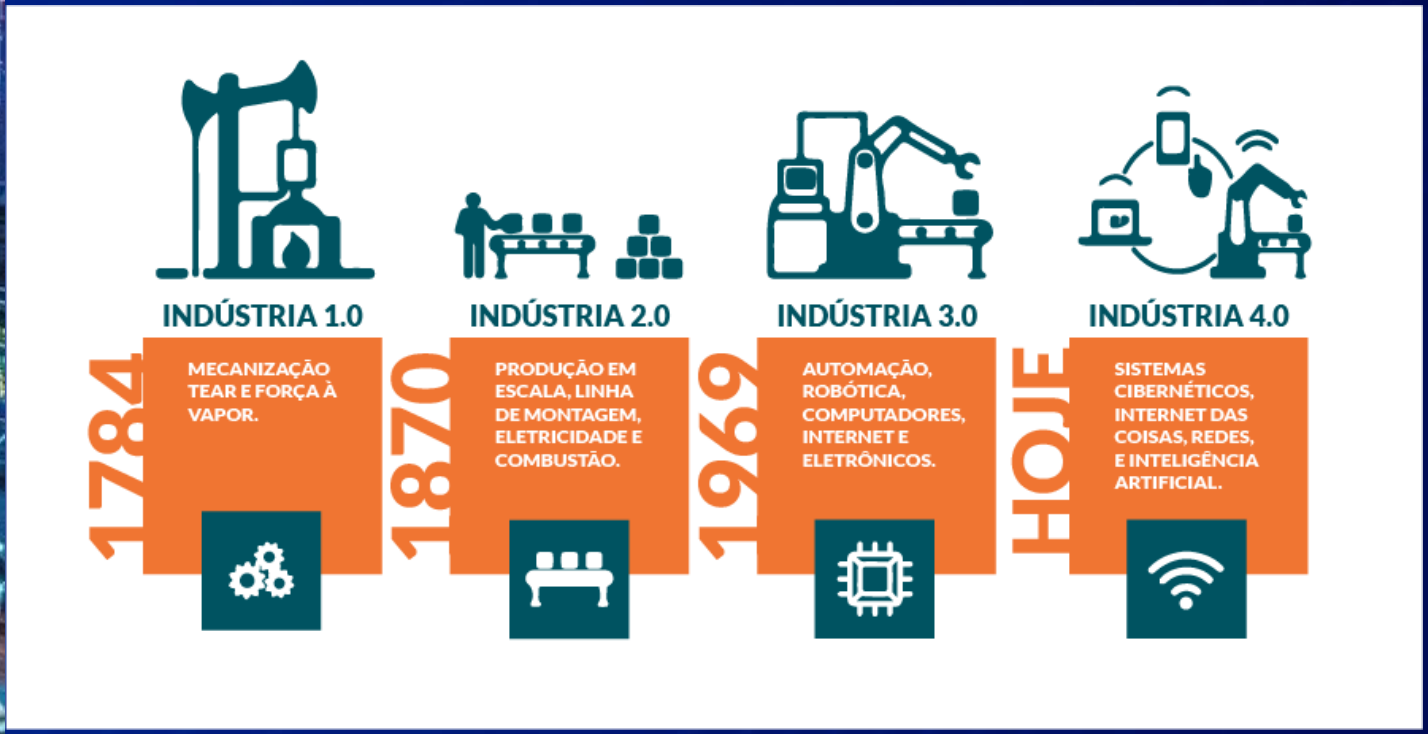

Instituto Nacional de Telecomunicações
510
Inatel

MINICURSO - INTRODUÇÃO À SISTEMAS EMBARCADOS

Aplicado a Automação Industrial com MicroPython

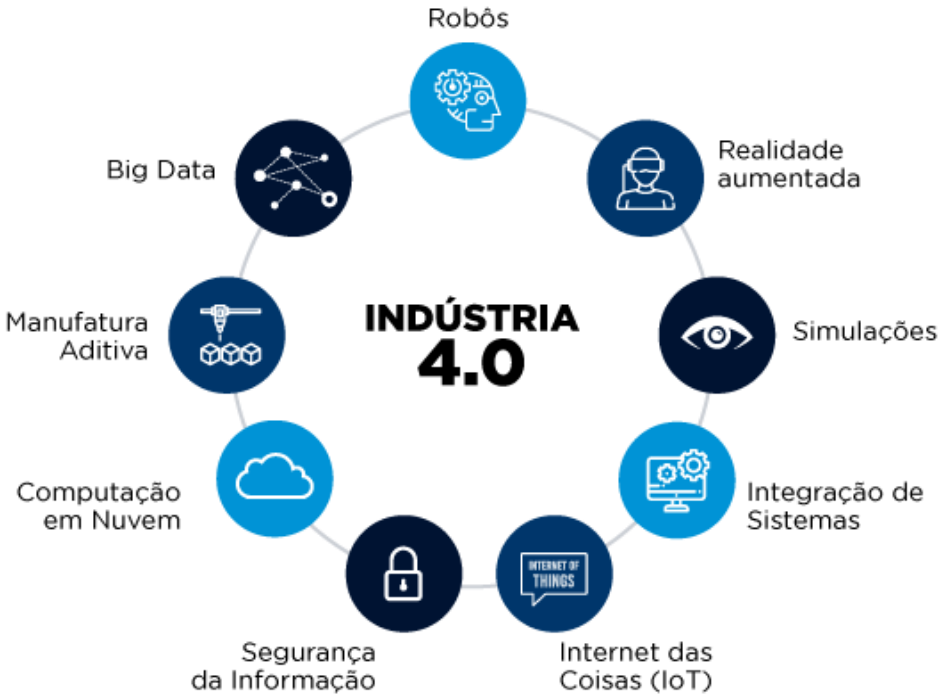
Inatel

INDÚSTRIA 4.0



Inatel

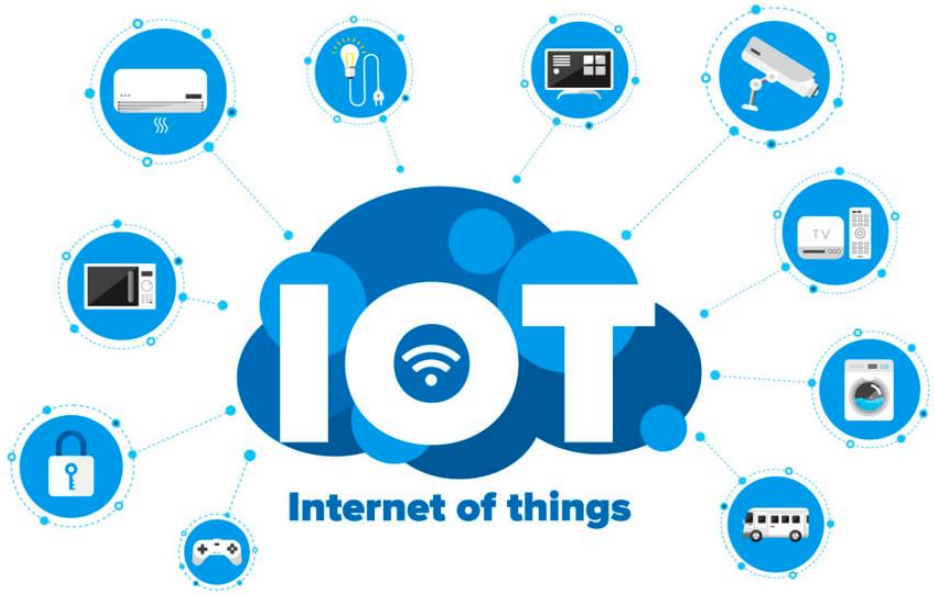
INDÚSTRIA 4.0



Inatel

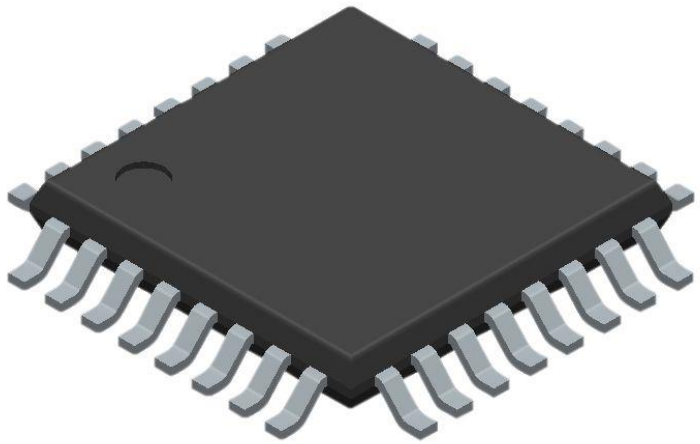
INTERNET DAS COISAS

- Comunicação sem fio
- Aumento de eficiência operacional
- Aumento na produtividade
- Novas oportunidades de negócios
- Redução de tempo ocioso
- Melhor agendamento para manutenção preditiva
- Melhor monitoramento de máquinas e dispositivos
- Maior segurança para os trabalhadores



