

Computação Física:

primeiros passos com ESP8266

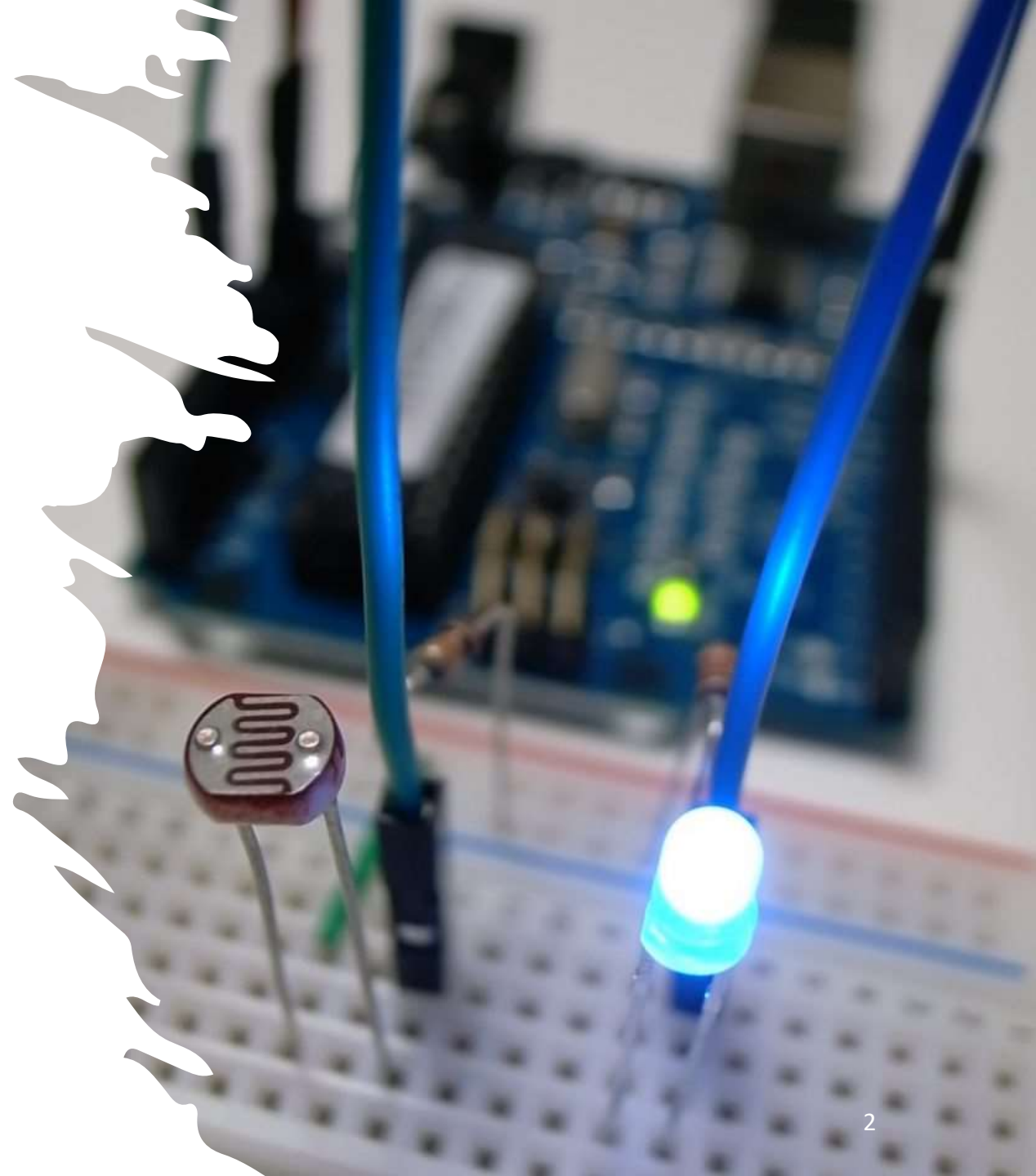
Prof. Me. Peter Jandl Junior
ADS | DC | GTI | SE



ARDUINO
DAY 2023

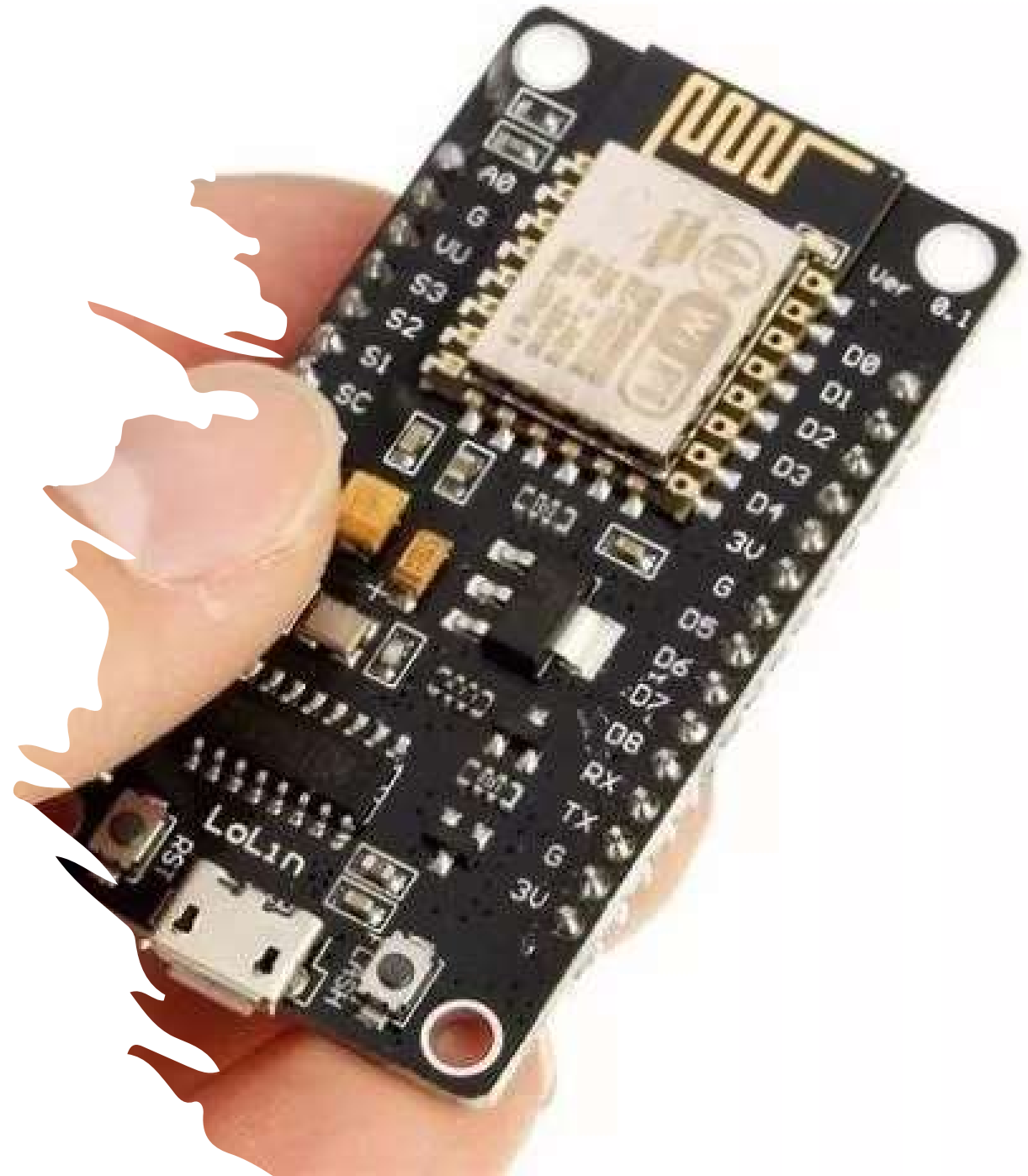
Computação física

- A computação física (CF) ou *physical computing* pode ser entendida como os sistemas interativos capazes de monitorar e atuar no ambiente circundante.
- A computação física refere-se ao uso de sistemas interativos baseados em microcontroladores incorporados tangíveis que podem sentir o mundo ao seu redor e/ou controlar saídas como luzes, telas e motores.



Computação física

- Computação Física é o uso combinado da computação (programação) e da eletrônica (sensores e atuadores) na prototipação de objetos físicos usados interativamente por seres humanos, cujo objetivo é interligar os mundos físico e virtual, assim demonstrar o uso da computação e a interação com a tecnologia para realização de suas atividades rotineiras.



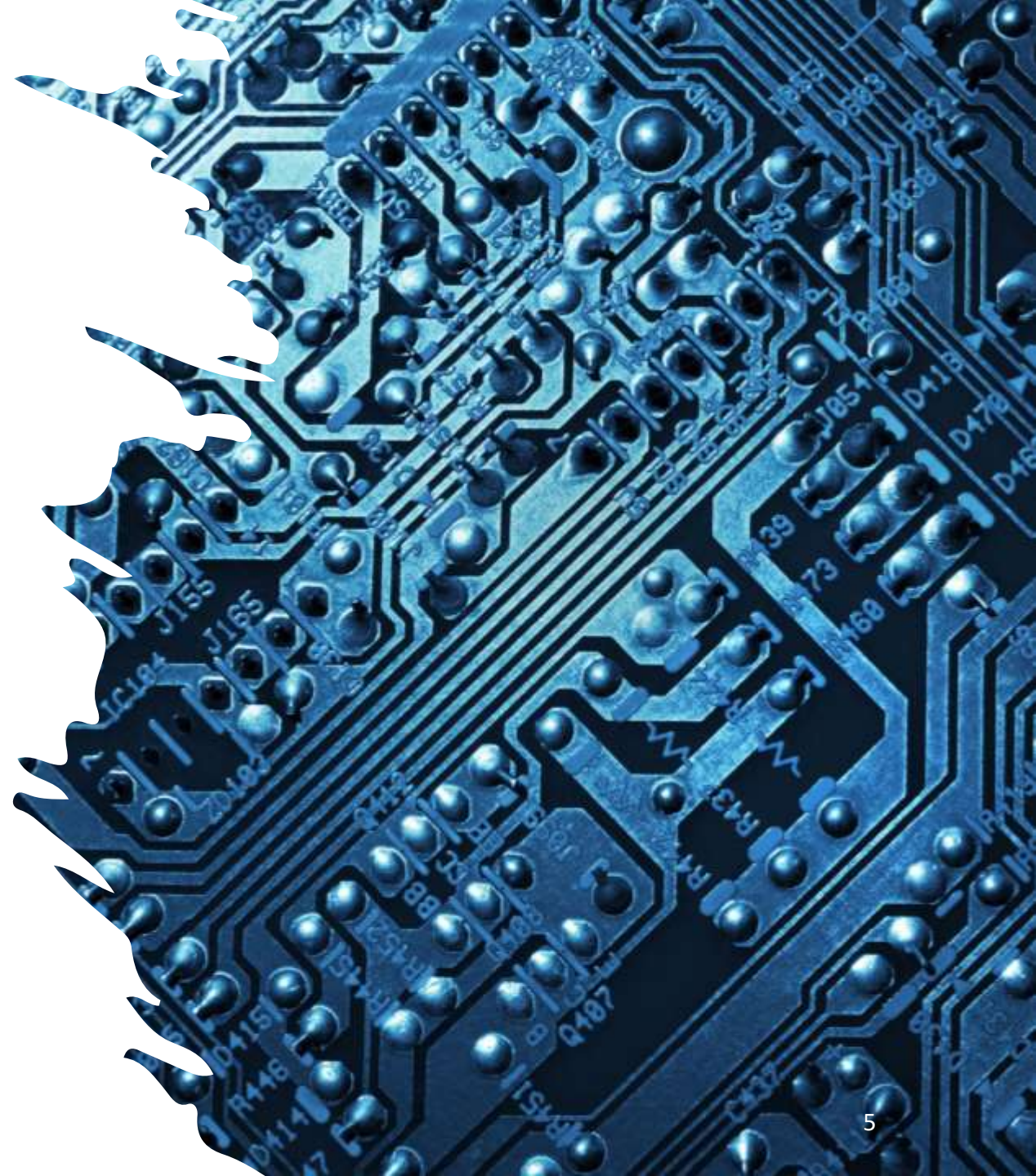
Microcontroladores

- **Microcontrolador** é geralmente um único circuito integrado que contém:
 - Um processador
 - Memória (RAM, FLASH e ROM)
 - Periféricos (E/S) programáveis
- São mais simples, de menor poder computacional, muito baratos e robustos.



