Introdução ao Desenvolvimento de Software Embarcado, Utilizando Linguagem C, com Microcontroladores

Revisão da Unidade 3 | Capítulo 4
Unidade 4 | Capítulo 1 - Aula síncrona (06/01/2024)
Prof. Wilton Lacerda Silva

Executores:
INSTITUTO FEDERAL Piauí
INSTITUTO FEDERAL Rio Grande do Norte
INSTITUTO FEDERAL Maranhão
INSTITUTO FEDERAL Ceará
cepedi
INSTITUTO HARDWARE BR

Coordenação: Softex
Iniciativa: MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO
GOVERNO FEDERAL BRASIL UNIÃO E RECONSTRUÇÃO

embarca tech
Residência Tecnológica em Sistemas Embarcados

Jornada até aqui...

Executores:

INSTITUTO FEDERAL Piauí
INSTITUTO FEDERAL Rio Grande do Norte
INSTITUTO FEDERAL Maranhão
INSTITUTO FEDERAL Ceará
cepedi
INSTITUTO HARDWARE BR

Coordenação: Softex

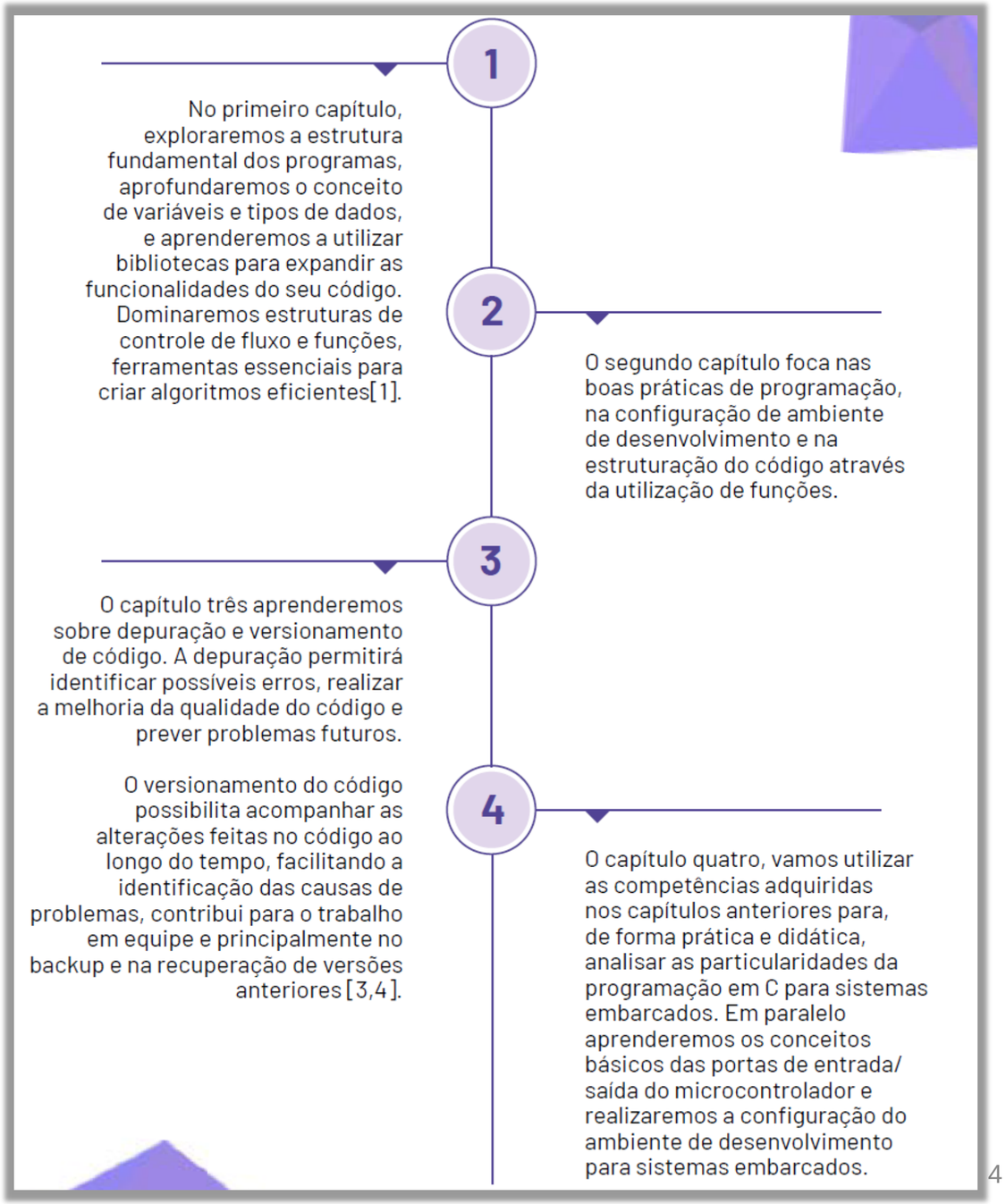
Iniciativa: MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO

GOVERNO FEDERAL BRASIL UNIÃO E RECONSTRUÇÃO

Sumário

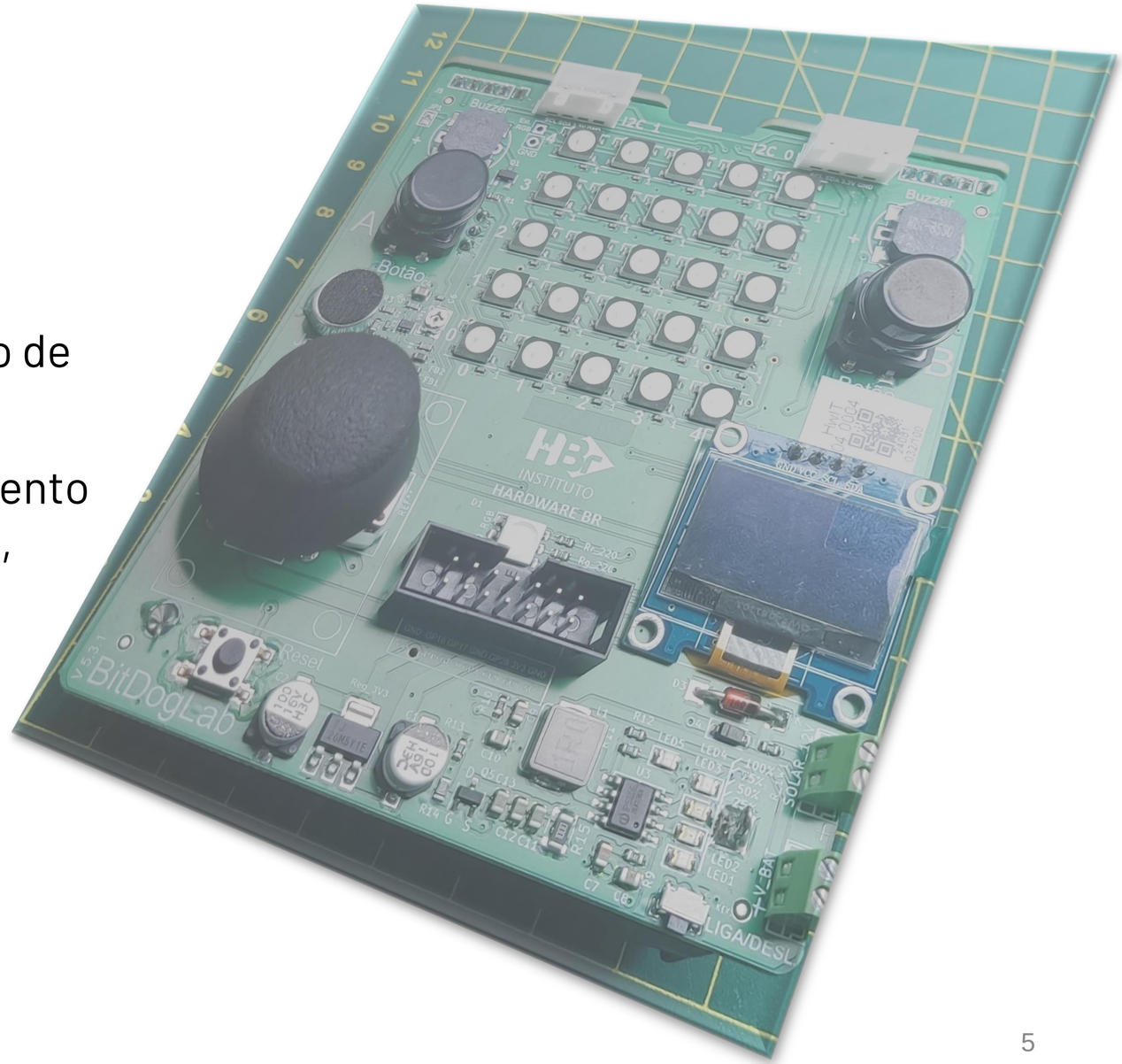
Pré-requisitos

- *Conhecimento básico de programação C, incluindo controle de fluxo, loops e funções.*
- *Noções básicas de microcontroladores.*
- *Familiaridade com a IDE do VS Code para programação em C.*
- *Ter assistido a videoaula do Capítulo 1 da Unidade 4.*



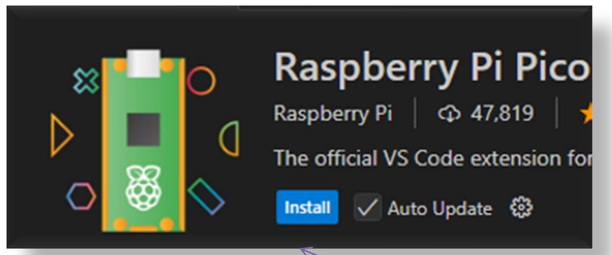
Objetivos

- Configurar a IDE para a utilização do RP2040.
- Entender as características do desenvolvimento de software em microcontroladores.
- Conhecer e caracterizar a placa de desenvolvimento BitDogLab e seus componentes principais: CPU, memória, interfaces de entrada/ saída.
- Compreender a organização das bibliotecas de desenvolvimento do Raspberry Pi Pico.