Inatel

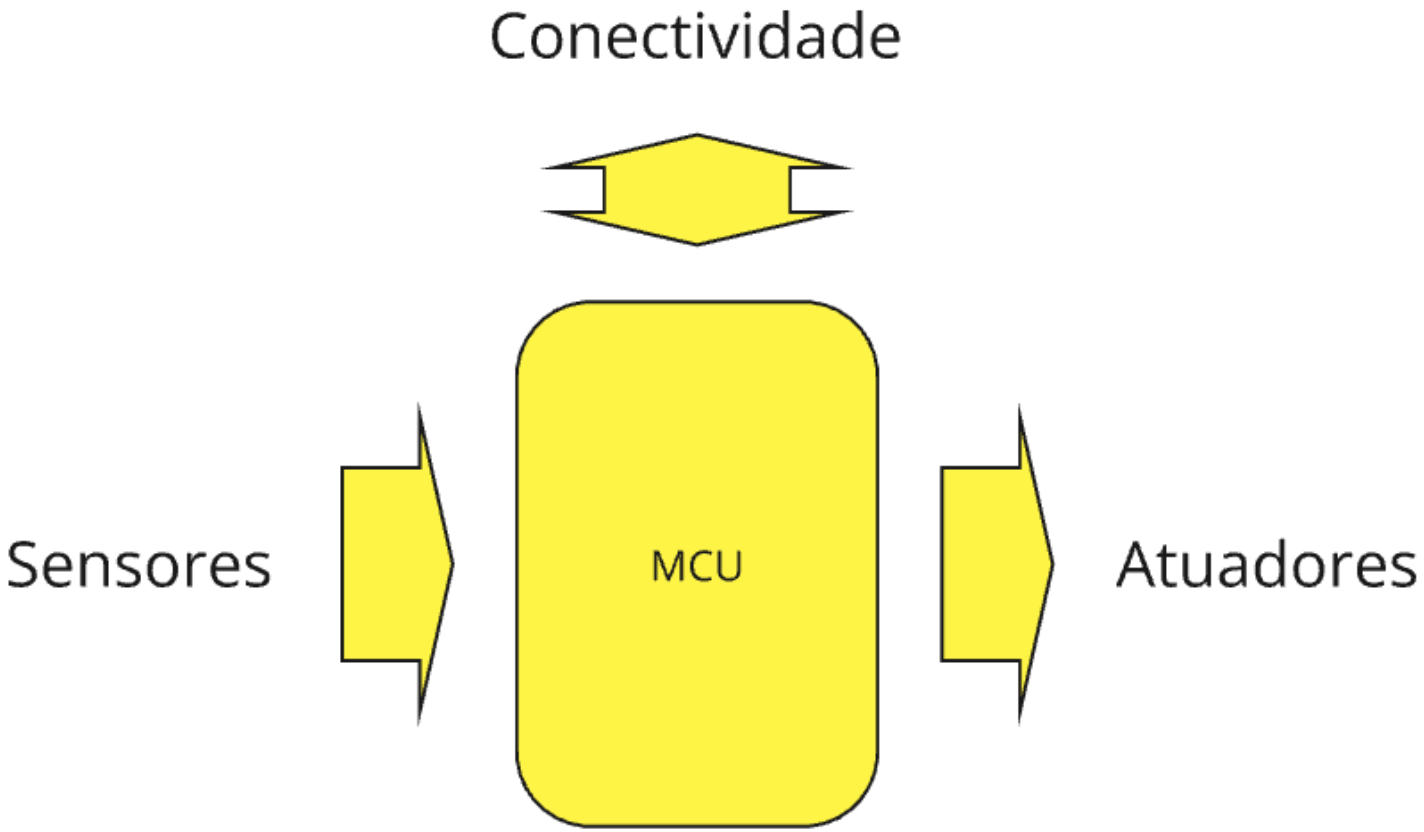
MICROCONTROLADORES



Circuito integrado (CI) que combina processador, memórias e periféricos em um único chip para controlar sistemas embarcados

Inatel

MICROCONTROLADORES



Inatel

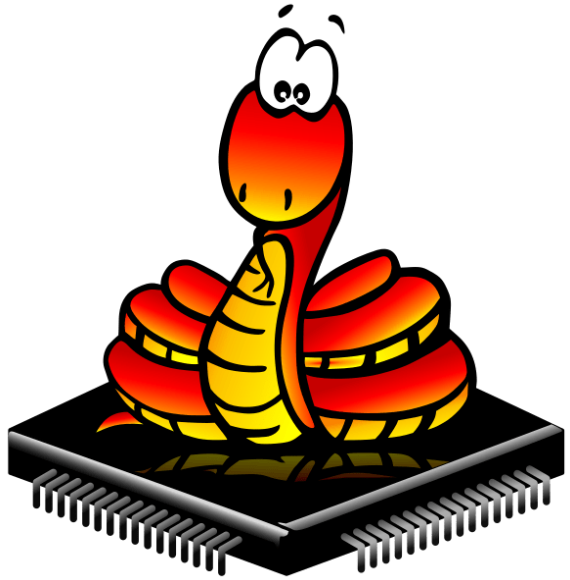
PROTOCOLOS DE COMUNICAÇÃO

- Internet
  - MQTT
  - HTTP
  - HTTPS
- Industrial
  - Modbus
  - PROFINET
  - Profibus
- Serial
  - I^2C
  - UART
  - SPI
- Outros protocolos sem fio
  - Bluetooth e Bluetooth Low Energy
  - Zigbee
  - LoRa

Inatel

MICROPYTHON

- Interpretador Python otimizado para dispositivos com recursos limitados.
- Compatível com bibliotecas Python padrão, mas adaptado para hardware embarcado.
- Suporte para comunicação com GPIOs, I²C, SPI, UART, PWM, entre outros.
- Código mais legível e fácil de desenvolver comparado a C/C++.



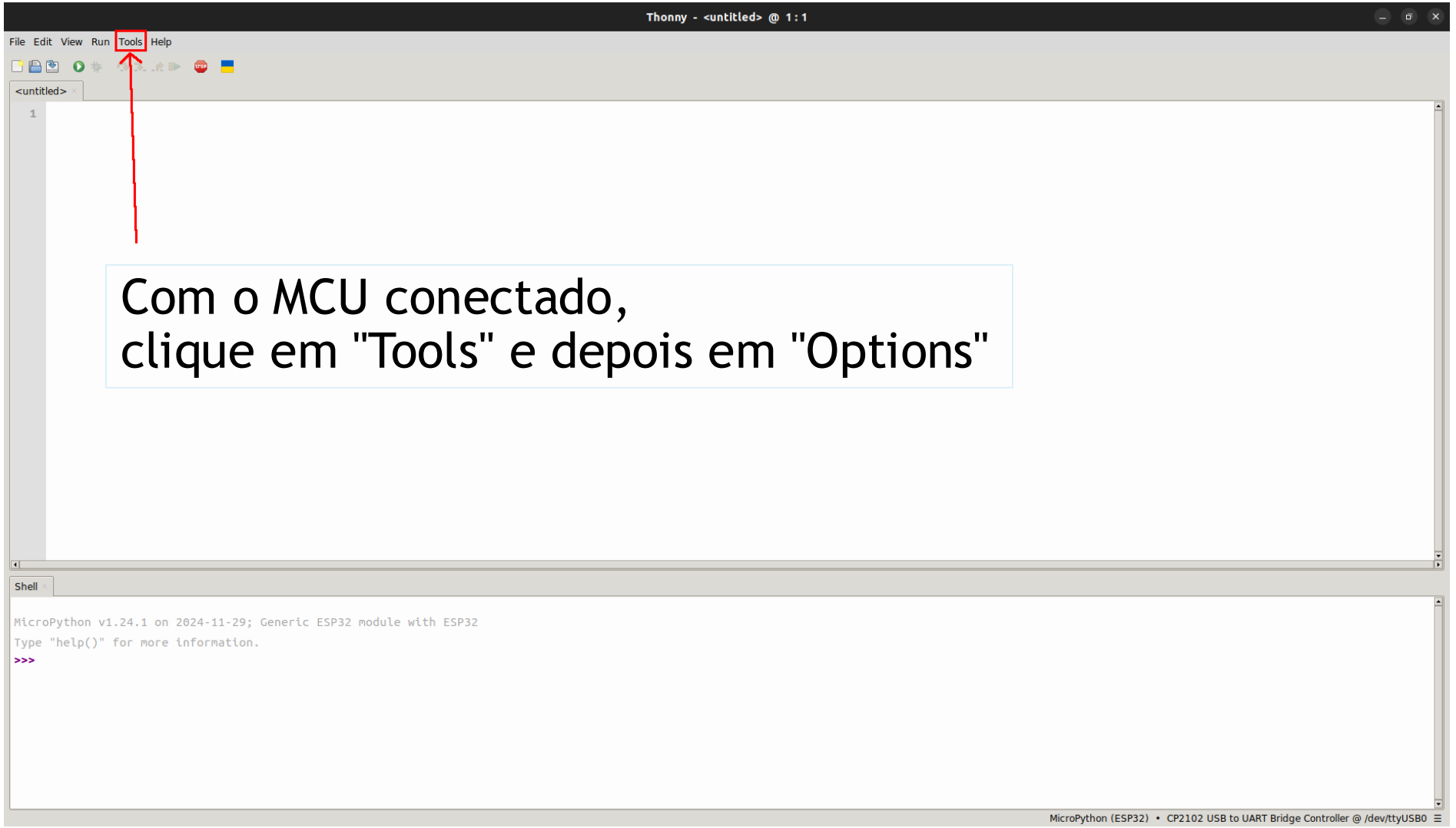
Inatel

THONNY IDE

- Thonny é uma IDE (Ambiente de Desenvolvimento Integrado) voltado para iniciantes em Python, com uma interface simples e intuitiva.
- Oferece suporte a MicroPython, permitindo desenvolvimento, depuração e execução do código diretamente em microcontroladores.

