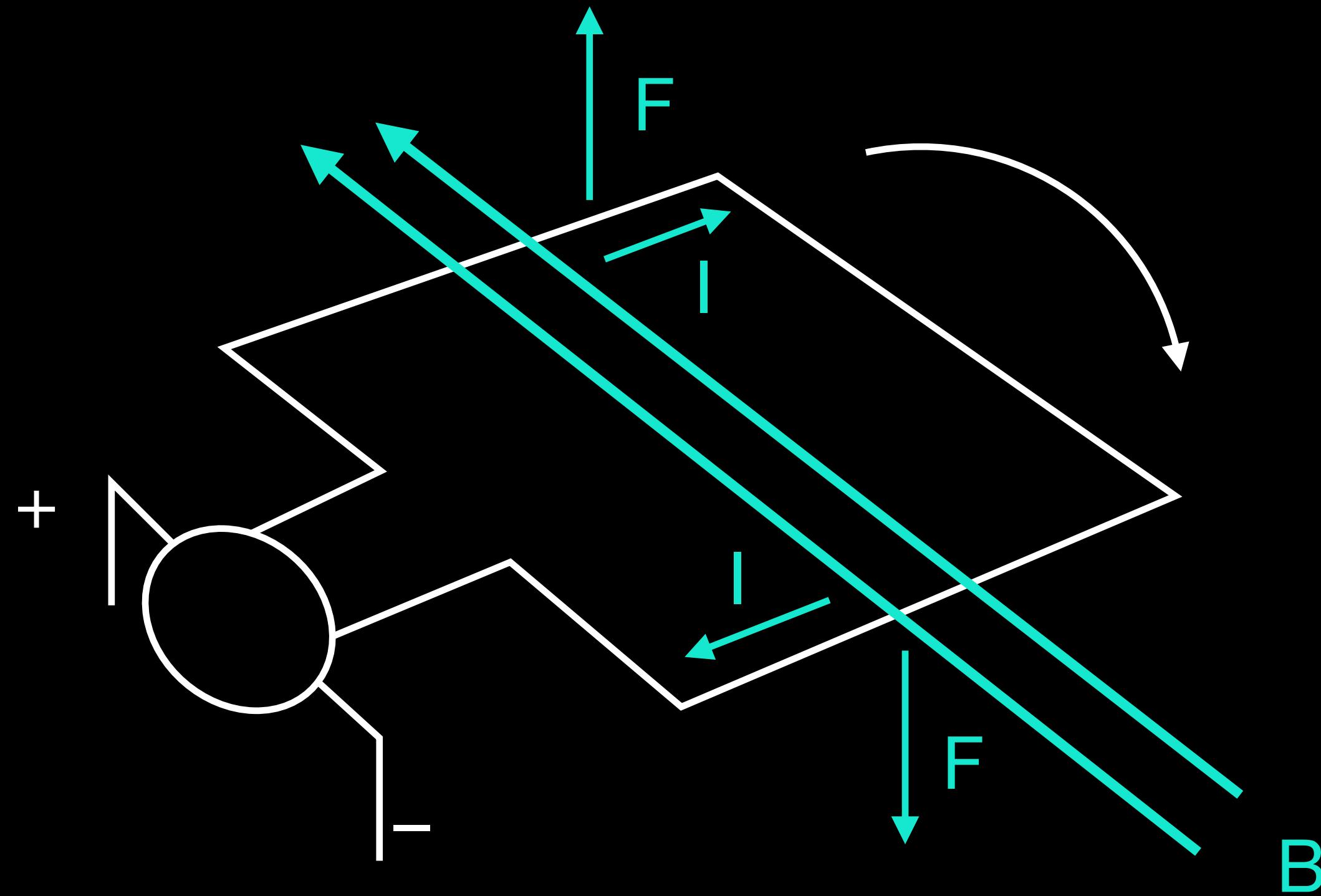


Motor DC (Motor de Corrente Contínua)

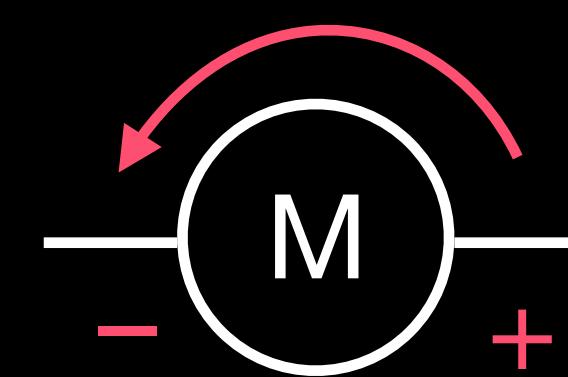
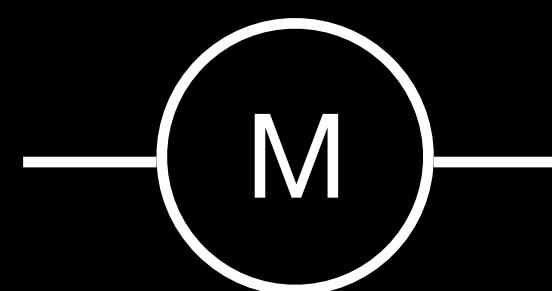
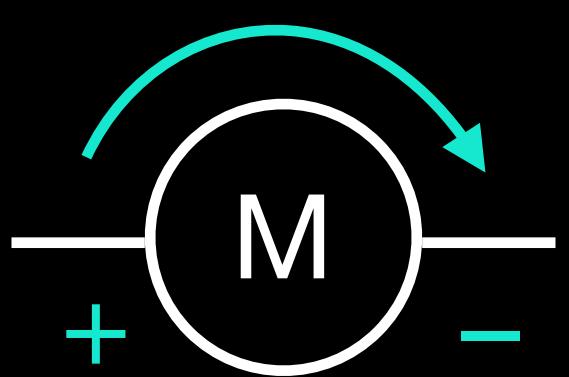


$$F = I \int dl \times B$$

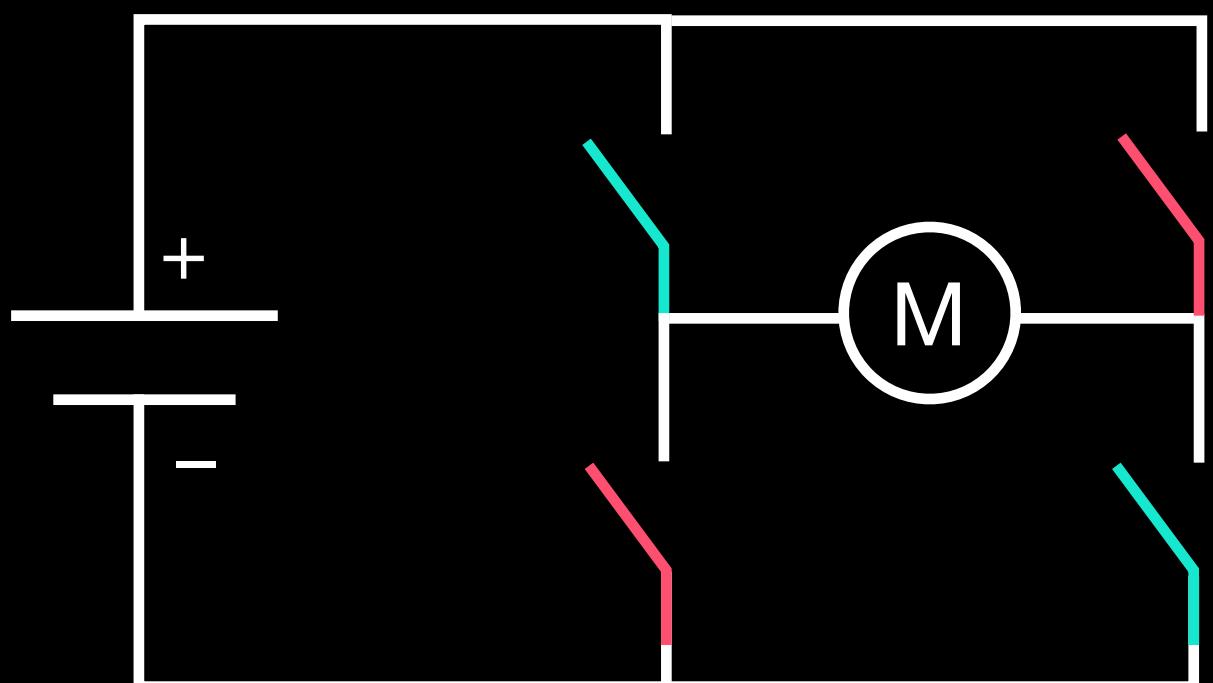
Força de Lorentz: Força Aplicada em uma Corrente num Campo Magnético