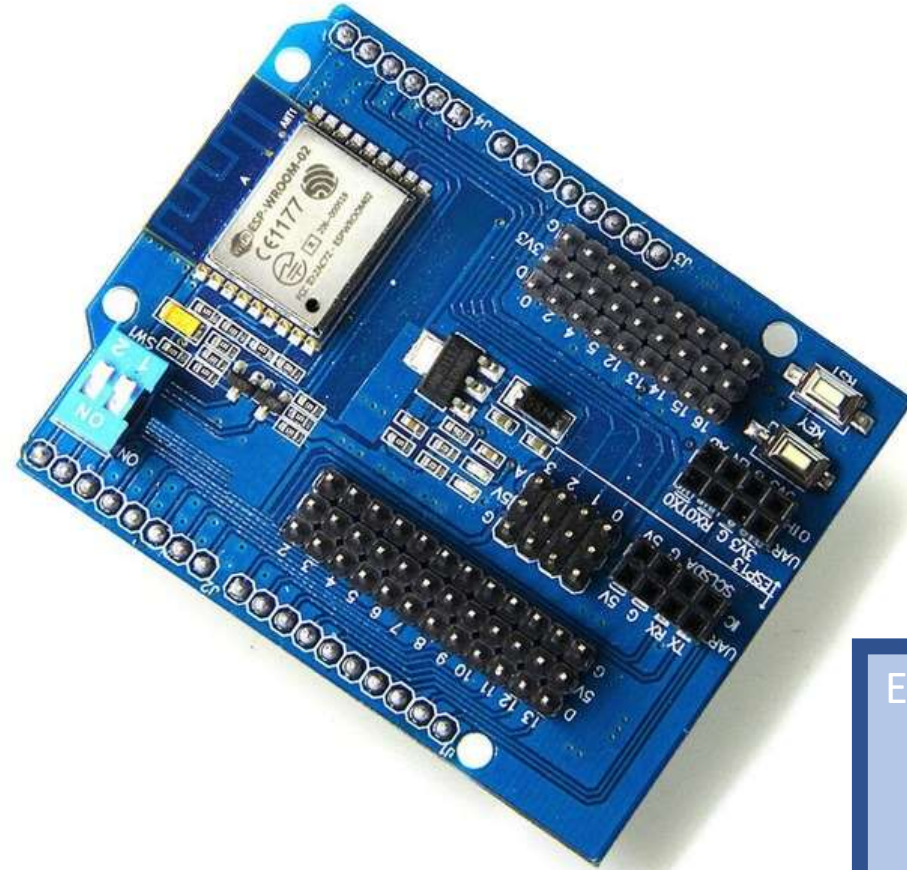
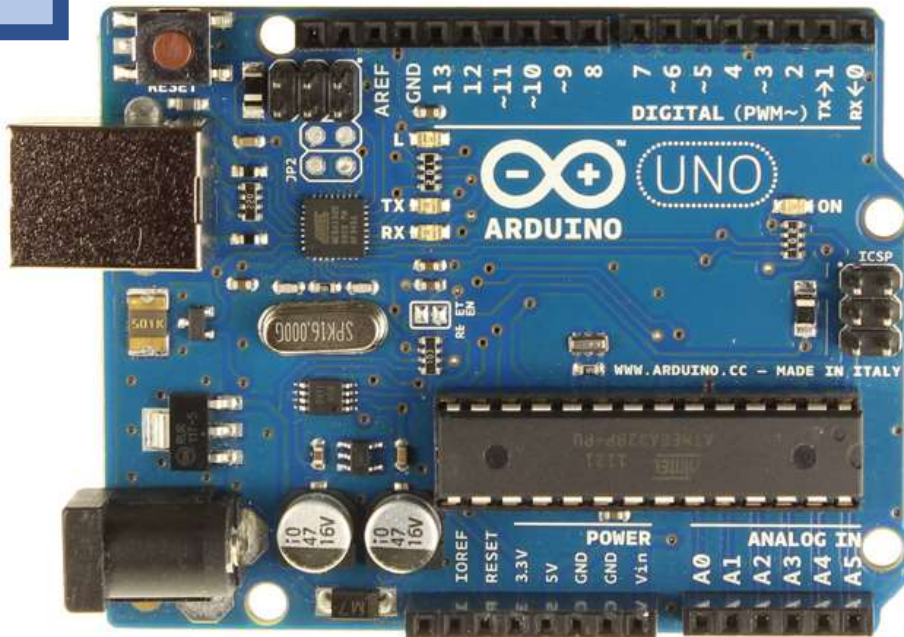
Placas de desenvolvimento

- Com a evolução das tecnologias de fabricação dos circuitos integrados, além de sua popularização nos microcomputadores e na telefonia celular, percebeu-se que sistemas microprocessados, isto é, dotados de microprocessadores ou microcontroladores são muito mais versáteis do que os dispositivos eletrônicos tradicionais, essencialmente por serem programáveis.
- Isto motivou o desenvolvimento de placas de desenvolvimento para facilitar o estudo destas tecnologias, assim como a construção de protótipos de sistemas mais simples.



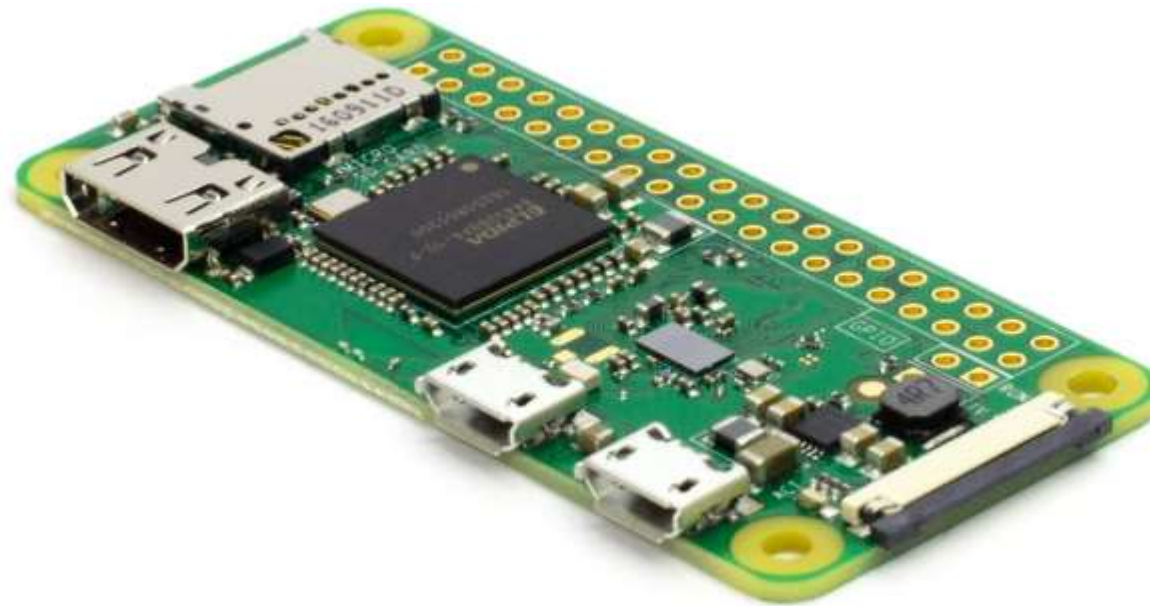
Placas de desenvolvimento: Arduino (uno, nano, mega, leonardo, ...)

Diversos
fabricantes
Originalmente
US\$ 10.00



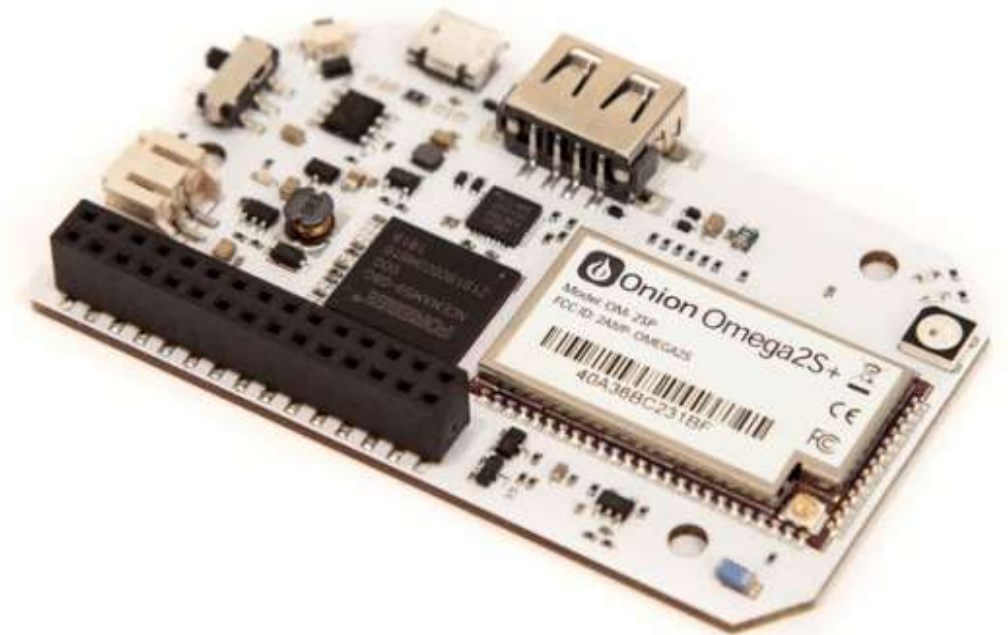
Existem muitos
shields
(módulos de
extensão)