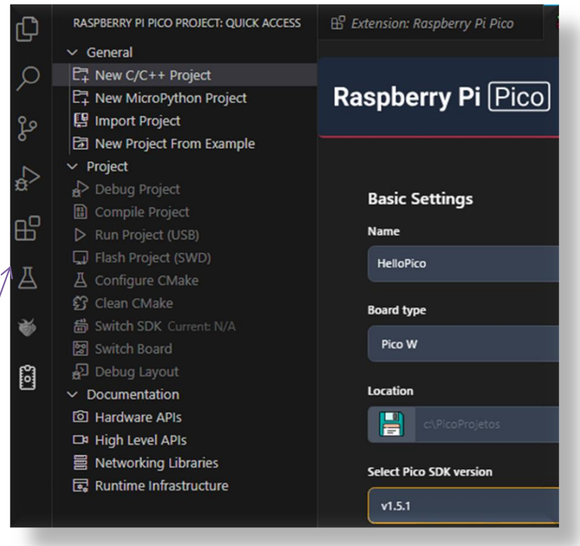
Configuração do VS Code para Raspberry Pi pico “RP2040”

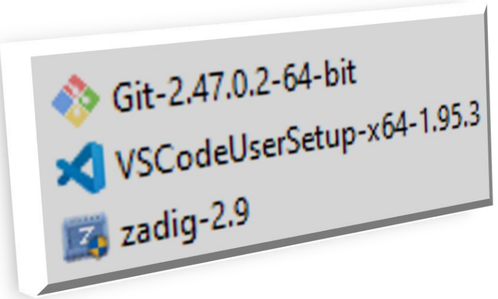
Aqui é considerado que nada está instalado no PC.



Neste processo o download de diversas ferramentas ocorrerão automaticamente.

O Pico SDK, desenvolvido pela Raspberry Pi, possui várias versões lançadas ao longo do tempo. Por exemplo, a versão **2.0.0**, lançada em 8 de agosto de 2024, introduziu suporte ao novo microcontrolador RP2350 e à compilação de código RISC-V, além de diversas melhorias e novas bibliotecas.

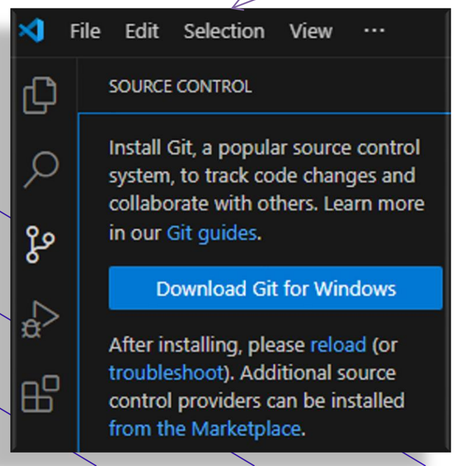




https://code.visualstudio.com

Instalação do VS Code → Instalação do Git → Instalar extensão: Raspberry Pi Pico → Instalação do SDK. Criar um novo Projeto (New C/C++ Project) → Compilar e Carregar o UF2 no RP2 → RUN Requer o USB driver ZADIG.

https://zadig.akeo.ie

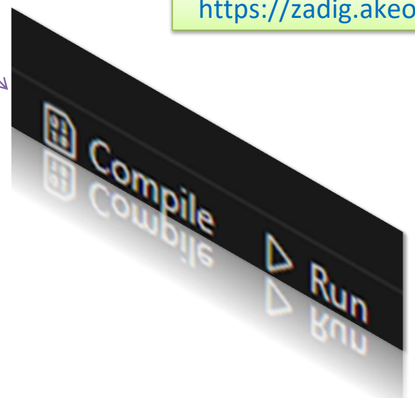


https://github.com/raspberrypi/pico-sdk

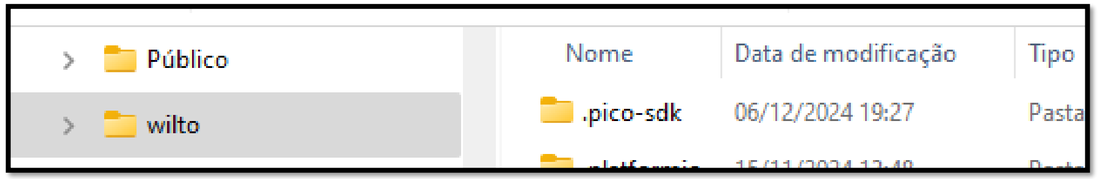
Serão instaladas 16 extensões, tais como:

Raspberry Pi Pico; Serial Monitor; RTOS Views; Python; Peripheral Viewer; MicroPico; MemoryView; debug-tracker-vscode; Cortex-Debug; Cmake Tools; Cmake; C/C++ Themes; C/C++ Extension Pack; C/C++

Carregar o arquivo UF2 para o RP2.
Para tanto é necessário colocar o RP2 no modo BOOTSEL.

