

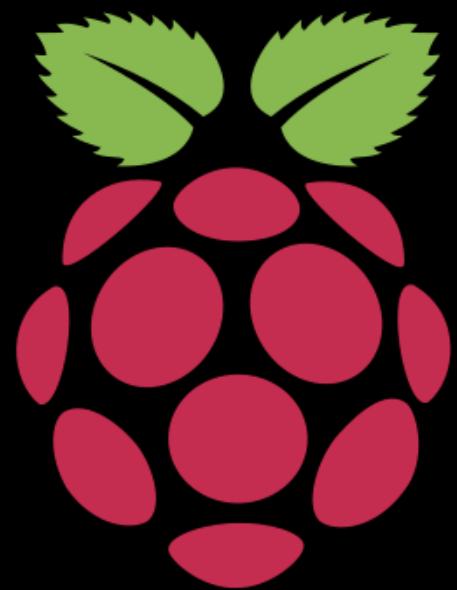


Projeto 06

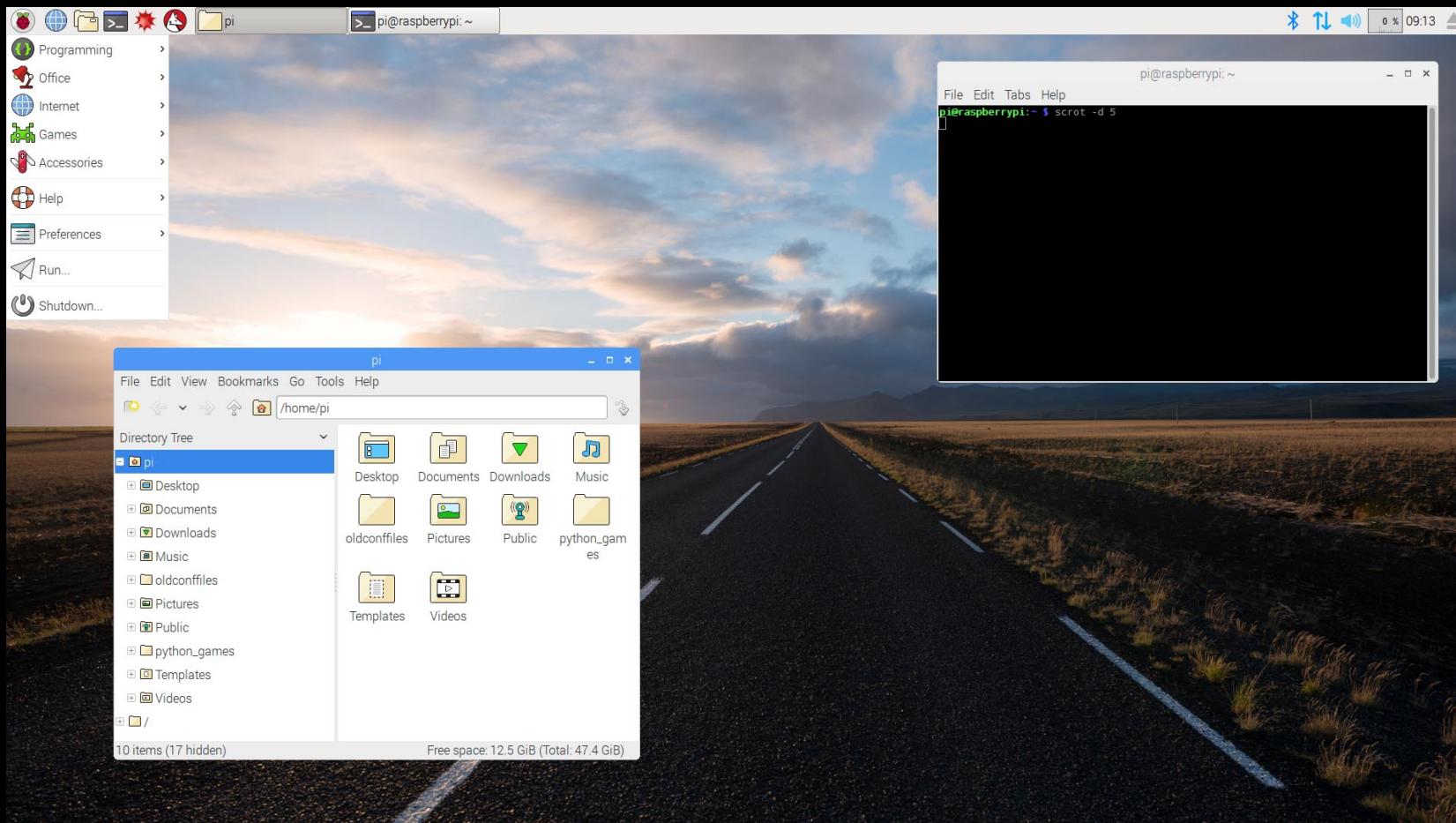
Controle de Tempo – Teoria

Jan K. S. – janks@puc-rio.br

ENG1419 – Programação de Microcontroladores



PC + microcontrolador



microcontrolador

A screenshot of the Arduino IDE. The title bar says "Timer | Arduino 1.6.13". The main area shows an open sketch named "Timer" with the following C++ code:

```
/*
ShiftDisplay example
by MiguelPynto
One minute clock timer, pausable with button
https://miguelpynto.github.io/ShiftDisplay/
*/
#include "ShiftDisplay.h"

const int DISPLAY_TYPE = COMMON_CATHODE; // COMMON_CATHODE or COMMON_ANODE
const int DISPLAY_SIZE = 4; // number of digits on display
const int BUTTON_PIN = A1; // connect one end of button to pin 2 and other to ground

volatile bool buttonPressed;

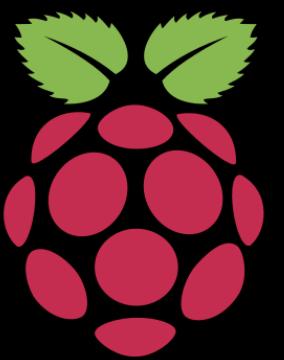
ShiftDisplay display(4, 7, 8, COMMON_ANODE, 4, true);

void debounce() {
  display.show(100); // execution will delay for 100ms
}

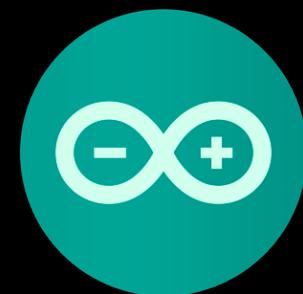
void buttonPressInterrupt() {
  buttonPressed = true;
}
```

37 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) em /dev/cu.usbmodem1441

Diferença Básica entre Raspberry Pi e Arduino



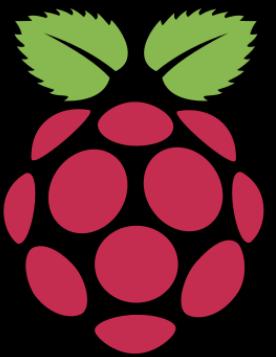
Raspberry Pi 3 B



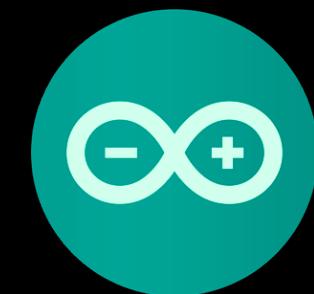
Arduino Mega 2560

Preço	R\$ 212,42	R\$ 67,90
Pinos Bem Identificados	não	sim
Pinos Digitais	26 (1 para PWM)	54 (15 para PWM)
Entradas Analógicas	0	16
Acessórios Shield	poucos	muitos
Potência em Repouso	1.5 W	0.425 W
Complexidade de Software	instalar e configurar Linux	nenhuma
Tempo de Inicialização	≈ 1 minuto	0s

Vantagens do Arduino



Raspberry Pi 3 B

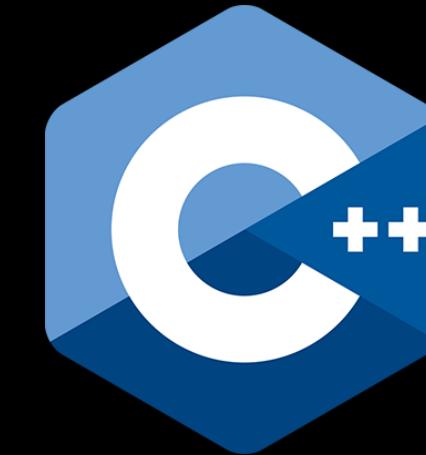


Arduino Mega 2560

Processador	4 núcleos de 1.2GHz	1 núcleo de 16 MHz
Memória RAM	1 GB	8 kB
Armazenamento	cartão SD até 64 GB	256 kB + 4 kB
WiFi e Bluetooth	incluídos na placa	só com acessórios extras
Sistema Operacional	Linux	(nenhum)
Precisa de computador?	não	sim
Linguagem de Programação	Python e muitas outras	C / C++
Roda Crysis?	não	não

Vantagens do Raspberry Pi

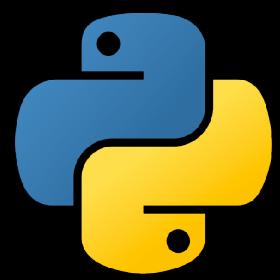
Revisão de C/C++



- operações
- variáveis
- listas
- textos
- funções
- condicionais
- loops



- operações
- variáveis
- listas
- textos
- funções
- condicionais
- loops



python

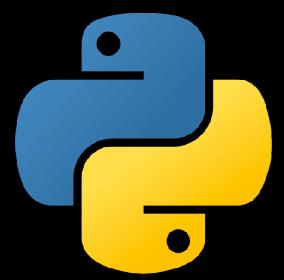


```
inteiro = 15  
decimal = 4.5  
lista = [1, 2, 3]  
texto = "Texto"  
booleano = True
```

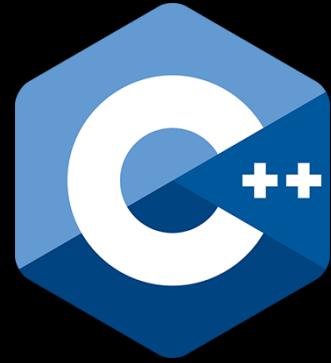
tipo de variável

```
int inteiro = 15;  
float decimal = 4.5;  
int lista[] = {1, 2, 3};  
char texto[] = "Texto";  
bool booleano = true;
```

ponto-e-vírgula

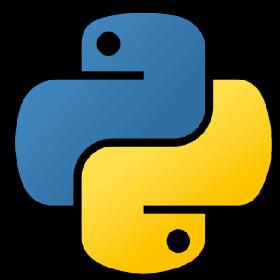


python

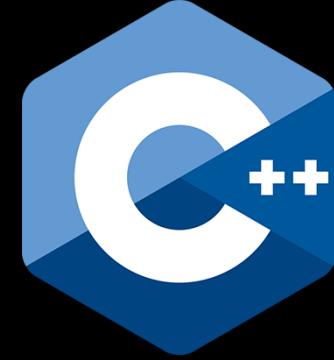


```
lista = [1, 2, 3]
x = lista[0]
lista[1] = -99
```

```
int lista[] = {1, 2, 3};
int x = lista[0];
lista[1] = -99;
```



python

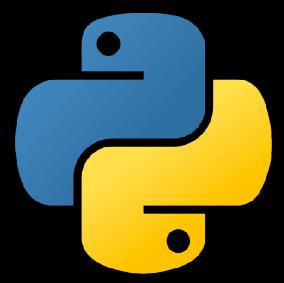


```
texto = "Texto"  
letra = texto[0]  
x = (letra == "T")
```

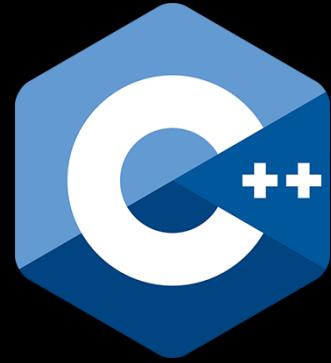
```
char texto[] = "Texto";  
char letra = texto[0];  
bool x = (letra == 'T');
```

aspas DUPLAS
para texto

aspas SIMPLES
para caracteres

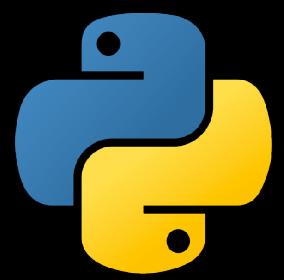


python



```
# linha de comentário  
from threading import Timer
```

```
// linha de comentário  
#include <TimerOne.h>
```



python

```
def soma2(x):  
    return x + 2  
  
y = soma2(4.5)
```

```
def imprime_oi():  
    print("oi!")
```

**tipo da variável
de RETORNO**

```
float soma2 (float x) {  
    return x + 2;  
}
```

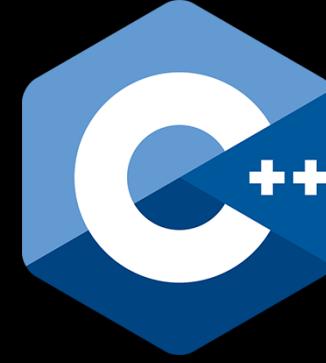
```
float y = soma2(4.5);
```

retorna "vazio"
void imprime0i () {
 Serial.print("oi!");
}

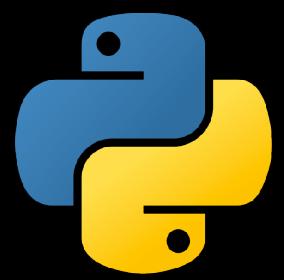


python

```
if x > 0 and y > 0:  
    z = 1  
elif x < 0 or y < 0:  
    z = 2  
elif x != 0:  
    z = 3  
else:  
    z = 4
```



```
if (x > 0 && y > 0) {  
    z = 1;  
}  
else if (x < 0 || y < 0) {  
    z = 2;  
}  
else if (x != 0) {  
    z = 3;  
}  
else {  
    z = 4;  
}
```



python

```
x = 50
while x > 0:
    x = x / 5
```

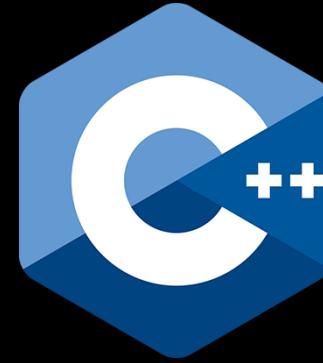
```
for i in range(0, 5):
    print(i)
```

```
float x = 50;
while (x > 0)
{
    x = x / 5;
}
```

```
for (int i = 0; i < 5; i++)
{
    Serial.println(i);
}
```

inicie i como 0;
a cada ciclo, enquanto i < 5,
faça i aumentar em 1 unidade

Loops



Hardware

pt.wikipedia.org/wiki/Arduino

Não autenticado Discussão Contribuições Criar uma conta Entrar

Artigo Discussão Ler Editar Editar código-fonte Ver histórico Pesquisar na Wikipédia

Participe do concurso cultural Wiki Loves Earth Brasil 2018 Confira o regulamento do concurso!

Arduino

Origem: Wikipédia, a enciclopédia livre.

Nota: Para outros significados, veja [Arduino \(desambiguação\)](#).

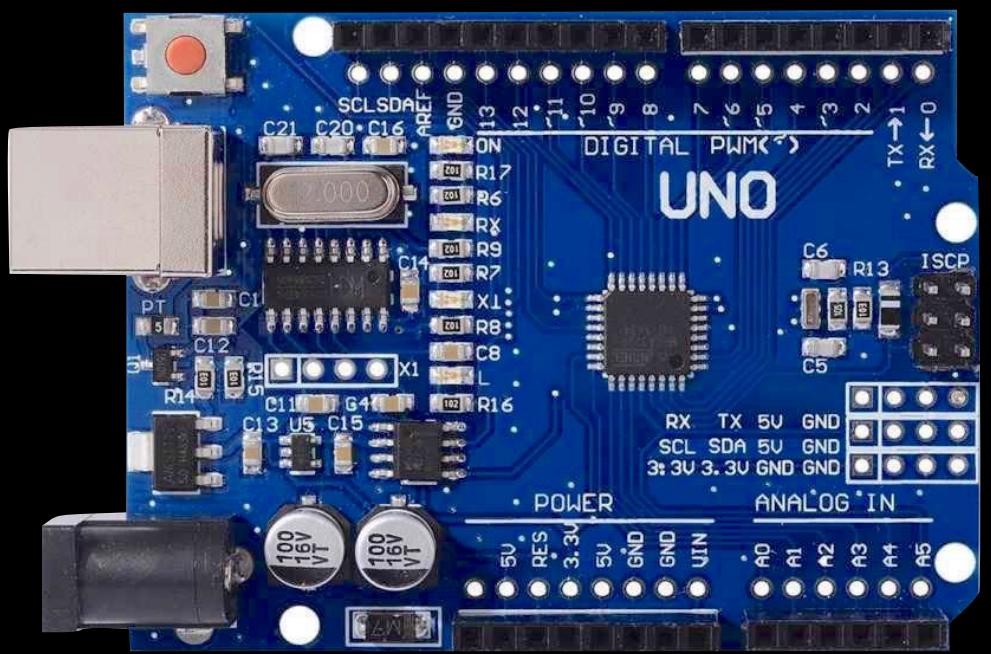
Arduino^{[2][4][5]} é uma plataforma de prototipagem eletrônica de hardware livre e de placa única,^[6] projetada com um microcontrolador Atmel AVR com suporte de entrada/saída embutido, uma linguagem de programação padrão,^[7] a qual tem origem em [Wiring](#), e é essencialmente C/C++.^[8] O objetivo do projeto é criar ferramentas que são acessíveis, com baixo custo, flexíveis e fáceis de se usar por novatos e profissionais. Principalmente para aqueles que não teriam alcance aos controladores mais sofisticados e de ferramentas mais complicadas.^[9]

Pode ser usado para o desenvolvimento de objetos interativos independentes, ou ainda para ser conectado a um computador hospedeiro. Uma típica placa Arduino é composta por um controlador, algumas linhas de E/S digital e analógica, além de uma interface serial ou USB, para interligar-se ao hospedeiro, que é usado para programá-la e interagi-la em tempo real. Ela em si não possui qualquer recurso de rede, porém é comum combinar um ou mais Arduinos deste modo, usando extensões apropriadas chamadas de shields^[10]. A interface do hospedeiro é simples, podendo ser escrita em várias linguagens. A mais popular é a Processing, mas outras que podem comunicar-se com a conexão serial são: Max/MSP,^[11] Pure Data,^[12] SuperCollider,^[13] ActionScript^[14] e Java.^[15] Em 2010 foi realizado um documentário sobre a plataforma chamado [Arduino: The Documentary](#).