Placas de desenvolvimento::Raspberry Pi

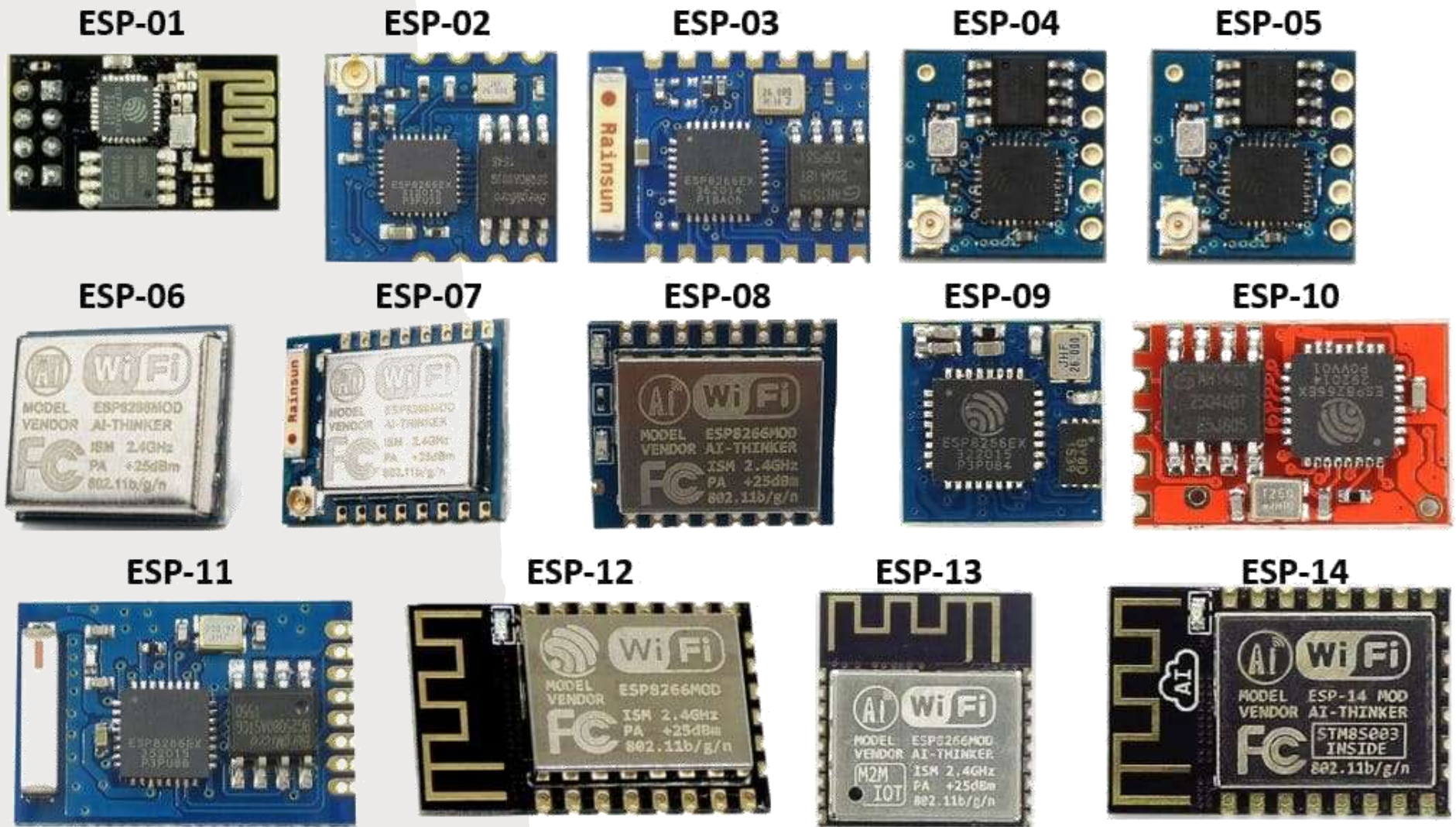


Computador de
US\$ 35,00,
tamanho de cartão
de crédito

Placas de desenvolvimento: Onion Omega2

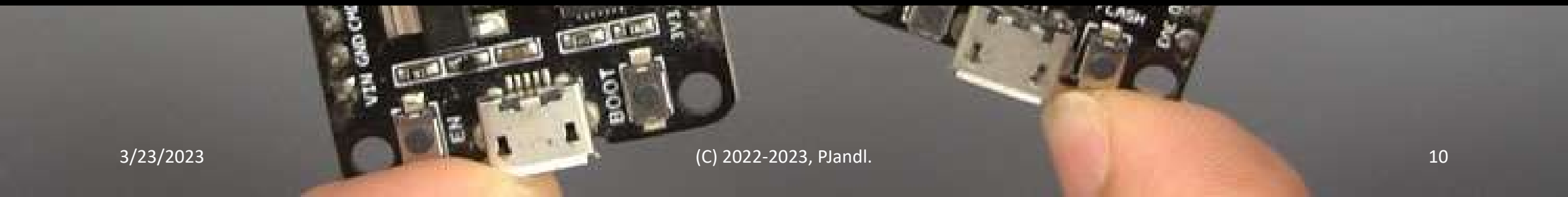


Placas de desenvolvimento ::família ESP





Placas de desenvolvimento::NodeMCU



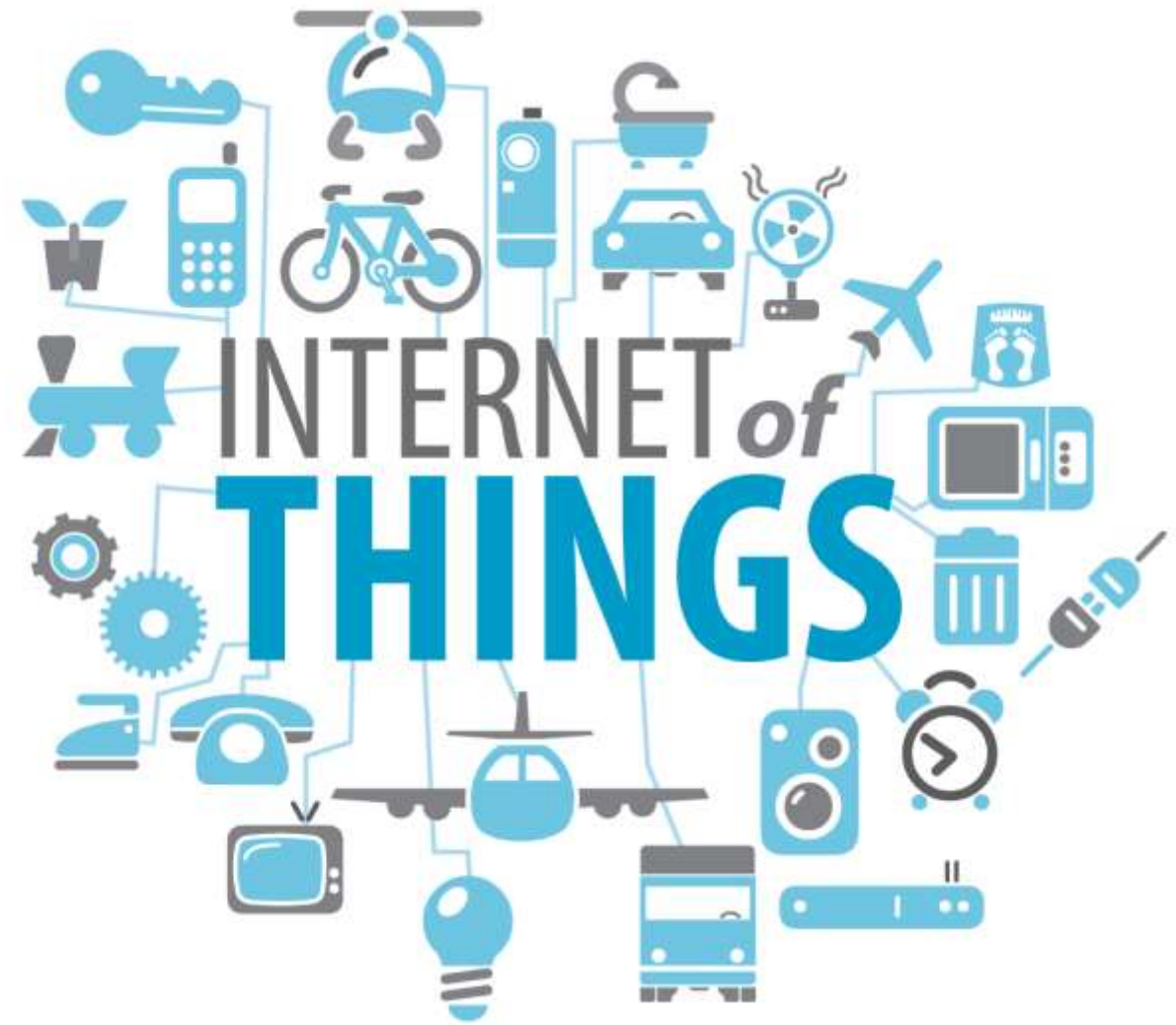
Placas de desenvolvimento:: existem muitas outras

- STM32 (blue pill, black pill)
- Raspberry Pi (3, 4, Pico etc)
- Banana Pi
- Adafruit Playground Express
- Teensy
- Digispark
- Micro:bit
- Wemos



IoT::Internet das Coisas

- A evolução e barateamento dos **microcontroladores** em associação com a popularização das redes de comunicação de dados são dos elementos que tornaram possível a Internet das Coisas.

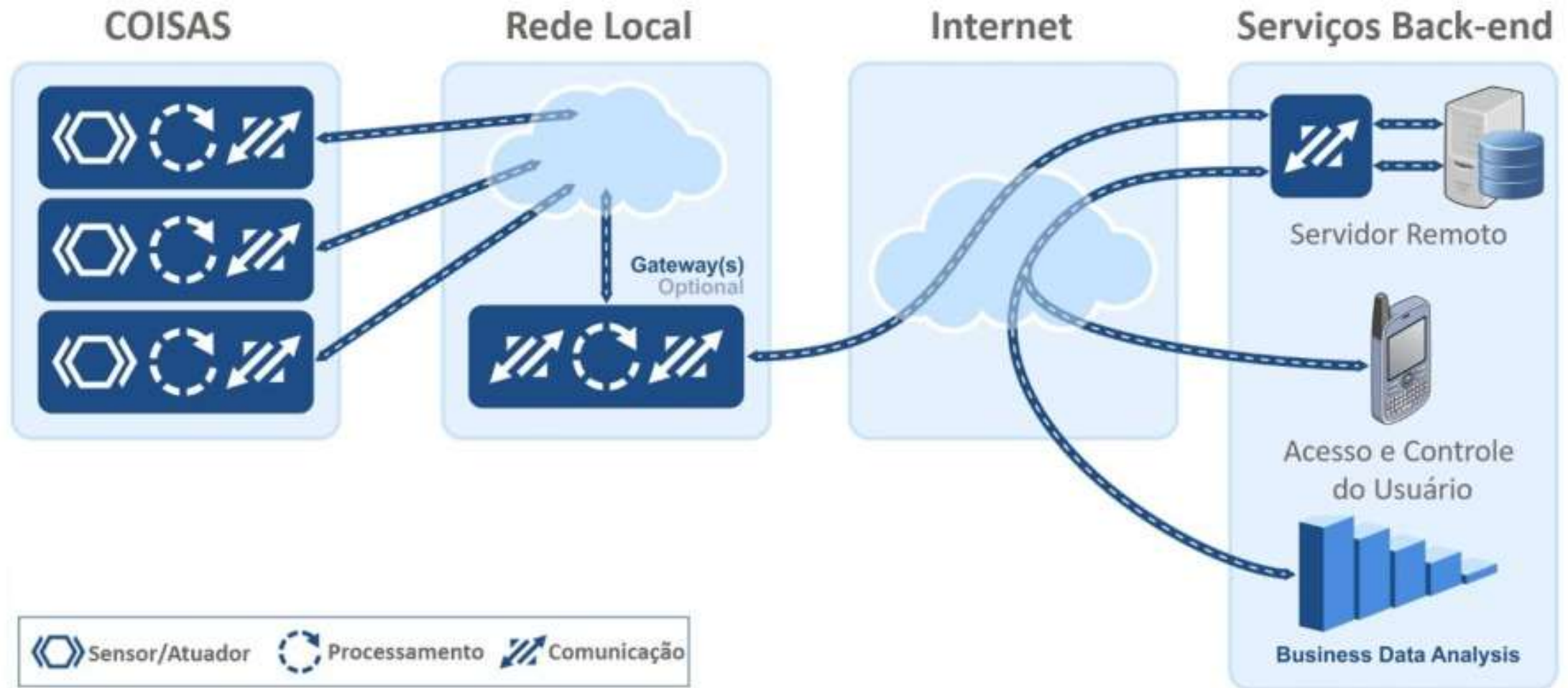


IoT::Internet das Coisas

Revolução tecnológica que objetiva conectar os itens usados do dia a dia à rede mundial de computadores.