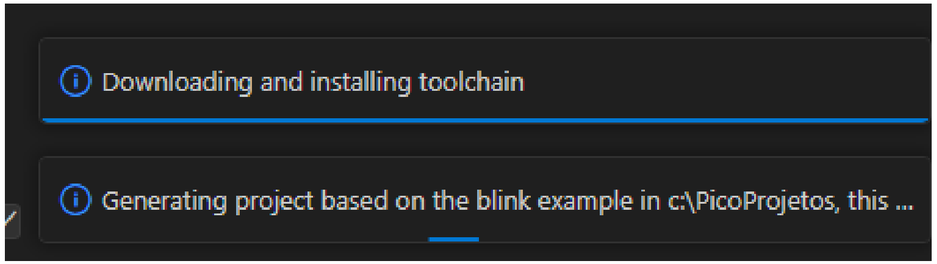


Configuração do VS Code para RP2

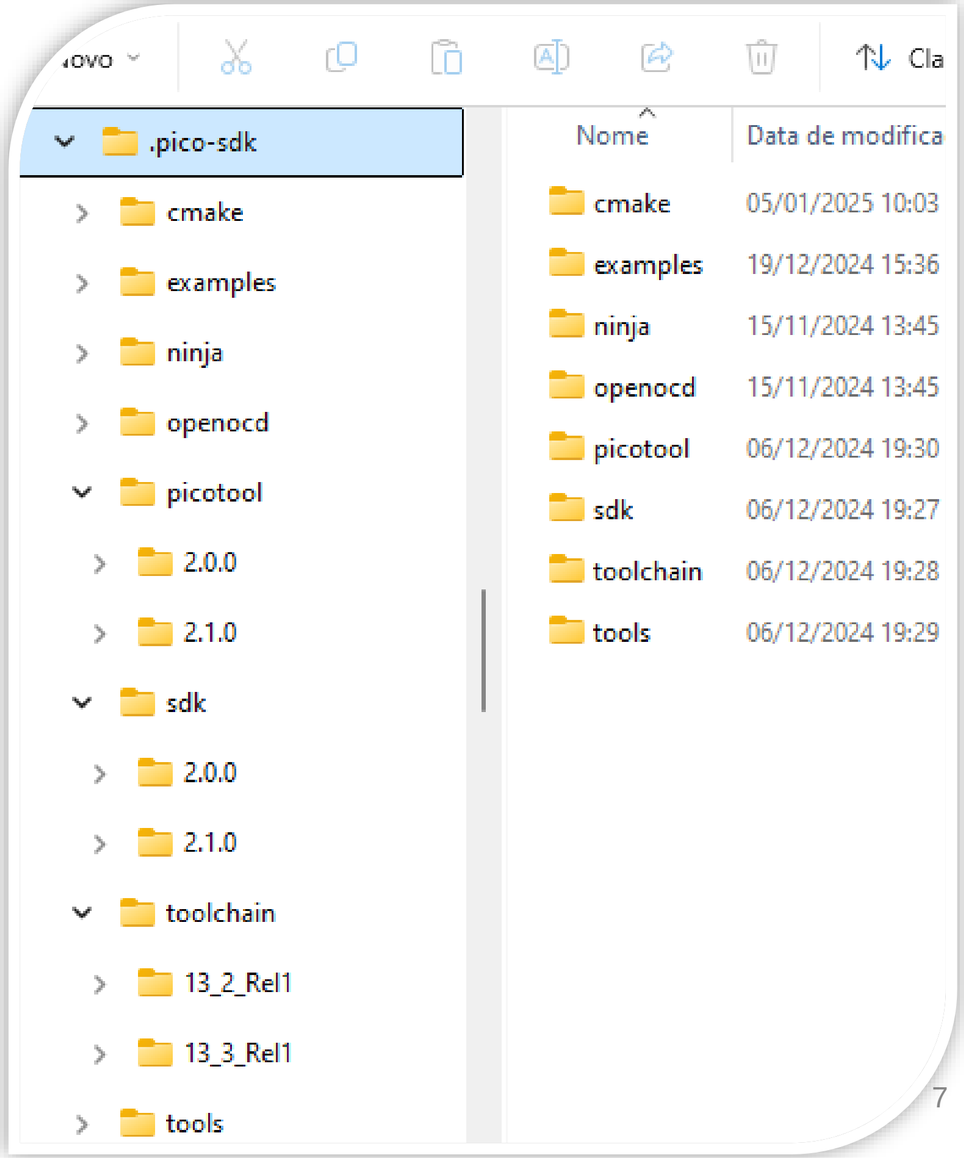


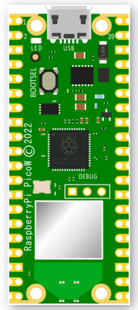
Apresentação da extensão “Raspberry Pi Pico” no VS Code

Solicitar a criação de um novo projeto. (SDK instalado?)



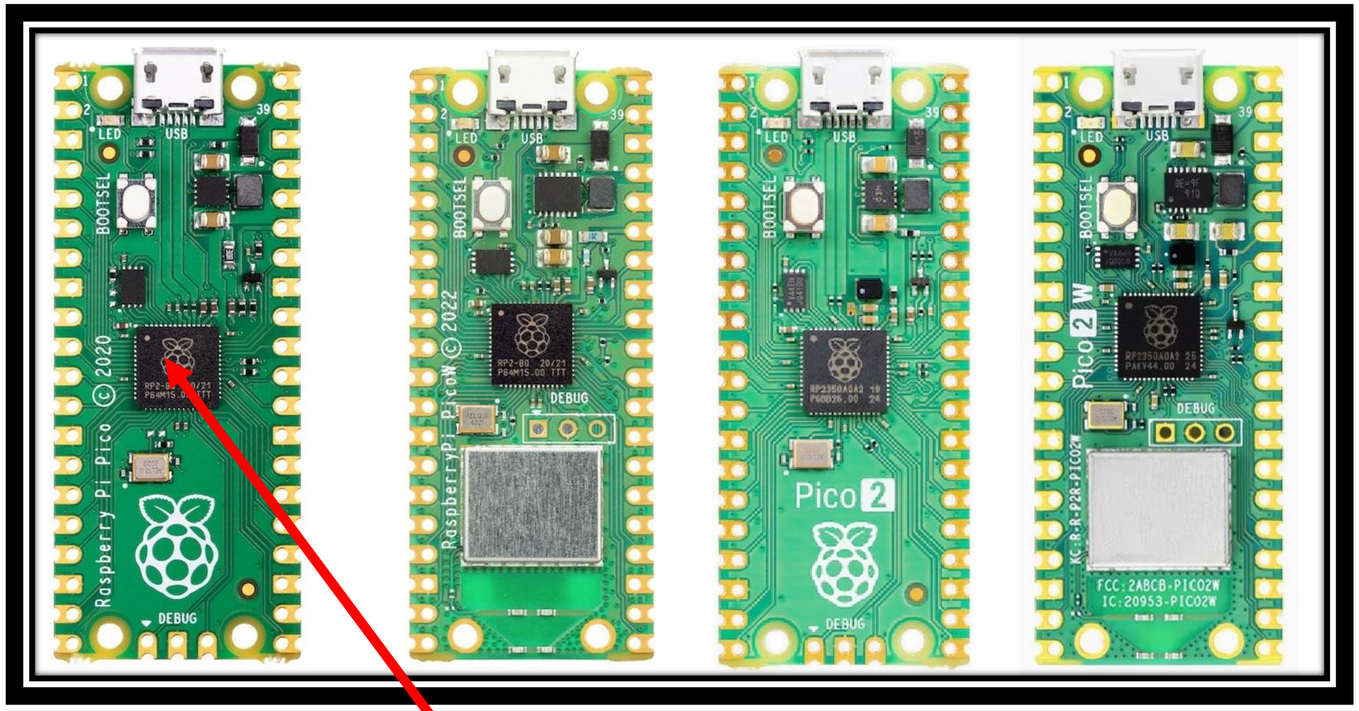
Conferir a extensão instalada...