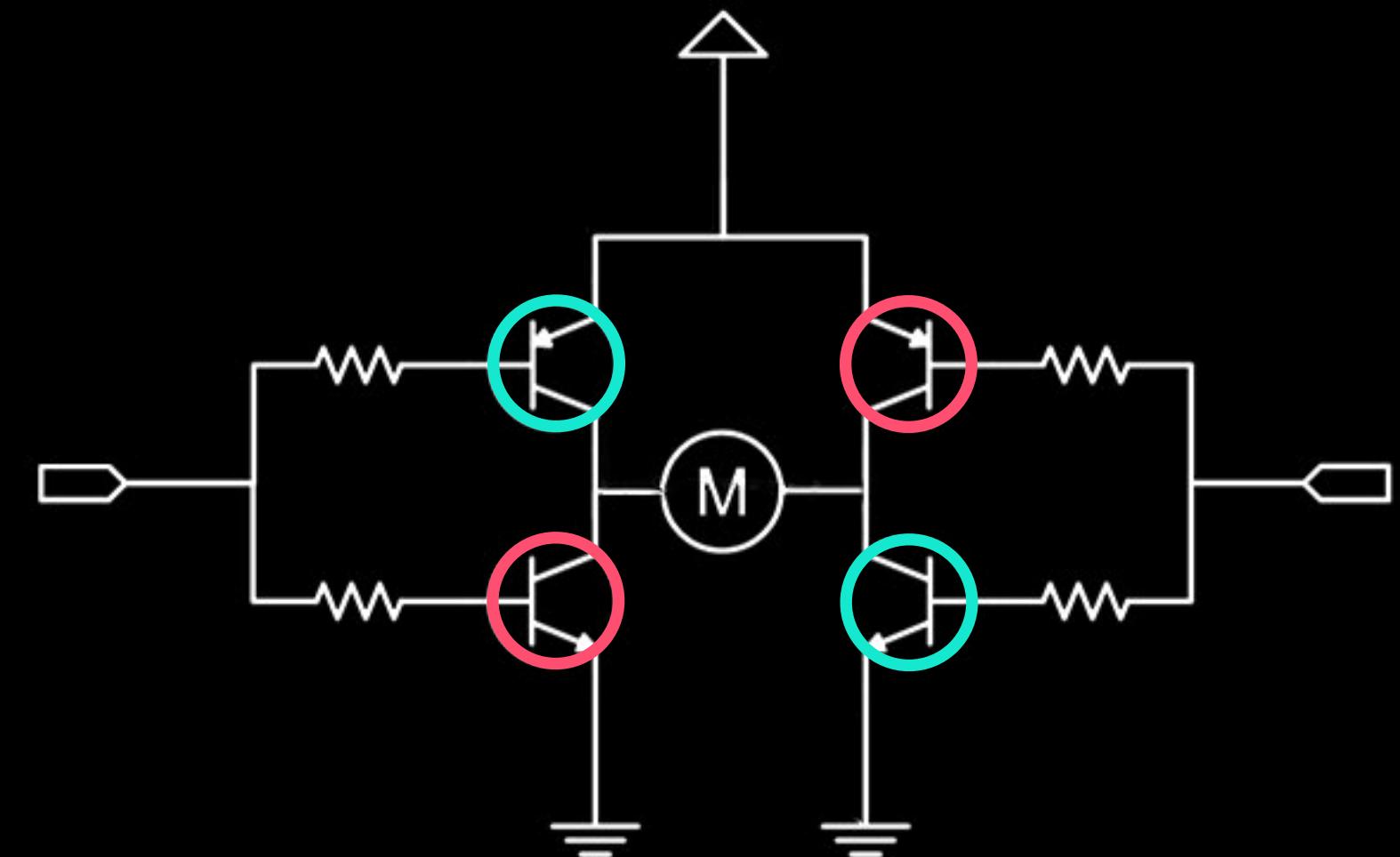
Princípio Básico de um Motor DC



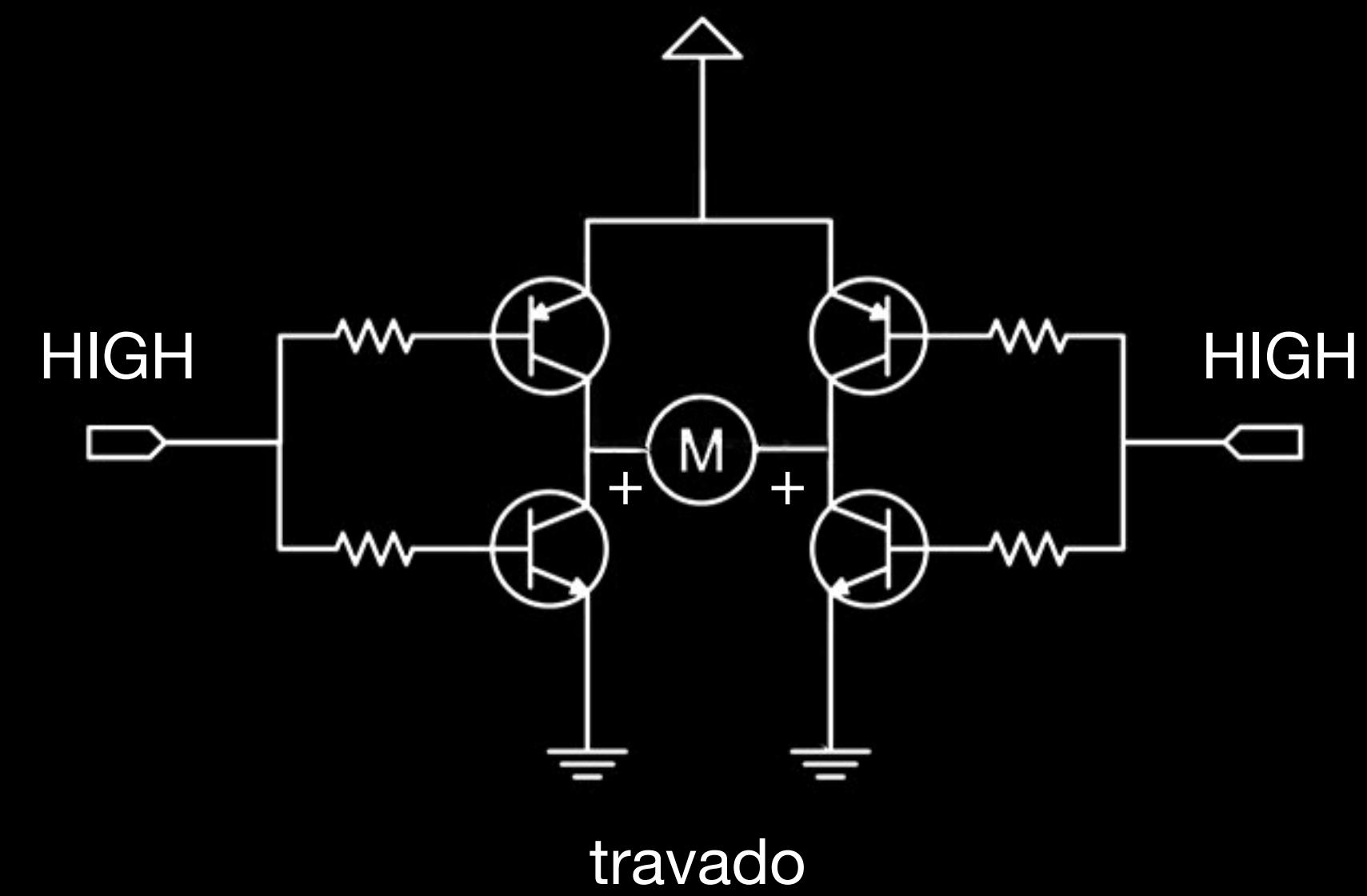
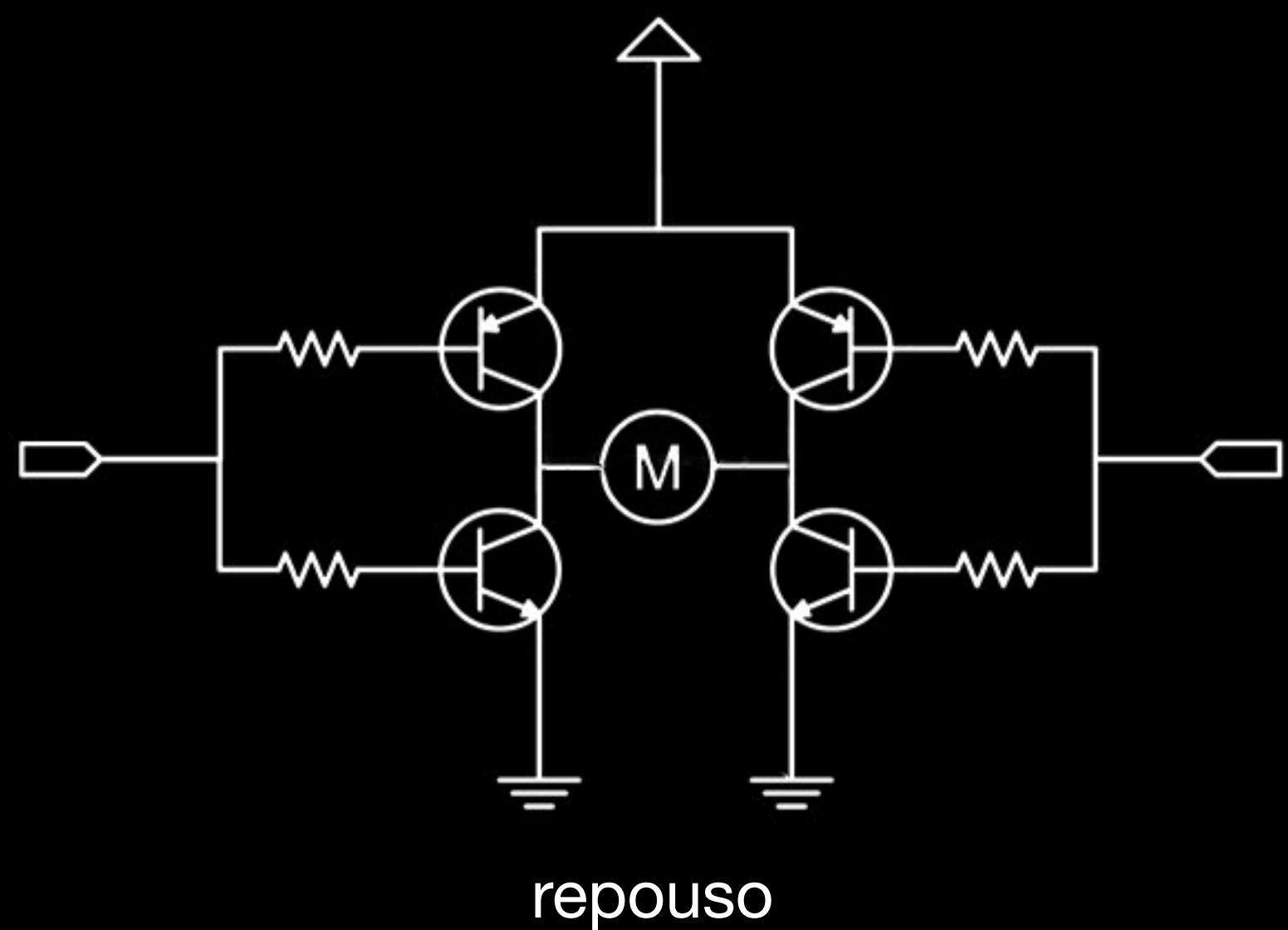
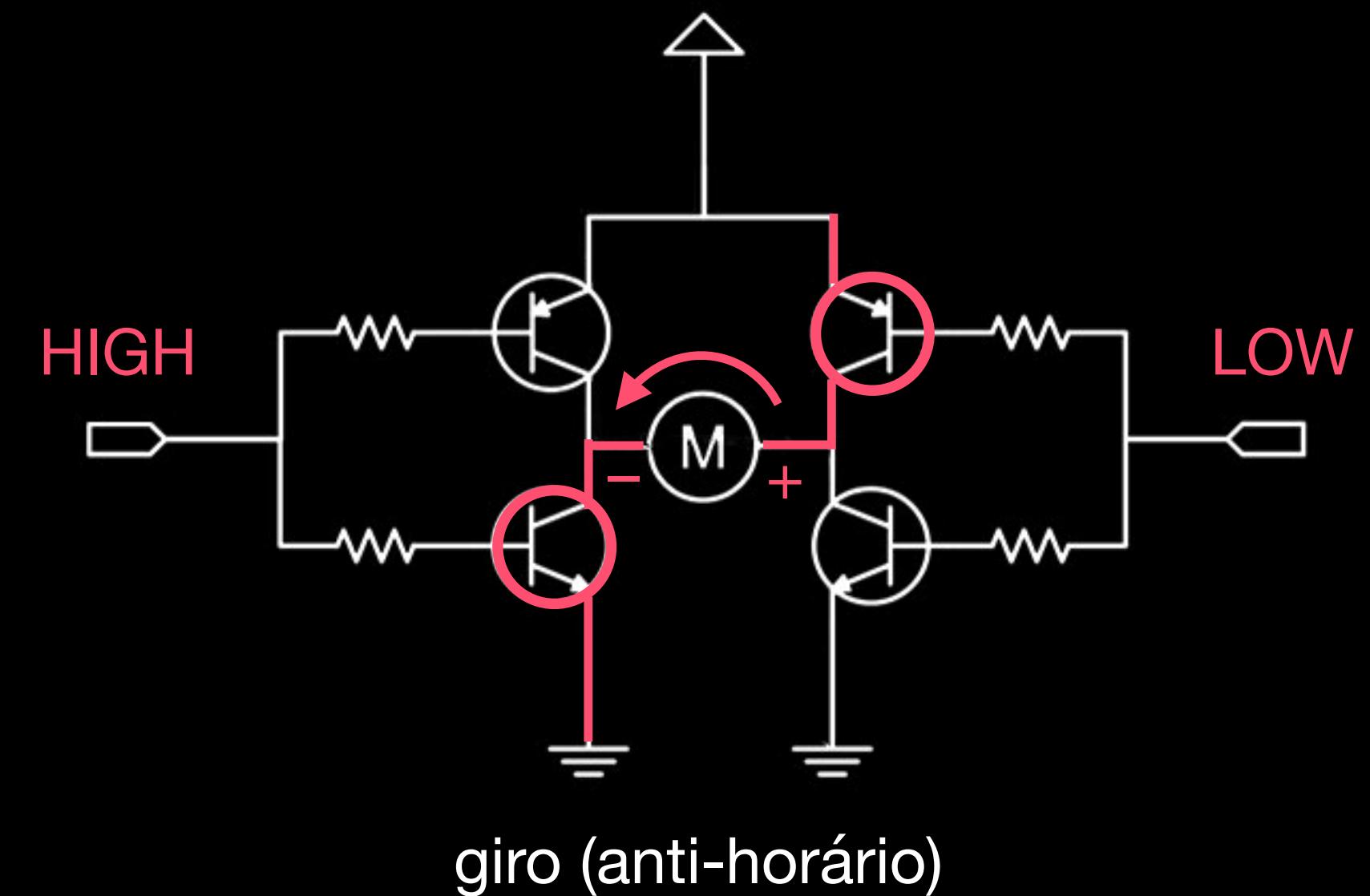
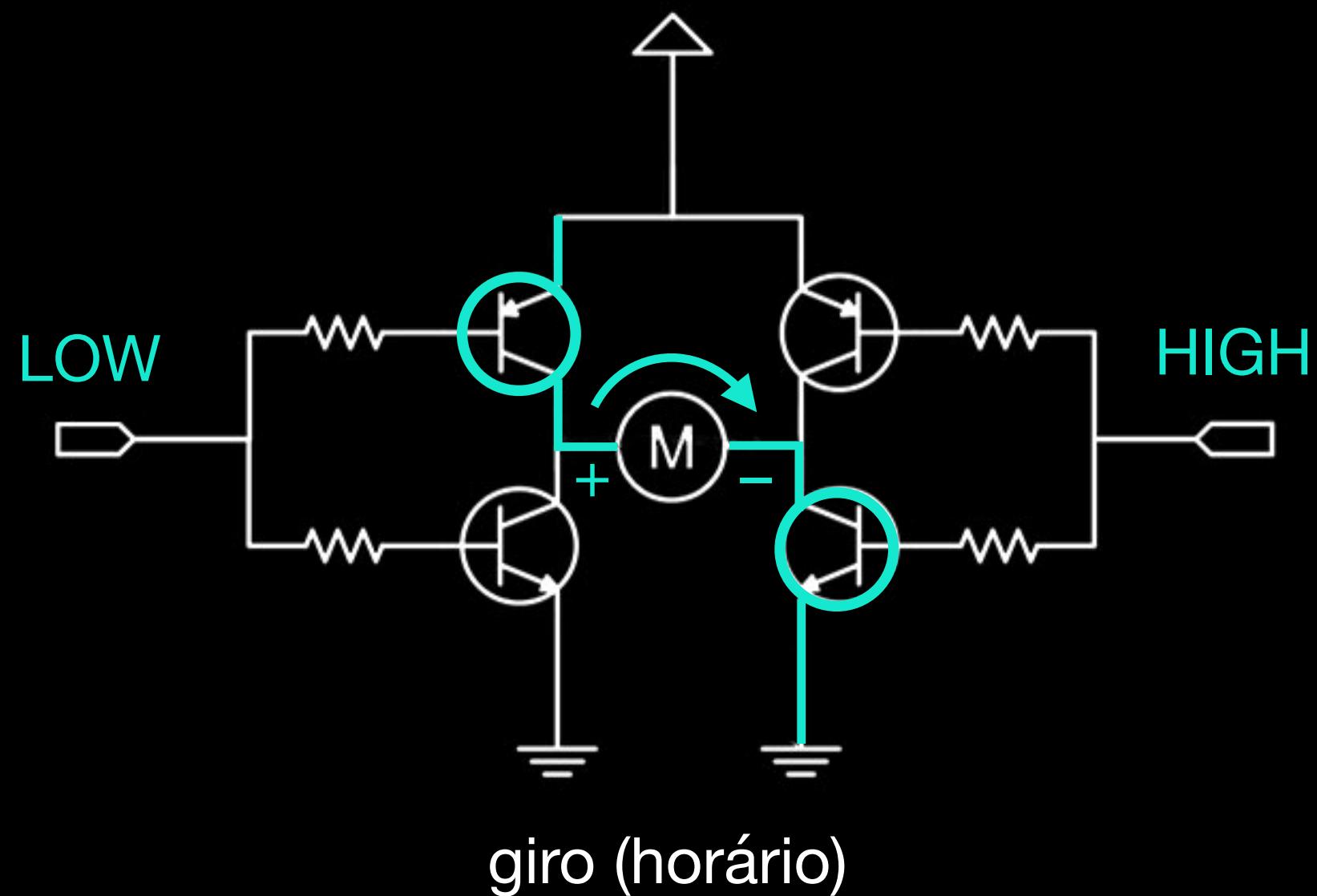
ponte H com chaves