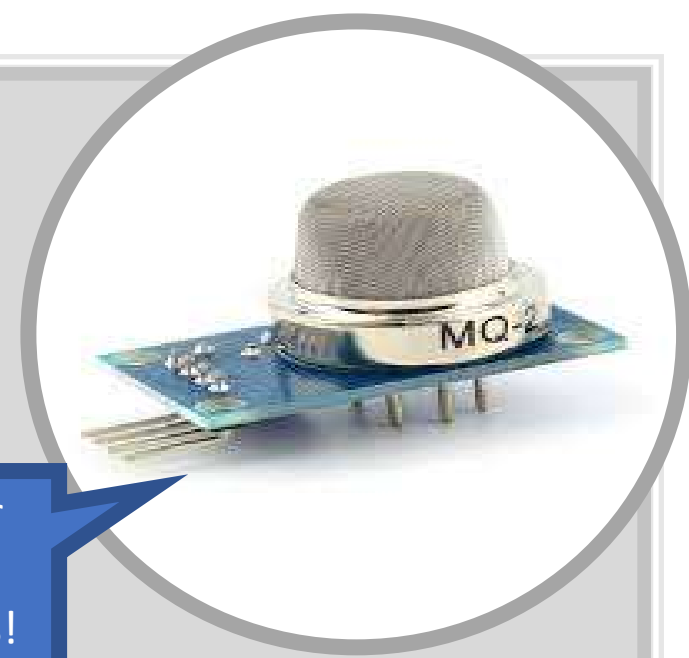
IoT::Internet das Coisas::como funciona?



Sensores e Atuadores

Transdutores e Sensores

- Transdutores que convertem uma forma de energia em outra.
- São comuns aqueles que produzem sinais elétricos.
- Exemplos: termistor, LDR, foto-diodo, antena, etc.



Pode conter um ou mais transdutores!

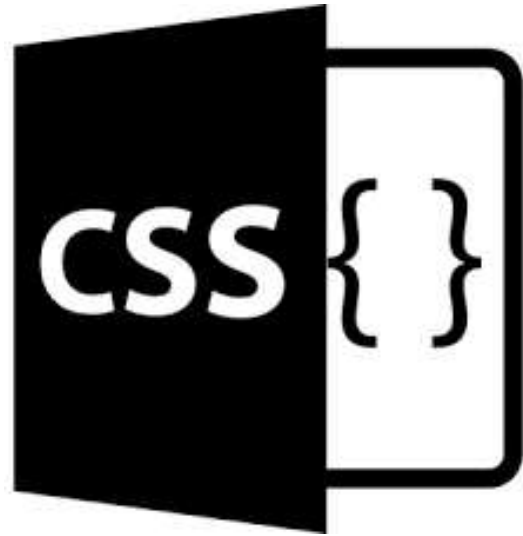
- Sensores detectam eventos ou mudanças ambientais e enviam dados (digitais) para outros sistemas.
- Exemplos: sensores de temperatura, cor, pressão, umidade, etc.

Atuadores

- Servem para controlar uma variável física, como:
 - uma válvula que regula o fluxo de água (ou outro líquido);
 - um motor com ajuste de velocidade ou posição;
 - uma resistência para aquecer um líquido ou câmara;
 - lâmpadas para iluminação de espaços.

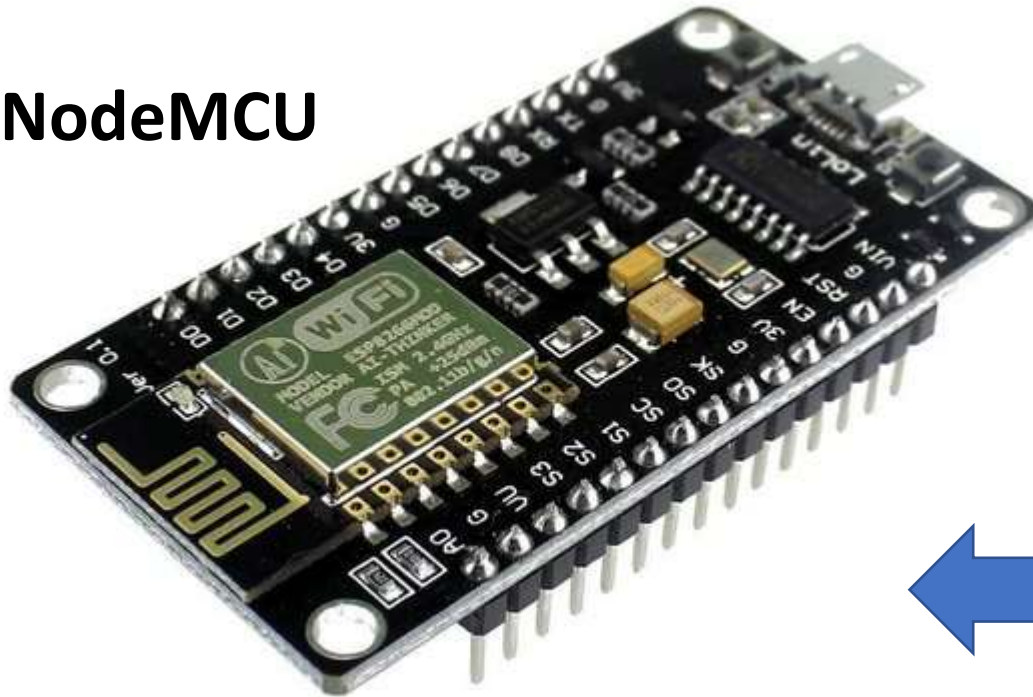


Linguagens de Programação



IoT::exemplo de aplicação

NodeMCU



DHT11



LDR

IoT::exemplo de aplicação



```
ESP8266_DTH11_LDR_ThingSpeak | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

ESP8266_DTH11_LDR_ThingSpeak

133 void envia_informacoes_thingspeak(String string_dados) {
134   if (client.connect(endereco_api_thingspeak, 80)) {
135     /* faz a requisição HTTP ao ThingSpeak */
136     client.print("POST /update HTTP/1.1\n");
137     client.print("Host: api.thingspeak.com\n");
138     client.print("Connection: close\n");
139     client.print("X-THINGSPEAKAPIKEY: " + chave_escrita_thingspeak + "\n");
140     client.print("Content-Type: application/x-www-form-urlencoded\n");
141     client.print("Content-Length: ");
142     client.print(string_dados.length());
143     client.print("\n\n");
144     client.print(string_dados);
145     last_connection_time = millis();
146     datacount++;
147     Serial.println("[ThingSpeak] Informação enviada: " + string_dados + " [" +
148                   String(datacount) + "]\n");
149   }
150 }
151
152 /*
153   ESP32 board setup
154 */
155 void setup() {
156   Serial.begin(115200);
157   /* LDR */

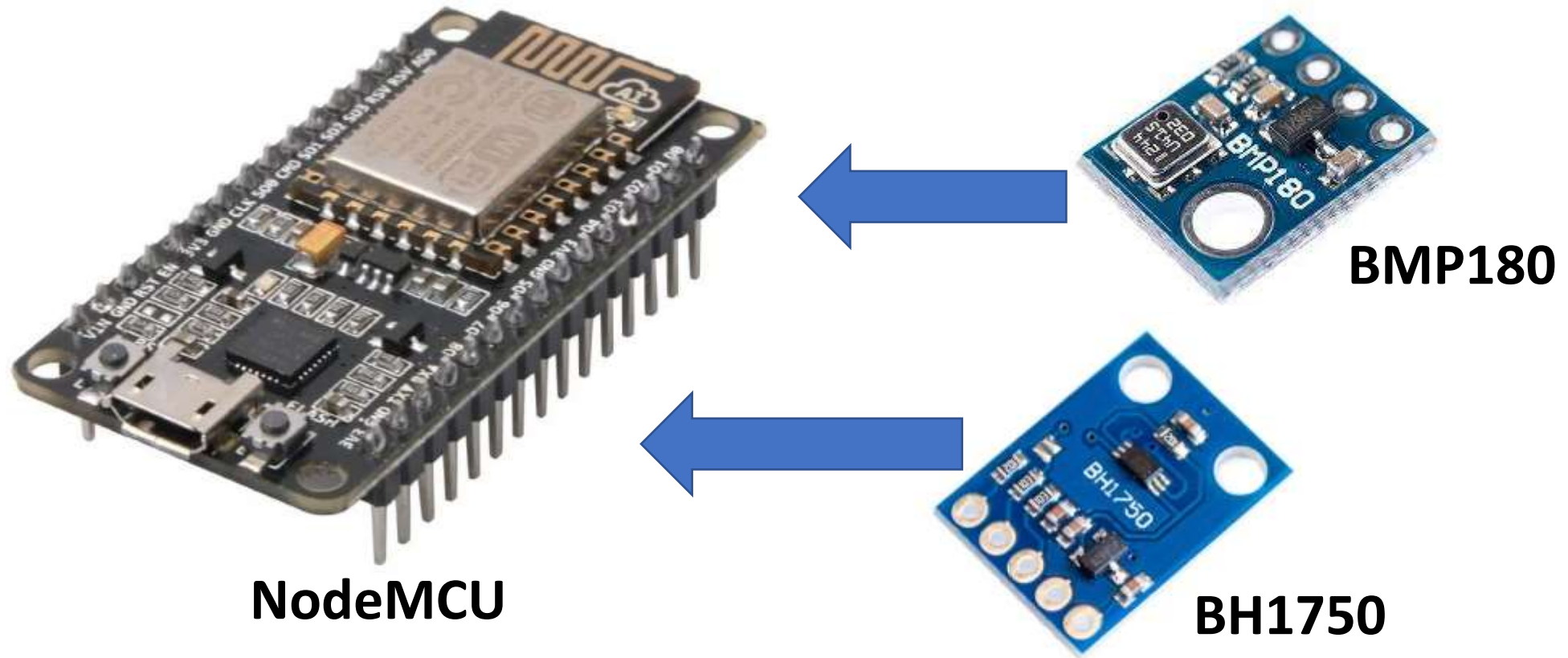
```

Done compiling.