Inatel

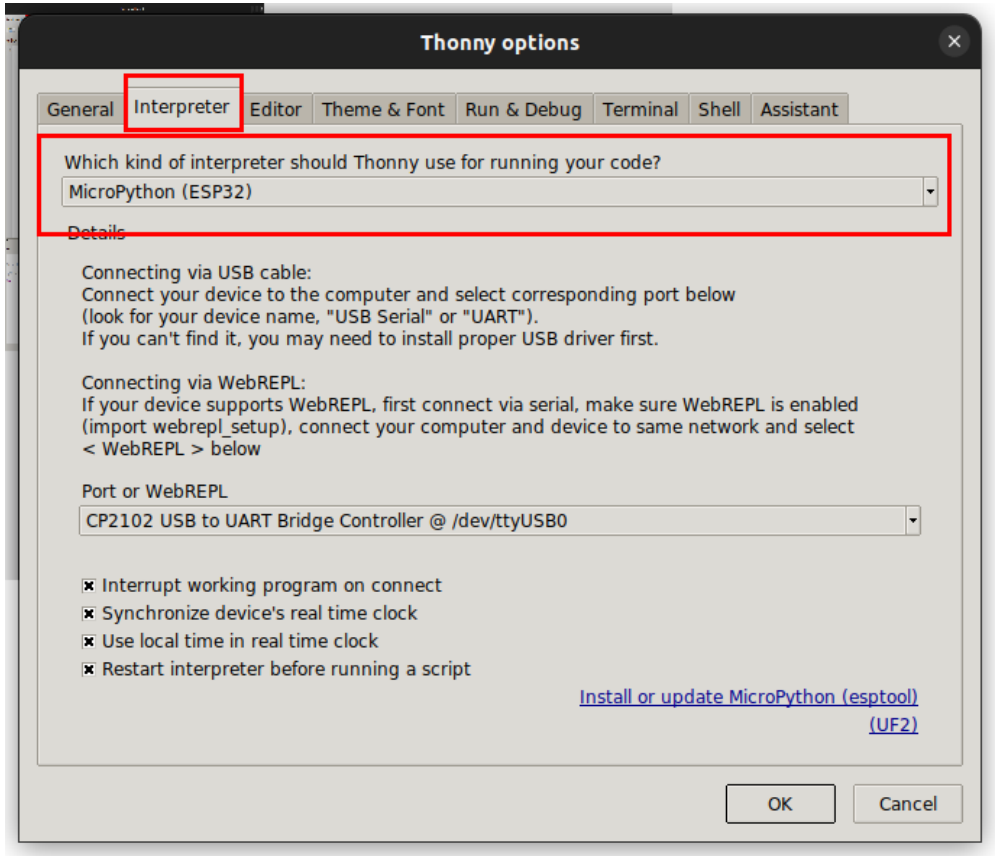
THONNY IDE — PREPARAÇÃO DO AMBIENTE



Com o MCU conectado,
clique em "Tools" e depois em "Options"

Inatel

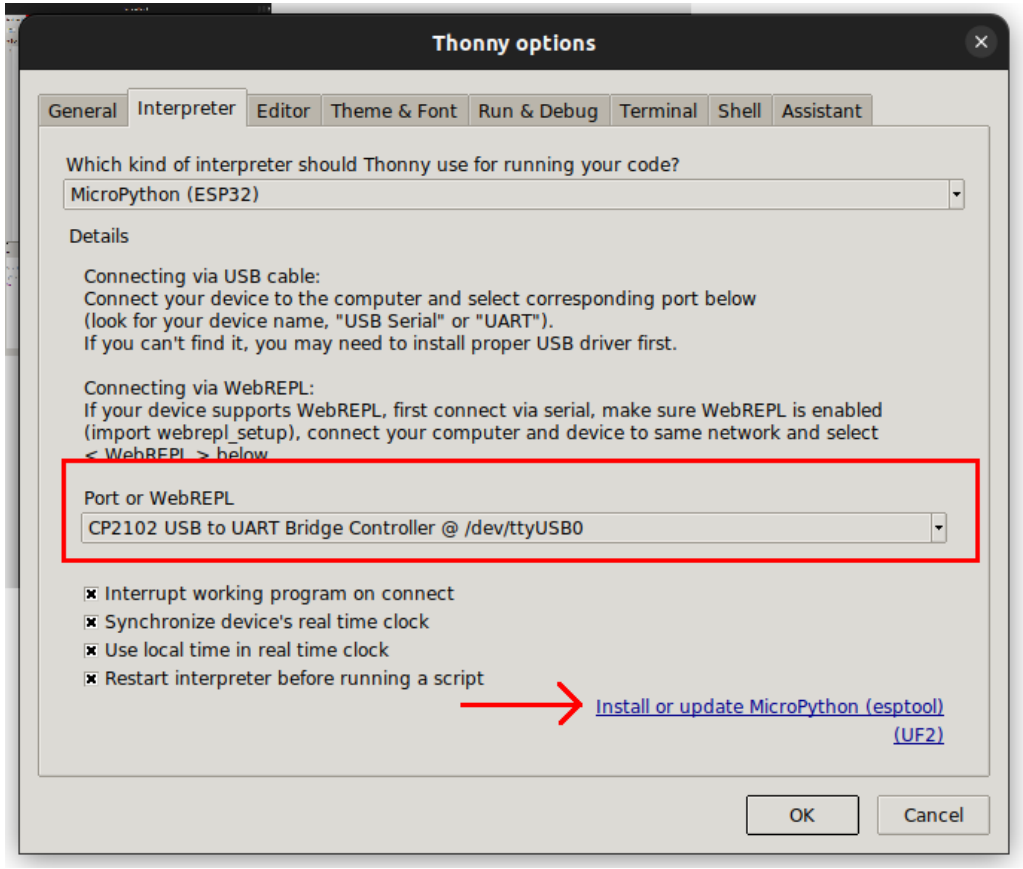
THONNY IDE — PREPARAÇÃO DO AMBIENTE



- Selecione a aba "Interpreter" e selecione o interpretador "MicroPython (ESP32)"
- Essa etapa pode variar de acordo com o microcontrolador utilizado

Inatel

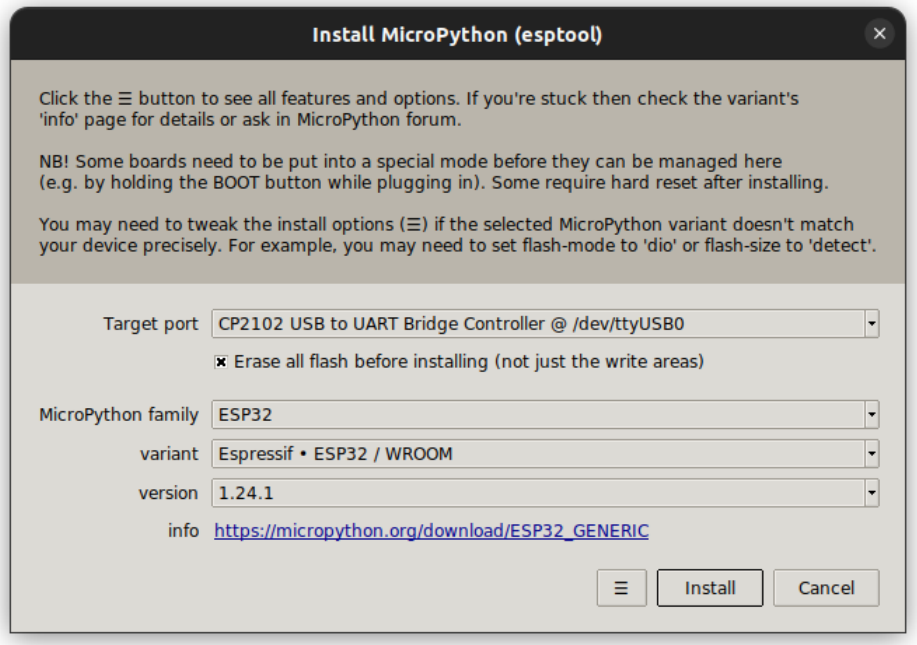
THONNY IDE — PREPARAÇÃO DO AMBIENTE



- Escolha a porta na qual o microcontrolador está conectado
- Por fim, clique em "Install or update MicroPython"

Inatel

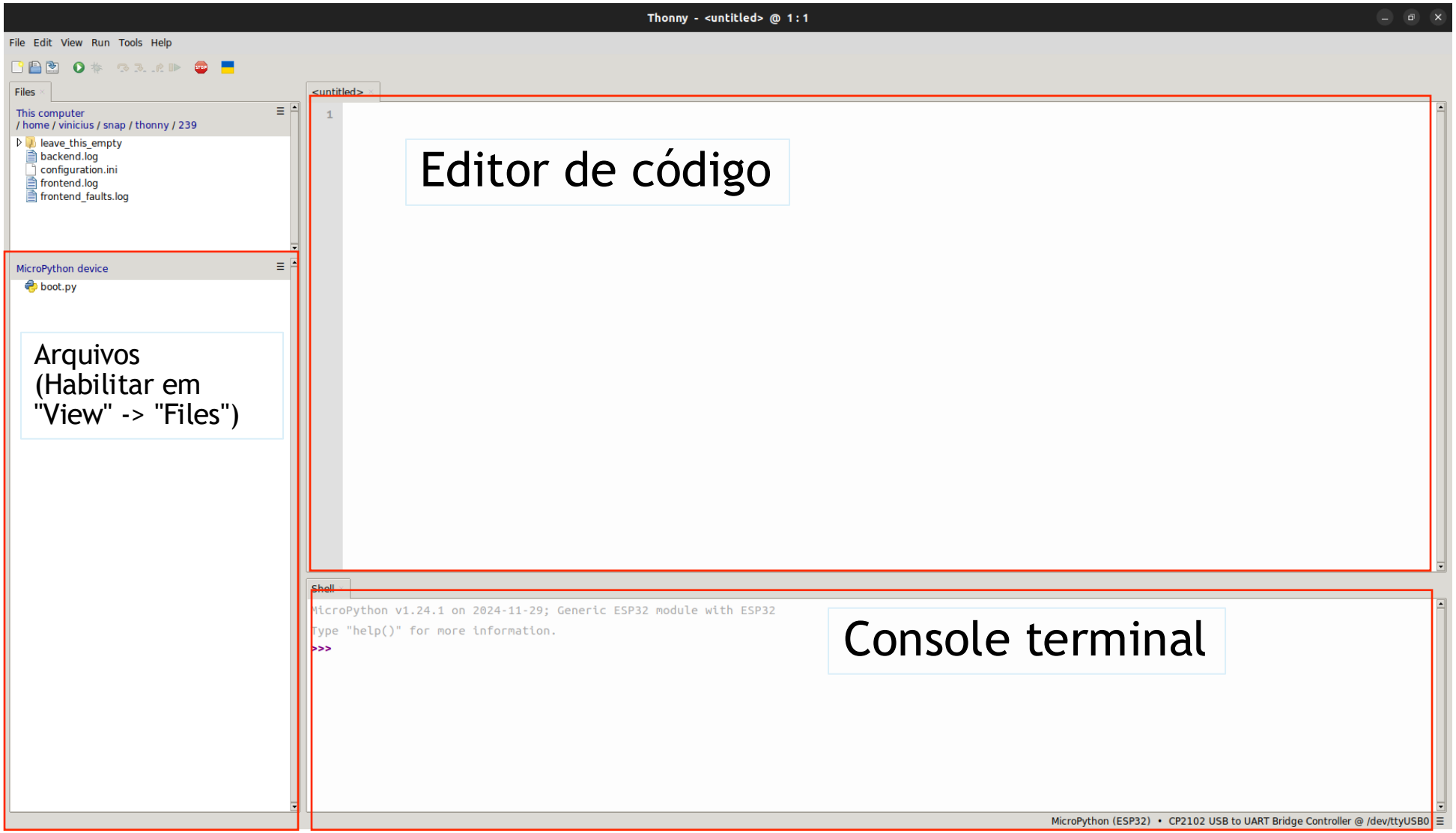
THONNY IDE — PREPARAÇÃO DO AMBIENTE



- Selecione a família e a variação do microcontrolador
- A versão a ser selecionada será a mais recente lançada
- Após o final do download, basta fechar as abas de configurações e o MicroPython já estará rodando no seu ESP32