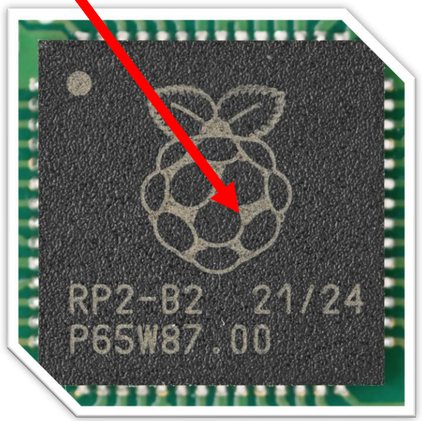




Placa de desenvolvimento Raspberry Pi Pico

A **Raspberry Pi Pico** é uma placa de desenvolvimento de microcontrolador criada pela **Raspberry Pi Foundation**. Projetado especialmente para tarefas embarcadas e IoT (Internet das Coisas).





Microcontroladores

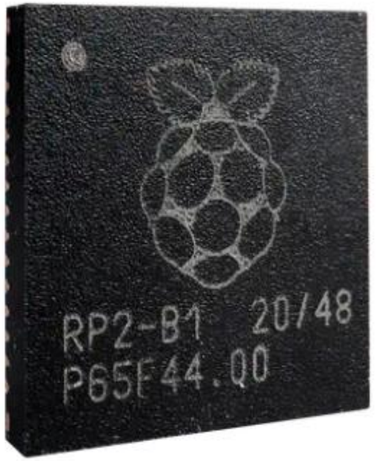
RP2040

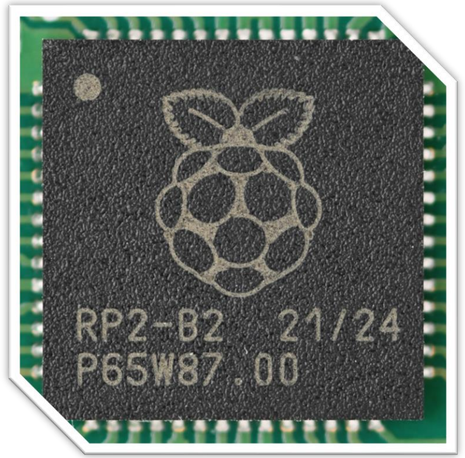
- Raspberry Pi Pico
- Raspberry Pi Pico W