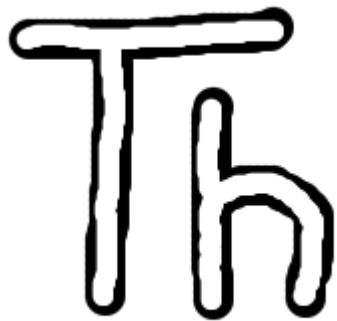
Sketch uses 278753 bytes (26%) of program storage space. Maximum is 1044464 bytes.
Global variables use 28748 bytes (35%) of dynamic memory, leaving 53172 bytes for local variables. Maximum is 81920 bytes.

<B48AM (balanced). Use pgm_read macros for IRAM/PROGMEM, 4MB (FS:2MB OTA:~1019KB), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM8

IoT::exemplo de aplicação



IoT::exemplo de aplicação



Thonny - D:\Claudio\Documents\Projetos\IoT\MicroPython\workspace\mqtt-bmp180-bh1750.py @ 52 : 16

Arquivo Editar Visualizar Executar Device Ferramentas Ajuda

Arquivos ×

- MicroPython device
 - bh1750.py
 - bmp180.py
 - boot.py
- This computer
 - D:\Claudio\Documents\Projetos\IoT\MicroPython\workspace
 - apds9960
 - ESP32MicroPython
 - oo
 - user_lib
 - 1-wire-scan.py
 - adc-ldr-nivel-esp32.py
 - adc-ldr-nivel-esp8266.py
 - adc-pot-esp32.py
 - adc-pot-esp8266.py
 - adc-pot-pwm-esp32.py
 - adc-pot-pwm-esp8266.py
 - adc-termistor-esp32.py
 - adc-termistor-esp8266.py
 - alexapy

mqtt-bmp180-bh1750.py × mqtt-led.py ×

```
1 # I2C -> ESP8266/ESP32
2 # VIN -> 3v3
3 # GND -> GND
4 # SCL -> GPIO4
5 # SDA -> GPIO5
6
7 from machine import Pin, I2C
8 from time import sleep
9 from bmp180 import BMP180
10 from bh1750 import BH1750
11 import network
12 from umqtt.simple import MQTTClient
13 import gc
14 gc.collect()
15
16 print('Iniciando...')
17 i2c = I2C(sda=Pin(5), scl=Pin(4))
18 bmp = BMP180(i2c) # 0x77
19 bmp.oversample_sett = 2
20 bmp.baseline = 101325
```

Shell ×

```
Publicando no servidor MQTT
('5.196.95.208', 1883)
Envio realizado.
Publicando no servidor MQTT
('5.196.95.208', 1883)
Envio realizado.
```

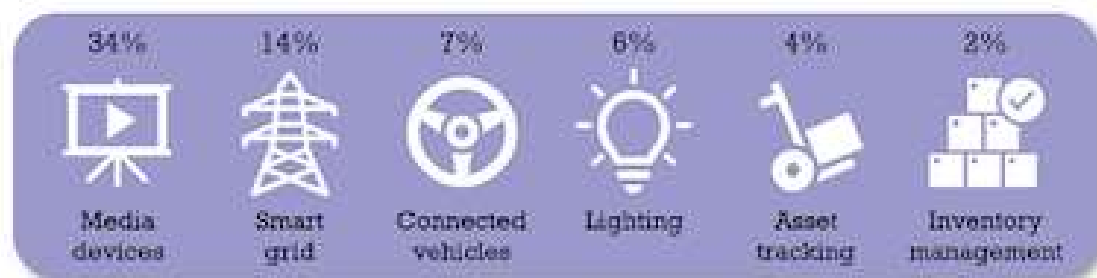
The Internet of Things (IoT) Market 2019-2030

24.1 billion

IoT connected devices in 2030 (7.6bn 2019)

\$1.5 trillion

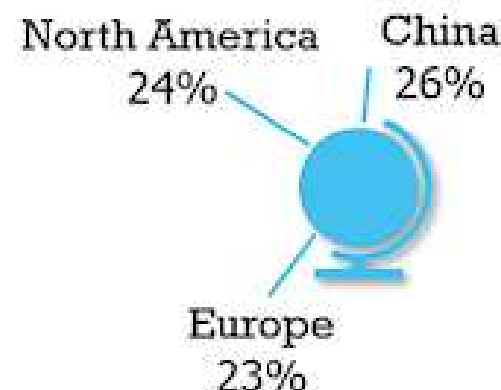
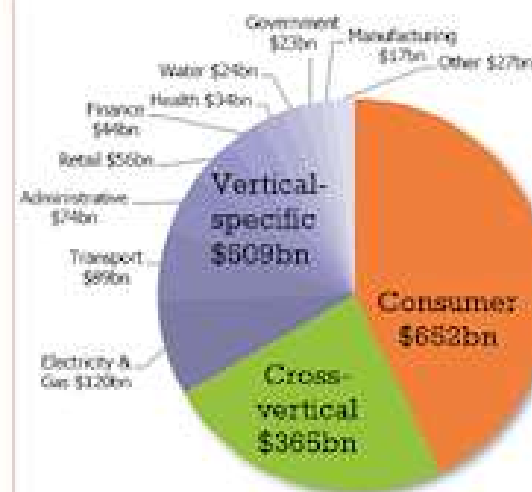
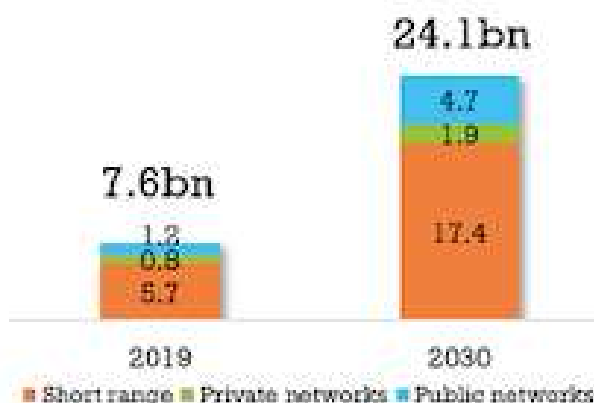
IoT revenue in 2030 (\$466bn 2019)



TRANSFORMA
INSIGHTS

transformainsights.com

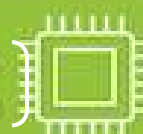
@transformatweet



1010
1010

66% services
\$1.0 trillion

34% hardware
\$520 billion



IoT::Brasil

- 70% da população já está conectada
- Estimativa de 724 milhões de dispositivos conectados até 2022
- Atenção do Governo Federal, Decreto no 9.894 de 25/06/2019 que:
 - *Institui o Plano Nacional de Internet das Coisas e dispõe sobre a Câmara de Gestão e Acompanhamento do Desenvolvimento de Sistemas de Comunicação Máquina a Máquina e Internet das Coisas.*
- IoT deve proporcionar crescimento de produtividade e acrescentar R\$ 122 bilhões ao PIB até 2025, além de gerar de 1.9 a 2.6 milhões de novos empregos.

E o salário?

De acordo com o site de empregos Glassdoor, a média da remuneração dos postos de trabalho em IoT é de:

5.000 REAISDESENVOLVEDORES DE DISPOSITIVOS

7.000 REAISESPECIALISTAS EM ANALYTICS

8.000 REAISESPECIALISTAS EM CONECTIVIDADE (TELECOMUNICAÇÕES)

15.000 REAISARQUITETOS IOT, QUE SÃO RESPONSÁVEIS PELO DESENHO DA SOLUÇÃO

2,3 milhões

de empregos deverão ser criados globalmente graças à inteligência artificial em 2020

FONTE: GARTNER

Existem mais de **7** bilhões de dispositivos IoT conectados no mundo atualmente, e esse número deve aumentar para