ponte H com transistores

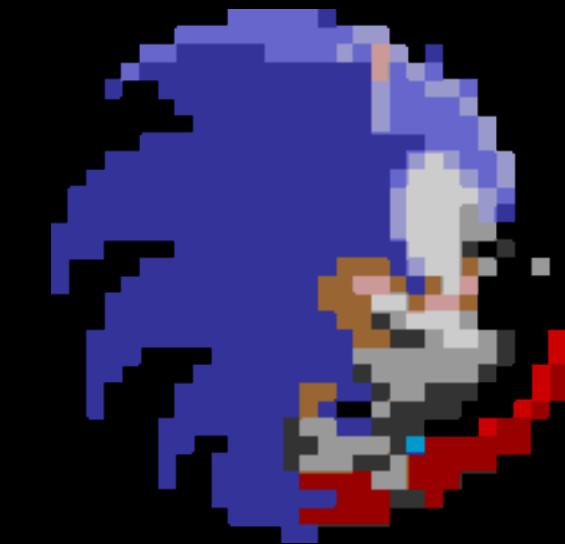
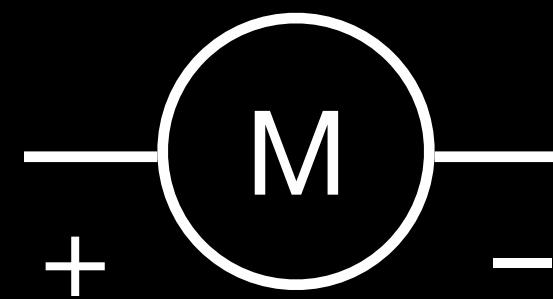
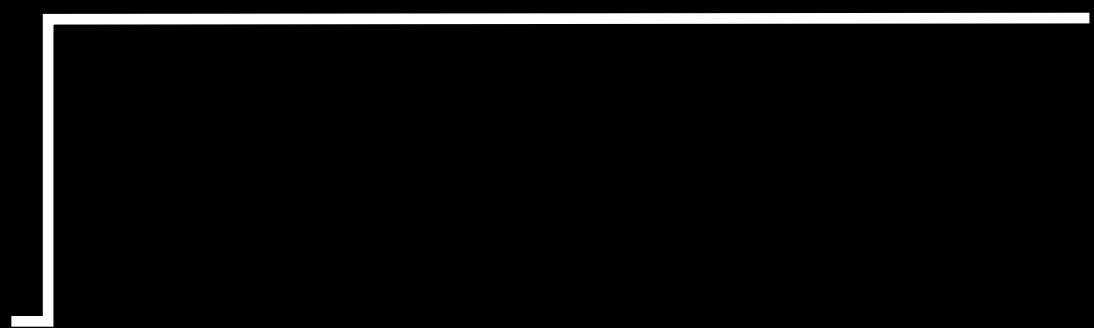