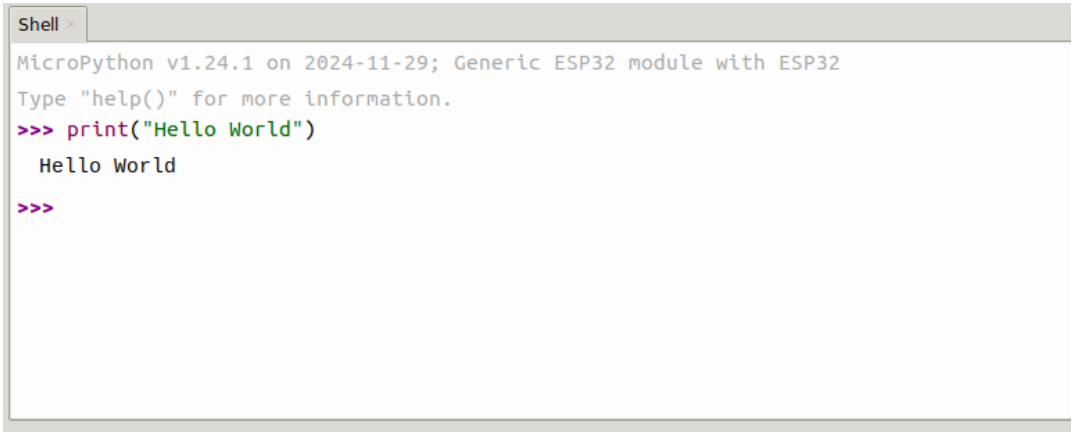
Inatel

THONNY IDE



Inatel

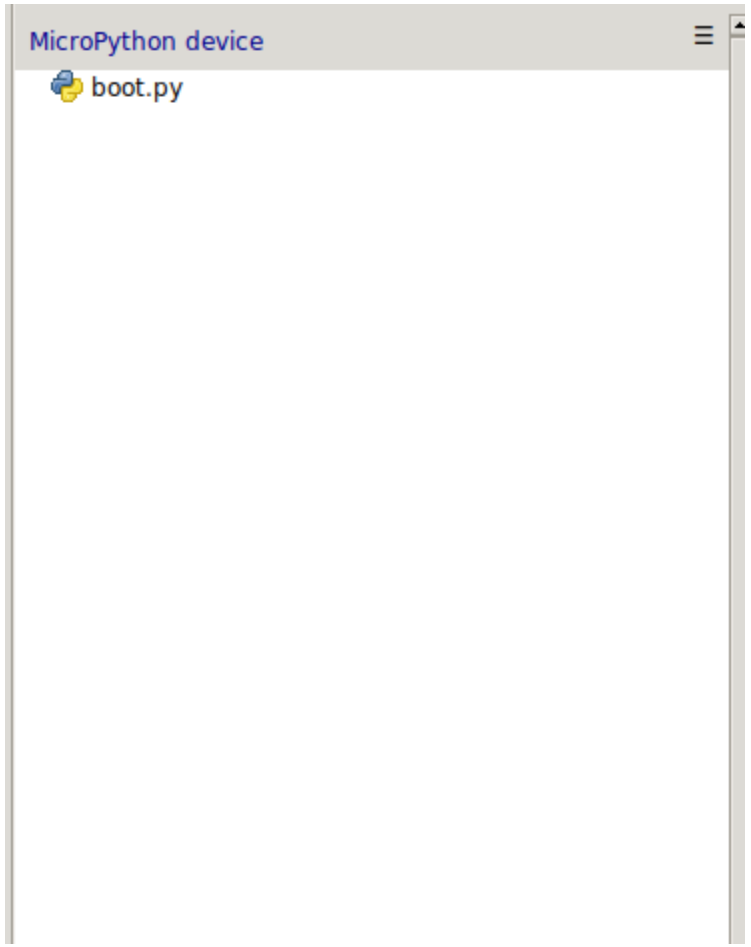
THONNY IDE — CONSOLE TERMINAL



- No console terminal é possível executar comandos Python dentro do ESP32
- Experimente executar o comando `print("Hello World")`

Inatel

THONNY IDE — ARQUIVOS



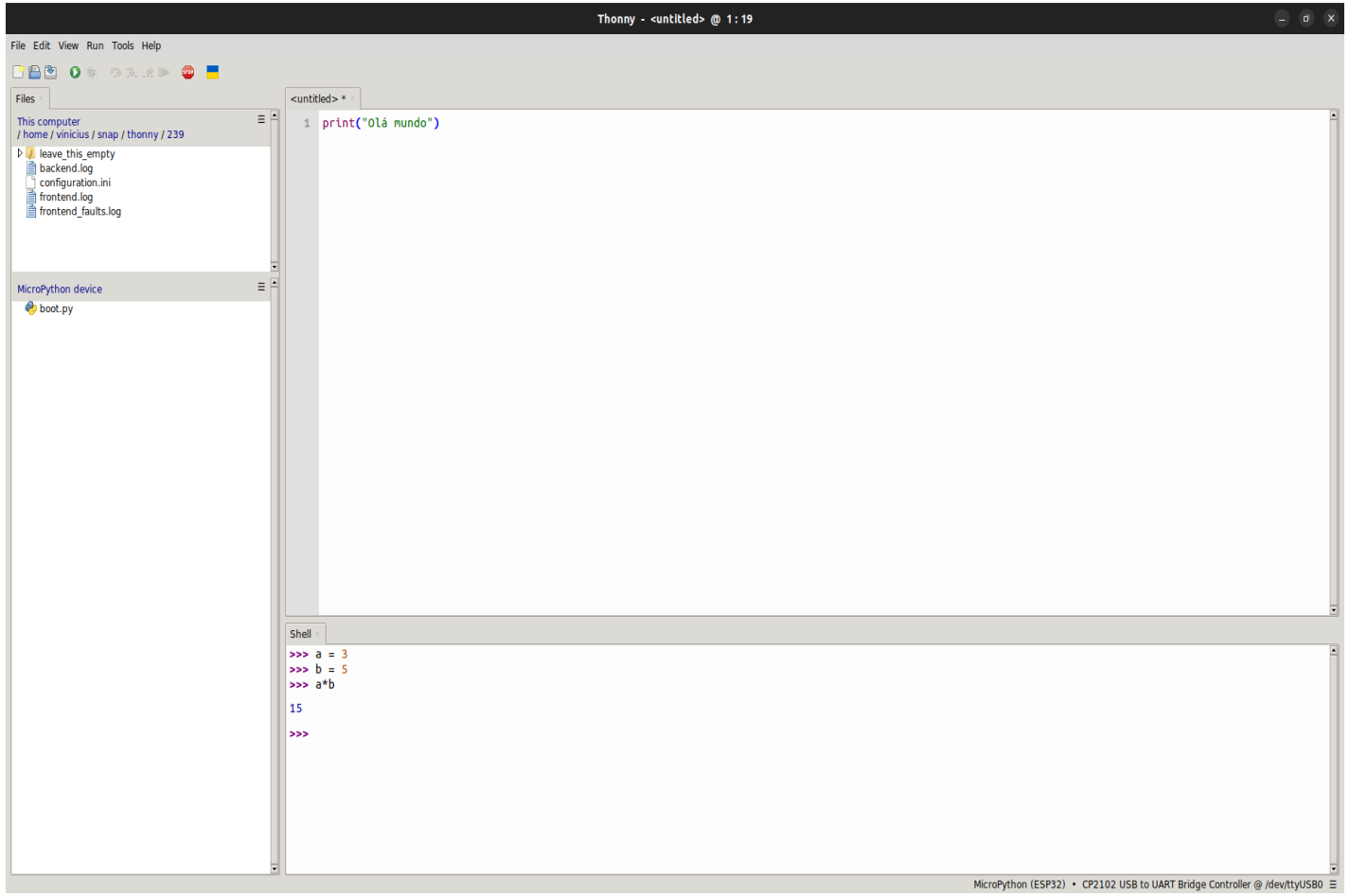
- Além de executar comandos através do terminal, é possível executar scripts pré-programados no microcontrolador.
- Os arquivos salvos no dispositivos são mostrados na aba "Files", em "MicroPython device".
- Os arquivos que sempre serão executados são nomeados como **`boot.py`** e **`main.py`**, nessa ordem.
- Caso a execução dos scripts presentes nesses arquivos se encerre, ou se algum desses não estiver presente, o modo interativo será inicializado automaticamente.