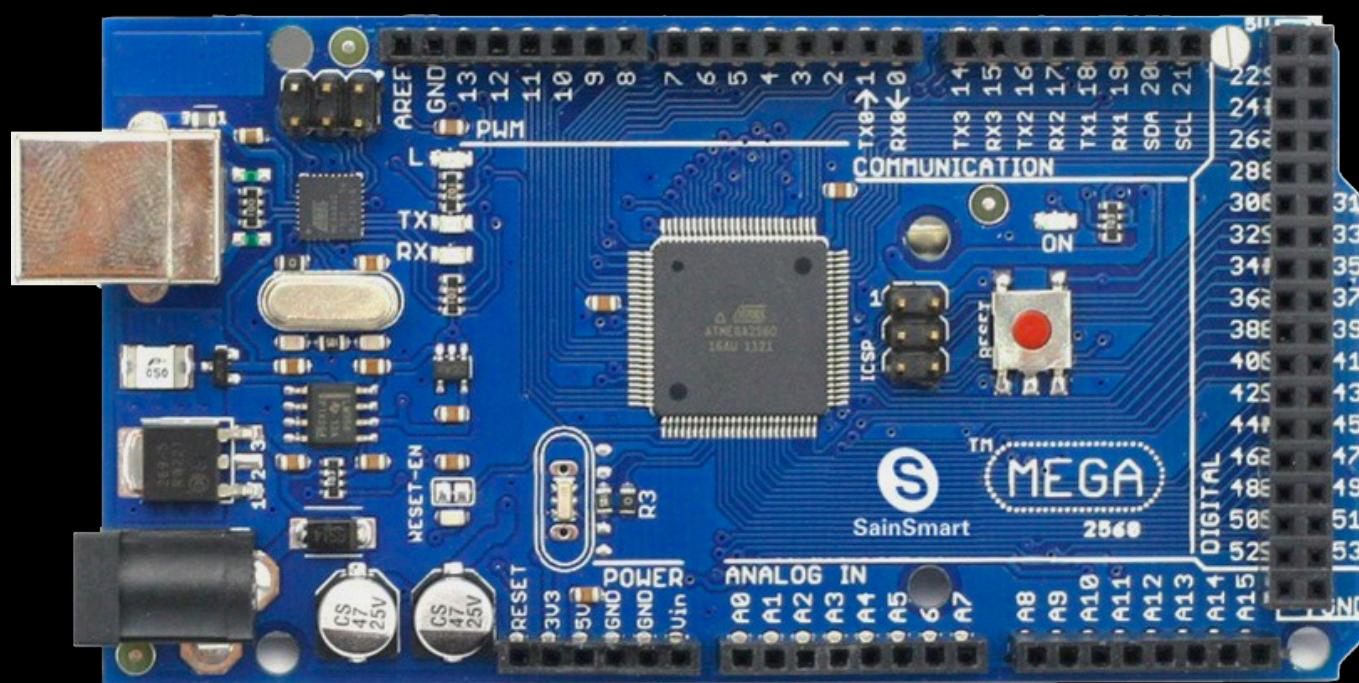


Dados sobre o Arduino

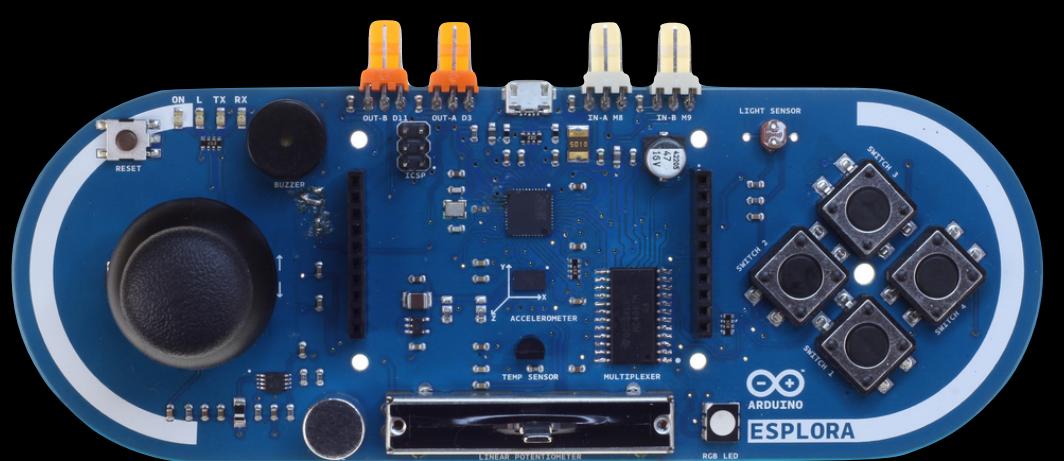
Arduino Uno



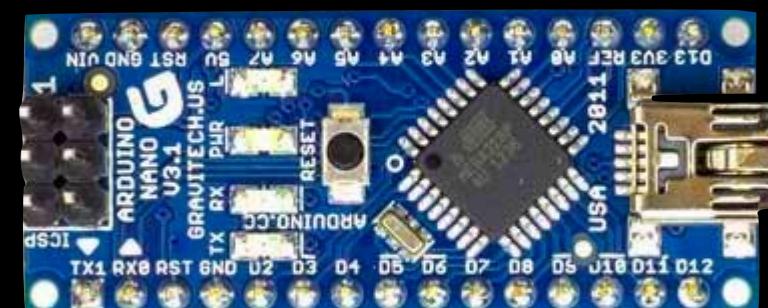
Arduino Mega 2560



Arduino Esplora



Arduino Nano

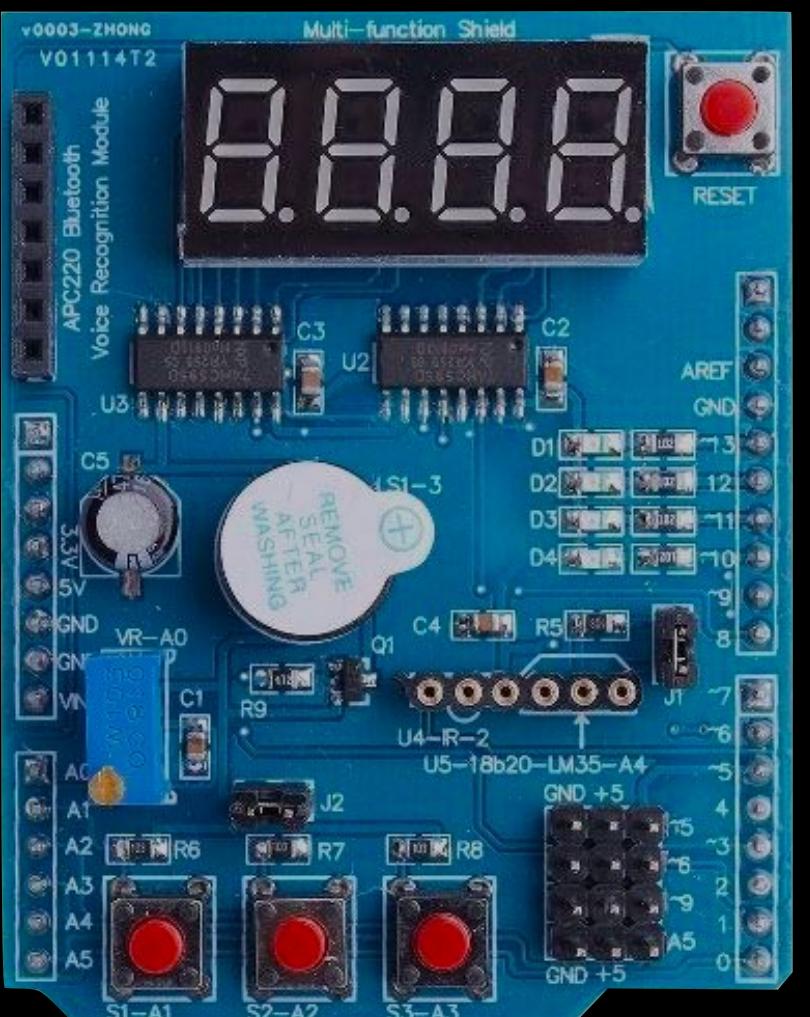


Exemplos de Modelos de Arduino

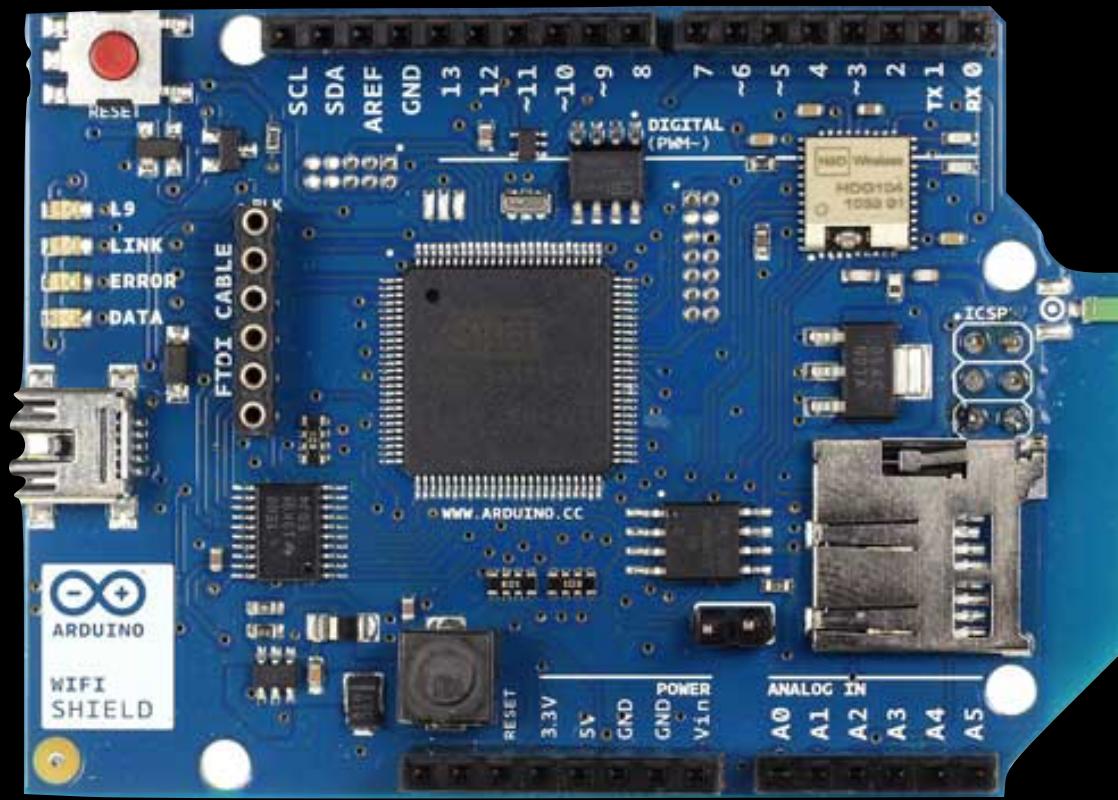


Arduino Mega 2560

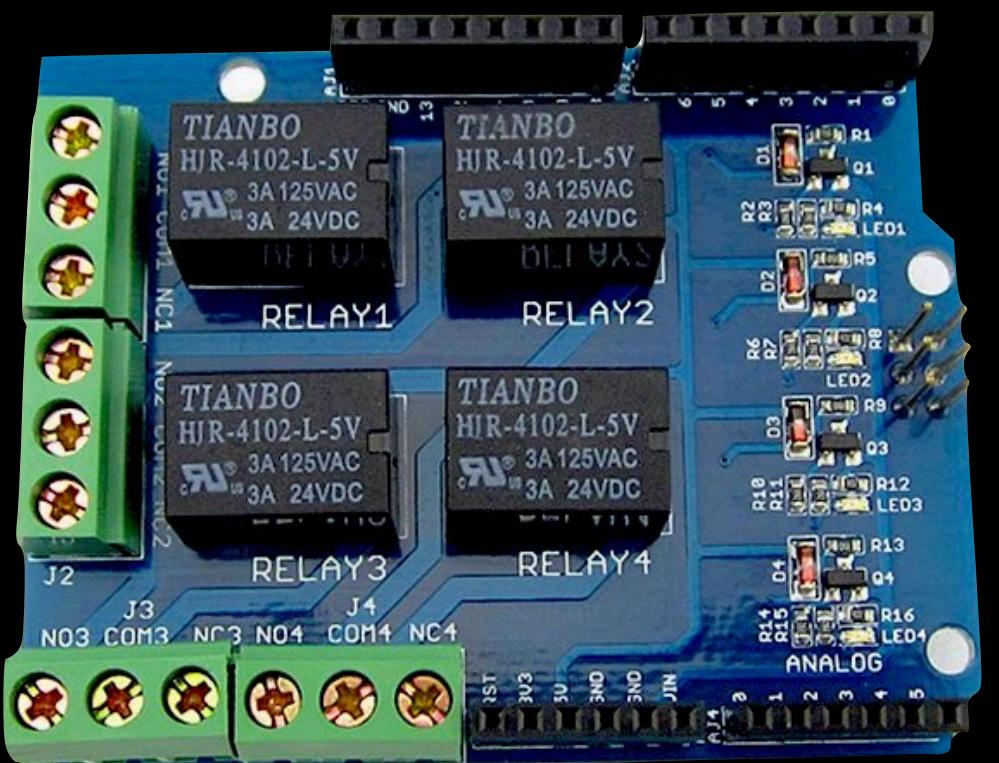
Shield Multifunção



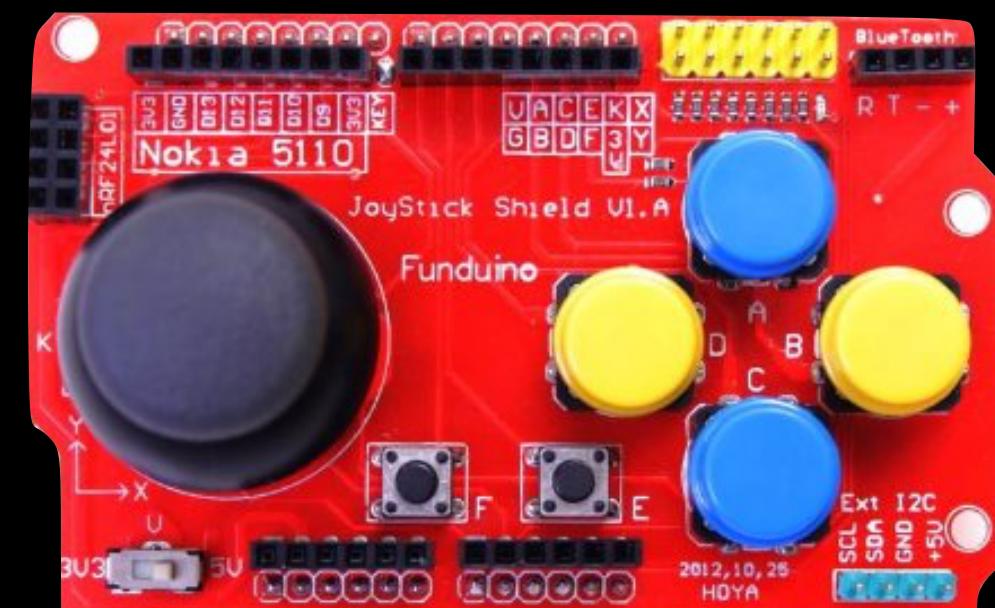
Shield com WiFi



Shield com Relés

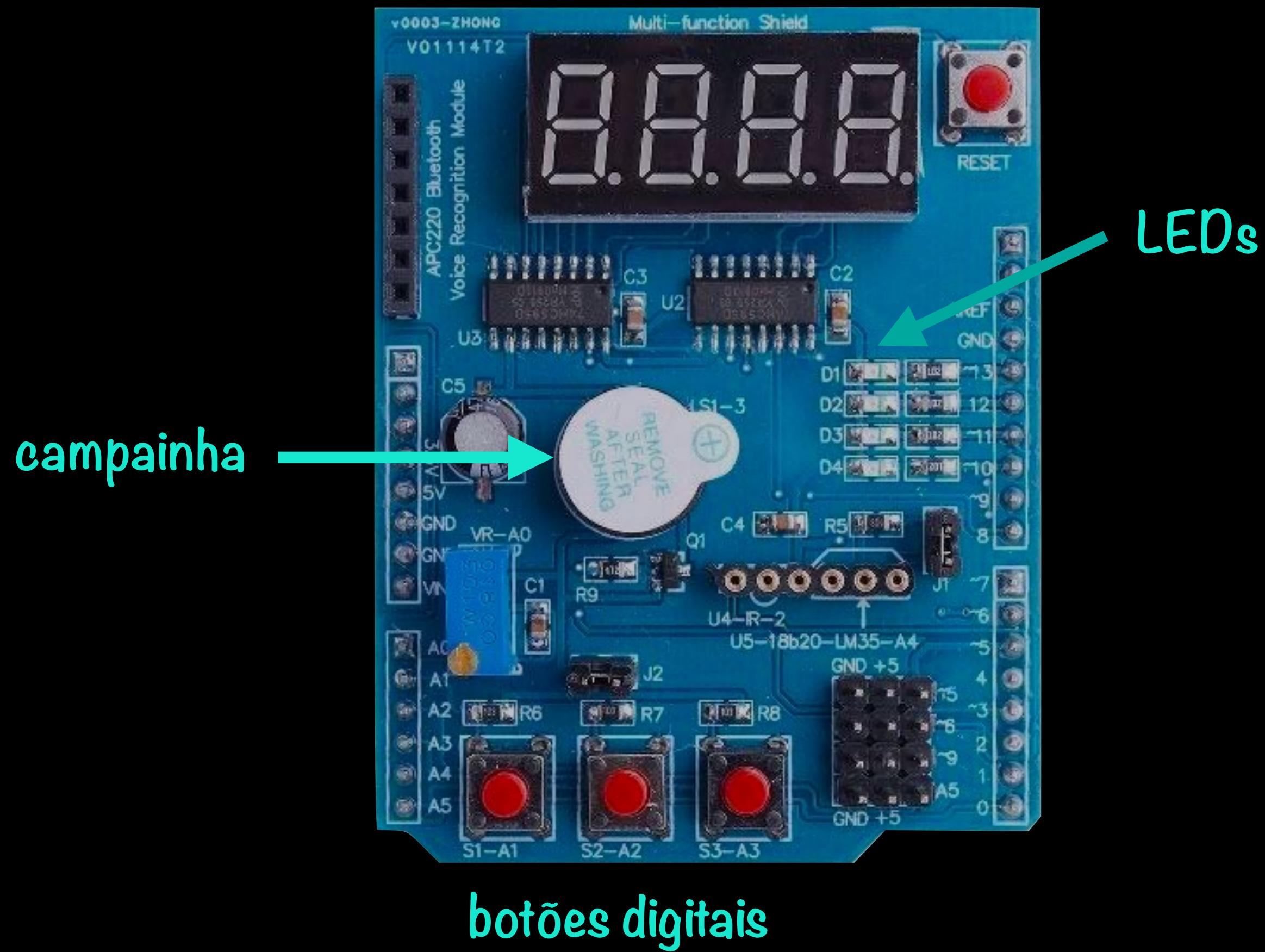


Shield com Joystick



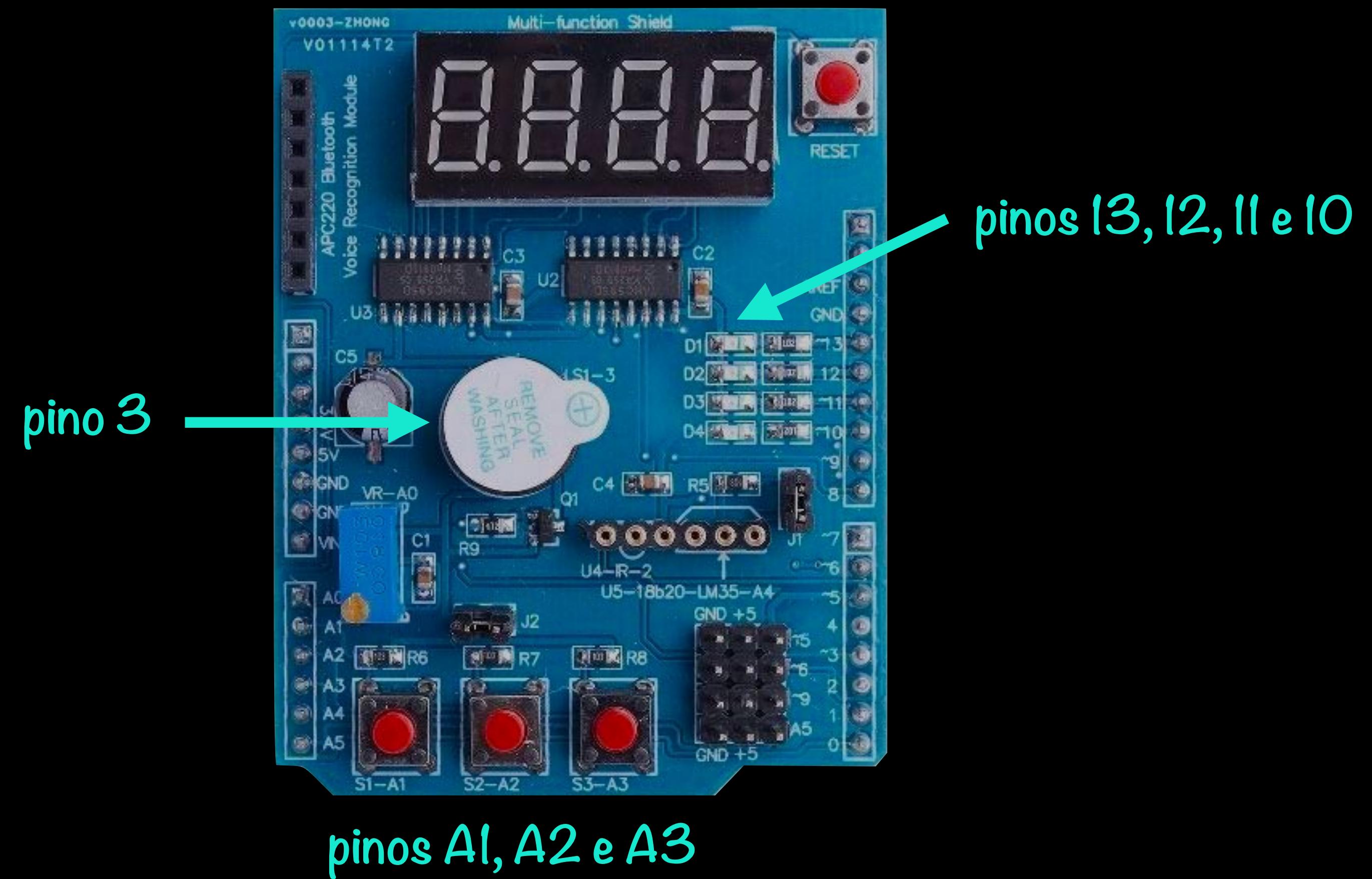
Exemplos de Shields para Arduino

displays de 7 segmentos

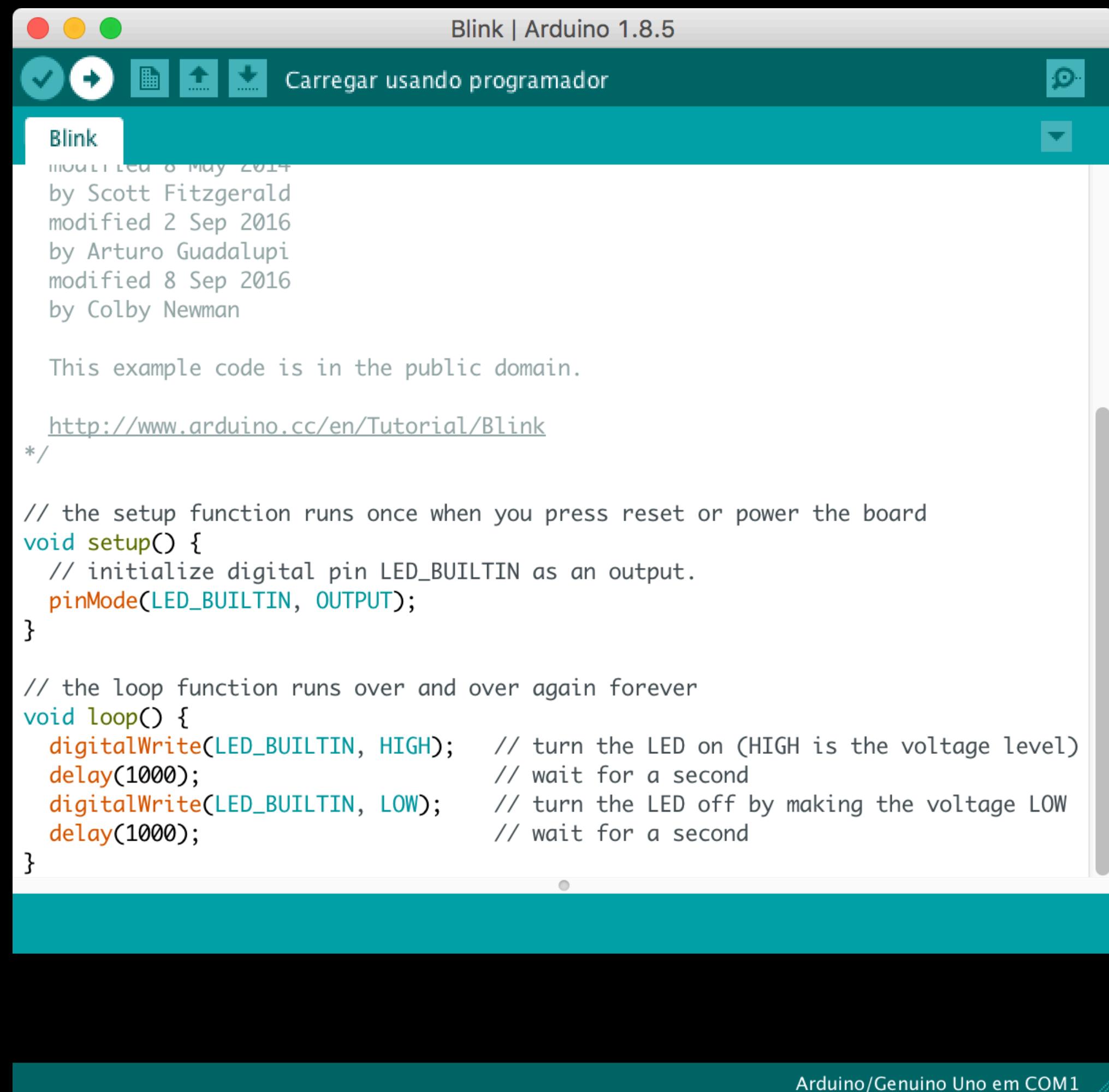


Shield Multifunção

pinos 4, 7 e 8



Pinos Usados pelo Shield Multifunção



Blink | Arduino 1.8.5

Carregar usando programador

Blink

modified 8 May 2014
by Scott Fitzgerald
modified 2 Sep 2016
by Arturo Guadalupi
modified 8 Sep 2016
by Colby Newman

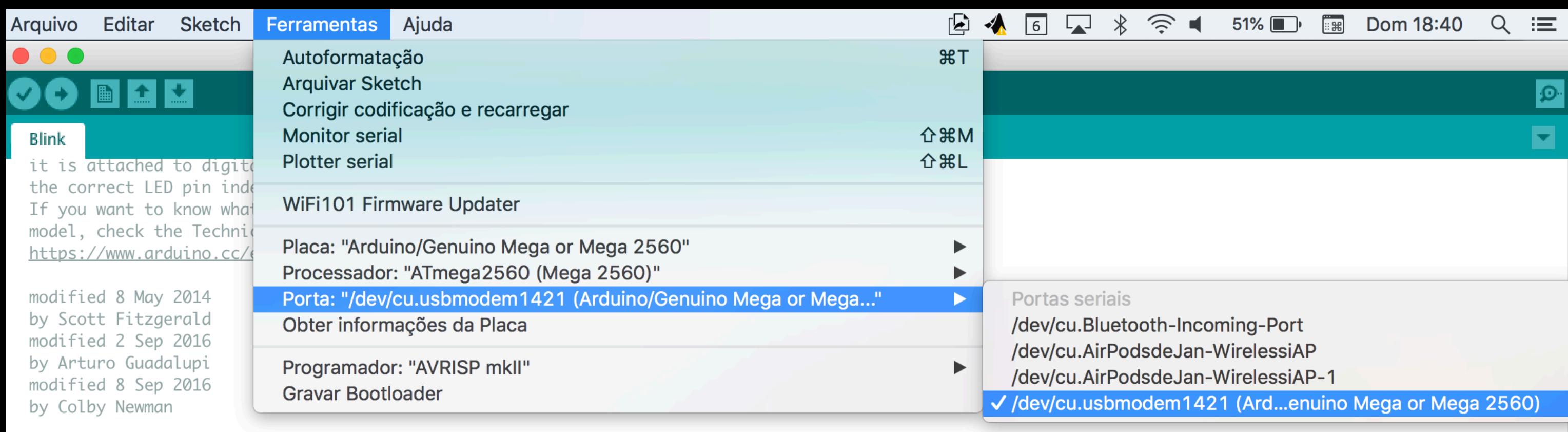
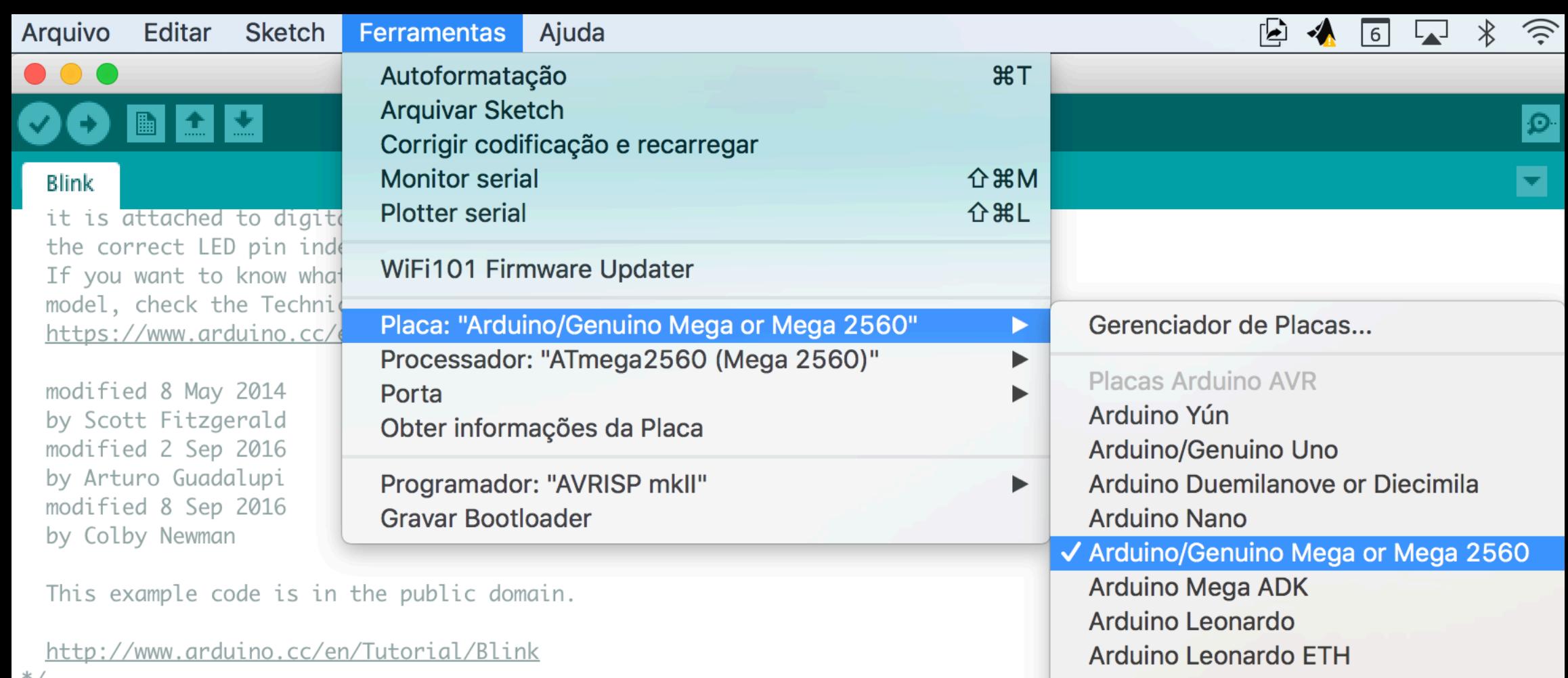
This example code is in the public domain.

<http://www.arduino.cc/en/Tutorial/Blink>

```
/*
 * the setup function runs once when you press reset or power the board
void setup() {
    // initialize digital pin LED_BUILTIN as an output.
    pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(LED_BUILTIN, HIGH);      // turn the LED on (HIGH is the voltage level)
    delay(1000);                        // wait for a second
    digitalWrite(LED_BUILTIN, LOW);       // turn the LED off by making the voltage LOW