Controle do Sentido de Giro com Ponte H

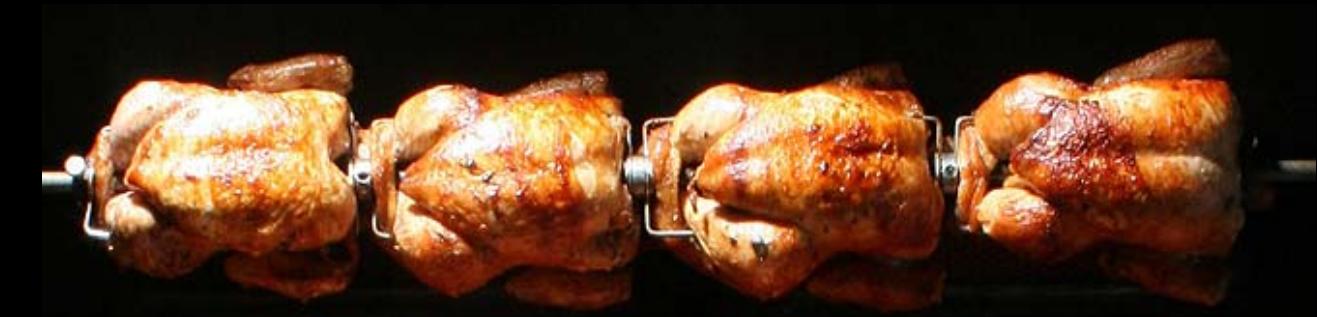
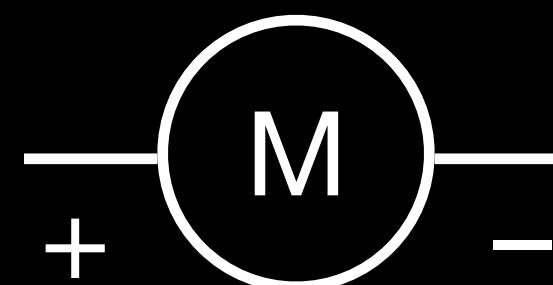
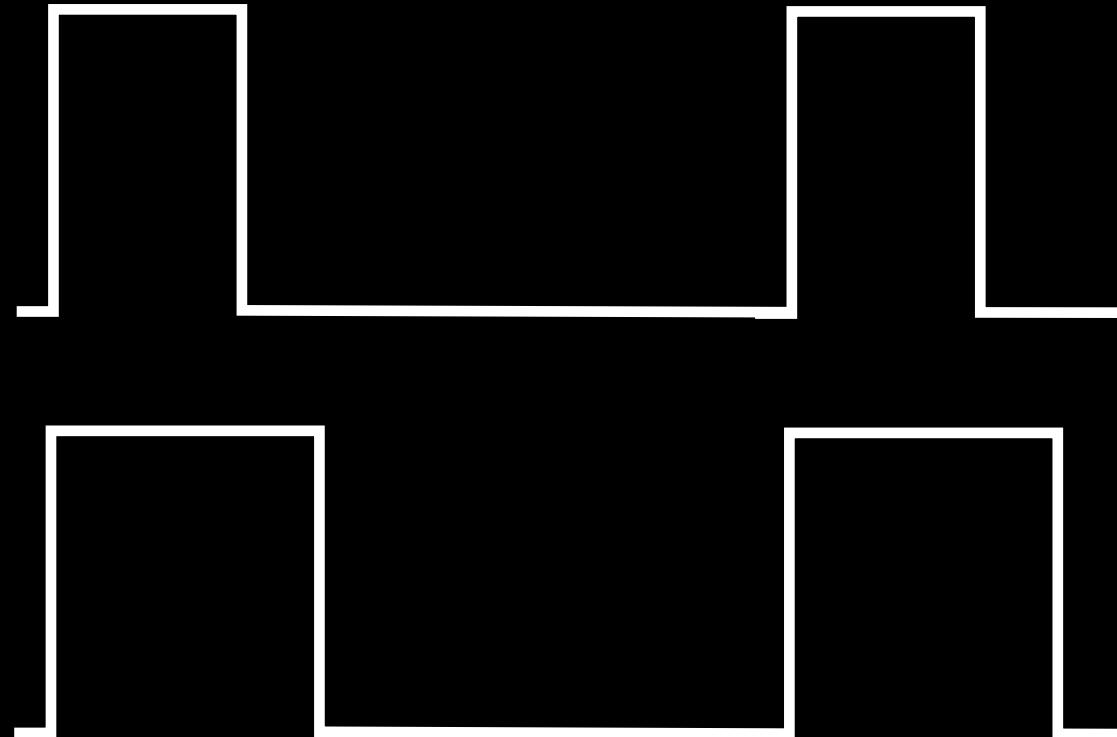


Controle do Sentido de Giro com Ponte H

velocidade total



velocidade parcial

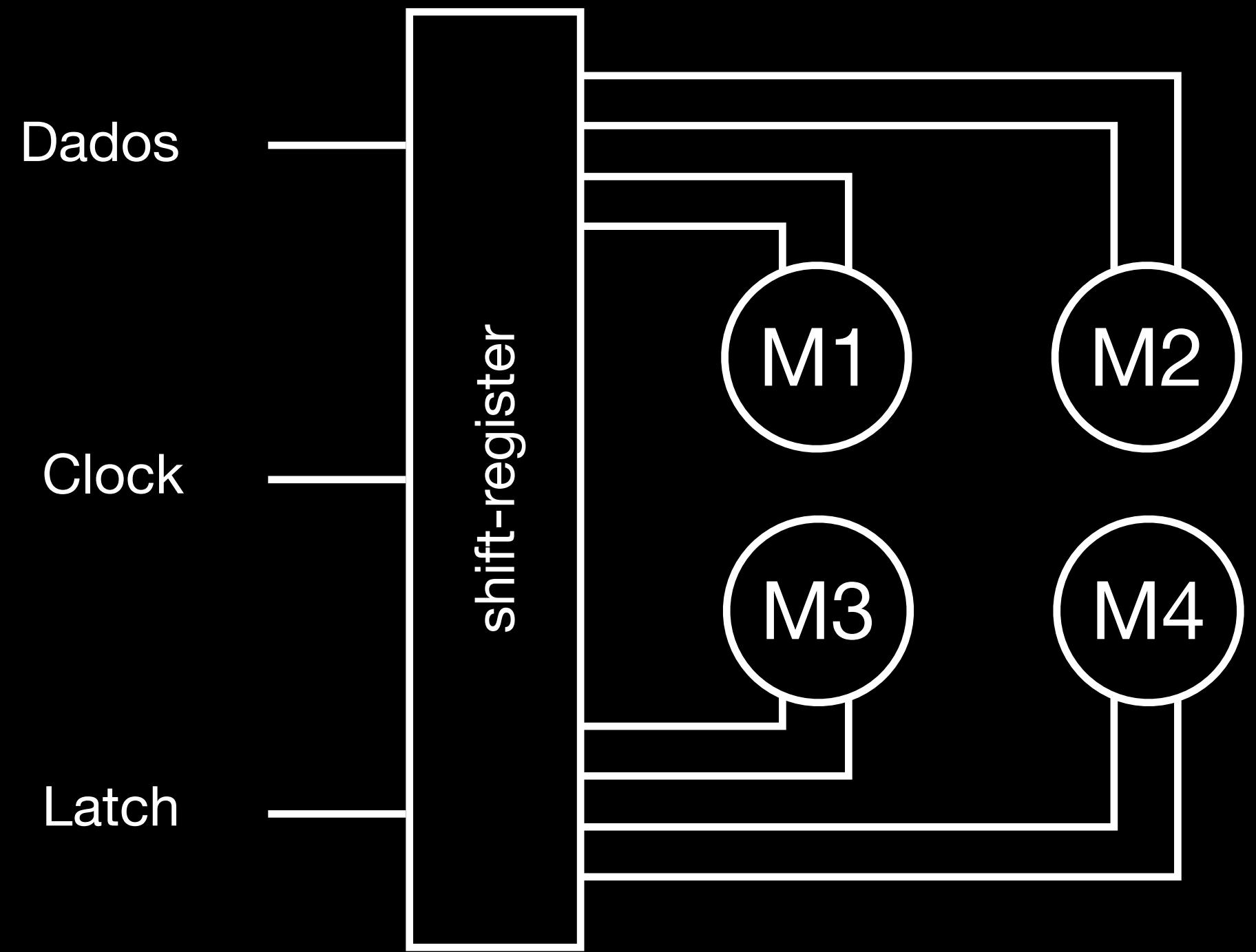
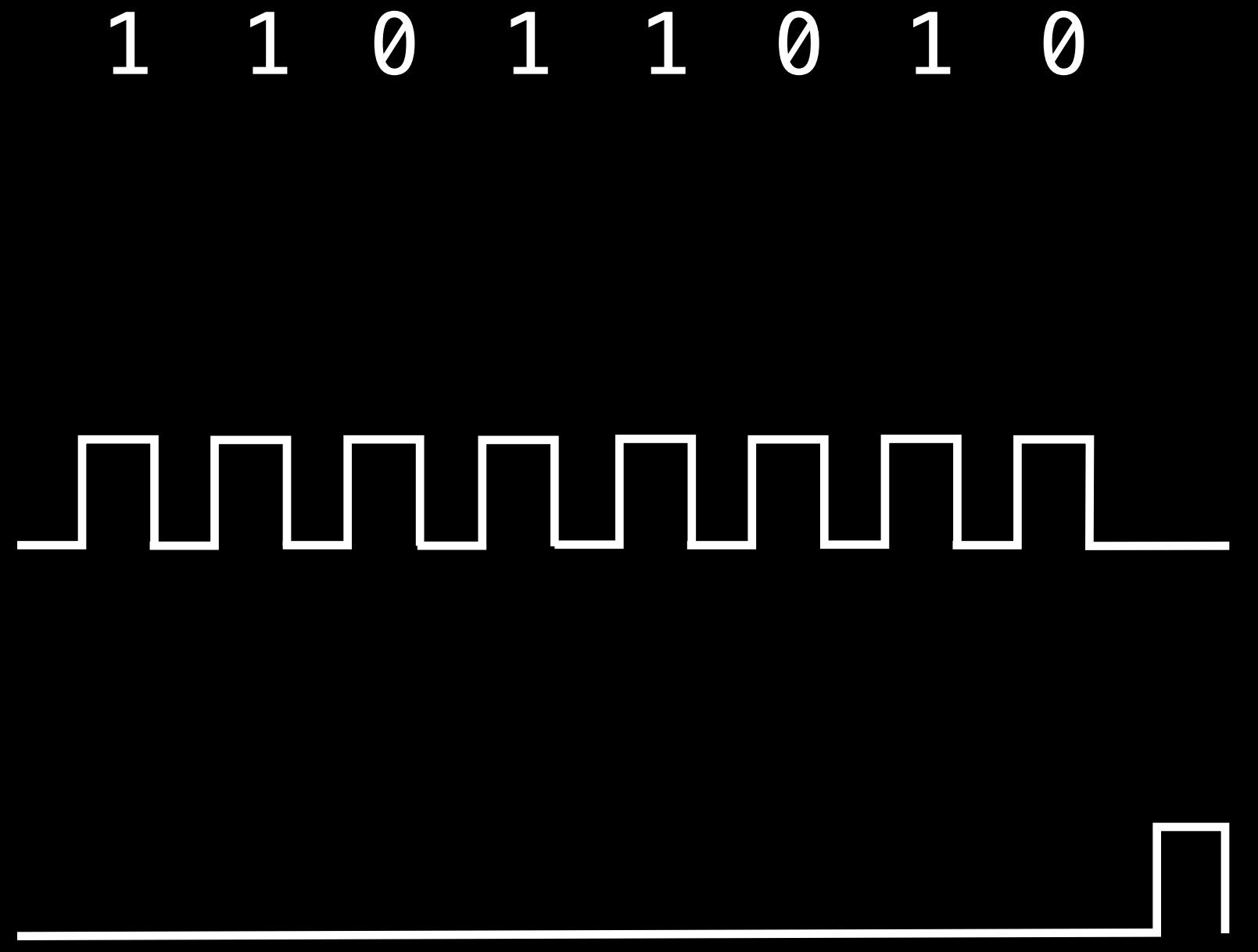