Inatel

THONNY IDE — EDITANDO ARQUIVOS

- A aba de edição de arquivos é utilizada para modificar ou escrever novos códigos

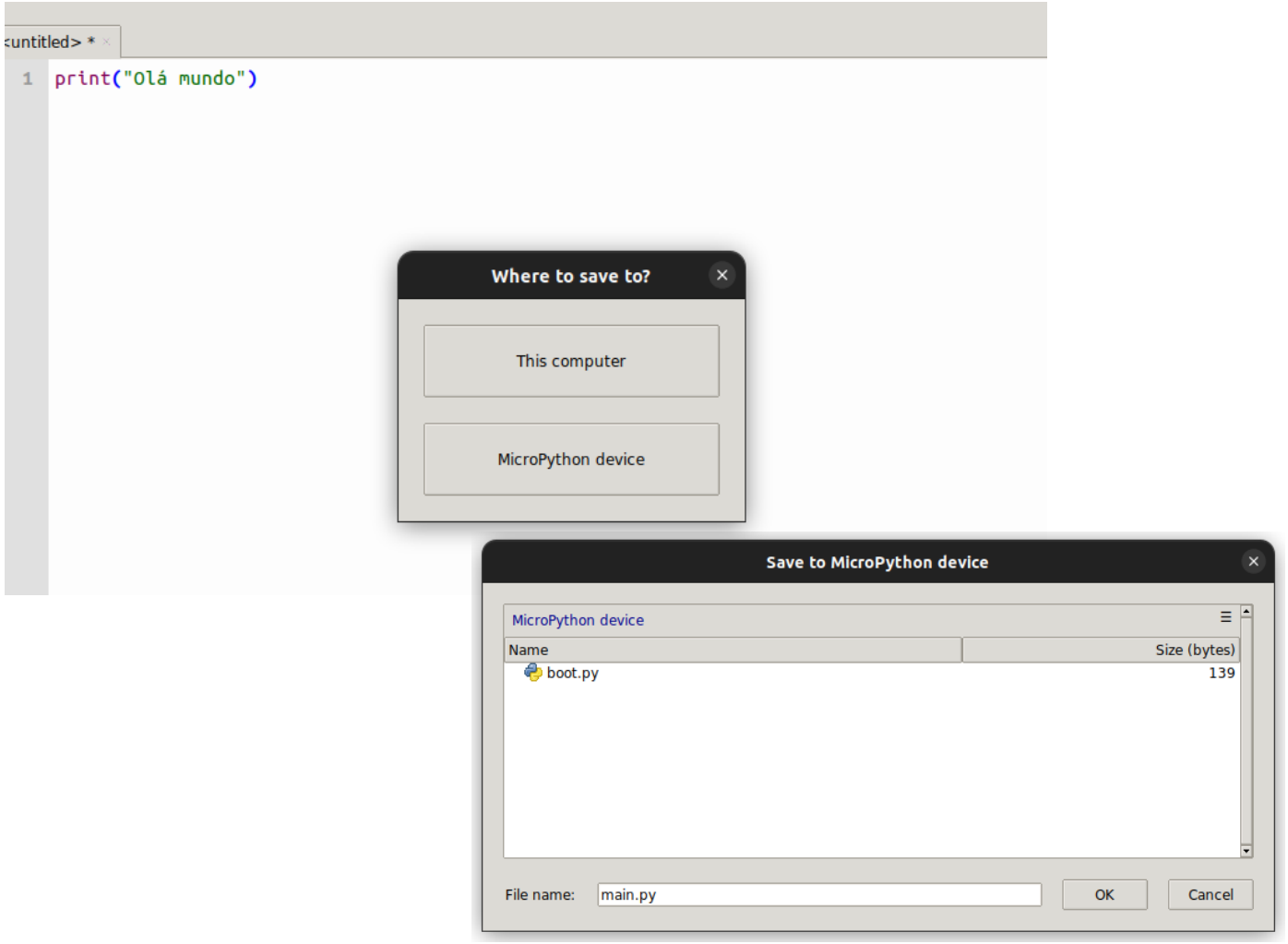
  print("Olá mundo!")



Inatel

THONNY IDE — EDITANDO ARQUIVOS

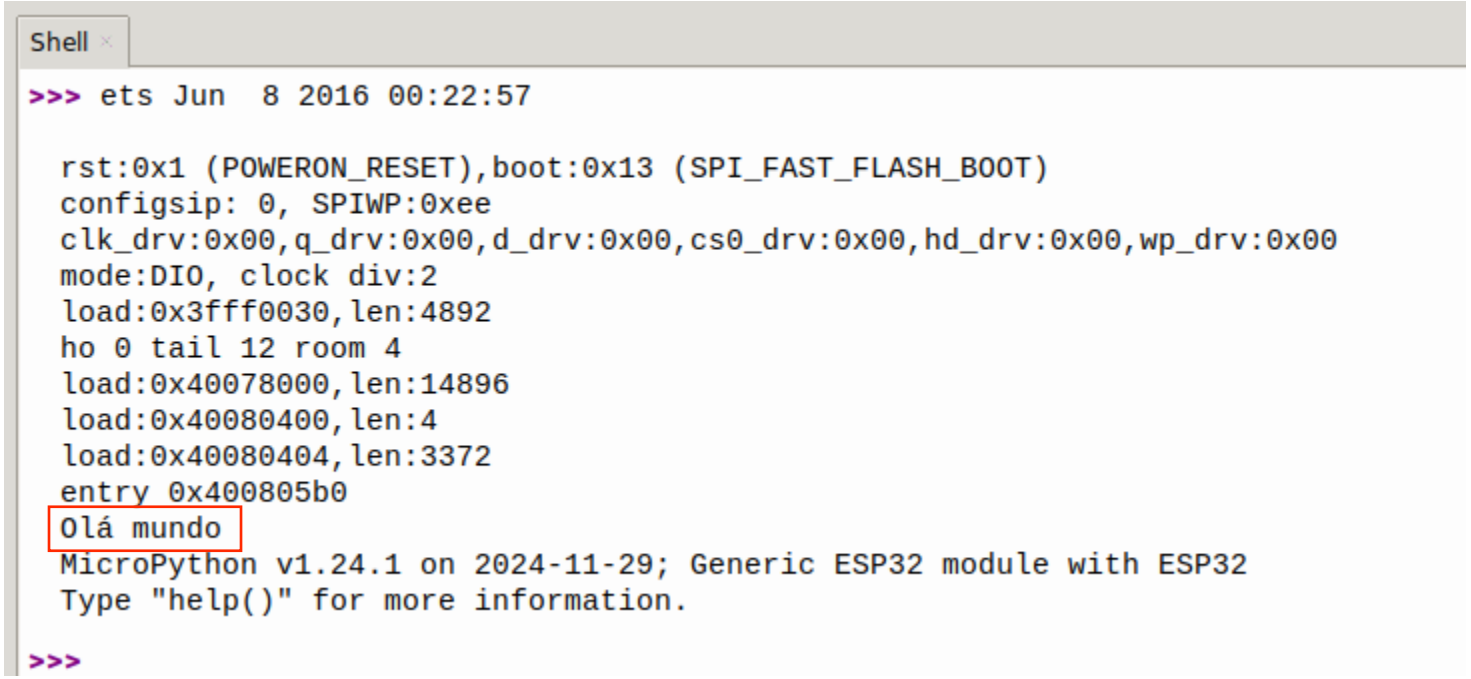
- Para salvar o arquivo no dispositivo, basta pressionar o botão "**Save**" ou o atalho **Ctrl + S**.
- Ao fazer isso, a IDE perguntara onde o arquivo deve ser salvo. Selecione "**MicroPython device**".
- Por fim, escolha o nome do arquivo (nesse caso **main.py**) e pressione "OK".