    delay(1000);                        // wait for a second
}
```

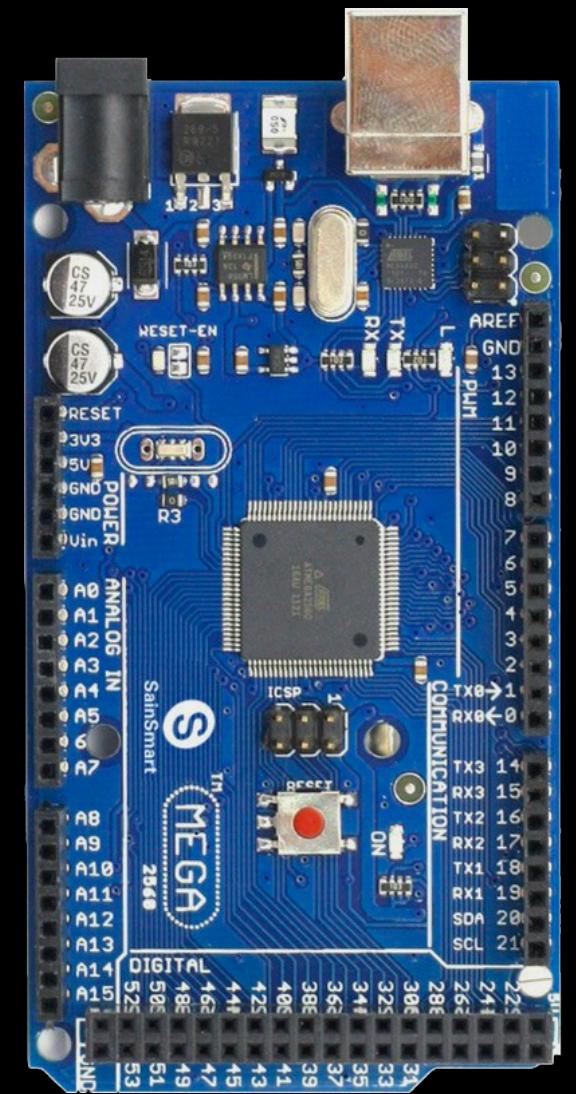
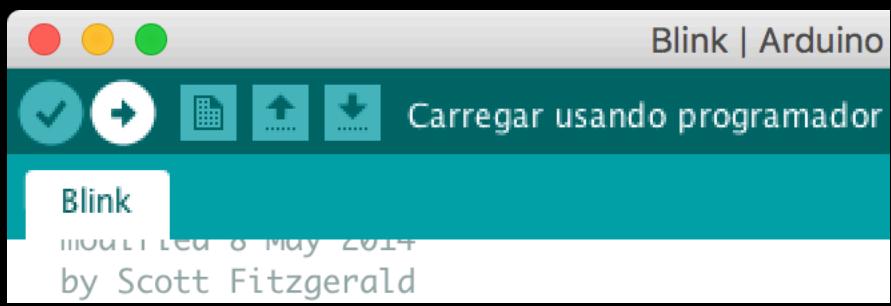
Arduino/Genuino Uno em COM1



Configuração de Comunicação com o Arduino

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);    // turn the LED on (HIGH is the voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);     // turn the LED off by making the voltage LOW
  delay(1000);                      // wait for a second
}
```



escreve programa
no computador

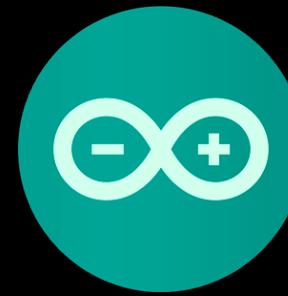
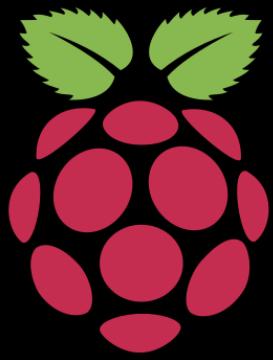


envia programa
pela USB



executa programa

Programação do Arduino



```
# bibliotecas
from gpiozero import LED, Button

# funções
def funcao1(x):
    return x + 2

# inicialização de componentes
led = LED(21)
botao = Button(11)
botao.when_pressed = funcao1

# loop infinito
while True:
    ...
```

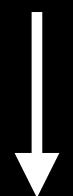
```
// bibliotecas
#include <GButton.h>

// funções
int funcao1(int x) {
    return x + 2;
}

// inicialização de componentes
int led = 21;
GButton botao(11);
void setup () {
    pinMode(led, OUTPUT);
    botao.setPressHandler(funcao1);
}

// loop infinito
void loop () {
    ...
}
```

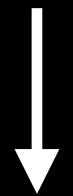
Início do Programa



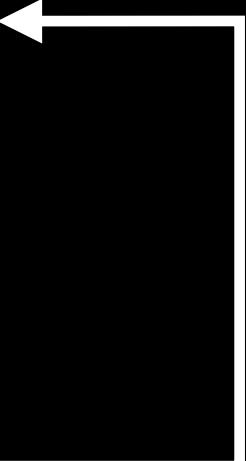
```
void setup () {  
    ...  
}
```

É possível reiniciar um programa no Arduino (RESET), mas não pará-lo.

Para remover um programa, é preciso inserir outro (ex: um em branco).