**PWM**

Controle de Velocidade com PWM



Shield para Motores



Controle dos 4 Motores via Shift Register

The screenshot shows a GitHub repository page for the Adafruit Motor shield library. The URL in the address bar is [GitHub, Inc. github.com/adafruit/Adafruit-Motor-Shield-Library](https://github.com/adafruit/Adafruit-Motor-Shield-Library). The repository name is **adafruit / Adafruit-Motor-Shield-Library**. The page includes navigation links for Pull requests, Issues, Marketplace, and Explore. It displays statistics: 24 commits, 1 branch, 1 release, and 6 contributors. A red bar highlights the commit count. Below this, a list of recent commits is shown, all made by user **ladyada**, with the latest commit being "Update README.txt" on 17 Oct 2016.

adafruit / Adafruit-Motor-Shield-Library

Pull requests Issues Marketplace Explore

Watch 61 Star 224 Fork 190

Code Issues 6 Pull requests 0 Projects 0 Wiki Insights

Adafruit Motor shield V1 firmware with basic Microstepping support. Works with all Arduinos and the Mega  
<http://www.ladyada.net/make/mshield>

24 commits 1 branch 1 release 6 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

ladyada	Update README.txt	Latest commit 99381df on 17 Oct 2016
📁 .github	Add GitHub issue template	2 years ago
📁 examples	Fixed to use built in Servo library	7 years ago
📄 AFMotor.cpp	Added support for cipKIT/MPIDE (PIC32 architecture) boards. Please se...	6 years ago
📄 AFMotor.h	Update AFMotor.h	4 years ago
📄 README.txt	Update README.txt	2 years ago
📄 keywords.txt	The latest version	8 years ago

Biblioteca da Adafruit para o Shield de Motor

```

#include <AFMotor.h>

AF_DCMotor motorA(1);
AF_DCMotor motorB(2);

void setup () {
    motorA.setSpeed(255); // valor entre 0 e 255
    motorB.setSpeed(128);
}

void loop () {
    motorA.run(FORWARD);
    motorB.run(BACKWARD);
    delay(1000);

    motorA.run(RELEASE);
    motorB.run(BRAKE);
    delay(1000);
}

```



A opção de freio não foi implementada direito, por algum motivo...

"Software"

```
// textos no estilo C  
char texto[] = "Olá!";
```

```
// textos no estilo C++  
string texto = "Olá!";
```

```
// textos no estilo Arduino  
String texto = "Olá!";
```

The screenshot shows a web browser displaying the Arduino Reference page for String variables. The URL in the address bar is [www.arduino.cc/reference/en/language/variables/data-types/](http://www.arduino.cc/reference/en/language/variables/data-types/). The page has a teal header with navigation links: HOME, BUY, SOFTWARE, PRODUCTS, EDU, RESOURCES, COMMUNITY, and HELP. On the far right of the header are icons for download, upload, and a plus sign. Below the header is a sidebar with categories: LANGUAGE, FUNCTIONS, VARIABLES (which is selected and highlighted in blue), and STRUCTURE. There are also links for LIBRARIES and GLOSSARY. A note below the sidebar states: "The Arduino Reference text is licensed under a [Creative Commons Attribution-Share Alike 3.0 License](#)". Another note says: "Find anything that can be improved? [Suggest corrections and new documentation via GitHub](#)". A link to a tutorial is provided: "Doubts on how to use Github? Learn everything you need to know in [this tutorial](#)". The main content area is titled "Example Code" and contains the following text: "All of the following are valid declarations for Strings." It then lists several examples of String declarations with comments explaining the methods used:

```
String stringOne = "Hello String";                                // using a constant String
String stringOne = String('a');                                     // converting a constant char into a String
String stringTwo = String("This is a string");                      // converting a constant string into a String
String stringOne = String(stringTwo + " with more"); // concatenating two strings
String stringOne = String(13);                                       // using a constant integer
String stringOne = String(analogRead(0), DEC);                     // using an int and a base
String stringOne = String(45, HEX);                                    // using an int and a base (hexadecimal)
String stringOne = String(255, BIN);                                   // using an int and a base (binary)
String stringOne = String(millis(), DEC);                           // using a long and a base
String stringOne = String(5.698, 3);                                 // using a float and the decimal places
```

The main content area also has a section titled "Functions" listing various String methods:

- LANGUAGE [charAt\(\)](#)
- LANGUAGE [compareTo\(\)](#)
- LANGUAGE [concat\(\)](#)
- LANGUAGE [c\\_str\(\)](#)
- LANGUAGE [endsWith\(\)](#)
- LANGUAGE [equals\(\)](#)
- LANGUAGE [equalsIgnoreCase\(\)](#)
- LANGUAGE [getBytes\(\)](#)

```
String texto1 = "Olá, mundo!";

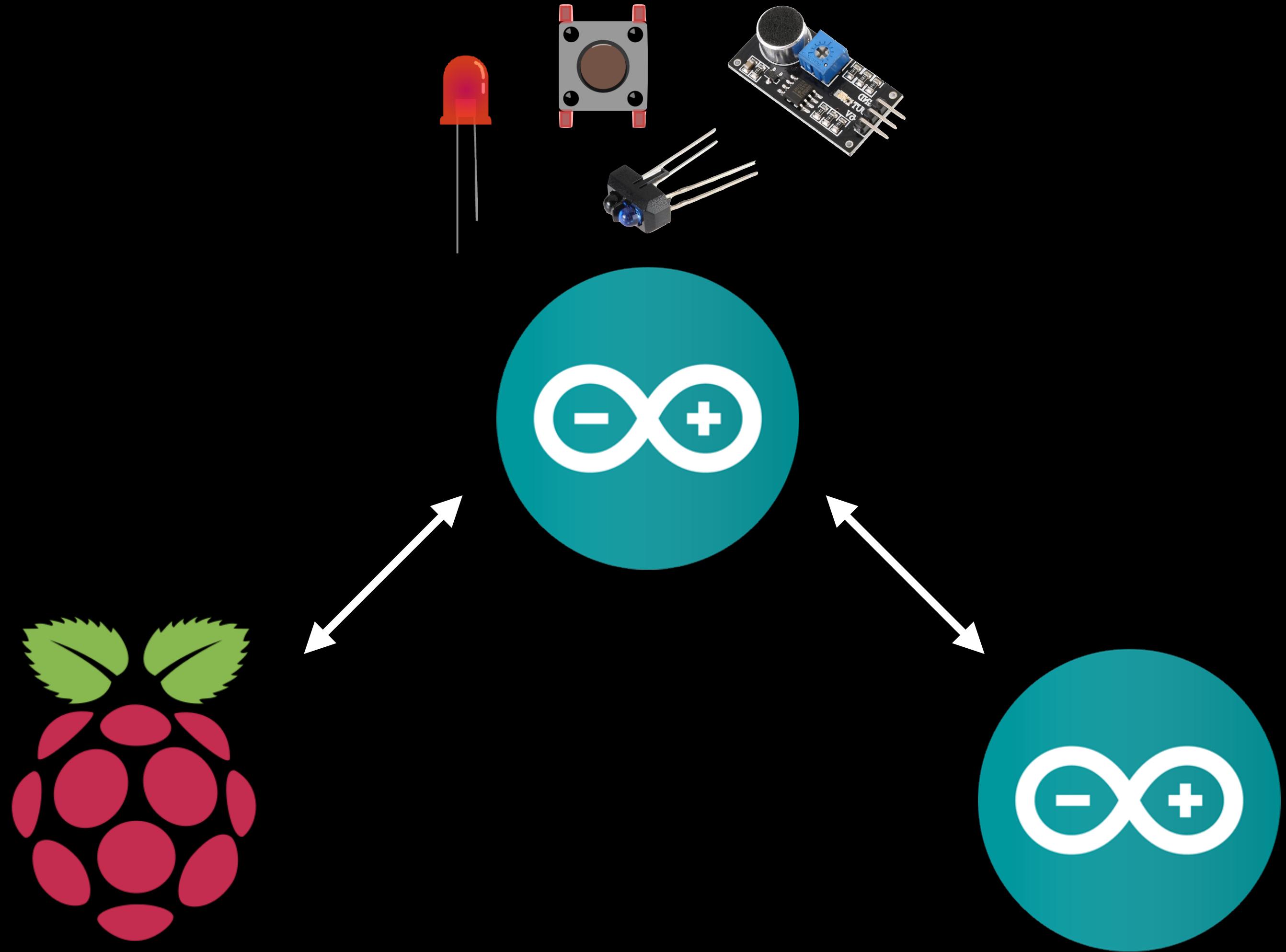
int numero = 100 * 2;
String texto2 = String(numero); // "200"
int numero2 = texto2.toInt() + 2; // 202

String texto3 = "aaa" + texto2; // "aaa200"

bool ehIgual = texto2 == texto3; // false
bool comecaComOla = texto1.startsWith("Olá"); // true

String trecho = texto1.substring(0, 3); // "Olá"
String trechoFinal = texto1.substring(5); // "mundo!"

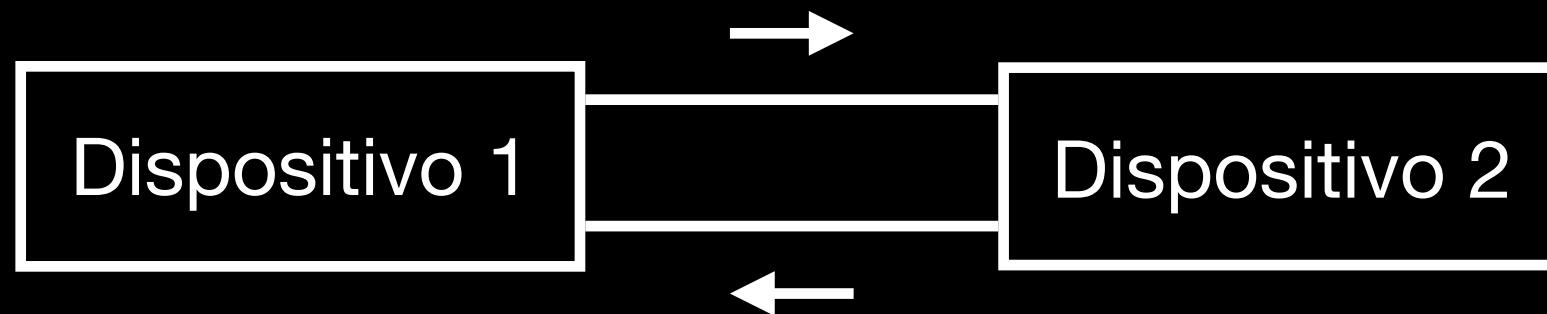
String texto4 = " abc abc \n";
texto4.trim(); // "abc abc"
texto4.replace("ab", "AB"); // "ABC ABC"
```