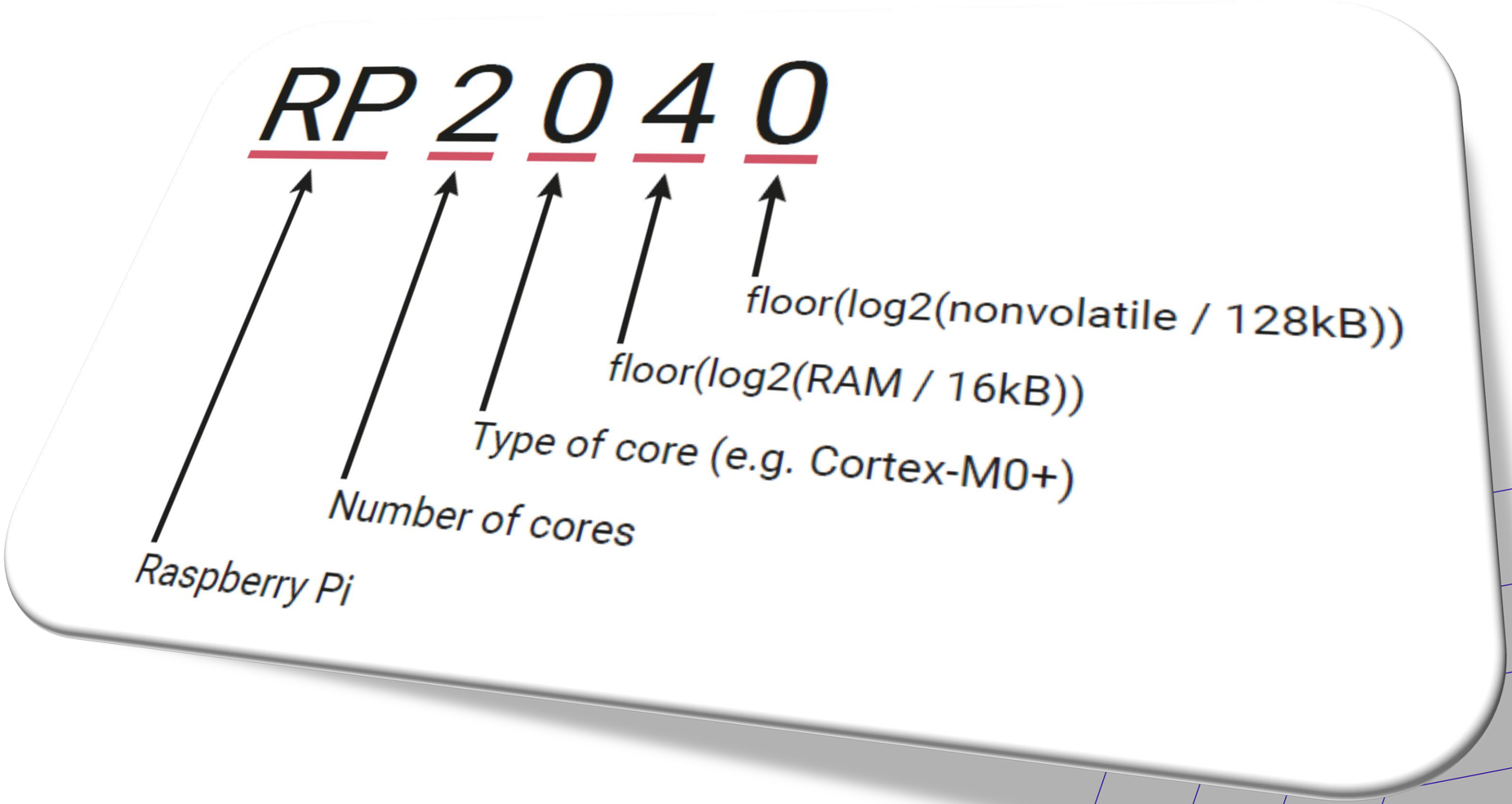
RP2350

- Raspberry Pi Pico 2
- Raspberry Pi Pico 2 W

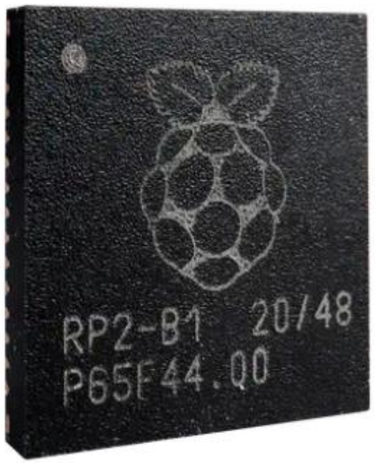
Microcontrolador RP2040

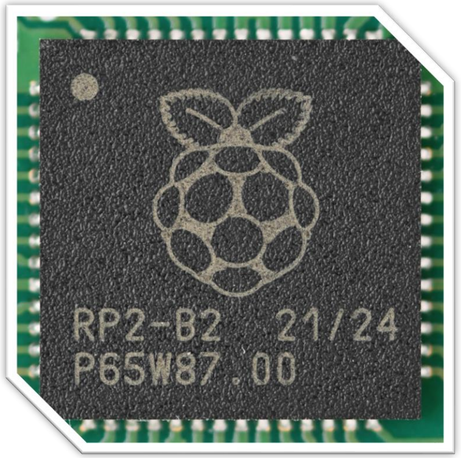