```
void loop () {  
    ...  
}
```



```
void setup () {  
    // inicia comunicação com taxa de 9600 bit/s  
    Serial.begin(9600);  
}  
  
void loop () {  
    Serial.println("Hey, listen!");  
  
    // aguarda tempo em MILISSEGUNDOS  
    delay(1000);  
}
```

Exemplo Básico de Setup/Loop para Imprimir Texto

teste | Arduino 1.8.5

```
teste
1 void setup () {
2     Serial.begin(9600);
3 }
4
5 void loop () {
6     Serial.println("Hey, listen!");
7     delay(1000); // aguarda tempo em MILISSEGUNDOS
8 }
9
10
```

Carregado.

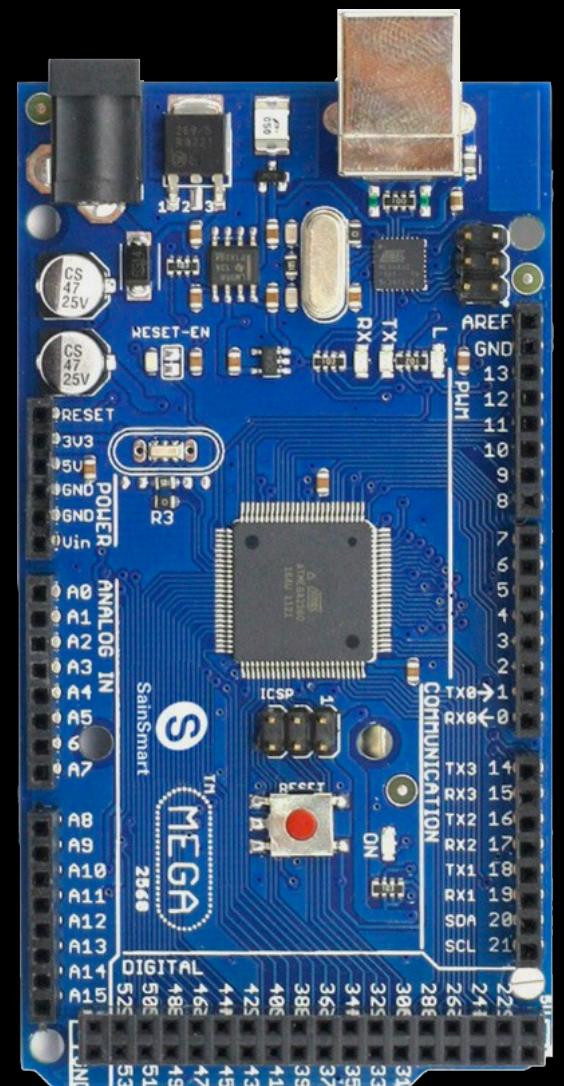
/dev/cu.usbmodem1421 (Arduino/Genuino Mega or Mega 2560)

Enviar

0 sketch usa Hey, listen!
Variáveis glo Hey, listen!
Hey, listen!

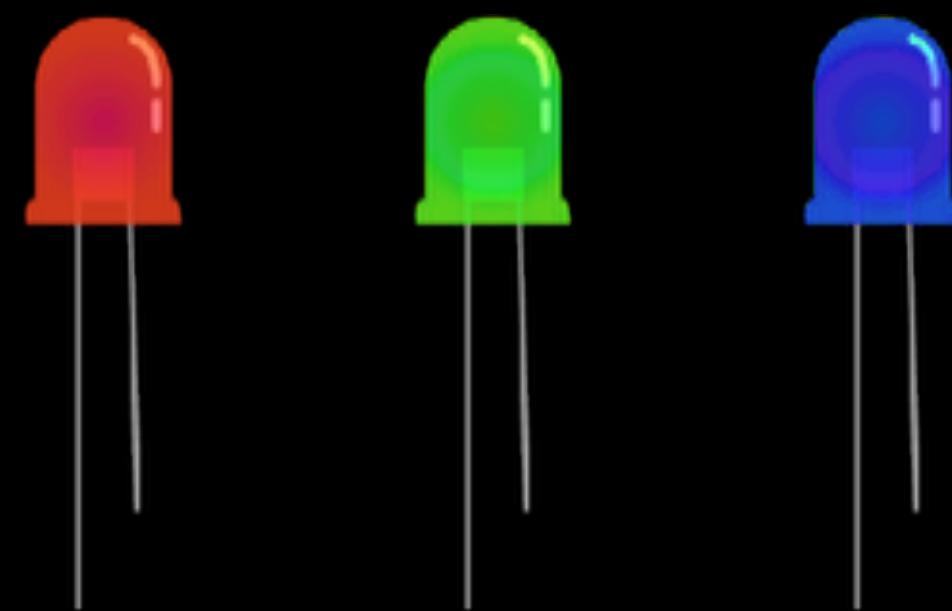
10 Arduino/G...

Auto-rolagem Nenhum final-de-linha 9600 velocidade Deleta a saída

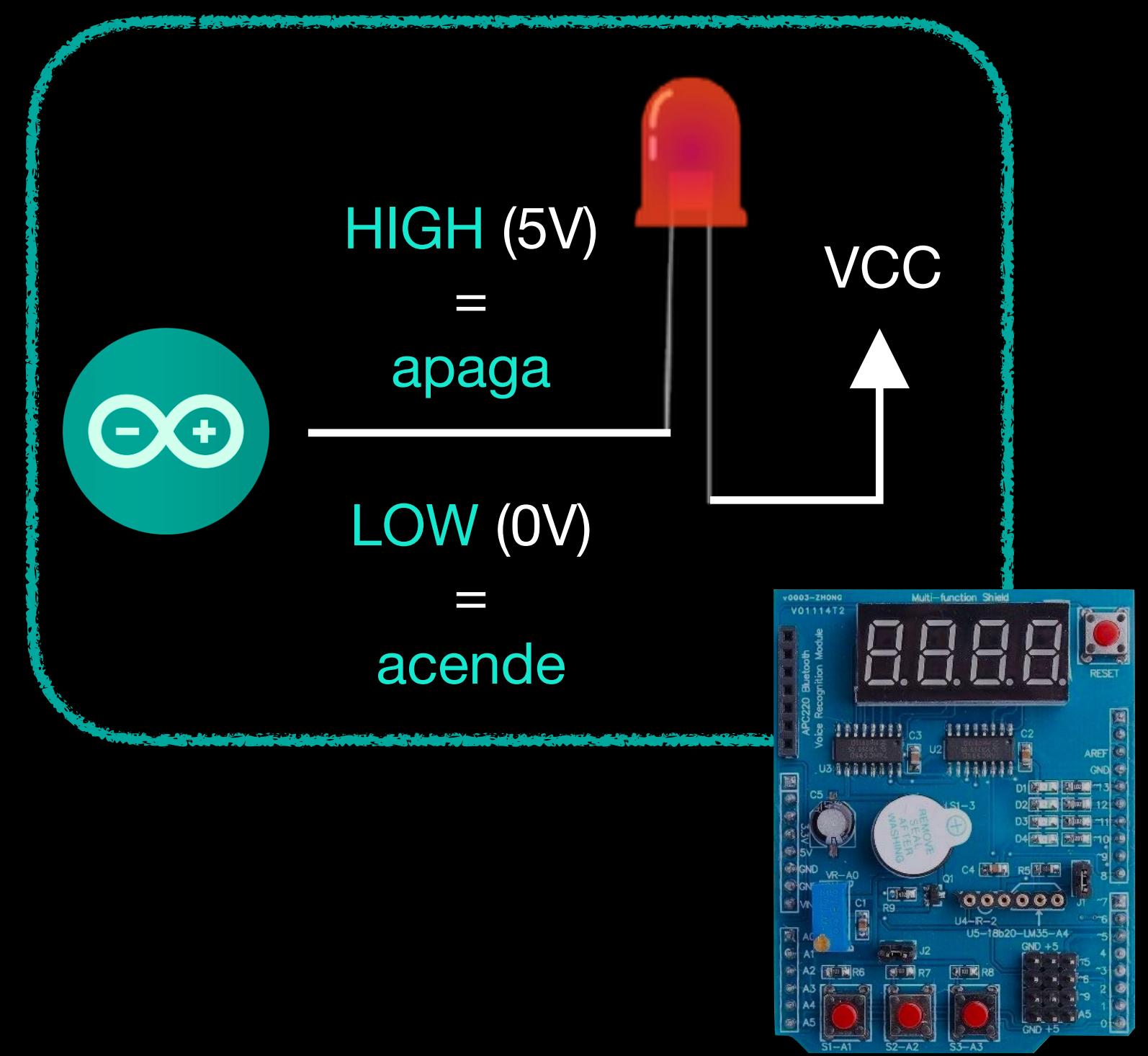
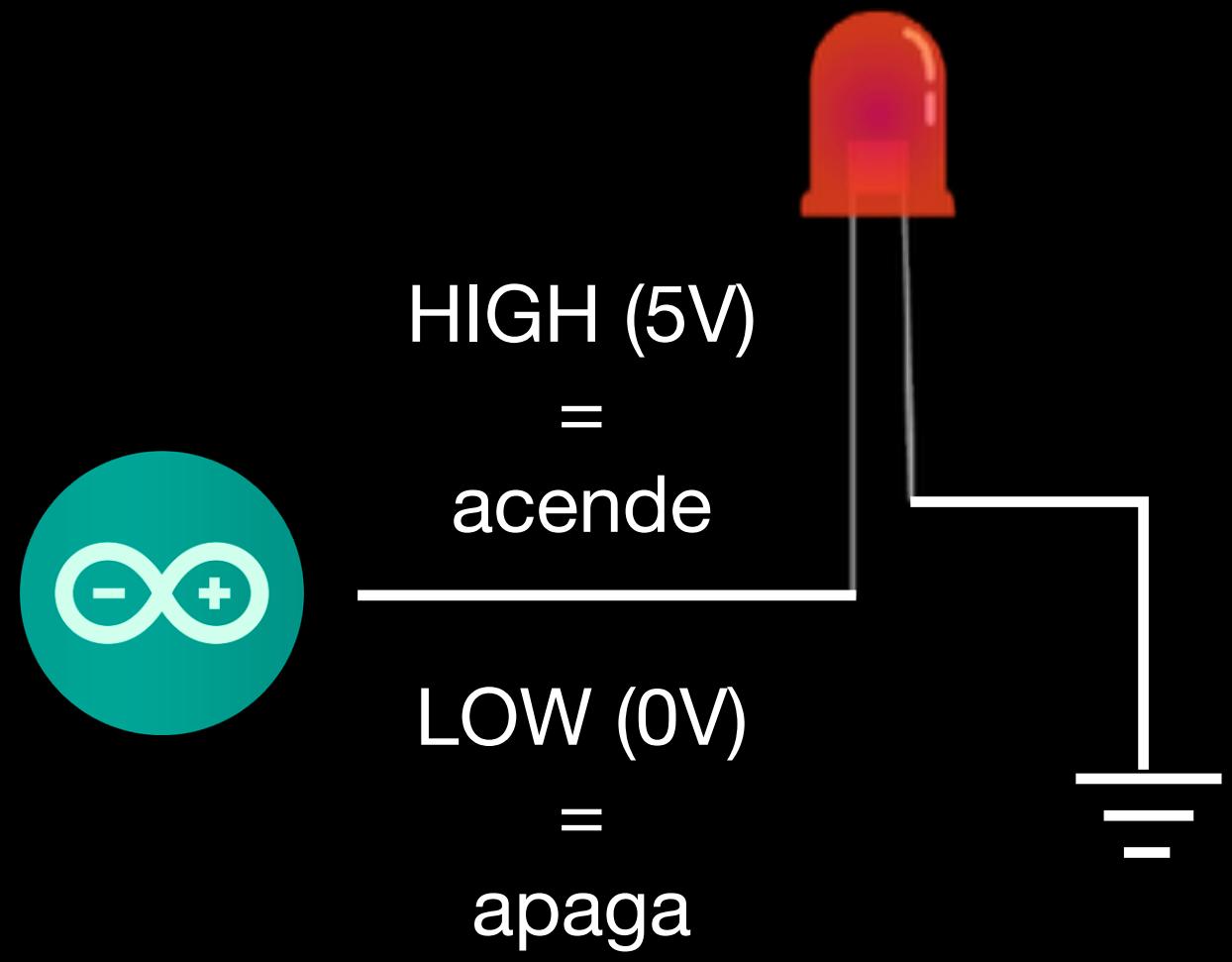


envio de textos
via USB (serial)

Impressão de Textos via Comunicação Serial



LEDs



Duas Formas de Usar um LED

```
int led = 13; // LED 1 está no pino 13
```

```
void setup () {  
    pinMode(led, OUTPUT);  
    digitalWrite(led, HIGH);  
}
```

```
void loop () {  
    digitalWrite(led, LOW);  
}
```



o primeiro digitalWrite não é necessário neste caso, mas é uma boa prática para casos gerais (por padrão, o pino começa com LOW)

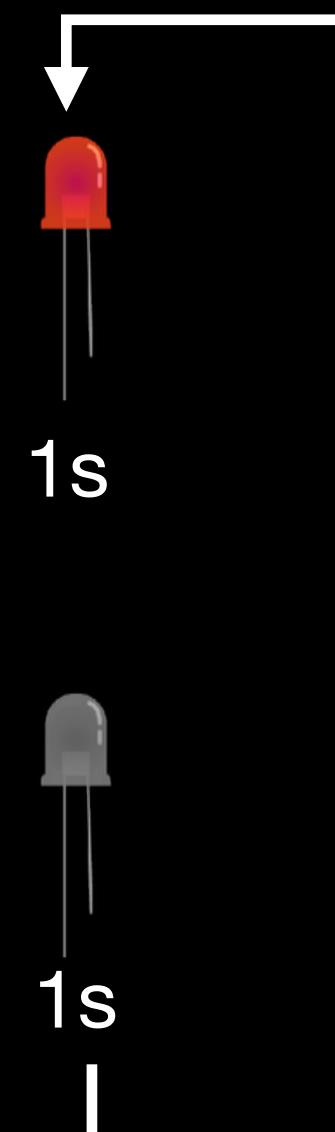


```
int led = 13;

void setup () {
    pinMode(led, OUTPUT);
    digitalWrite(led, HIGH);
}

void loop () {
    digitalWrite(led, LOW);
    delay(1000);

    digitalWrite(led, HIGH);
    delay(1000);
}
```





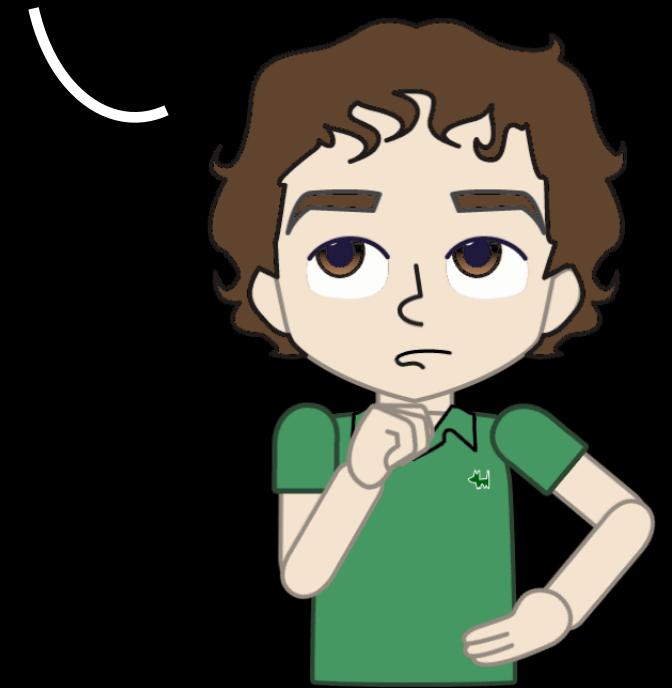
Campainha Ativa