Comunicação Serial

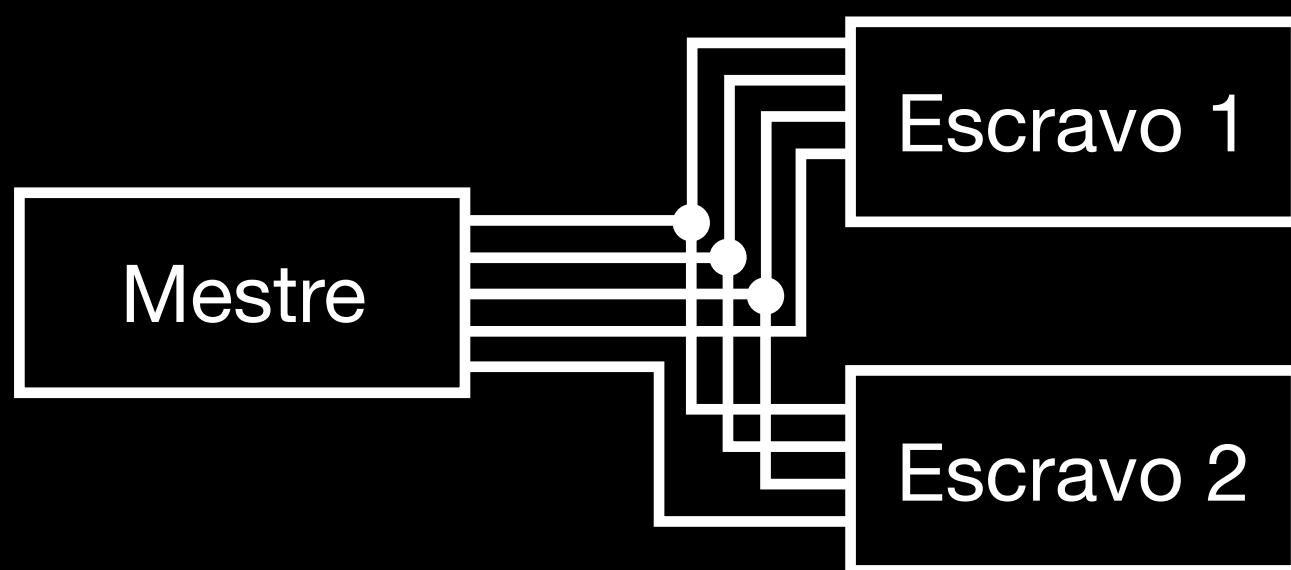
## UART

Universal Asynchronous  
Receiver/Transmitter



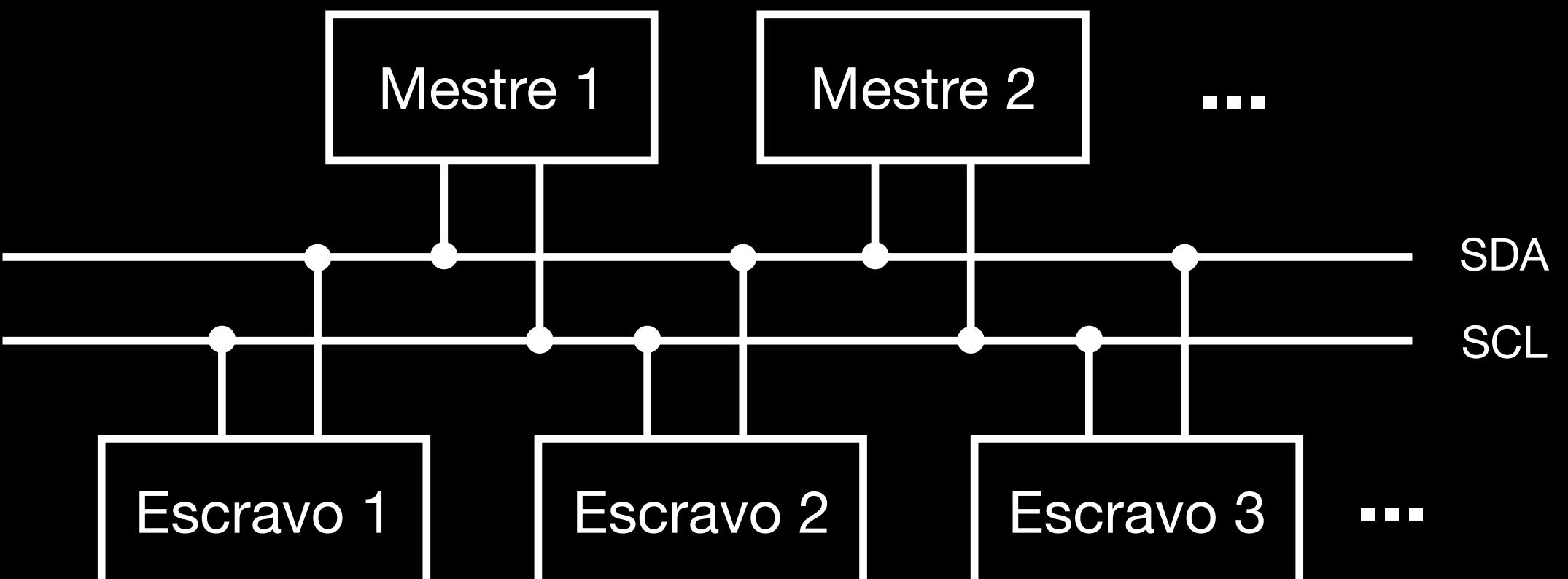
## SPI

Serial Peripheral Interface

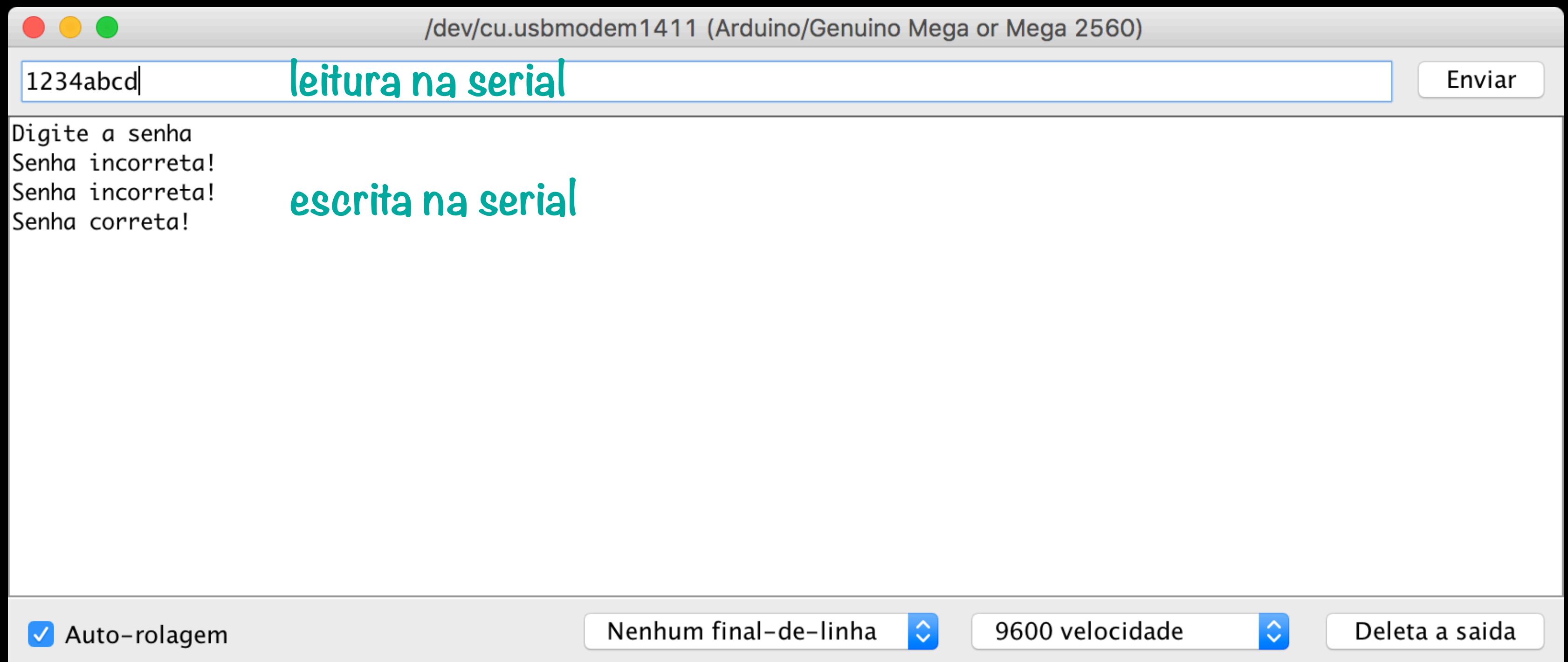


## I2C

Inter-Integrated Circuit



Tipos de Comunicação Serial



Comunicação Serial com o Computador

```

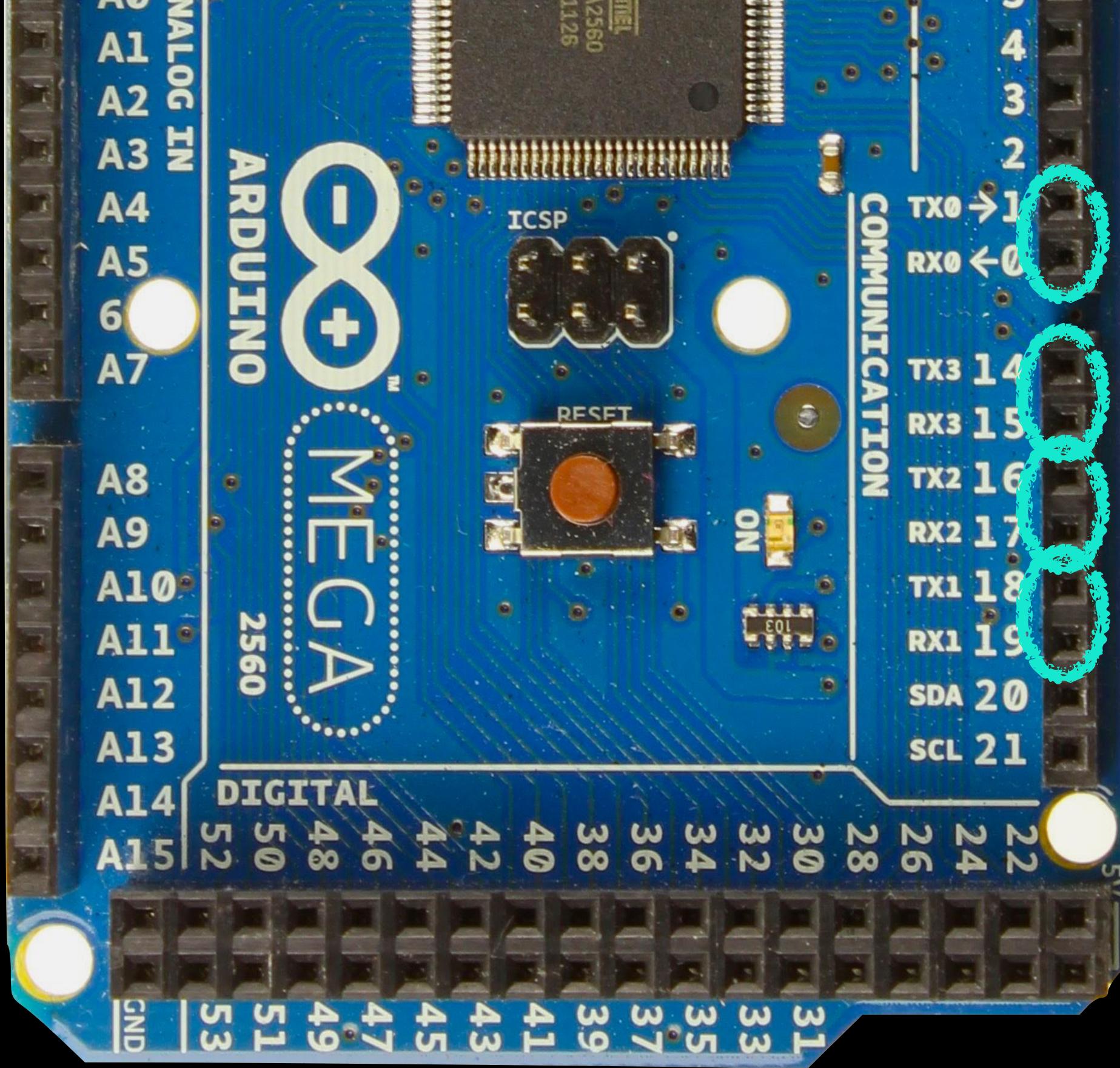
void setup () {
    Serial.begin(9600);
    Serial.println("Digite a senha");
    Serial.setTimeout(10); // espera máxima de 10ms na leitura
}

void loop () {
    String texto = Serial.readString();
    texto.trim();           // remove quebra de linha
    if (texto != "") {
        if (texto == "1234abcd") {
            Serial.println("Senha correta!");
        }
        else {
            Serial.println("Senha incorreta!");
        }