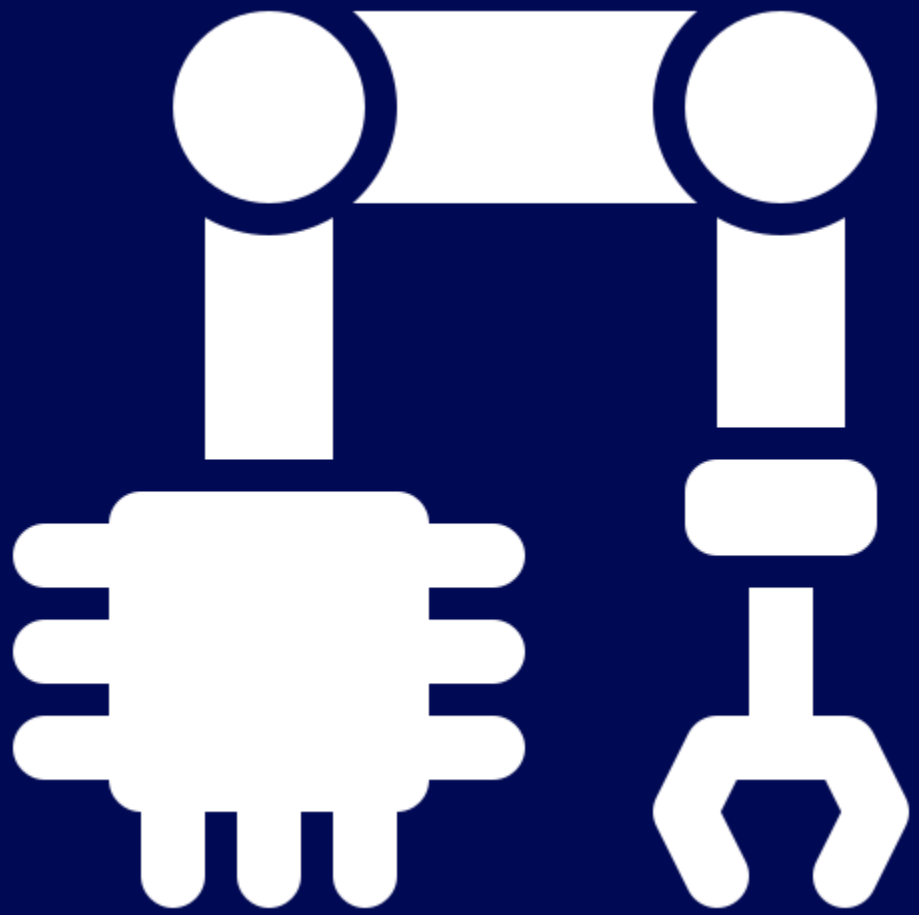
Inatel

THONNY IDE — EDITANDO ARQUIVOS

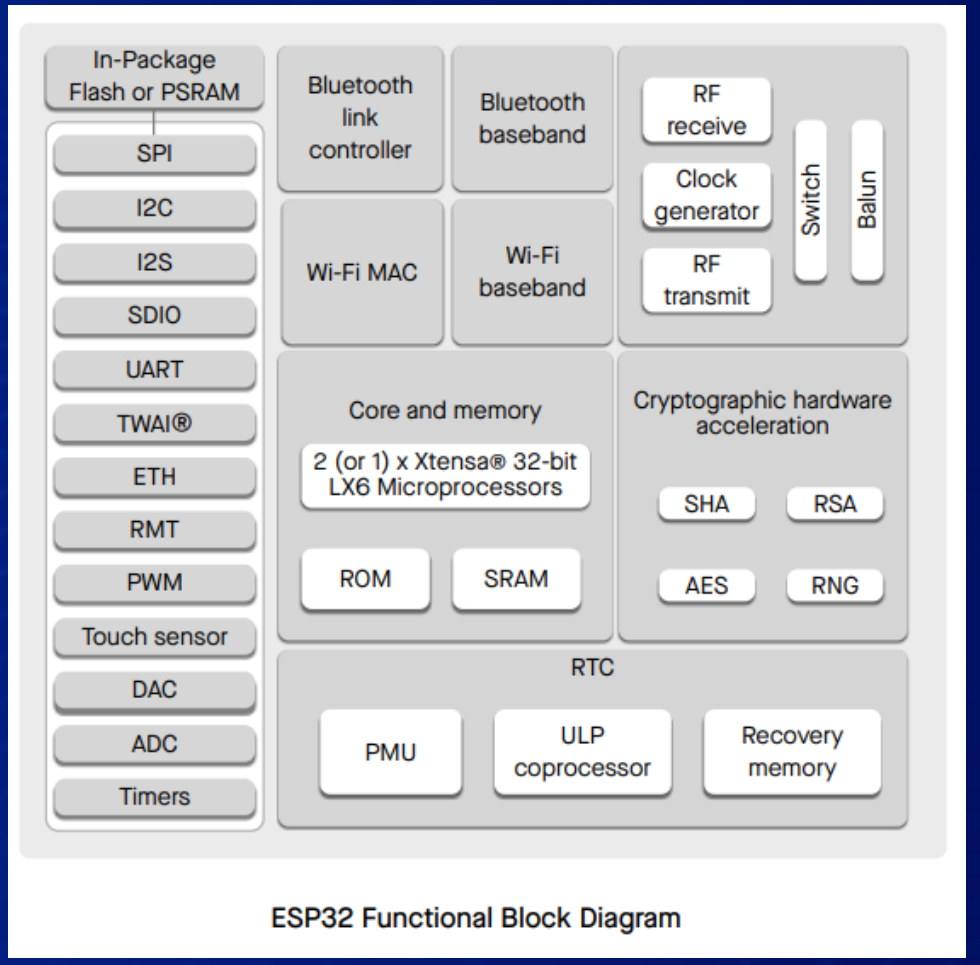
- Agora, ao resetar o dispositivo pode-se perceber que o script foi executado e então voltou pro modo interativo



Inatel



PERIFÉRICOS

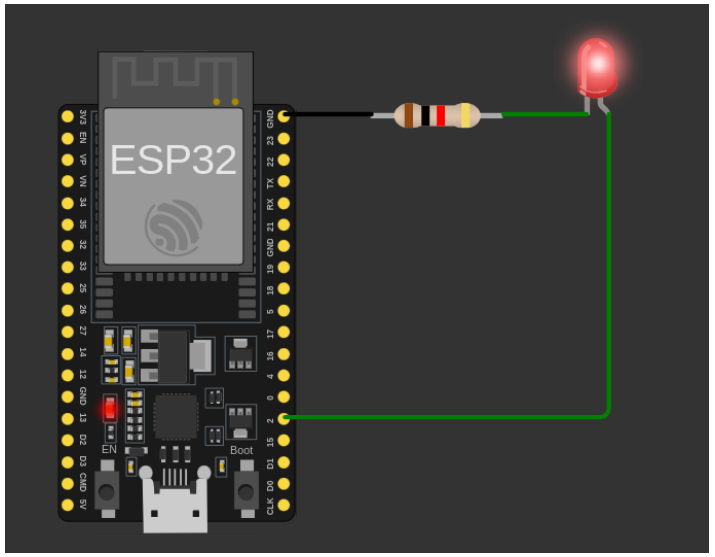


ESP32 Functional Block Diagram

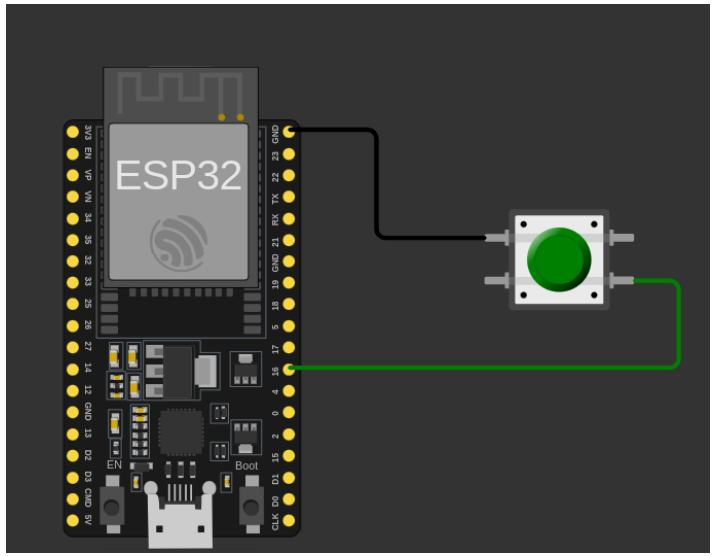
Inatel

GPIO

GPIO (General Purpose Input/Output), ou Entrada e Saída de Propósito Geral, são pinos presentes no microcontrolador que podem ser configurados para diferentes funções. Ou seja, esses pinos podem ser usados para ler informações (entrada) ou enviar sinais (saída). No ESP32, por exemplo, você pode usar os GPIOs para ligar e desligar LEDs, ler o estado de botões, controlar motores e até para se comunicar com outros dispositivos.



- GPIO2 como saída para um led.



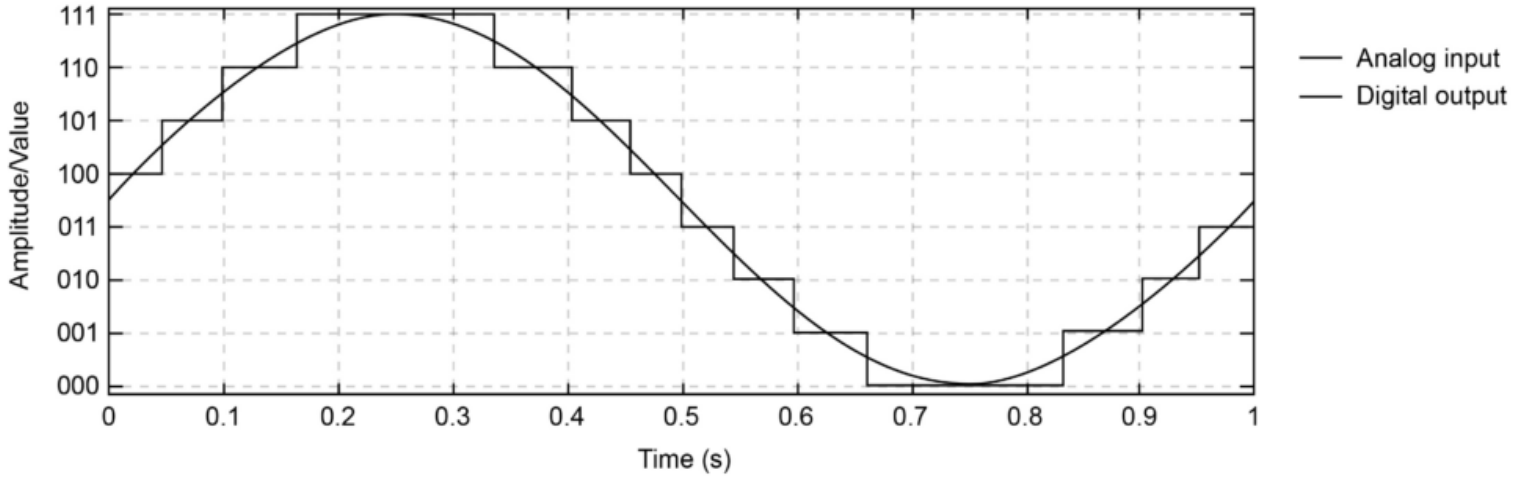
- GPIO16 como entrada para um botão.

ADC PWM I2C UART

Inatel

ADC

ADC (Analog-to-Digital Converter) de um microcontrolador é um periférico que converte sinais analógicos (tensão variável) em valores digitais, permitindo que o microcontrolador trabalhe com esses dados. O ESP32 possui **12 bits de resolução**, o que significa que o valor digital retornado pela conversão pode variar de **0 a 4095**. Esse valor representa a amplitude do sinal analógico em relação à referência de tensão (geralmente 3.3V, que é a tensão máxima que o ADC pode ler).



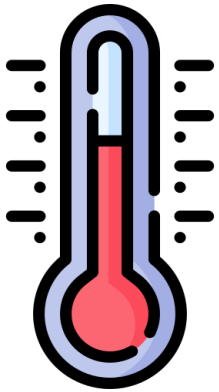
Resolução: Determina o número de valores que o ADC gera para representar uma tensão analógica.