```
int campainha = 3;
```

```
void setup () {  
    pinMode(campainha, OUTPUT);  
    digitalWrite(campainha, HIGH);  
}
```

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

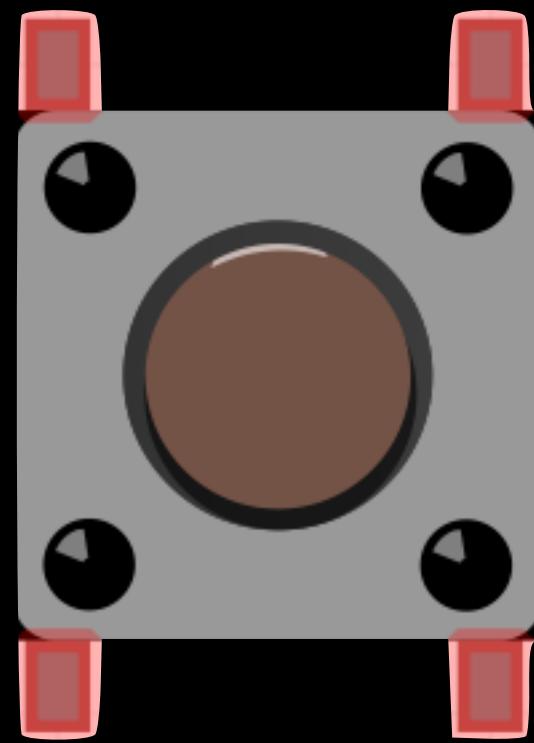
o controle da campainha é
exatamente igual ao de um LED



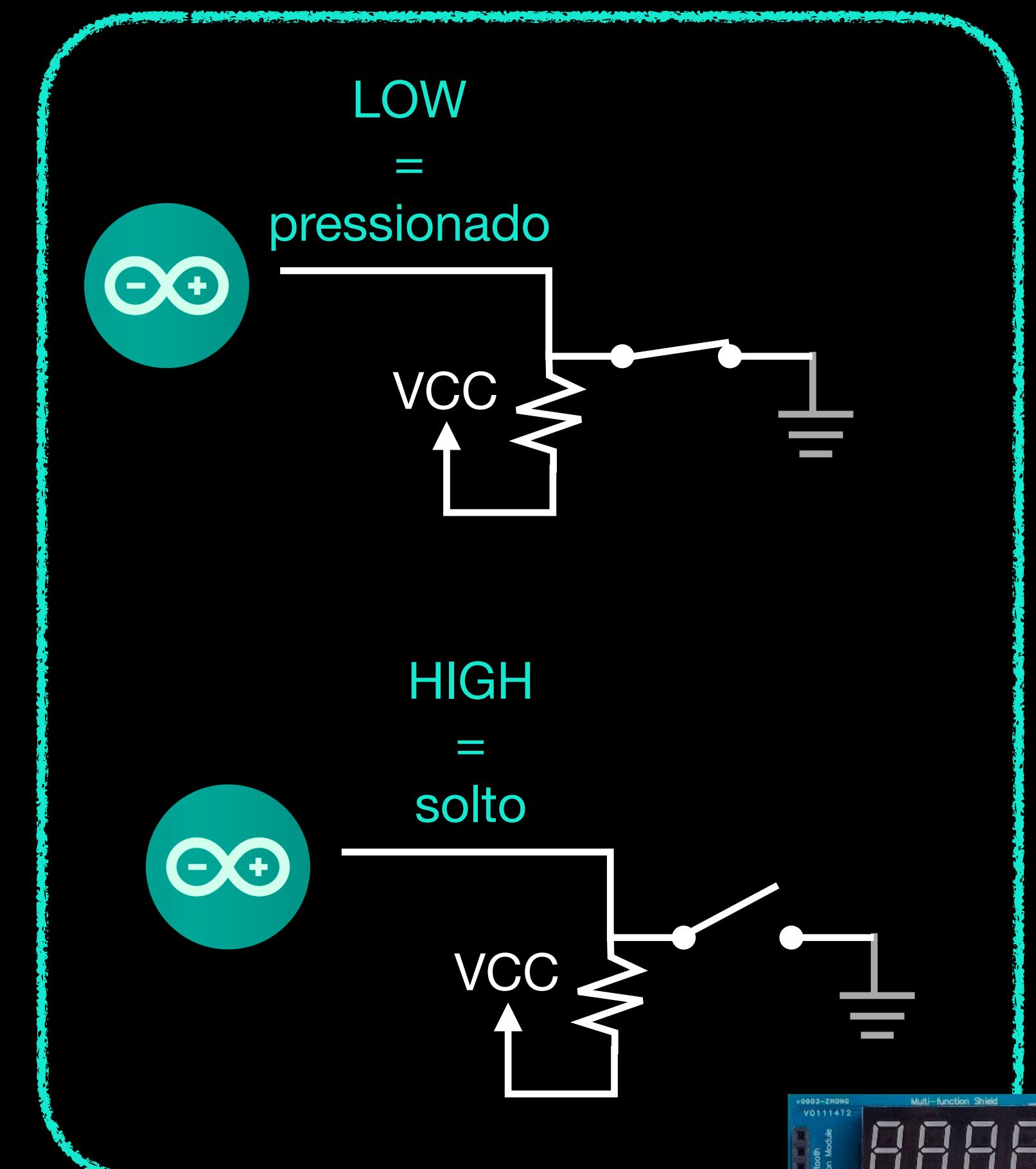
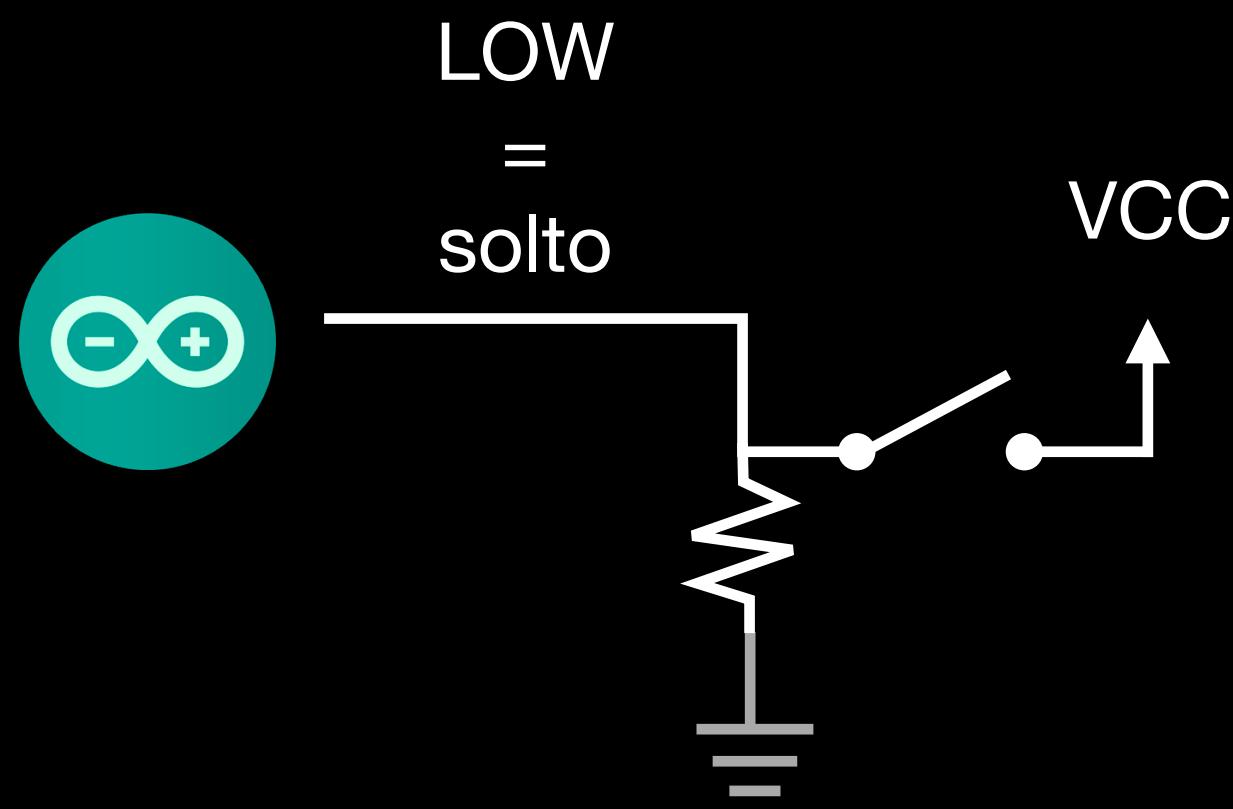
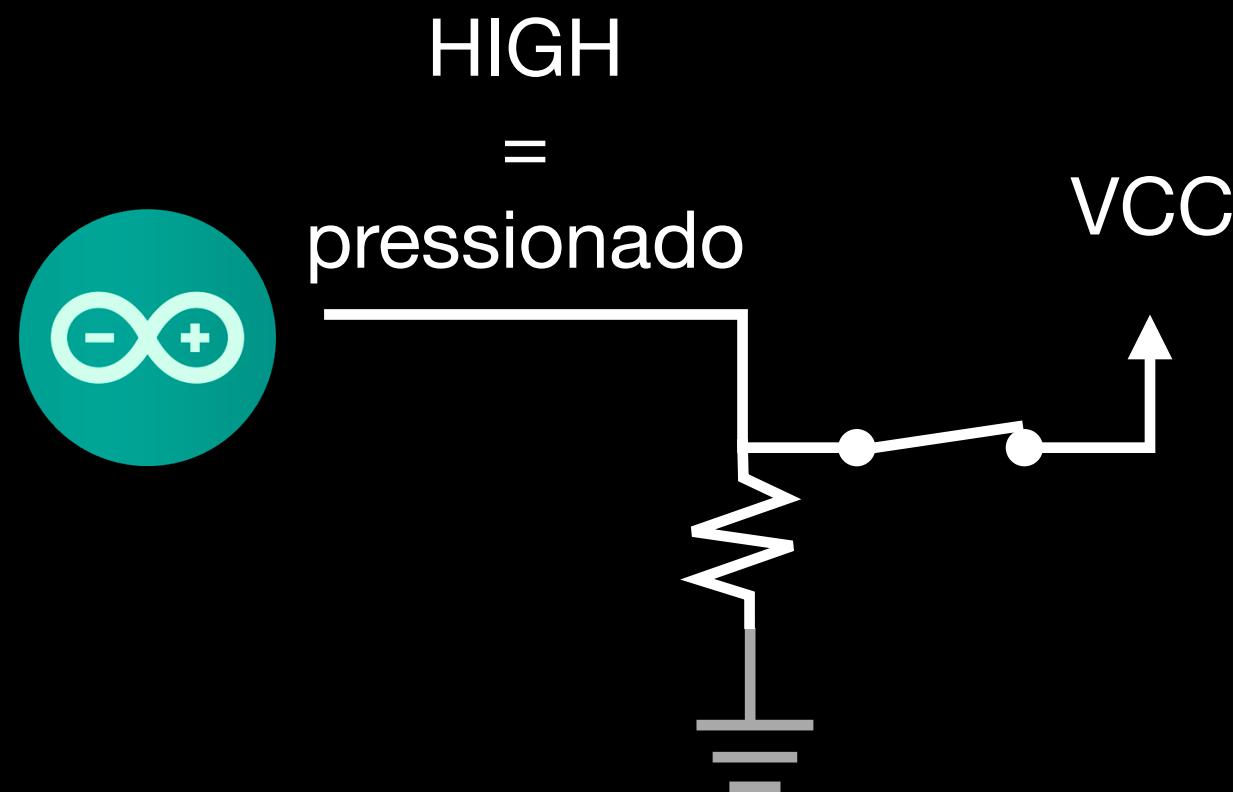
beep



"Piscando" uma Campainha (Como um Alarme)



Botão



Duas Formas de Usar um Botão

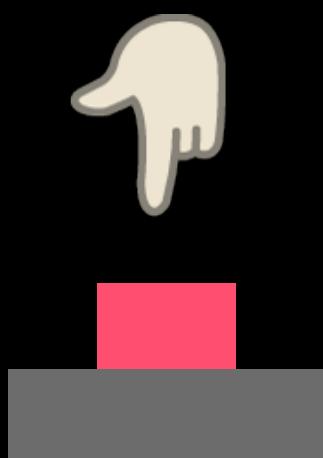
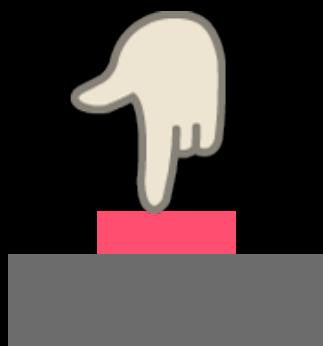
pinos analógicos também podem ser usadas como portas digitais

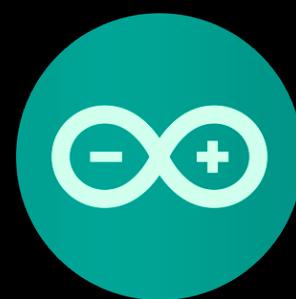
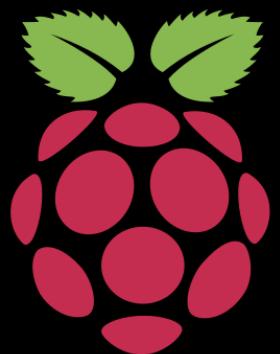
```
int led = 13;  
int botao = A1; // botão 1 está no pino Analógico 1
```



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

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





```
# bibliotecas
from gpiozero import Button

# inicialização de componentes
botao = Button(11)
botao.when_pressed = funcao1
botao.when_released = funcao2
botao.when_held = funcao3
```



?

Eventos de Botão no Arduino?

Atlassian, Inc. bitbucket.org/geekfactory/gfbutton/overview

GFButton Library

Arduino library to easily manage buttons and keys as objects on the arduino sketch. The GFButton class provides methods for polling and event based programming styles.

The objective of this library is to move the button logic outside of the arduino sketch in order to keep the code organized.

Basic library usage

The following example illustrates the library usage. One button is used to turn on the led and other button is used to turn it off. When this example runs, pressing the ON button will print the message to the serial monitor many times, whereas the OFF button will print a message only once. See src/GFButton.h for full member documentation.

```
/**  
 * GeekFactory - "INNOVATING TOGETHER"  
 * Distribucion de materiales para el desarrollo e innovacion  
 * www.geekfactory.mx  
  
 * Basic example for the GFButton library. This example shows how to use  
 * the polling (synchronous) API. This is the easiest way to use the library.  
 */  
  
#include "GFButton.h"  
  
// Create two button instances on pins 2 & 3  
GFButton buttonOn(2);  
GFButton buttonOff(3);
```

Jesus Ruben Santa Anna Zamudio · 2017-11-05

1 commit
Pushed to geekfactory/gfbutton
6967241 README.md edited online with ...

Jesus Ruben Santa Anna Zamudio · 2017-11-05

1 commit
Pushed to geekfactory/gfbutton
1abb33e Change version number

Jesus Ruben Santa Anna Zamudio · 2017-11-05

1 commit
Pushed to geekfactory/gfbutton
949c58f Added license information

Jesus Ruben Santa Anna Zamudio · 2017-11-05

2 commits
Pushed to geekfactory/gfbutton
a045beb Merge branch 'master' of https://bitbucket.org/geekfactory/gfbutton
bc00663 Improve library description

Jesus Ruben Santa Anna Zamudio · 2017-11-05

1 commit
Pushed to geekfactory/gfbutton
f42e116 README.md edited online with ...