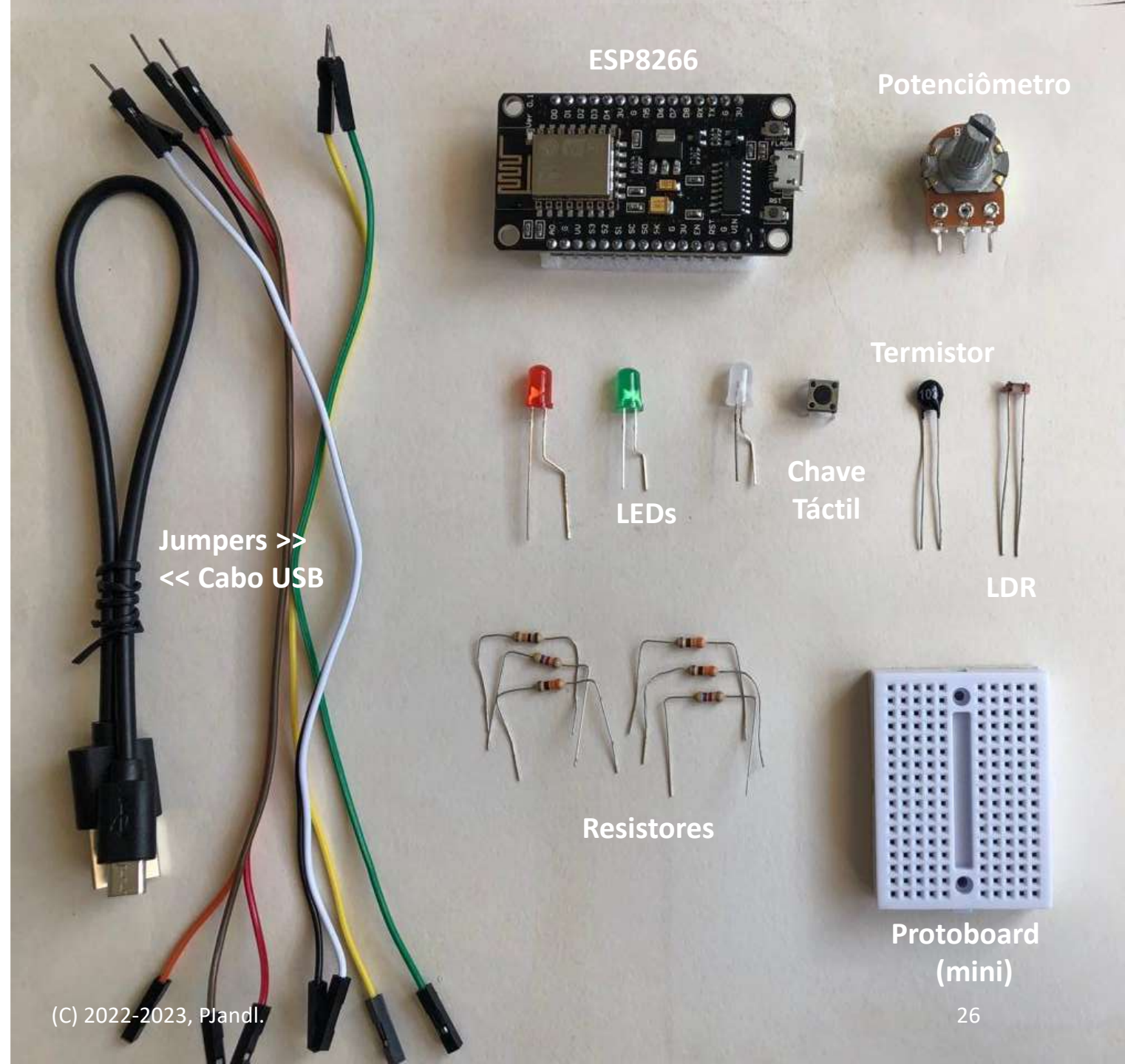
22 bilhões até 2025

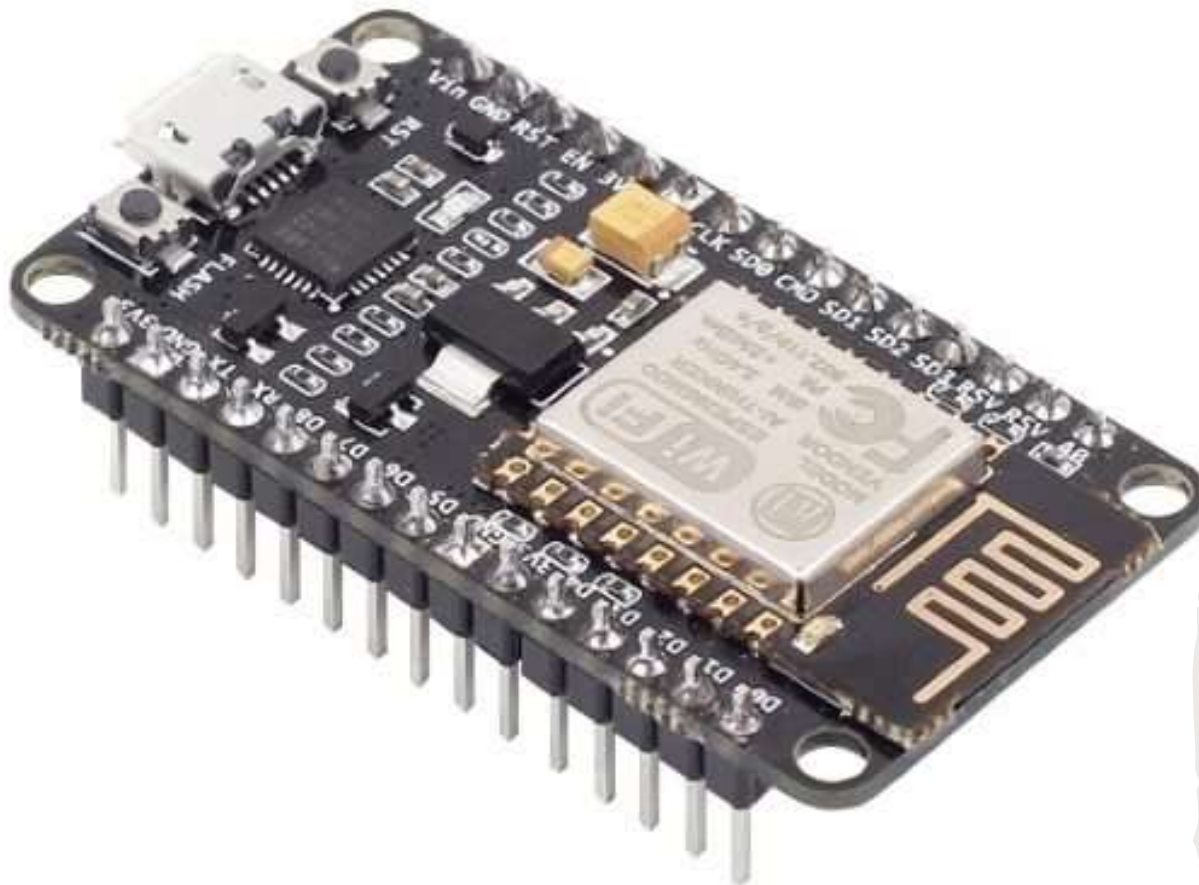
FONTE: ORACLE

Nosso materiais

- ESP8266 ou ESP32
- Protoboard
- Leds
- Resistores
- Fotoresistor (LDR)
- Chave táctil (botão)
- Jumpers e cabo USB A-C

23/03/2023





NodeMCU ESP8266

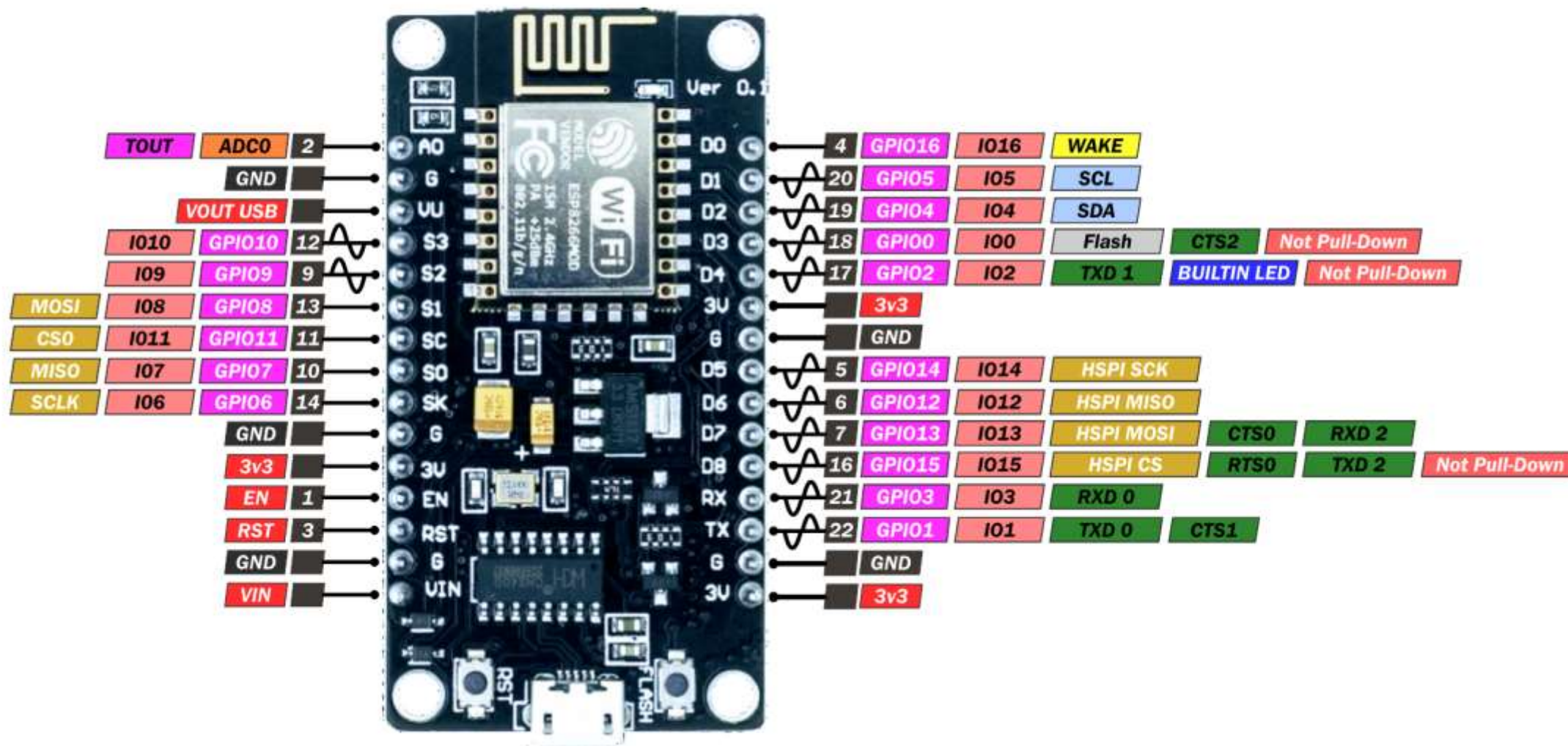
- System-On-Chip baseado no ESP-12 E/F
- CPU Xtensa single core RISC 32bits operando à 80MHz
- Wi-Fi 802.11 b/g/n (STA/AP/STA+AP)
- 17 GPIO (08 saídas PWM)
- 01 entrada ADC (10bits resolução)
- Suporte aos barramentos I2C, SPI, UART
- Memória:
 - RAM: 32Kb (instruções) + 96Kb (dados)
 - ROM: 64Kbytes (boot)
 - Flash SPI Winbond (de 512Kb à 4Mb)
- Vários fabricantes

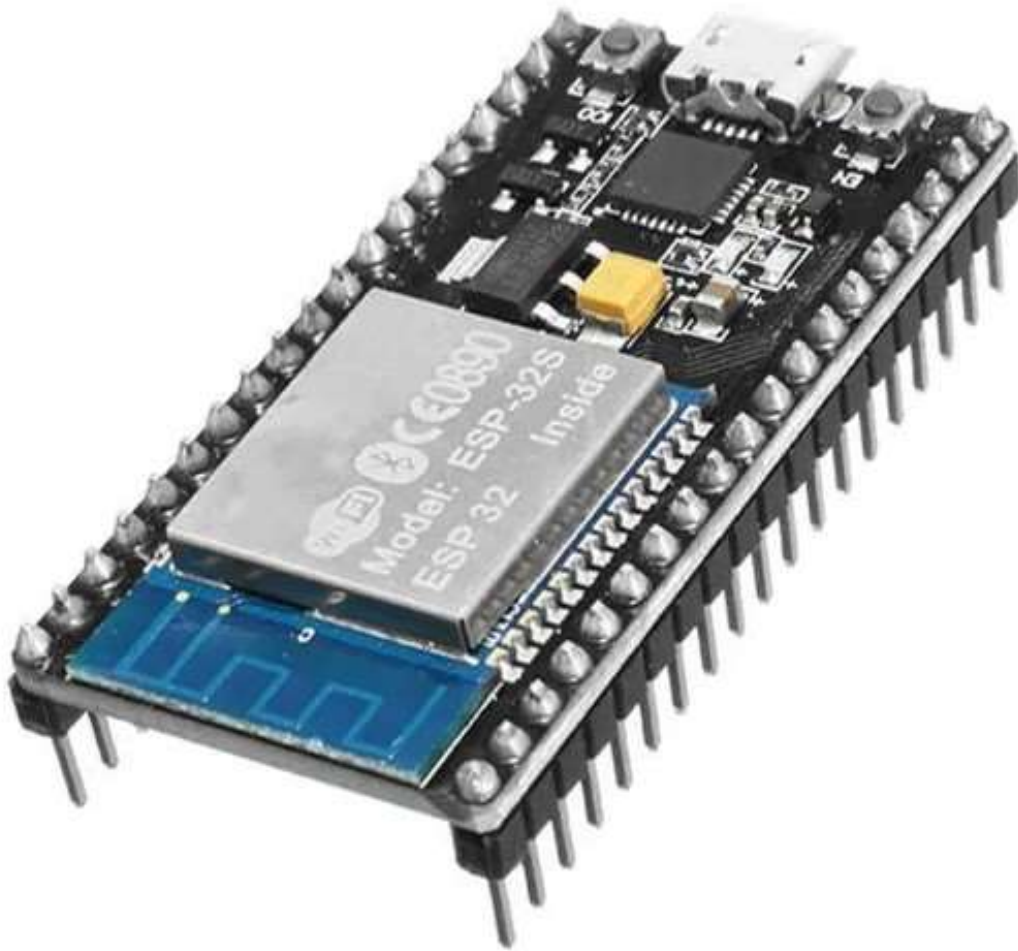
Preços de
R\$30 a R\$60

NodeMCU v3 CH340

PINOUT

ESP8266





NodeMCU ESP32

- System-On-Chip baseado no ESP-12 E/F
- CPU Xtensa dual core RISC 32bits operando à 160MHz
- Wi-Fi 802.11 b/g/n (STA/AP/STA+AP)
- 36 GPIO (16 saídas PWM)
- 02 entradas ADC (12bits resolução)
- Suporte aos barramentos I2C, SPI, UART, CAN, I2S
- Memória:
 - RAM: 32Kb (instruções) + 96Kb (dados)
 - ROM: 64Kbytes (boot)
 - Flash SPI Winbond (de 512Kb à 4Mb)
- Vários fabricantes