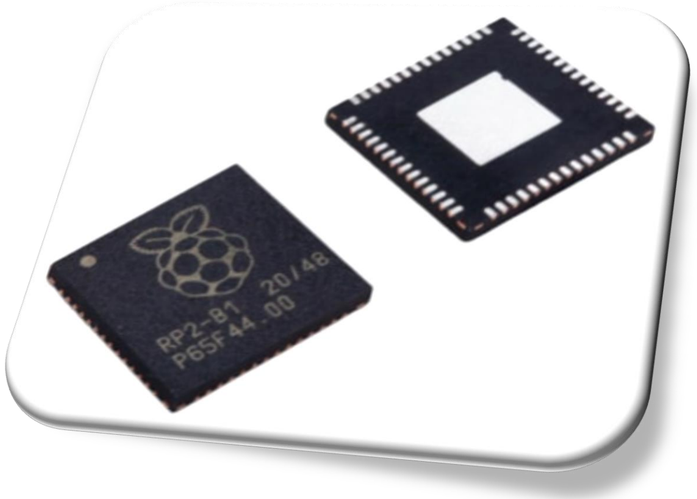




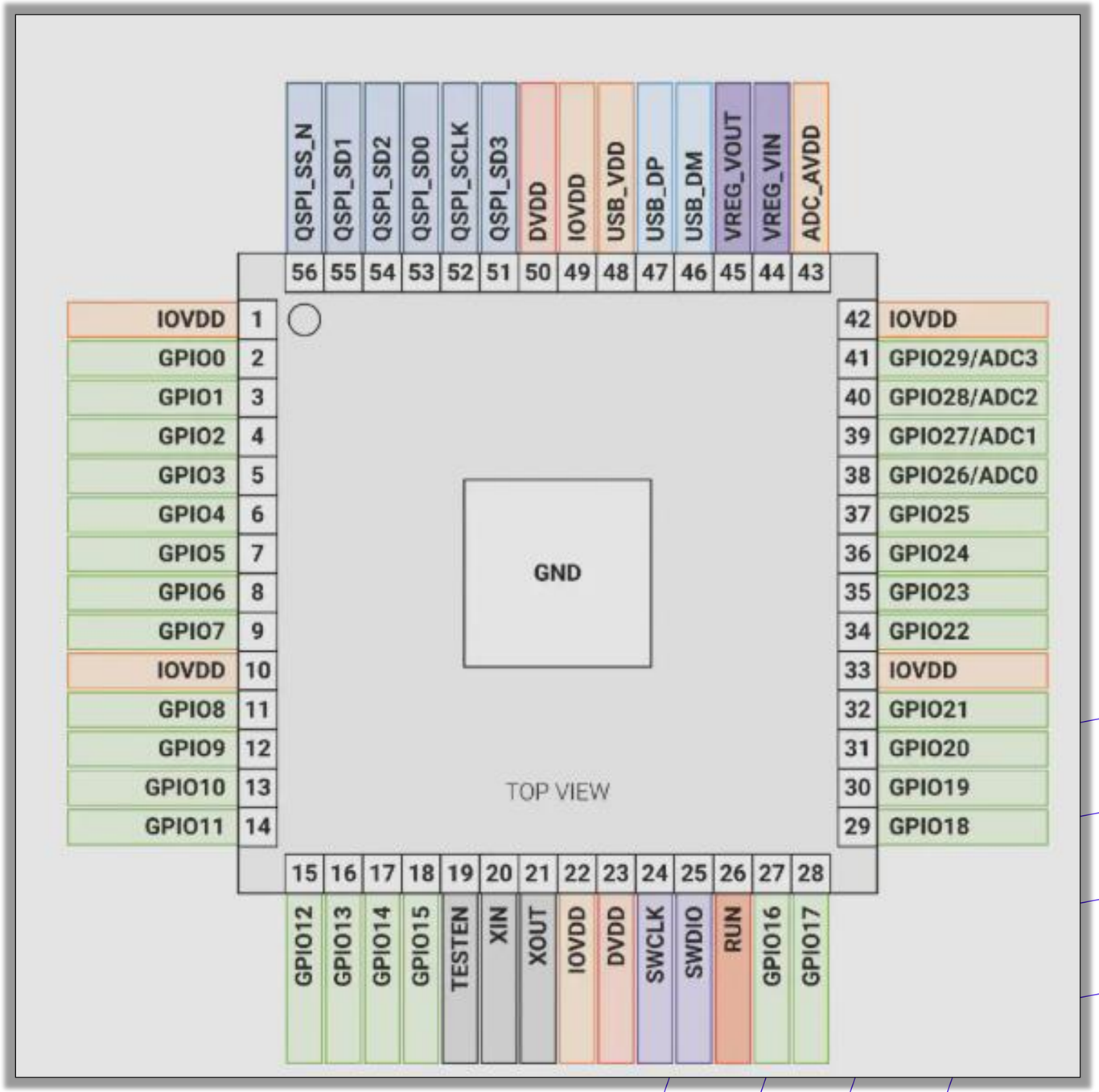
RP2040



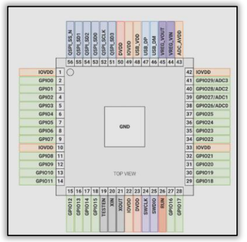




- 30 GPIOs
- 4 ADCs (3 disponíveis no KIT RP2 W)