```
#include <GFButton.h>

GFButton botao(A1);
void setup () {
    Serial.begin(9600);
    botao.setPressHandler(botaoPressionado);
    botao.setReleaseHandler(botaoSolto);
}

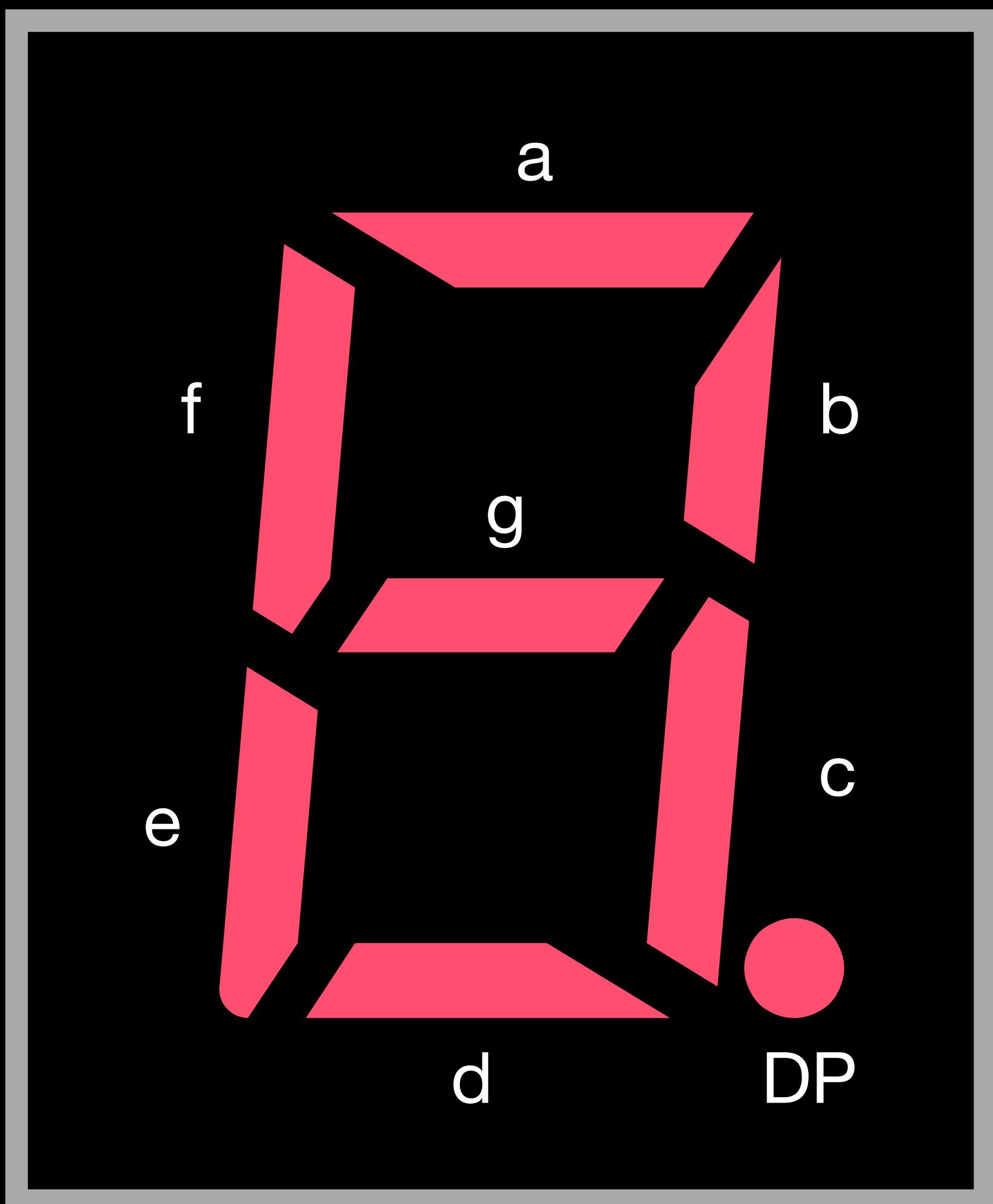
void loop () {
    botao.process(); ← ATENÇÃO! Não esqueça desta chamada!
}

void botaoPressionado (GFButton& botaoDoEvento) {
    Serial.println("Botão foi pressionado!");
}