Preços de
R\$35 a R\$75

Protoboard

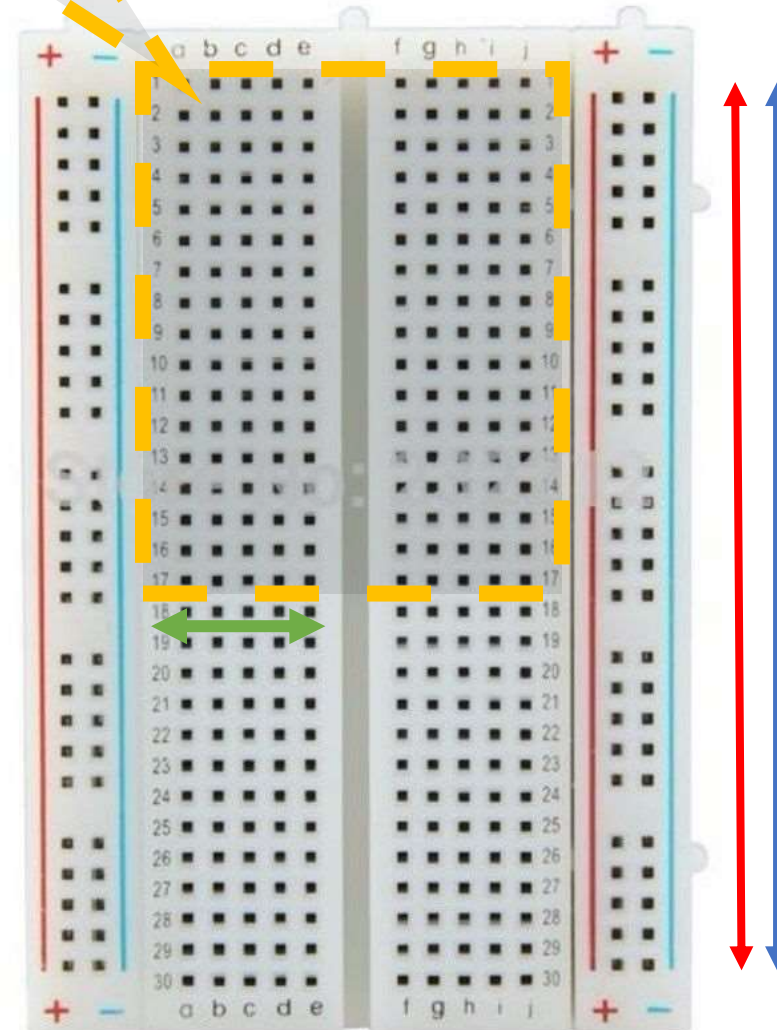
170, 400, 830 pontos



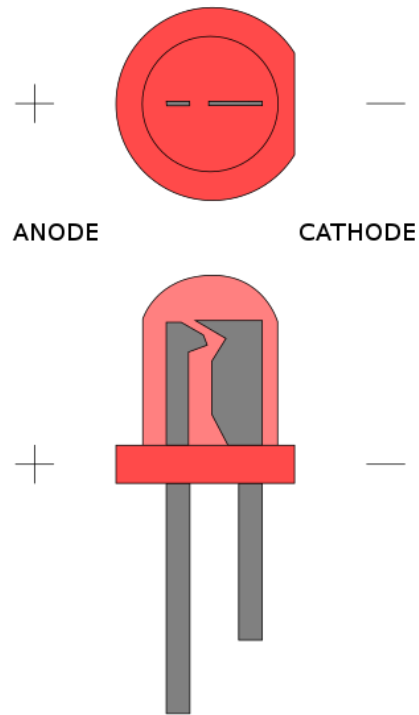
Permite montagens eletrônicas sem necessidade de soldagem.

Mini
protoboard

Linhas de
Trabalho
(horizontais)



Linhas de
Alimentação
(verticais)



LED

Light Emitter Diode

- Componente que produz luz quando alimentado por uma corrente elétrica adequada.
- Possui polaridade.
- Disponível em várias cores, tamanhos e potências.

Resistores

- Componente passivo usado para limitar a passagem da corrente elétrica ou ajustar a tensão elétrica sobre outros componentes do circuito.
- Seu valor (resistência ôhmica) é indicado por um código de cores.
- Disponível em vários valores e potências.



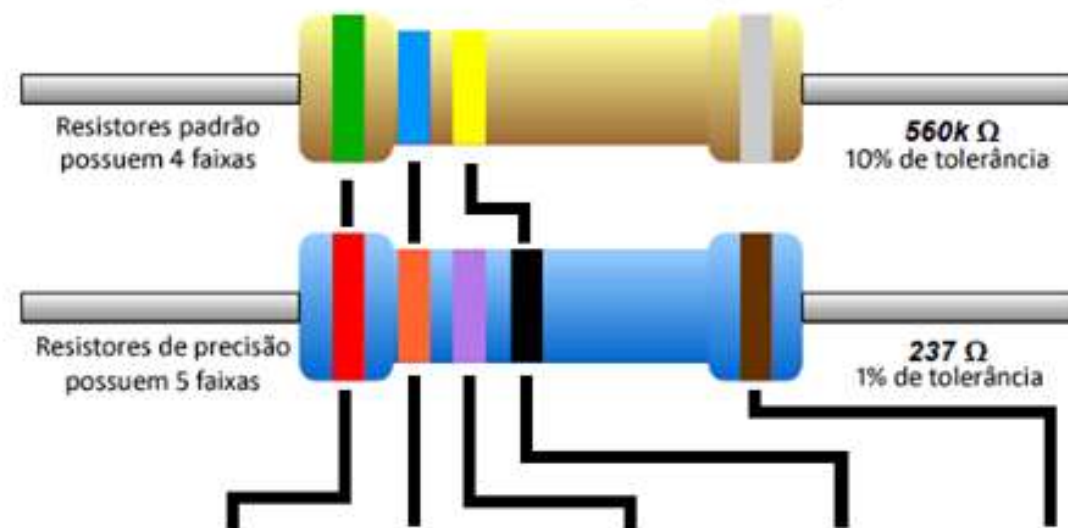
Código de Cores

Polícia **Military** Vai **LAVA**
Viatura Com **Bombril**

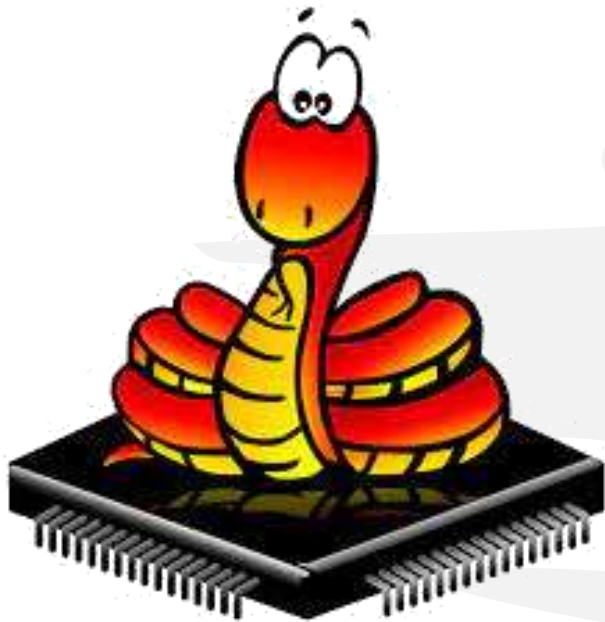
23/03/2023

Código de Cores

A extremidade com mais faixas deve apontar para a esquerda



Cor	1ª Faixa	2ª Faixa	3ª Faixa	Multiplicador	Tolerância
Preto	0	0	0	x 1 Ω	
Marrom	1	1	1	x 10 Ω	+/- 1%
Vermelho	2	2	2	x 100 Ω	+/- 2%
Laranja	3	3	3	x 1K Ω	
Amarelo	4	4	4	x 10K Ω	
Verde	5	5	5	x 100K Ω	+/- 5%
Azul	6	6	6	x 1M Ω	+/- 25%
Violeta	7	7	7	x 10M Ω	+/- .1%
Cinza	8	8	8		+/- .05%
Branco	9	9	9		
Dourado				x .1 Ω	+/- 5%
Prateado				x .01 Ω	+/- 10%



MicroPython

MicroPython

- É uma implementação de software de uma linguagem de programação amplamente compatível com Python 3, escrita em C, que é otimizada para rodar em um microcontrolador. O MicroPython consiste em um compilador Python para bytecode e um interpretador de tempo de execução desse bytecode.