    }
}

```



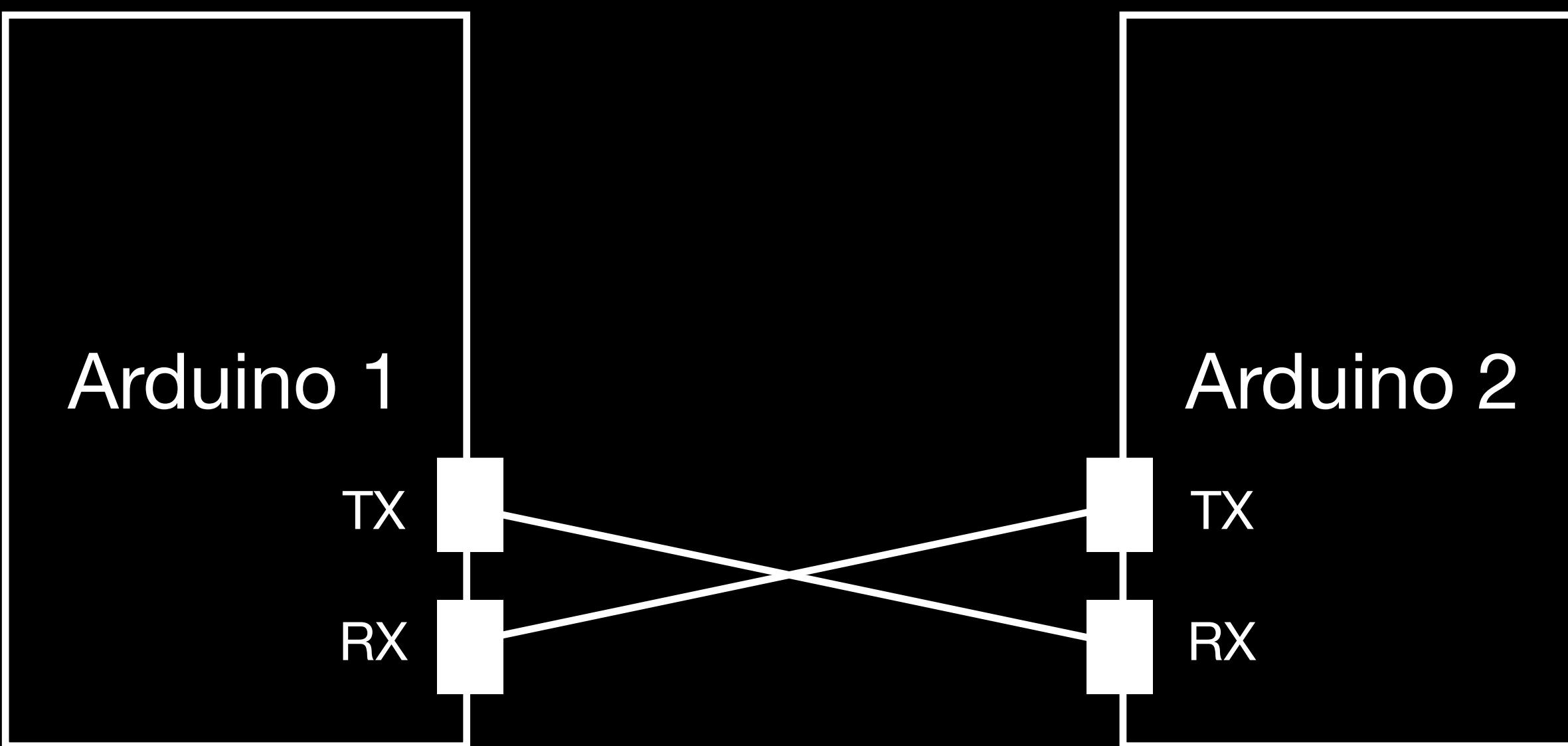
Essa lógica é bem parecida com a leitura do controle remoto do Projeto 02!



Serial  
Serial 3  
Serial 2  
Serial 1

TX = transmissão  
RX = recepção

Portas Seriais do Arduino Mega



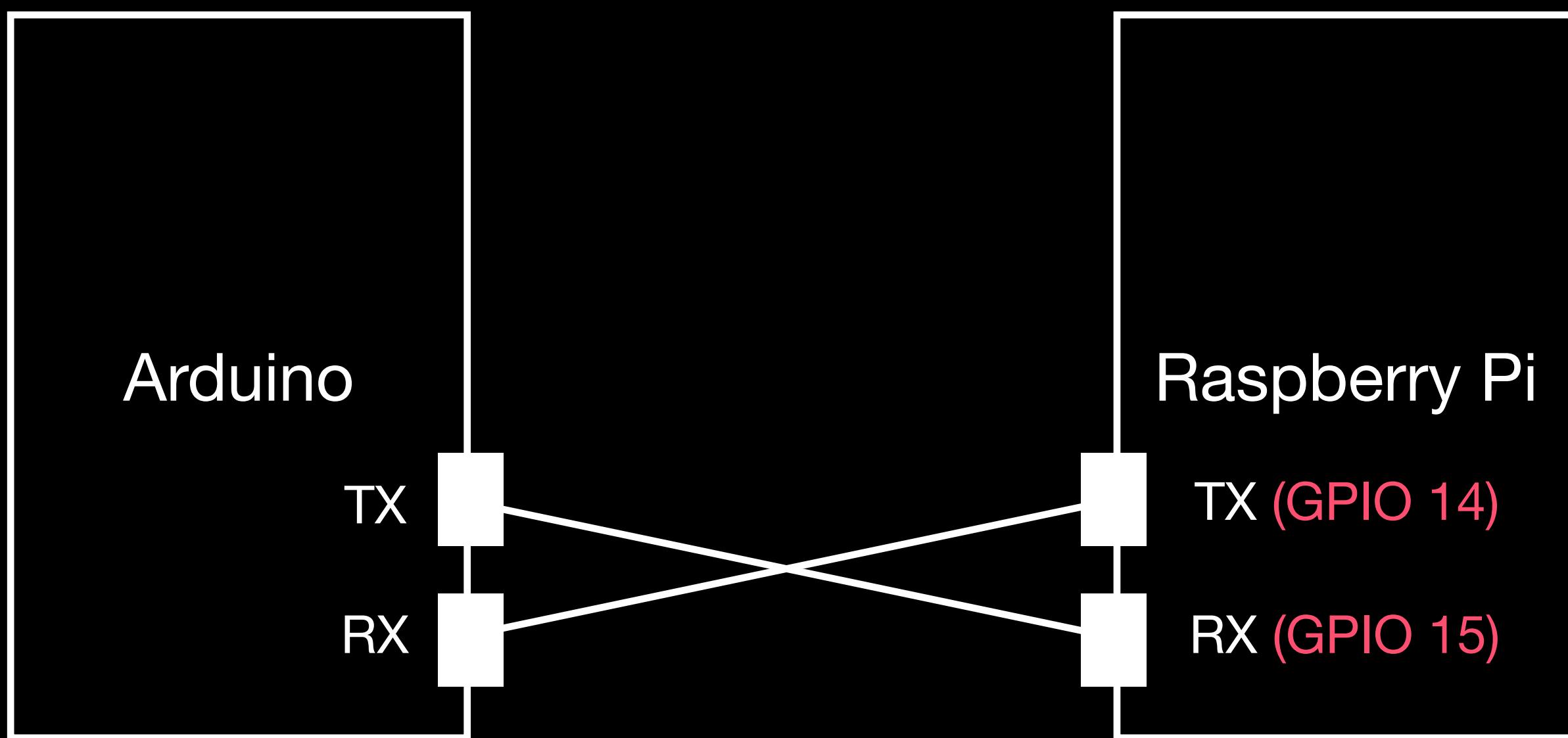
Comunicação Serial entre Arduinos

```
void setup () {  
    // conexão serial "0" com o computador  
    Serial.begin(9600);  
    Serial.setTimeout(10);  
  
    // conexão serial "1" com o outro Arduino  
    Serial1.begin(9600);  
    Serial1.setTimeout(10);  
}  
  
void loop () {  
    String texto = Serial1.readString();  
    texto.trim();  
    if (texto != "") {  
        if (texto == "1234abcd") {  
            Serial1.println("Ok!");  
            Serial.println("Texto recebido: " + texto);  
        }  
    }  
}
```