Números de valores de ADC = (2^Resolução)

Exemplo:
Tensão de Entrada: (Valor Lido / (2^Resolução)-1) x Tensão de Referência

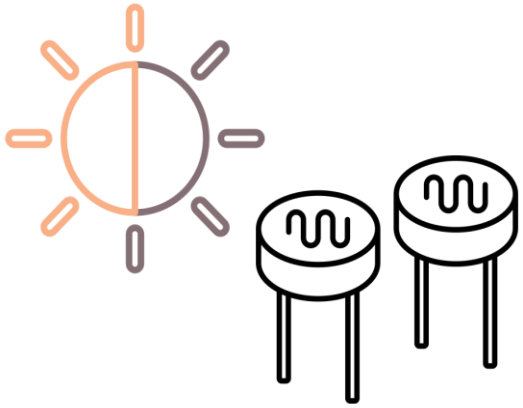
Inatel

APLICAÇÕES DE ADC



Sensor temperatura

Sensor ECG



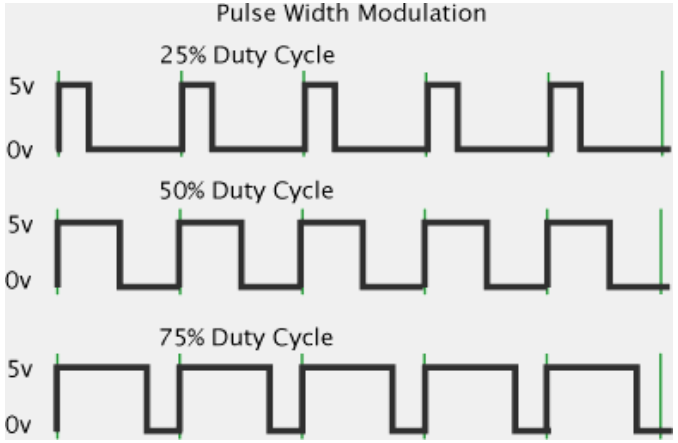
Sensor Luminosidade

Joystick

Inatel

PWM

PWM (**Modulação por Largura de Pulso**) é um método para **modular sinais analógicos** usando um **sinal digital de onda quadrada**. Ele controla a **potência efetiva entregue a um dispositivo** variando o tempo em que o sinal permanece no estado **ALTO (ON)** dentro de um período fixo.



Período(T):é o tempo total do ciclo do sinal PWM
Frequência(f): é o número de ciclos por segundos

Duty Cycle: Representa a porcentagem do tempo que o sinal fica no estado **ALTO** dentro de um período.
DC(%) = (Ton/T)*100
Valor PWM = DC*((2^Resolução)-1)

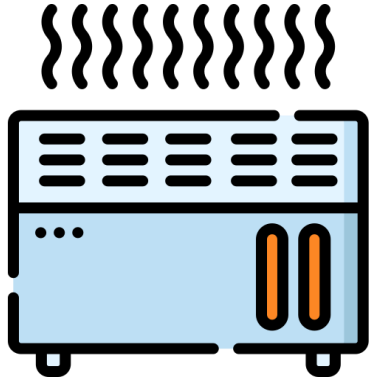
Exemplo:

Resolução = 16 bits
DC = 50% (ou seja, metade do tempo ligado e metade desligado)
valor PWM = $0.5*(2^{16} -1)$ -> 32767.5

Inatel

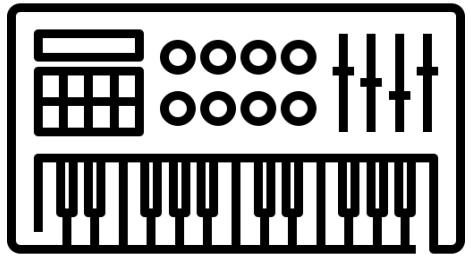
APLICAÇÕES DE PWM

Servo Motor



Sistema de aquecimento

Controle de luminosidade

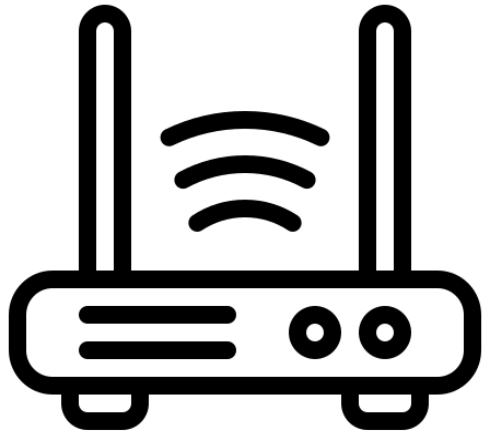


Instrumentos músicais

Inatel

WIFI

O **Wi-Fi** é um dos principais recursos do **ESP32**, permitindo conectividade sem fio para comunicação com servidores, troca de dados entre dispositivos e criação de redes locais. O ESP32 suporta os modos **STA (Station)**, **AP (Access Point)** e **AP+STA (Modo Duplo)**, oferecendo flexibilidade para diversas aplicações.

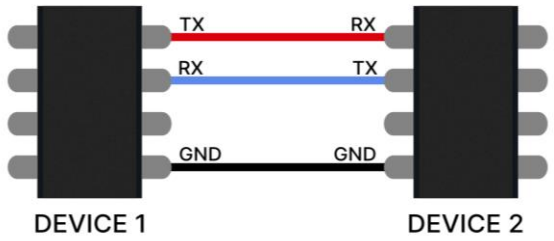


```
import network

wlan = network.WLAN(network.WLAN.IF_STA) # create station interface
wlan.active(True)       # activate the interface
wlan.scan()             # scan for access points
wlan.isconnected()      # check if the station is connected to an AP
wlan.connect('ssid', 'key') # connect to an AP
wlan.config('mac')      # get the interface's MAC address
wlan.ipconfig('addr4')  # get the interface ipv4
```

Inatel

COMUNICAÇÕES

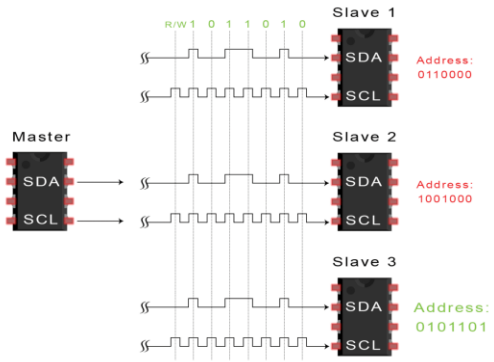


UART
(Transmissor/Receptor Assíncrono Universal)

Tx -> Rx e Rx -> TX

Comunicação entre dispositivos seriais.
Utilizado em comunicação de microcontroladores com modens LTE, Wifi, e Computadores de desenvolvimento (DEBUG). Além de ser utilizado como interface para Modbus e outras comunicações.

Velocidade máxima :
BaudRate 9600,15200bps

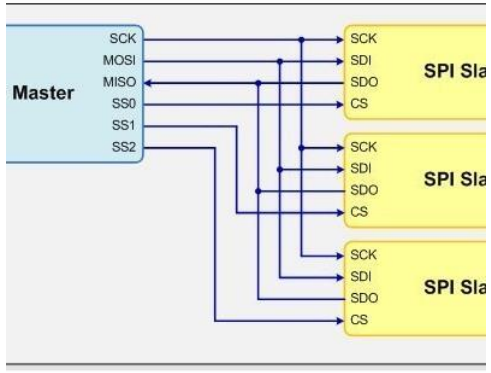


I2C (Inter-Integrated Circuit)

SDA -> Dados
SCL -> Clock

Comunicação Half-duplex entre dispositivo mestre e escravo via endereçamento. Comumente utilizado em sensores.

Velocidade máxima 5Mhz
Mais utilizada é 100khz



SPI (Serial Peripheral Interface)

MISO,MOSI, SCK, SS.

Comunicação Full-duplex entre dispositivo mestre e escravo comutado via pino de controle CS.
Utilizado em periféricos de alta velocidade como memórias, adc externo

Velocidade variável com dispositivo > 100Mbps

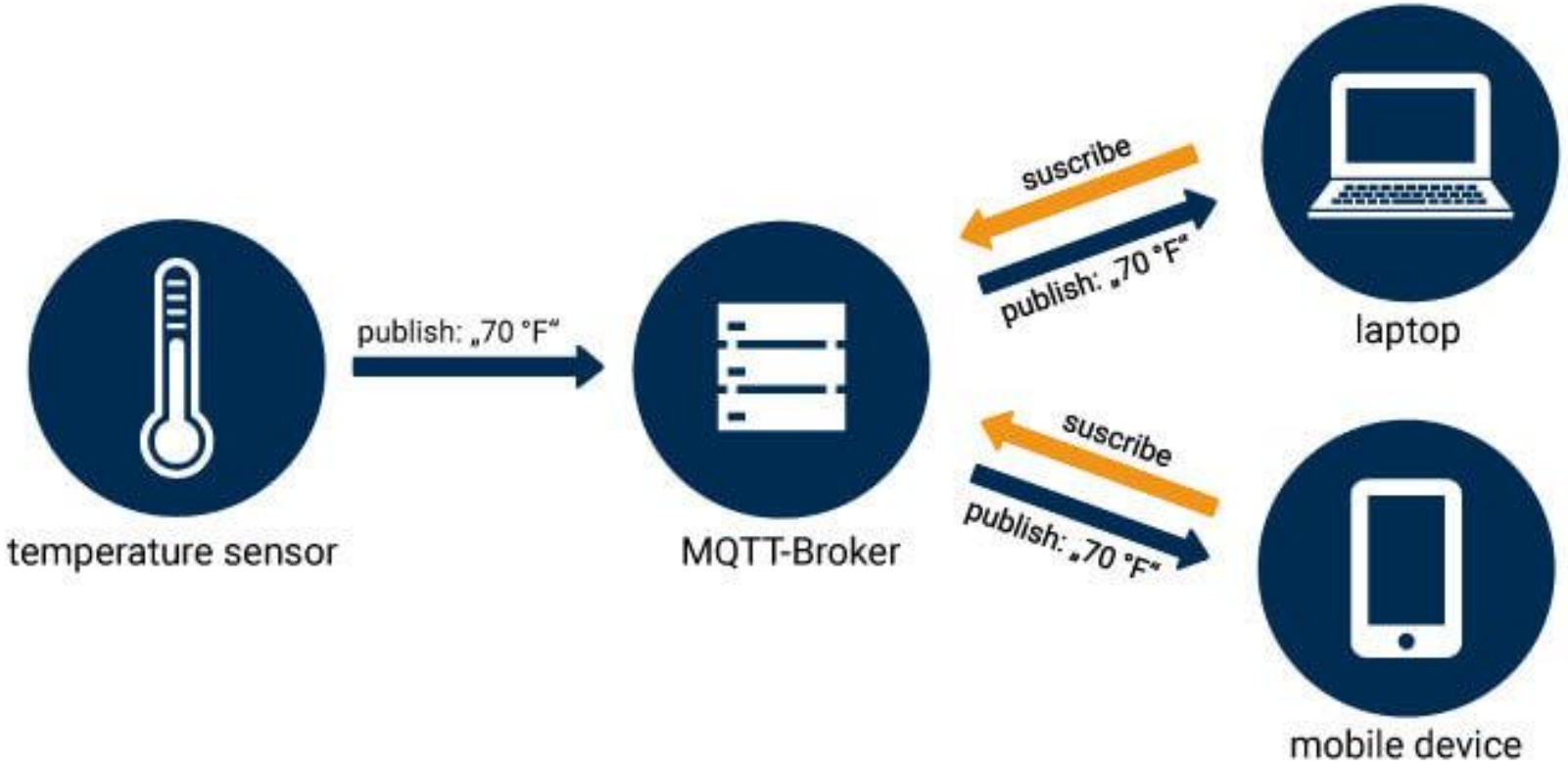
Inatel

MQTT
(MESSAGE QUEUING TELEMETRY TRANSPORT)