<https://micropython.org/>

Python Quick Reference

Variable

```
# Assign an number
x = 5
# Assign String variable
maker = "I am a maker "
# Make a number into a string
str(x)
# Combine strings
newString = maker + " More string"
```

Strings

```
# Output a string
print("Hello World ")
# output a number
print(x)
# output string and number
print("hello "+str(x)+ " world")
#input
myVariable = input("message")
#to input number eval output
myVariable = eval(myVariable)
```

Lists

```
# Make a list
myList = ["item0 ", "item1 ", "item2 "]
# Add item to list
myList.append("item3")
# Select first item in a list
myList[0]
# Select last item in the list
myList[-1]
```

Import

```
import time
# Call function from time
time.sleep(1) # Wait for 1 second
# Import single function
from math import sqrt
# Call
sqrt(2)
```

Maths

```
# Addition
1+3
# Exponent
2**3
# Subtraction
1-3
# Modulus
2%3
# Multiplication
1*3
# Integer Division
2//3
# Division
1/3
```

Logic

```
# Return True
True
not False
(True | False) # or
(True & True) #and
# Return False
False
not True
(False | False) # or
(True & False) #and
```

Compare

```
# Return True if comparison holds
x > y # Greater than
x < y # Less
x <= y # Less or equal
x >= y # Greater or equal
x == y # Equal
x != y # Not equal
```

If..else

```
if(x > y):
    print("x is larger ")
elif(x < y):
    print("y is larger ")
else:
    print("Equality! ")
```

Functions

```
# define a function
def threeTimes(xIn):
    return xIn*3
# Call with input=5
threeTimes(5)
# Define Lambda function
t = 6
tMult= lambda a : a*t
# Call with a = 5 t=6
tMult(5)
t = 1
# Call with a = 5 t=1
tMult(5)
```

Loops

```
# While less than
while(x < y):
    x += 1
    print(x)
# While true with break
while(True):
    x += 1
    print(x)
    if(x>y):
        break
```

```
# For loop with default count
for i in range(1,10):
    print(2**i)
# For loop with custom count
for i in range(10,-10,-1):
    print(2**i)
# For loop over list elements
for i in myList:
    print(i)
```

Classes

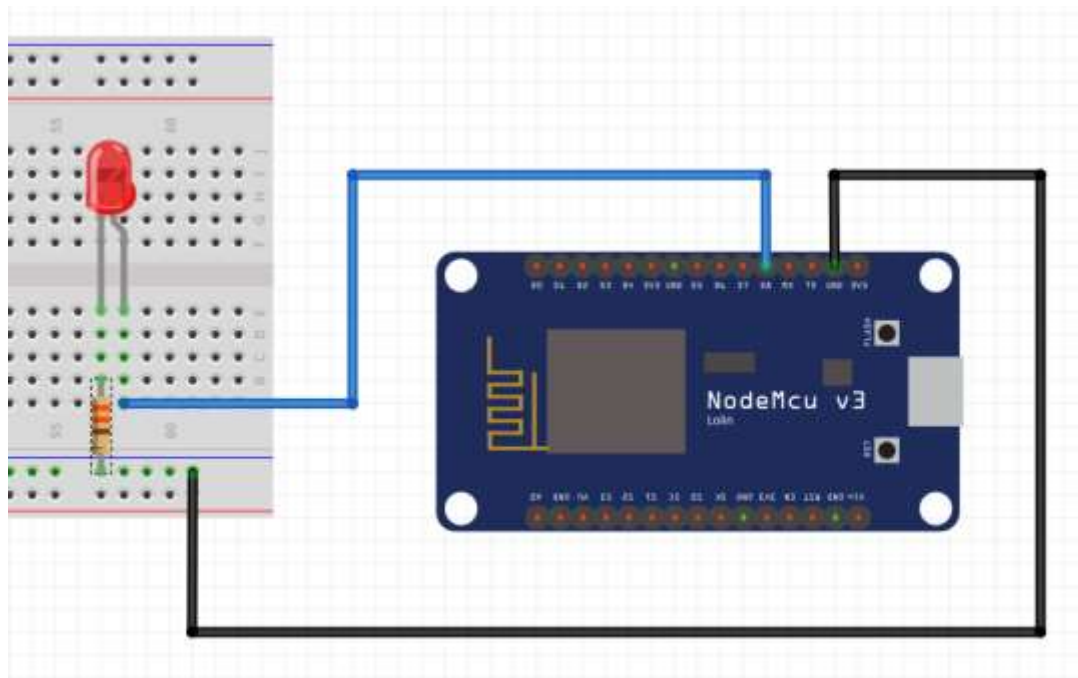
```
# define Class
class myClass:
    # Initialisation
    def __init__(self,in1,in2):
        self.input1 = in1
        self.input2 = in2
    def add(self): # Class method
        return self.input1+self.input2
# Run the program
myObject = myClass(1,2)
myObject.add ()
```


Vamos trabalhar?

<https://github.com/pjandl/ocf>

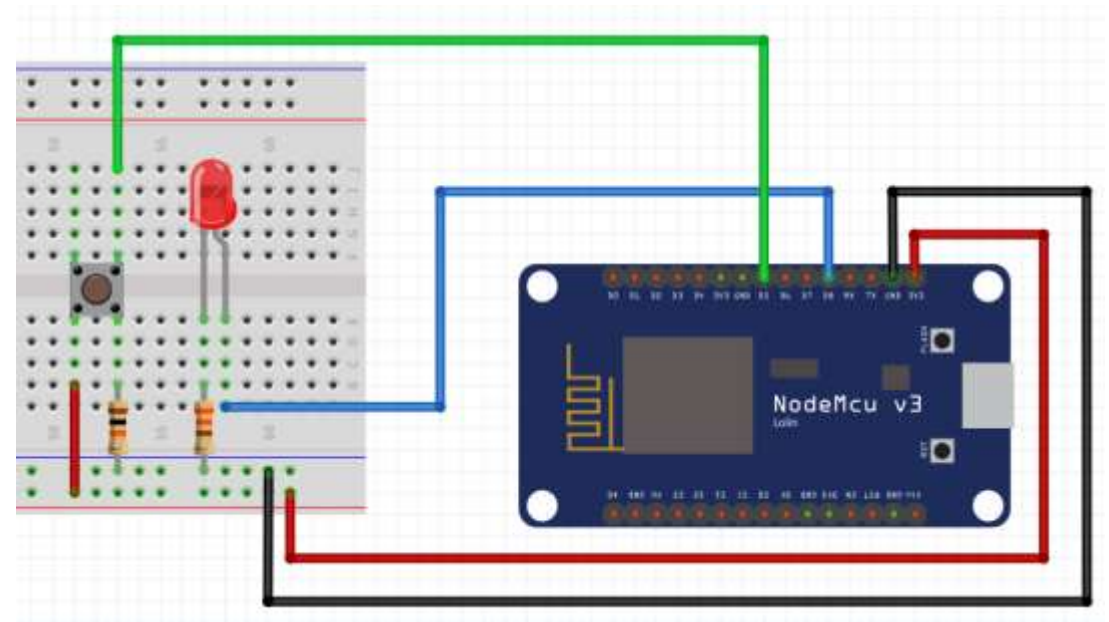
Led externo

T-2022-2 → 03_led_externo.md



Led externo, botão e retenção de estado

T-2022-2 → 05_botao_led_estado.md

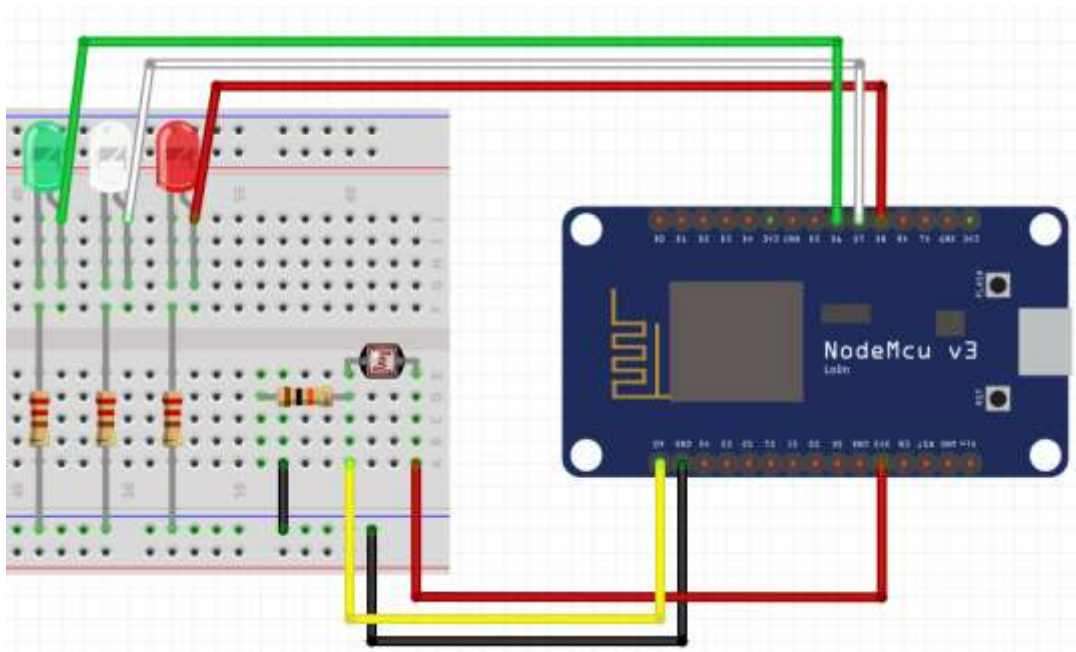


Vamos trabalhar?

<https://github.com/pjandl/ocf>

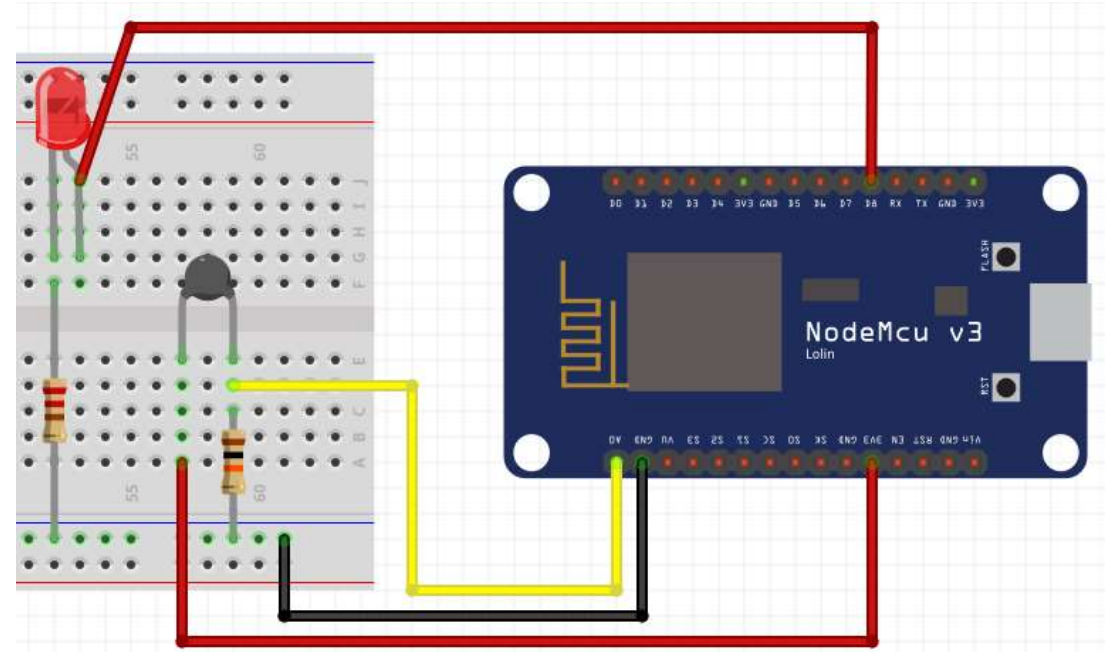
LDR e escala de Leds

T-2022-2 → 09_ADC_ldr_leds.md



NTC e led

T-2022-2 → 10_ADC_ntc.md



ESP8266 ::Primeiros Passos

Peter Jandl Junior

peter.jandl@fatec.sp.gov.br

<https://github.com/pjandl/ocf>

<https://tecnopode.blogspot.com/>