void botaoSolto (GFButton& botaoDoEvento) {
    Serial.println("Botão foi solto!");
}
```



Displays de 7 Segmentos

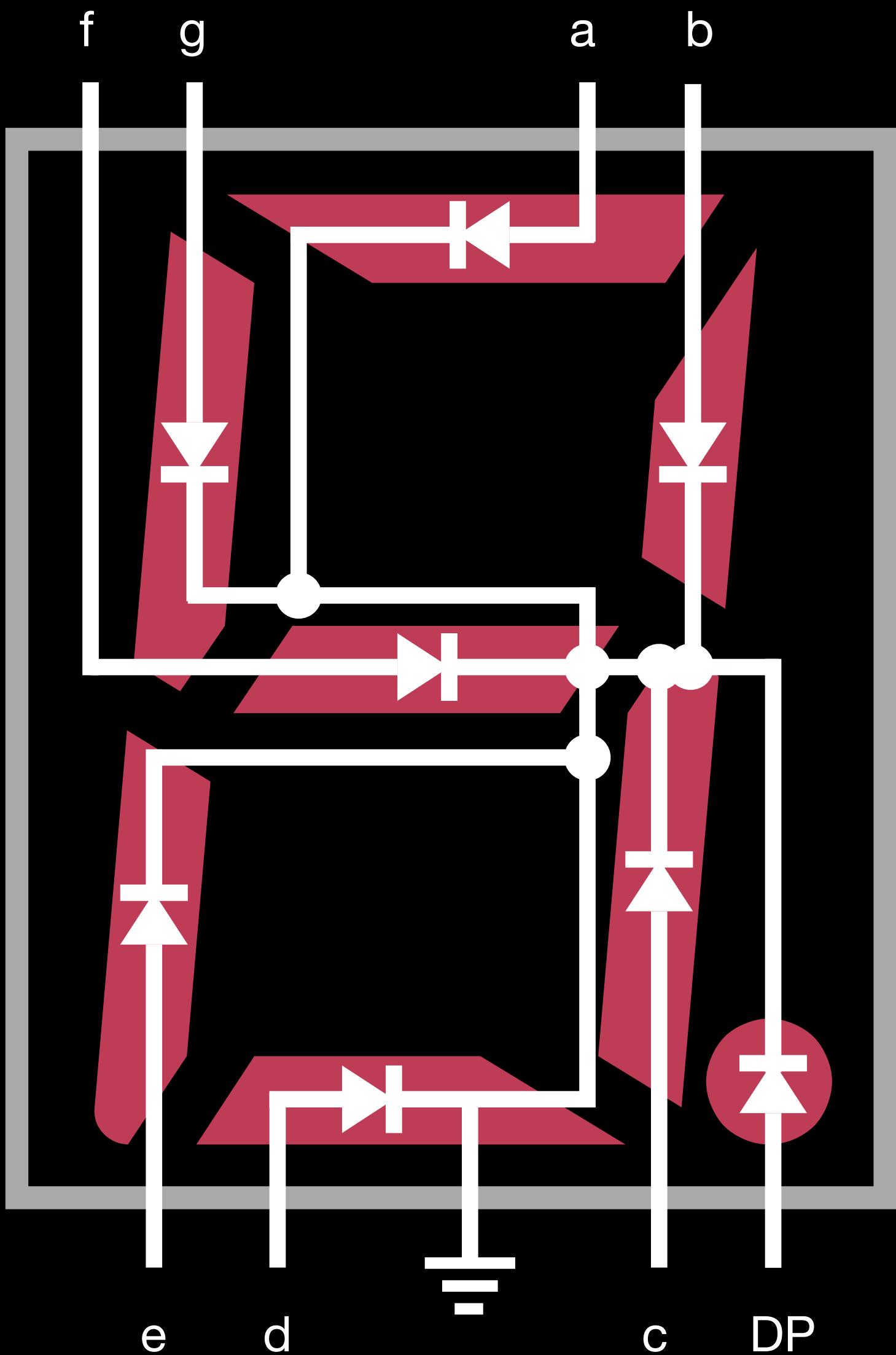


a	b	c	d	e	f	g	DP	
1	1	1	1	1	1	1	1	8.
0	0	0	0	0	0	0	0	9.
1	1	0	1	1	0	1	1	2.
0	1	1	1	1	0	0	0	1.

Elementos de um Display de 7 Segmentos

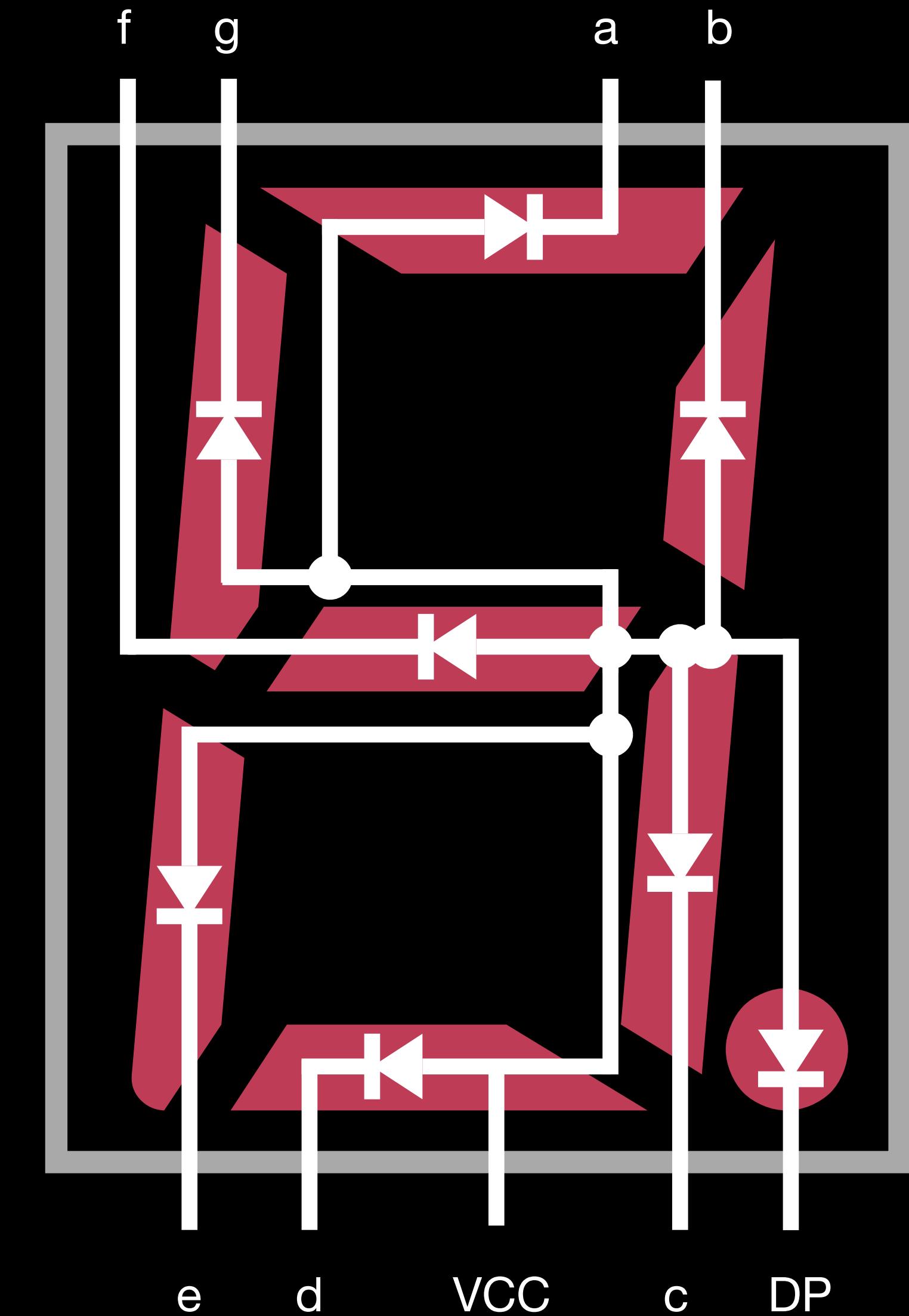
catodo comum

acende segmentos com HIGH (3.3V)

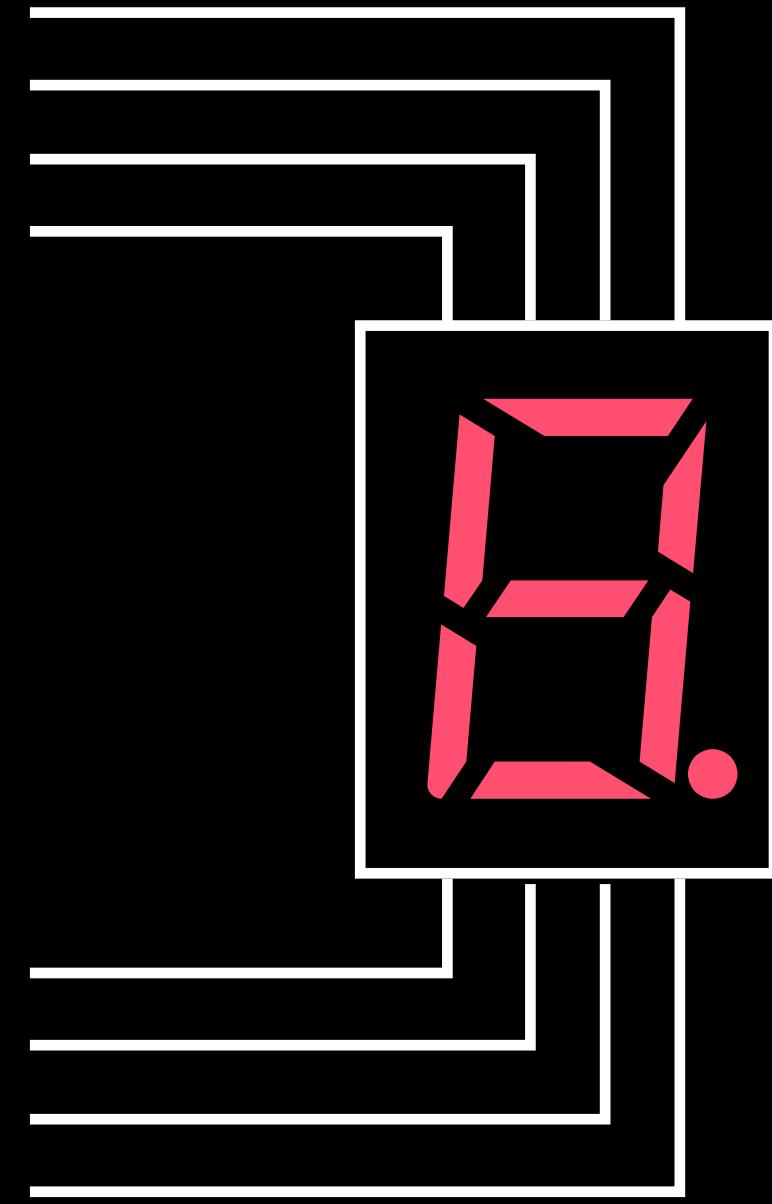
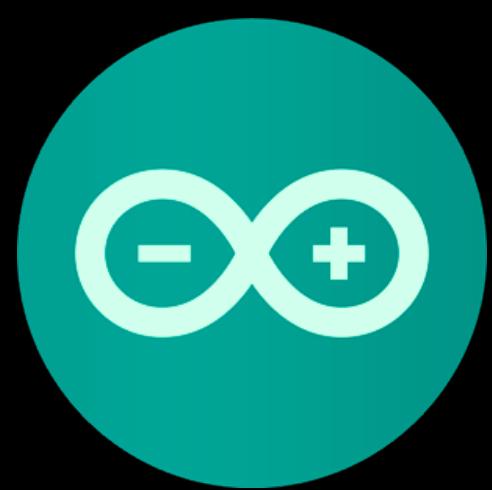


anodo comum

acende segmentos com LOW (0V)

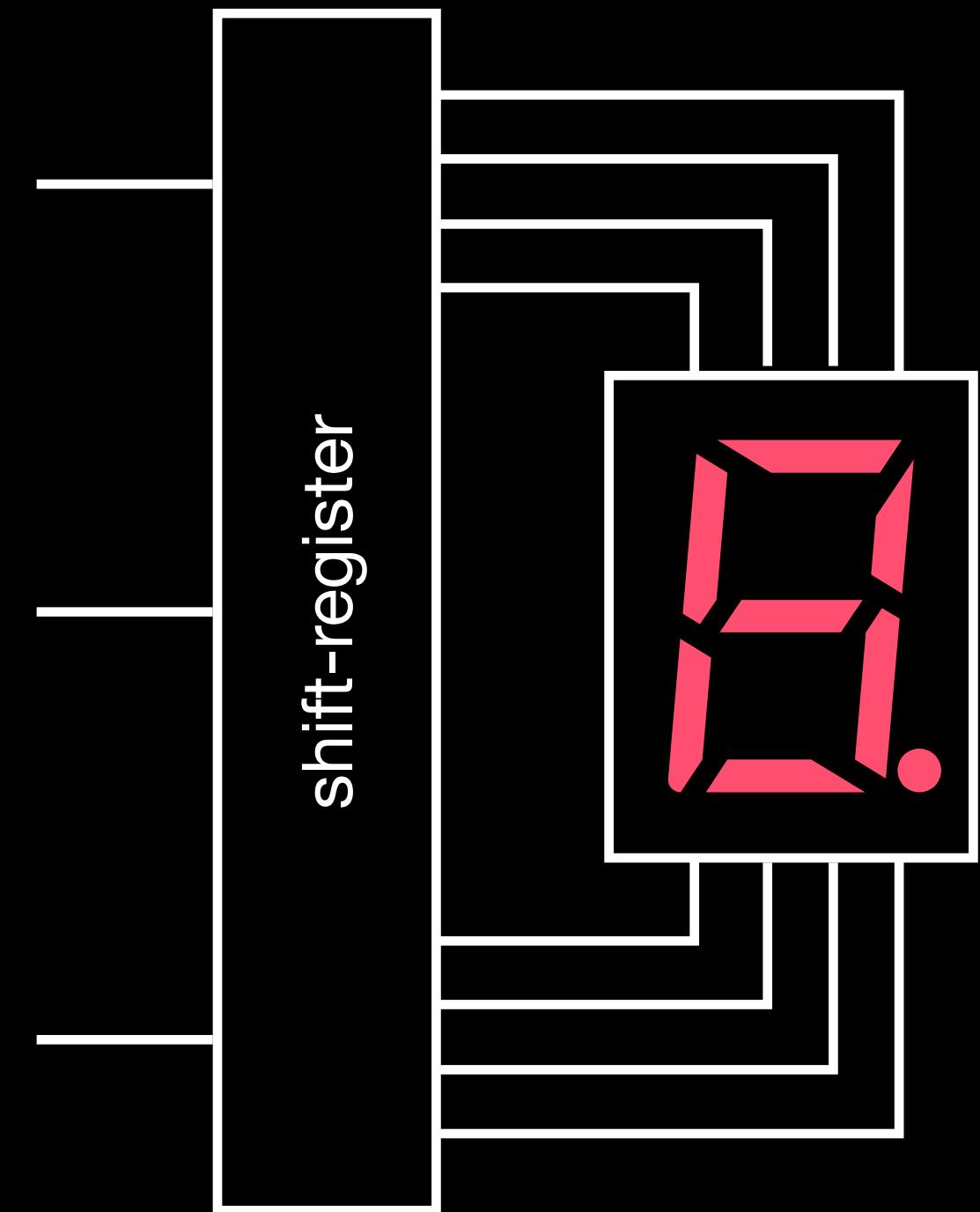
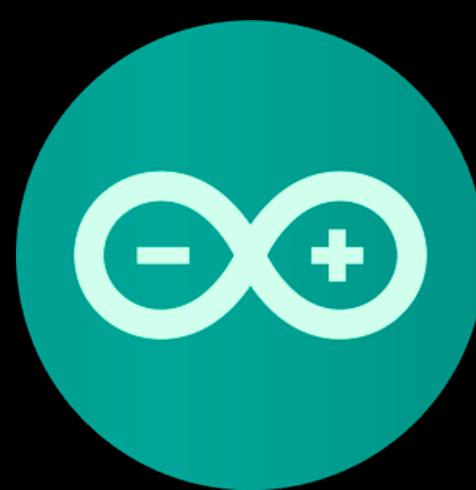


LEDs Dentro de um Display de 7 Segmentos



controle paralelo

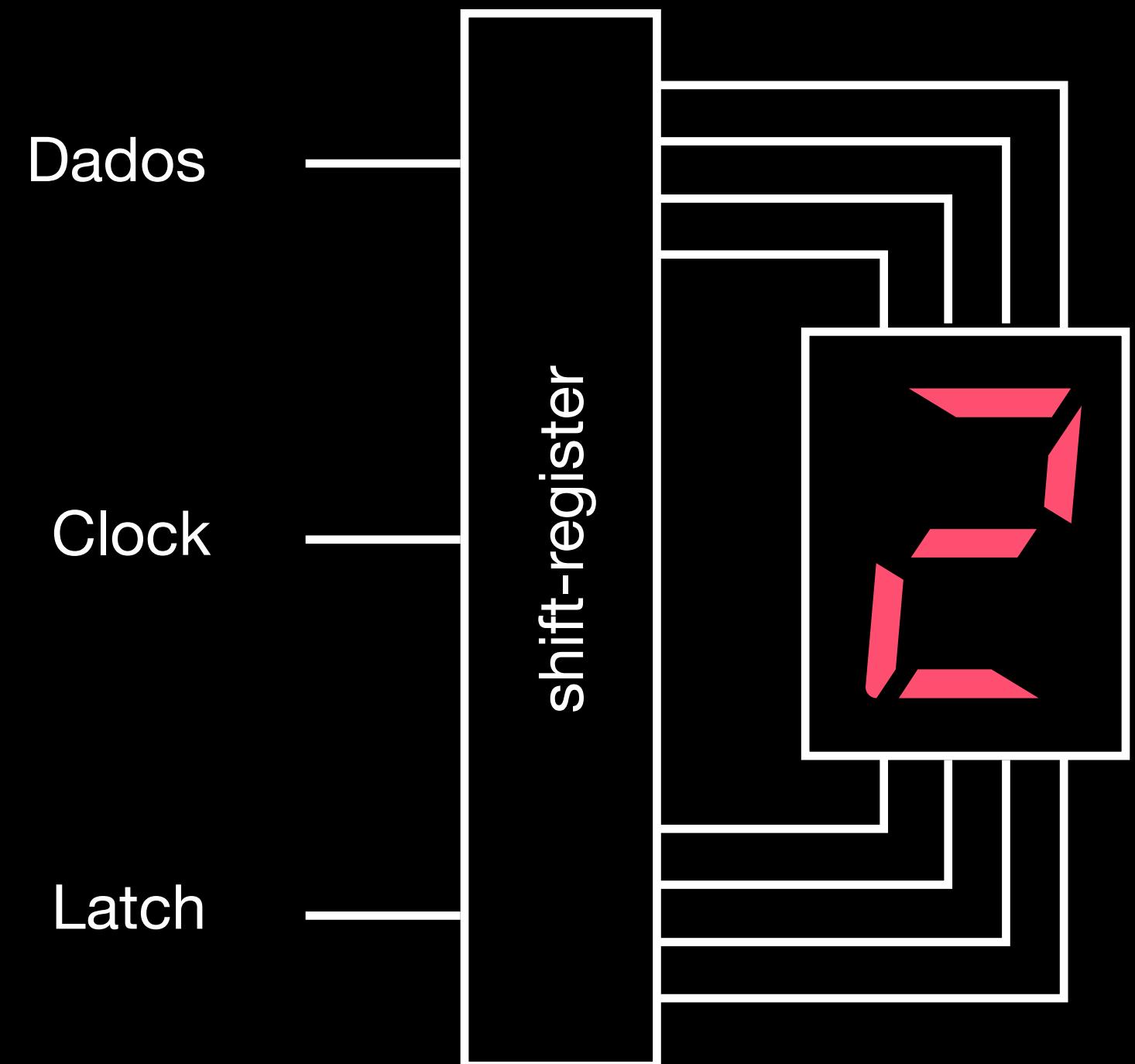
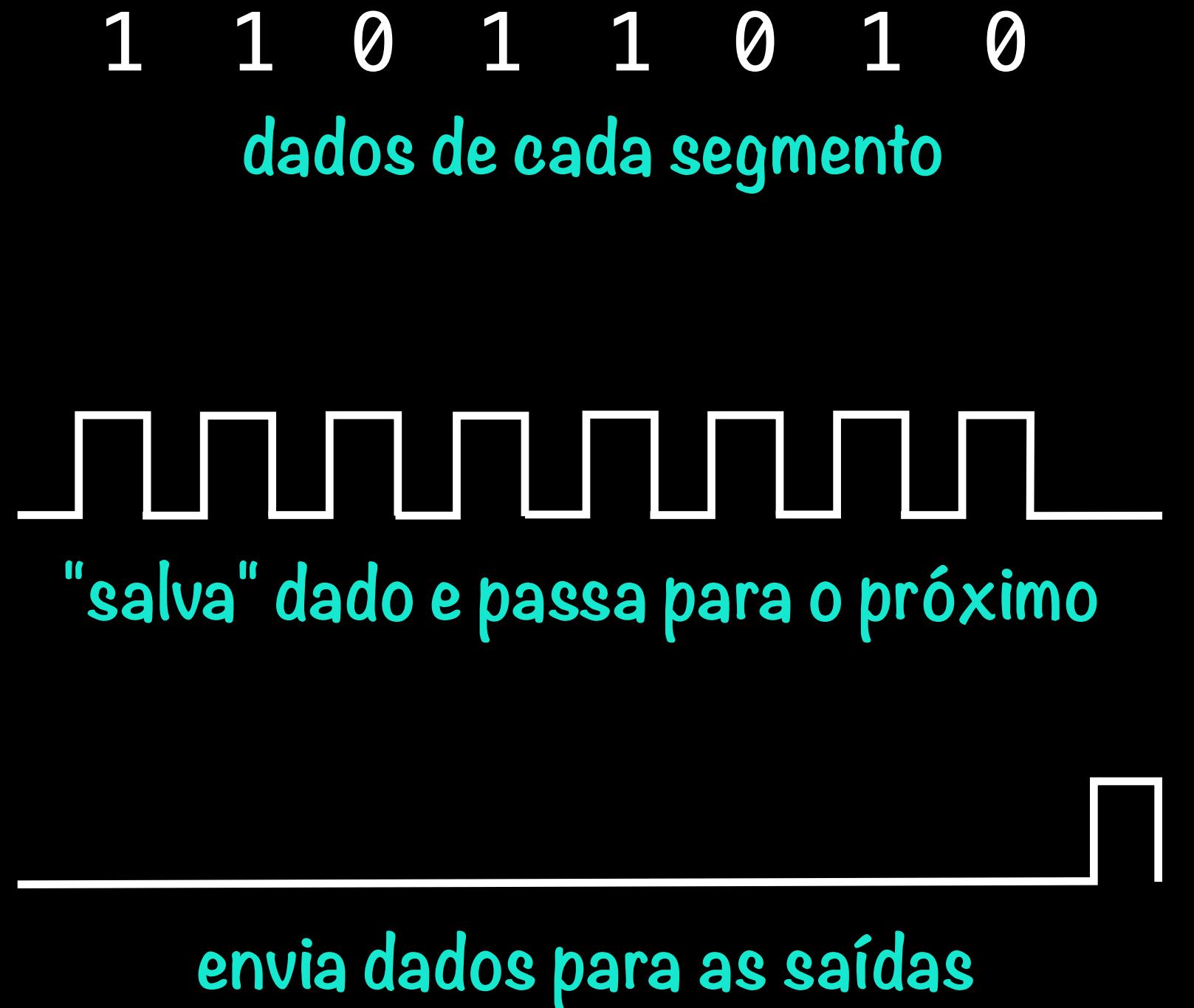
8 conexões + terra/VCC



controle serial

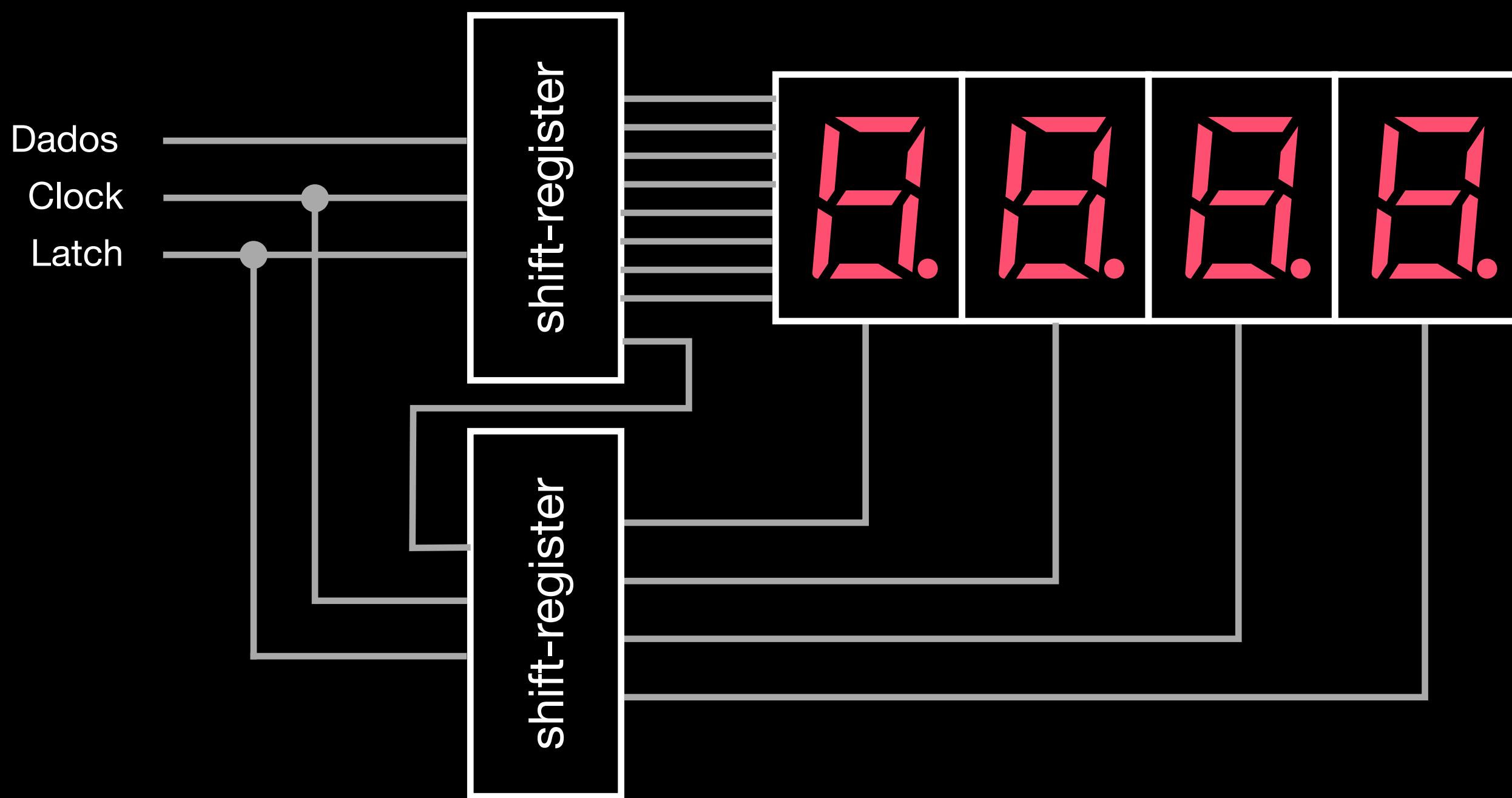
3 conexões + terra/VCC

Controle Paralelo vs Serial



Envio Serial de Dados para o Display

envia os segmentos



escolhe um display de cada vez

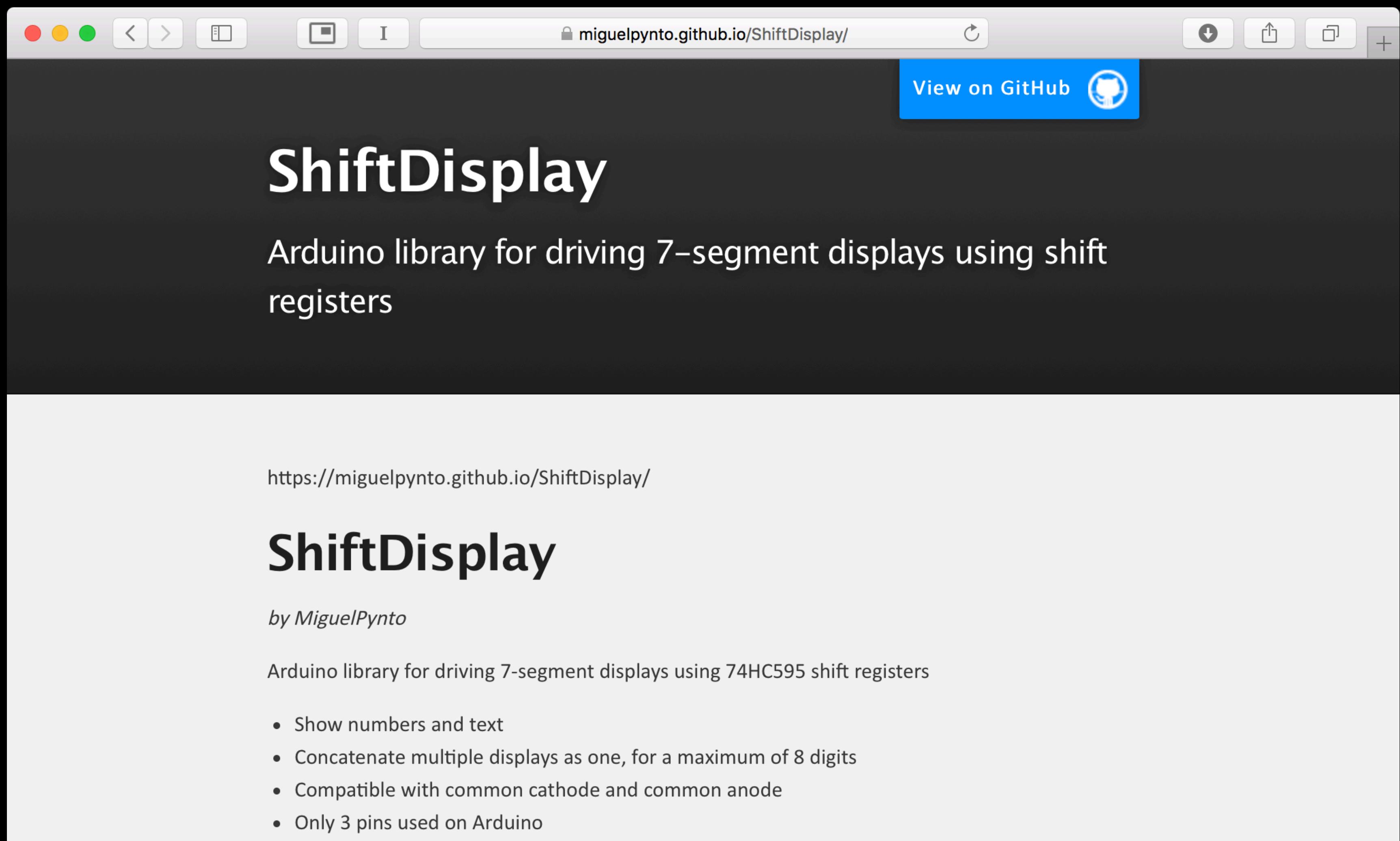
Controle Serial de Vários Displays



≈

7 9 0 6

Alternância Rápida de Displays para Exibir os Dígitos

A screenshot of a web browser window displaying the ShiftDisplay library page. The address bar shows the URL: miguelpynto.github.io/ShiftDisplay/. The browser interface includes standard controls like back, forward, and search, along with GitHub integration buttons for viewing the repository on GitHub. The main content features a large, bold title "ShiftDisplay" followed by a subtitle: "Arduino library for driving 7-segment displays using shift registers". Below this, there is a link to the GitHub repository: <https://miguelpynto.github.io/ShiftDisplay/>.

ShiftDisplay

Arduino library for driving 7-segment displays using shift registers

<https://miguelpynto.github.io/ShiftDisplay/>

ShiftDisplay

by MiguelPynto

Arduino library for driving 7-segment displays using 74HC595 shift registers

- Show numbers and text
- Concatenate multiple displays as one, for a maximum of 8 digits
- Compatible with common cathode and common anode
- Only 3 pins used on Arduino

```
ShiftDisplay display(4, 7, 8,  
                    COMMON_ANODE  
                    ou  
                    COMMON_CATHODE  
                    , 4, true);
```

pino do latch → pino do dados ↓
pino do clock → tipo de display de 7 segmentos ↑
quantidade de displays ↓

envie os bits do display antes
de selecionar o display

Inicialização do ShiftDisplay

```
// apenas salva o que será exibido  
display.set(1234);
```

8888

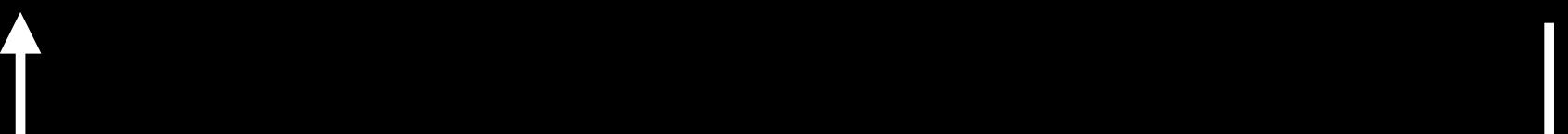
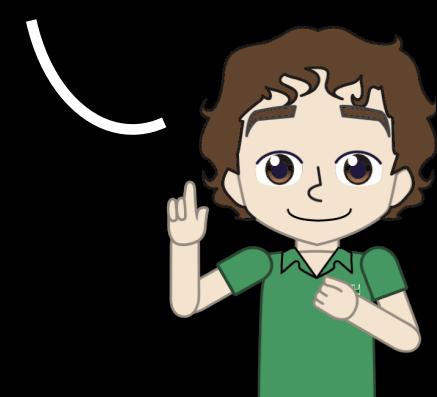
```
// alterna uma vez os dígitos  
display.update();
```

1888 → 8288 → 8838 → 8884

```
// alterna dígitos durante 1000 ms  
display.show(1000);
```

1888 → 8288 → 8838 → 8884

ATENÇÃO: a função show trava
a execução do programa!