## Raspberry Pi 3 GPIO Header

<i>Pin#</i>	<i>NAME</i>		<i>NAME</i>	<i>Pin#</i>
01	3.3v DC Power		DC Power 5v	02
03	GPIO02 (SDA1 , I <sup>2</sup> C)		DC Power 5v	04
05	GPIO03 (SCL1 , I <sup>2</sup> C)		Ground	06
07	GPIO04 (GPIO_GCLK)		(TXD0) GPIO14	08
09	Ground		(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)		(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)		Ground	14
15	GPIO22 (GPIO_GEN3)		(GPIO_GEN4) GPIO23	16
17	3.3v DC Power		(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)		Ground	20
21	GPIO09 (SPI_MISO)		(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)		(SPI_CE0_N) GPIO08	24
25	Ground		(SPI_CE1_N) GPIO07	26
27	ID_SD (I <sup>2</sup> C ID EEPROM)		(I <sup>2</sup> C ID EEPROM) ID_SC	28
29	GPIO05		Ground	30
31	GPIO06		GPIO12	32
33	GPIO13		Ground	34
35	GPIO19		GPIO16	36
37	GPIO26		GPIO20	38
39	Ground		GPIO21	40

Portas Seriais do Raspberry Pi



Comunicação Serial entre Arduino e Raspberry Pi

A screenshot of a web browser displaying the pySerial 3.0 documentation. The URL in the address bar is [pythonhosted.org/pyserial/](http://pythonhosted.org/pyserial/). The page title is "Welcome to pySerial's documentation". On the left sidebar, there is a logo for "PY SERIAL" featuring a stylized serial port icon. The sidebar also contains links for "Table Of Contents", "Welcome to pySerial's documentation", "Indices and tables", "Next topic", "pySerial", "This Page", "Show Source", and a "Quick search" input field with a "Go" button. The main content area starts with a paragraph about the module's purpose and backends. It then lists "Other pages (online)" and "Contents:" with a detailed list of topics.

pySerial 3.0 documentation » [next](#) | [modules](#) | [index](#)

# Welcome to pySerial's documentation

This module encapsulates the access for the serial port. It provides backends for [Python](#) running on Windows, OSX, Linux, BSD (possibly any POSIX compliant system) and IronPython. The module named “serial” automatically selects the appropriate backend.

Other pages (online)

- [project page on GitHub](#)
- [Download Page with releases](#)
- This page, when viewed online is at <https://pyserial.readthedocs.org/en/latest/> or <http://pythonhosted.org/pyserial/>.

Contents:

- [pySerial](#)
  - [Overview](#)
  - [Features](#)
  - [Requirements](#)
  - [Installation](#)
  - [References](#)
  - [Older Versions](#)
- [Short introduction](#)
  - [Opening serial ports](#)
  - [Configuring ports later](#)
  - [Readline](#)
  - [Testing ports](#)

```
>> from serial import Serial  
  
>> meu_serial = Serial("/dev/serial0", baudrate=9600,  
timeout=0.01)  
  
>> texto = "Olá!" + "\n"  
>> meu_serial.write(texto.encode("UTF-8"))  
  
>> while True:  
...     retorno = meu_serial.readline()  
...     if retorno != "":  
...         texto_recebido = retorno.decode().strip()  
...         print(texto_recebido)
```

# Resumo da Ópera

## Funcionalidade

## Comandos

Sensor Ótico

```
int sensor = A11; pinMode(sensor, INPUT);
digitalRead(sensorOtico) == LOW // superfície é clara?
int valorAnalogico = analogRead(sensorOtico);
```

Shield Motor  
[acessar documentação](#)

```
#include <AFMotor.h>
AF_DCMotor motorA(1); AF_DCMotor motorB(2);
motorA.setSpeed(255); motorB.setSpeed(128);
motorA.run(FORWARD); motorB.run(BACKWARD);
motorA.run(RELEASE);
```

Strings  
[acessar documentação](#)

```
String texto = "Olá!", trecho = texto.substring(0, 3);
String texto1 = String(numero), texto2 = "aaa" + texto2;
int numero2 = texto2.toInt() + 2;
texto2 == texto3; bool comecaCom = texto.startsWith("Olá");
texto.trim(); texto.replace("a", "A");
```

Serial no Arduino  
[acessar documentação](#)

```
Serial.begin(9600); Serial1.begin(9600);
Serial.setTimeout(10);
Serial.print("Olá"); Serial1.println(2);
String texto = Serial.readString().trim();
```

PySerial  
[acessar documentação](#)

```
from serial import Serial
meu_serial = Serial("/dev/serial0", baudrate=9600)
texto = "Olá!" + "\n"
meu_serial.write(texto.encode("UTF-8"))
texto_recebido = meu_serial.readline().decode().strip()
```

Resumo para o Projeto 09

## Funcionalidade

Leitura Analógica  
[acessar documentação](#)

Servomotor  
[acessar documentação](#)

Braço Mecânico  
[acessar documentação](#)

EEPROM  
[acessar documentação](#)

## Comandos

```
int potenciometro = A10;  
pinMode(potenciometro, INPUT);  
int valorAnalogico = analogRead(potenciometro);  
int valorMapeado = map(valorAnalogico, 0, 1023, min, max);  
  
#include <Servo.h>  
Servo servo;  
servo.attach(pino); servo.detach();  
servo.write(anguloEmGraus);  
  
#include <meArm.h>  
int base = 12, ombro = 11, cotovelo = 10, garra = 9;  
meArm braco(  
    180, 0, -pi/2, pi/2, // ângulos da base  
    135, 45, pi/4, 3*pi/4, // ângulos do ombro  
    180, 90, 0, -pi/2, // ângulos do cotovelo  
    30, 0, pi/2, 0 // ângulos da garra  
);  
braco.begin(base, ombro, cotovelo, garra);  
braco.goDirectlyTo(x, y, z); braco.gotoPoint(x, y, z);  
braco.openGripper(); braco.closeGripper();  
braco.getX(); braco.getY(); braco.getZ(); braco.end();  
  
#include <EEPROM.h>  
EEPROM.get(endereco, minhaVariavel);  
EEPROM.put(endereco, minhaVariavel);
```

## Funcionalidade

Campainha Passiva  
[acessar documentação](#)

Interrupção  
[acessar documentação](#)

Contagem  
de Tempo  
[acessar documentação](#)

Encoder Rotativo  
[acessar documentação](#)

## Comandos

```
int campainhaPassiva = 5;  
pinMode(campainhaPassiva, OUTPUT);  
int frequencia = 220; int duracaoEmMs = 500;  
tone(campainhaPassiva, frequencia);  
tone(campainhaPassiva, frequencia, duracaoEmMs);  
noTone(campainhaPassiva);
```

```
int sensorDeSom = 19;  
pinMode(sensorDeSom, INPUT);  
int origem = digitalPinToInterrupt(sensorDeSom);  
attachInterrupt(origem, minhaFuncao, RISING);
```

```
unsigned long instanteAnteriorDeDeteccao = 0;  
  
if (millis() > instanteAnteriorDeDeteccao + 10) {  
    instanteAnteriorDeDeteccao = millis();  
}
```

```
#include <RotaryEncoder.h>  
RotaryEncoder encoder(20, 21);  
attachInterrupt(digitalPinToInterrupt(20), funcao, CHANGE);  
attachInterrupt(digitalPinToInterrupt(21), funcao, CHANGE);  
encoder.tick(); int posicao = encoder.getPosition();
```

## Funcionalidade

### Revisão de C++

### Print Serial

### Escrita/Leitura acessar documentação

### GButton acessar documentação

### ShiftDisplay acessar documentação

### Timer1 acessar documentação

## Comandos

```
int inteiro = 2; float decimal = 4.5; bool booleano = true;  
char texto[] = "Olá"; int listaDeInteiros[] = {1, 2, 3, 4};  
  
if (x > 0 && y > 0) {  
    z = 1;  
}  
else if (x < 0 || y < 0) {  
    z = 2;  
}
```

```
for (int i = 0; i < 5; i++) {  
    Serial.println(i);  
}  
float soma (float x) {  
    return x + 2;  
}
```

```
Serial.begin(9600); Serial.println("Olá"); Serial.println(2);
```

```
int led = 13; pinMode(led, OUTPUT); digitalWrite(led, LOW);  
int campainha = 3; digitalWrite(campainha, HIGH);  
int botao = A1; pinMode(botao, INPUT); digitalRead(botao) == LOW
```

```
#include <GButton.h>  
GButton botao(A1); botao.isPressed(); botao.process();  
botao.setPressHandler(funcao); botao.setReleaseHandler(funcao);
```

```
#include <ShiftDisplay.h>  
ShiftDisplay display(4, 7, 8, COMMON_ANODE, 4, true);  
ShiftDisplay display(4, 7, 8, COMMON_CATHODE, 4, true);  
display.set(1234); display.set(4.21, 2); display.set("Erro");  
display.update(); display.show(1000);
```

```
#include <TimerOne.h>  
Timer1.initialize(1000000); Timer1.attachInterrupt(funcao);
```