- Bem leve e eficiente
- Projetado para troca de mensagens com baixa largura de banda e alta latência
- Modelo Publicador/Assinante
- Um *Server* centralizado (Broker)
- *Clients* não se comunicam diretamente
  (no modelo clássico)

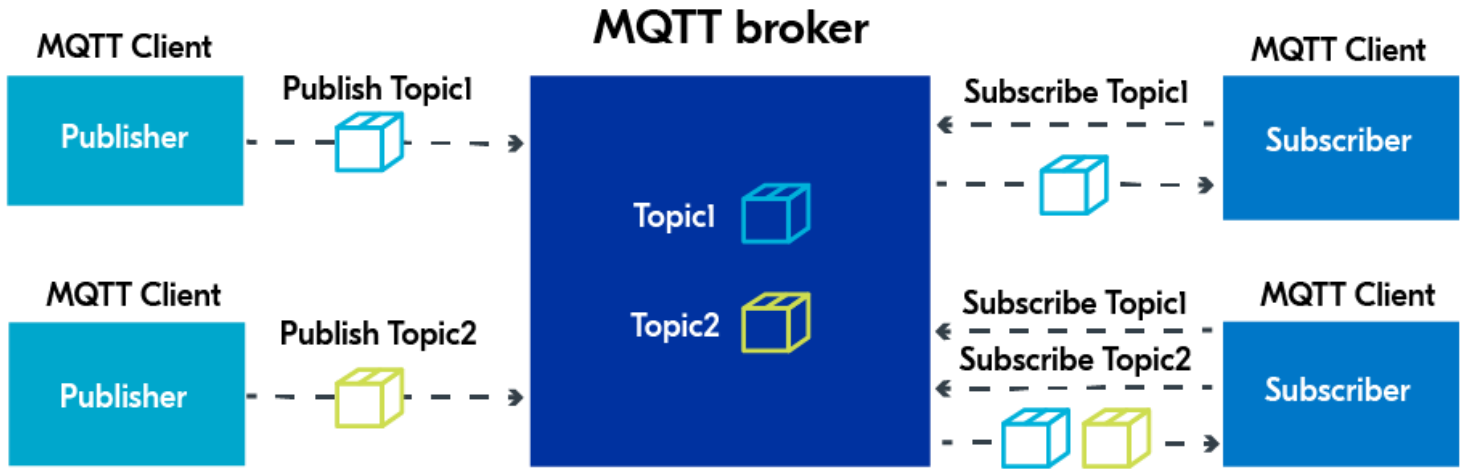
Inatel

MQTT
(MESSAGE QUEUING TELEMETRY TRANSPORT)



Inatel

MQTT
(MESSAGE QUEUING TELEMETRY TRANSPORT)



Inatel

MQTT

Conexão:

- Utiliza-se o ID do client e o endereço do broker

Publicação:

- Utiliza-se o tópico em que será publicado e a respectiva mensagem


```
from umqtt.simple import MQTTClient

MQTT_BROKER = mybroker.mqtt.io'
MQTT_CLIENT_ID = ''

client = MQTTClient(MQTT_CLIENT_ID, MQTT_BROKER)
client.connect()

client.publish('Topico', 'mensagem')
```

Inatel

MQTT

Inscrição:

- Define-se uma função de *callback*, isto é, uma função que será chamada quando receber uma informação.
- Um loop infinito deve ser definido para verificar se há mensagens recebidas


```
from umqtt.simple import MQTTClient

def message_callback(topic, msg):
    print(f'Message on topic {topic.decode()}')
    print(f'Message:{msg.decode()}')

# Define a funcao message_callback como funcao de callback de
mensagens
client.set_callback(message_callback)

# Faz a inscricao no topico
client.subscribe('meu_topico')

# Loop infinito
while True:
    # Espera uma mensagem
    if client.check_msg():
        print('Message received!')
```

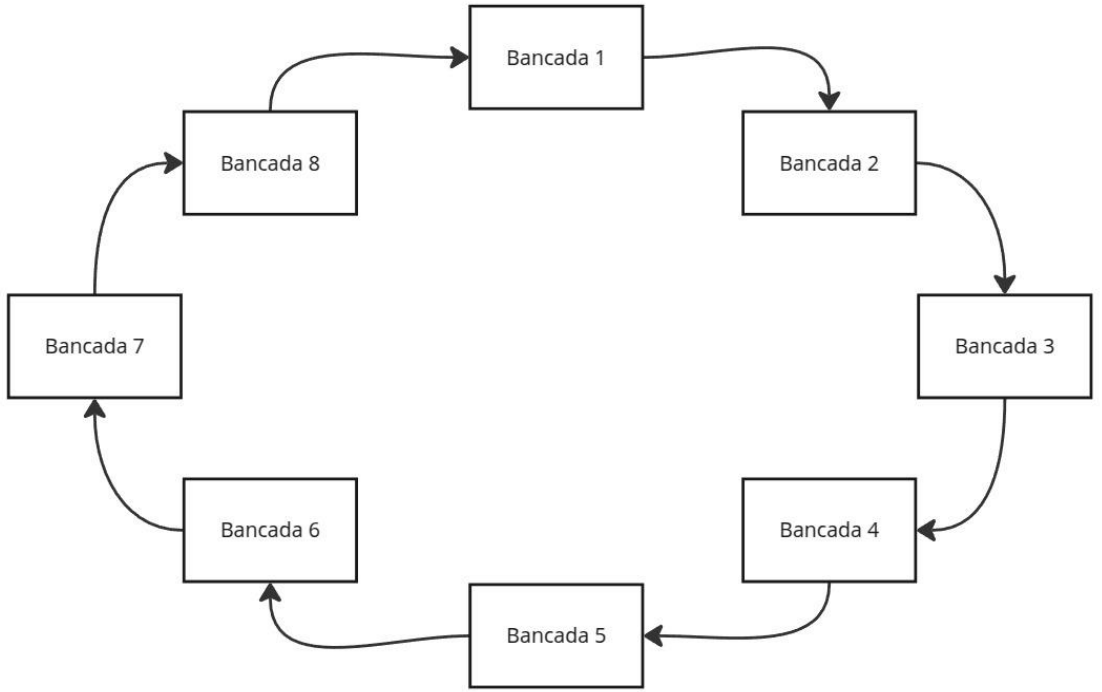
Inatel

EXPERIÊNCIA

Inatel

EXPERIÊNCIA

- A meta é fazer uma rede interativa, onde uma publicação na Bancada 1, ativará algo na Bancada 2, que por sua vez deverá ativar algo na Bancada 3, e assim por diante



Inatel

EXPERIÊNCIA

Bancada	Tópico de publicação	Mensagem	Evento de disparo da mensagem	Tópico de inscrição	Evento
1	"IWCAE2/1/botao"	"0" – botão solto "1" - botão pressionado	Mudança de estado do botão (pressionado ou solto)	"IWCAE2/8/botao"	O servomotor deve "acenar" enquanto o botão estiver pressionado (30° a 150°)
2	"IWCAE2/2/LDR"	Valor do LDR lido	A cada 1 segundo	"IWCAE2/1/botao"	Ligar o LED quando receber "1", desligar, caso contrário
3	"IWCAE2/3/botao"	"0" – botão solto "1" - botão pressionado	Mudança de estado do botão (pressionado ou solto)	"IWCAE2/2/LDR"	Acender um LED, caso o LDR detectar escuridão.
4	"IWCAE2/4/pot"	Valor lido do potenciômetro	A cada 1 segundo	"IWCAE2/3/botao"	Alternar o brilho do LED toda vez que o botão for pressionado (10% em 10%)
5	"IWCAE2/5/botao"	"0" – botão solto "1" - botão pressionado	Mudança de estado do botão (pressionado ou solto)	"IWCAE2/4/pot"	Se a posição do potenciometro for maior que 50%, o LED deve piscar na frequência de 1 Hz
6	"IWCAE2/6/LDR"	Valor do LDR lido	A cada 1 segundo	"IWCAE2/5/botao"	Ao pressionar o botão, o buzzer deve "Tocar uma musica"
7	"IWCAE2/7/pot"	Valor lido do potenciômetro	A cada 1 segundo	"IWCAE2/6/LDR"	Ao detectar um ambiente muito claro, piscar um LED.
8	"IWCAE2/8/botao"	"0" – botão solto "1" - botão pressionado	Mudança de estado do botão (pressionado ou solto)	"IWCAE2/7/pot"	Alterar a luminosidade de um LED proporcionalmente ao potenciômetro

Inatel

Carlos Eduardo da Silva Inácio
carlos.silva@inatel.br

Vinícius Lemos de Aguiar
vinicius.lemos@inatel.br

inatel.tecnologias
inatel.tecnologias
inateloficial
company/inatel
www.inatel.br

Campus em Santa Rita do Sapucaí
Minas Gerais - Brasil
Av. João de Camargo, 510
Centro - 37536-001

Inatel

_o futuro
não tem hora,
mas tem lugar.