```
#include <ShiftDisplay.h>

ShiftDisplay display(4, 7, 8, COMMON_ANODE, 4, true);

void setup () {
    display.set(1234);
```

```
}

void loop () {
    display.update();
```

8888

1888 → 8288 → 8838 → 8884

≈

1234

```
#include <ShiftDisplay.h>

ShiftDisplay display(4, 7, 8, COMMON_ANODE, 4, true);

void setup () {
    display.set(1234);
    display.show(1000);                                1234
}
                                         (exibe 1 vez durante 1000 ms)

void loop () {
    display.set(4321);
    display.update();                                 4321
}

```

```
display.set(4321);
```

4321

```
display.changeDot(0, true); // ponto no índice 0
```

4.321

```
display.set(4.56);
```

0046

```
display.set(4.56, 2); // 2 casas decimais
```

0456

```
display.set(4.56, 2, true); // zeros à esquerda
```

0456

```
display.set("ERRO");
```

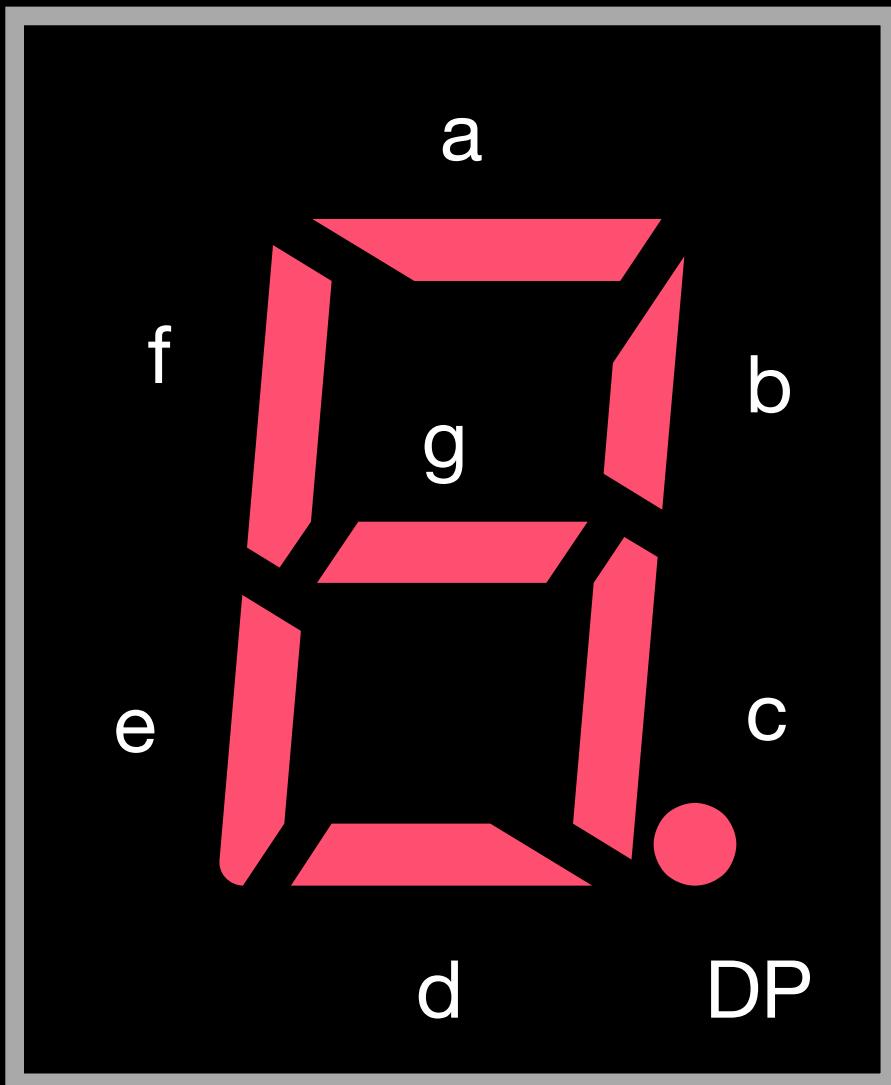
Erro

```
display.set("oi");
```

0000

```
display.set("oi", ALIGN_CENTER);
```

0000



a	b	c	d	e	f	g	DP	
1	1	1	1	1	1	1	1	8.
0	0	0	0	0	0	0	0	8
1	1	0	1	1	0	1	1	2.
0	1	1	1	1	0	0	0	2

```
byte x[] = {0b11111111, 0b00000000, 0b01101111, 0b01111000};  
display.set(x);  
...
```

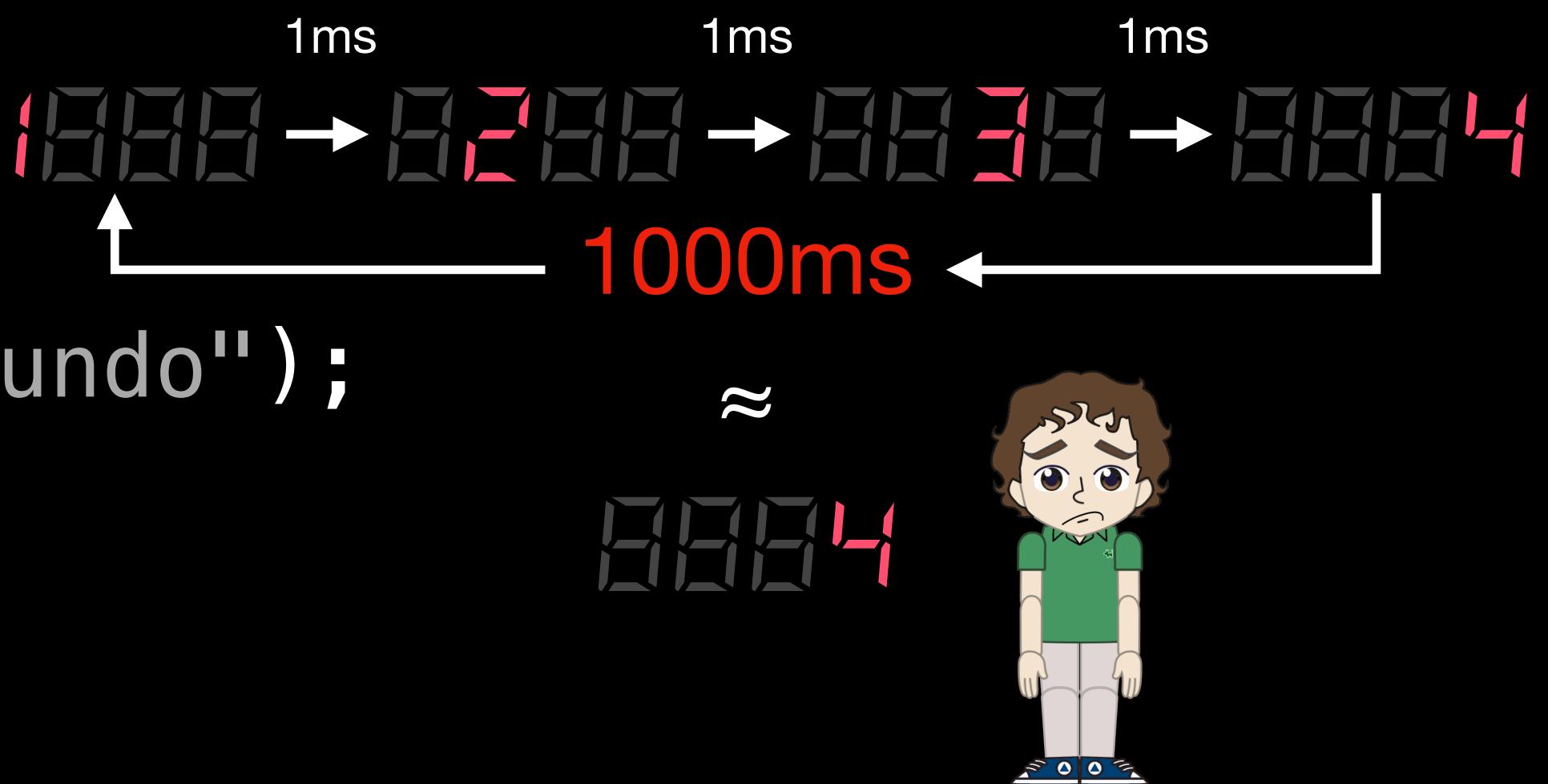
8.82.0

"Software"

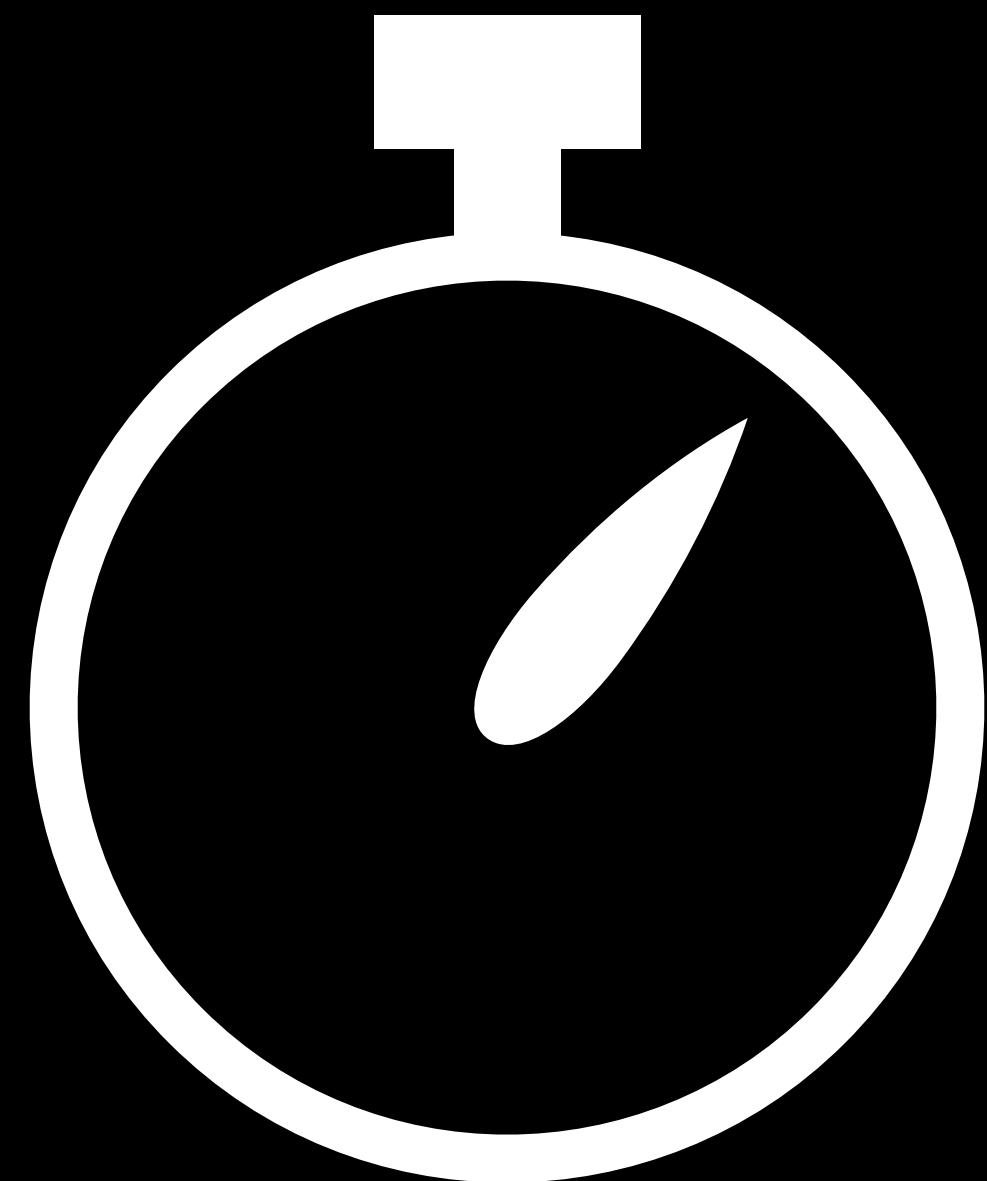
```
#include <ShiftDisplay.h>
```

```
ShiftDisplay display(4, 7, 8, COMMON_ANODE, 4, true);
void setup () {
    Serial.begin(9600);
    display.set(1234);
}

void loop () {
    display.update();
    delay(1000);
    Serial.println("+1 segundo");
}
```



Exemplo de Problema com a Função Delay



Timer Recorrente

Início do Programa



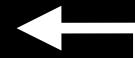
```
void setup () {  
    ...  
}
```

Inicia Timer Recorrente



```
void loop () {  
    ...  
}
```

```
void loopDoTimer () {  
    ...  
}
```



período do timer

Loop do Timer "Paralelo" ao Loop do Programa

The screenshot shows a web browser window with the URL www.pjrc.com/teensy/td_libs_TimerOne.html. The page content is as follows:

Basic Usage

The timer is configured to repetitively measure a period of time, in microseconds. At the end of each period, an interrupt function can be run. The PWM pins can also be configured to switch during a portion of the period.

If using TimerThree, replace "Timer1" with "Timer3".

Configuration

Run Control

Code Examples and Descriptions

- [LeaControl](#)
- [DogLcd](#)
- [ST7565](#)
- [AltSoftSerial](#)
- [NewSoftSerial](#)
- [SoftwareSerial](#)
- [MIDI](#)
- [PS2Keyboard](#)
- [DmxSimple](#)
- [Firmata](#)
- [Wire](#)
- [SPI](#)
- [OneWire](#)
- [XBee](#)
- [VirtualWire](#)
- [X10](#)
- [IRremote](#)
- [TinyGPS](#)
- [USBHostShield](#)
- [Ethernet](#)
- [Bounce](#)
- [Keypad](#)
- [Audio](#)
- [Encoder](#)
- [Ping](#)
- [CapacitiveSensor](#)
- [FreqCount](#)
- [FreqMeasure](#)
- [Servo](#)
- [PulsePosition](#)
- [Stepper](#)
- [AccelStepper](#)
- [FrequencyTimer2](#)
- [Tlc5940](#)
- [SoftPWM](#)
- [ShiftPWM](#)
- [Time](#)
- [TimeAlarms](#)

```

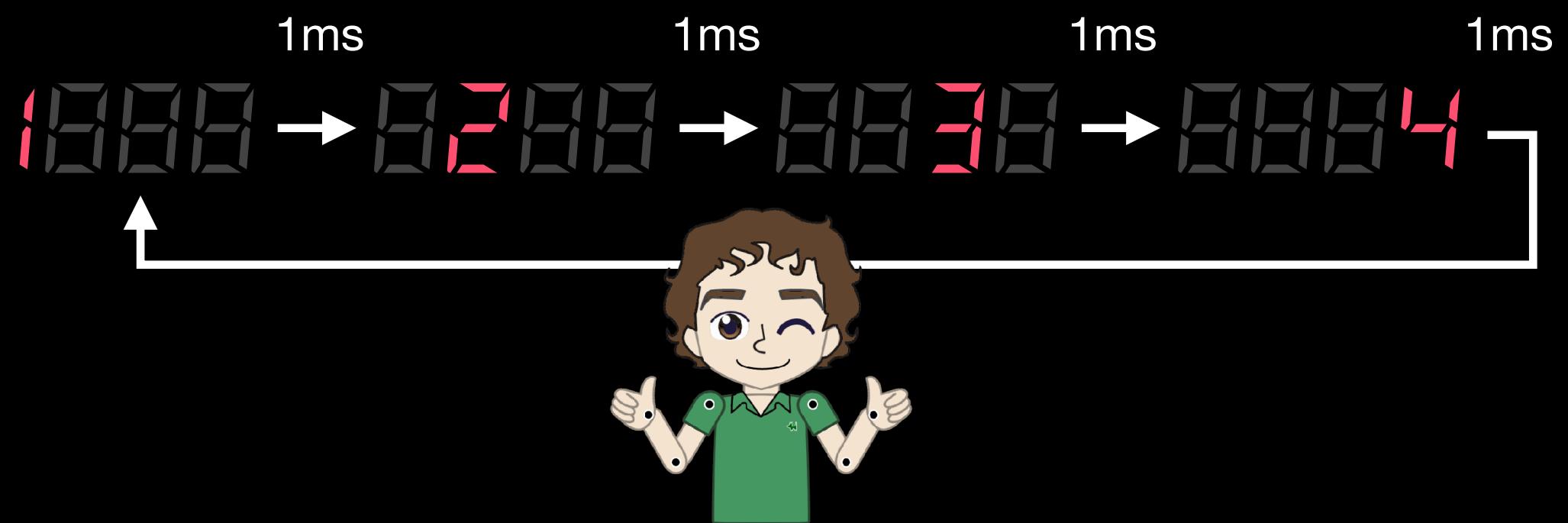
#include <ShiftDisplay.h>
#include <TimerOne.h>

ShiftDisplay display(4, 7, 8, COMMON_ANODE, 4, true);
void setup () {
    Serial.begin(9600);
    display.set(1234);
    Timer1.initialize(1000000); ← tempo em MICROsegundos
    Timer1.attachInterrupt(loopDoTimer);
}

void loop () {
    display.update();
}

void loopDoTimer () {
    Serial.println("+1 segundo");
}

```



Note

Inside the attached function, delay() won't work

```
void setup() {  
    Timer1.initialize(1000000);  
    Timer1.attachInterrupt(loopDoTimer);  
}
```

```
void loopDoTimer () {  
    Serial.println("Legen... wait for it...");  
    delay(1000);  
    Serial.println("dary!");  
}
```

← NÃO FUNCIONA!

Resumo da Ópera

Funcionalidad

Revisão de C++

```
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;  
}
```

Print Serial

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

Escrita/Leitura documentação

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

GButton documentação

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

ShiftDisplay documentação

```
#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); display.changeDot(0, true)
```

Timer1 documentação

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