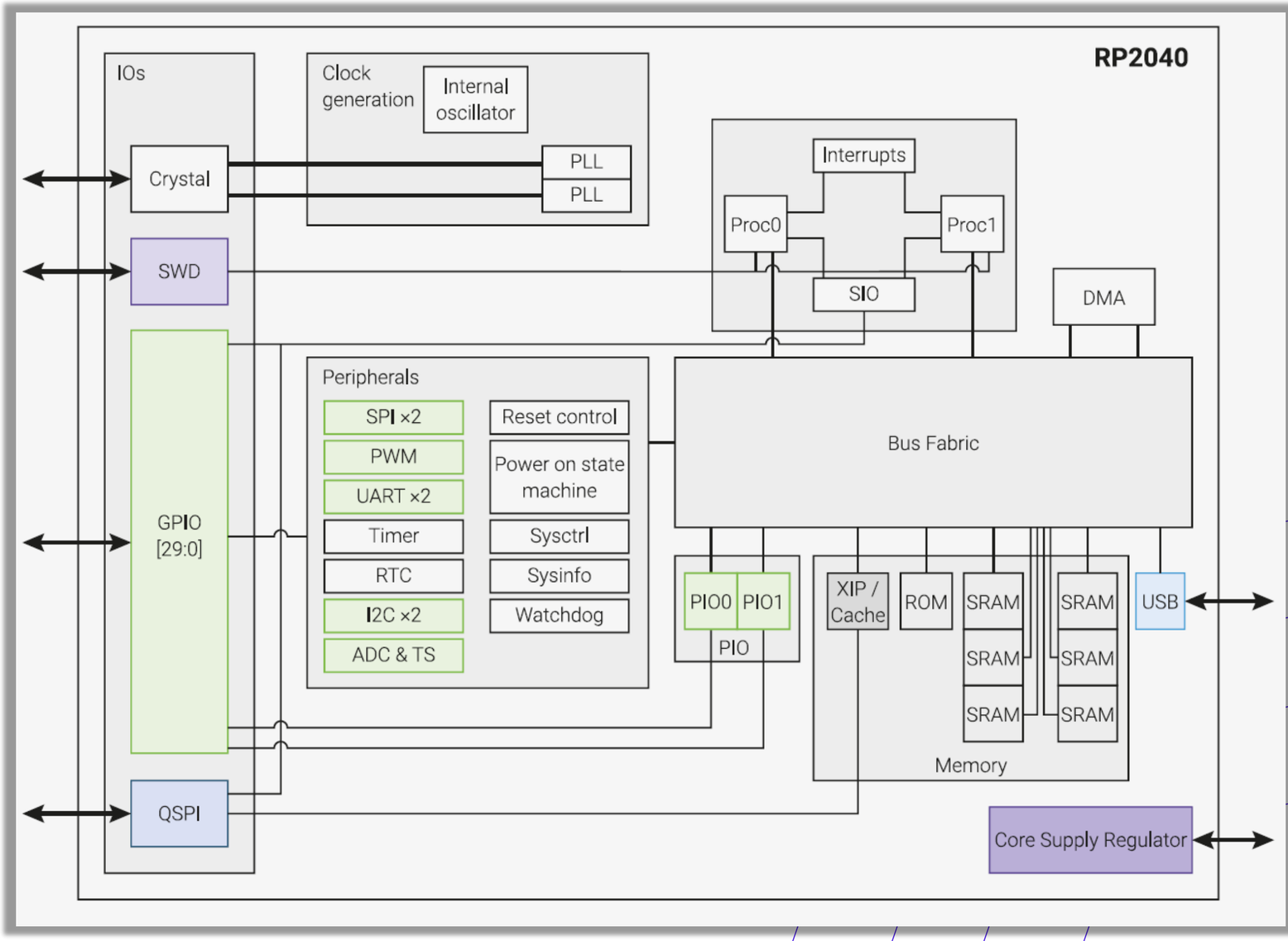


Microcontrolador RP2040

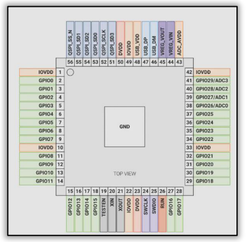


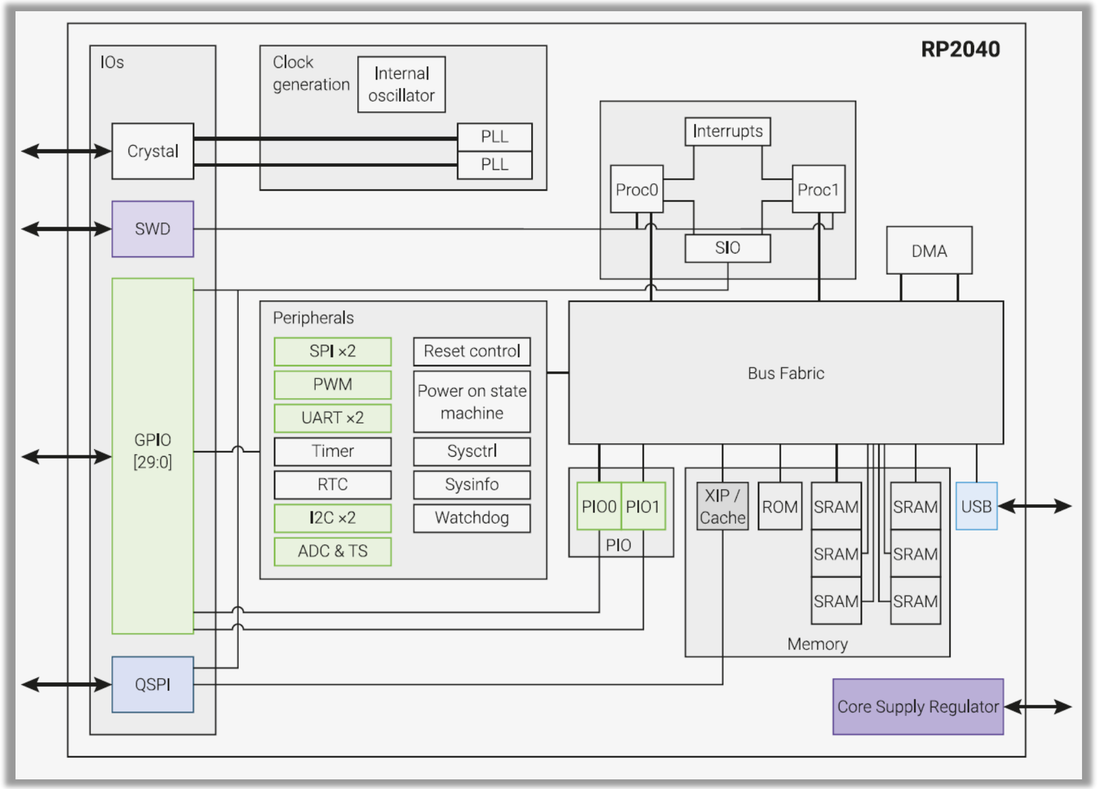
Características principais:

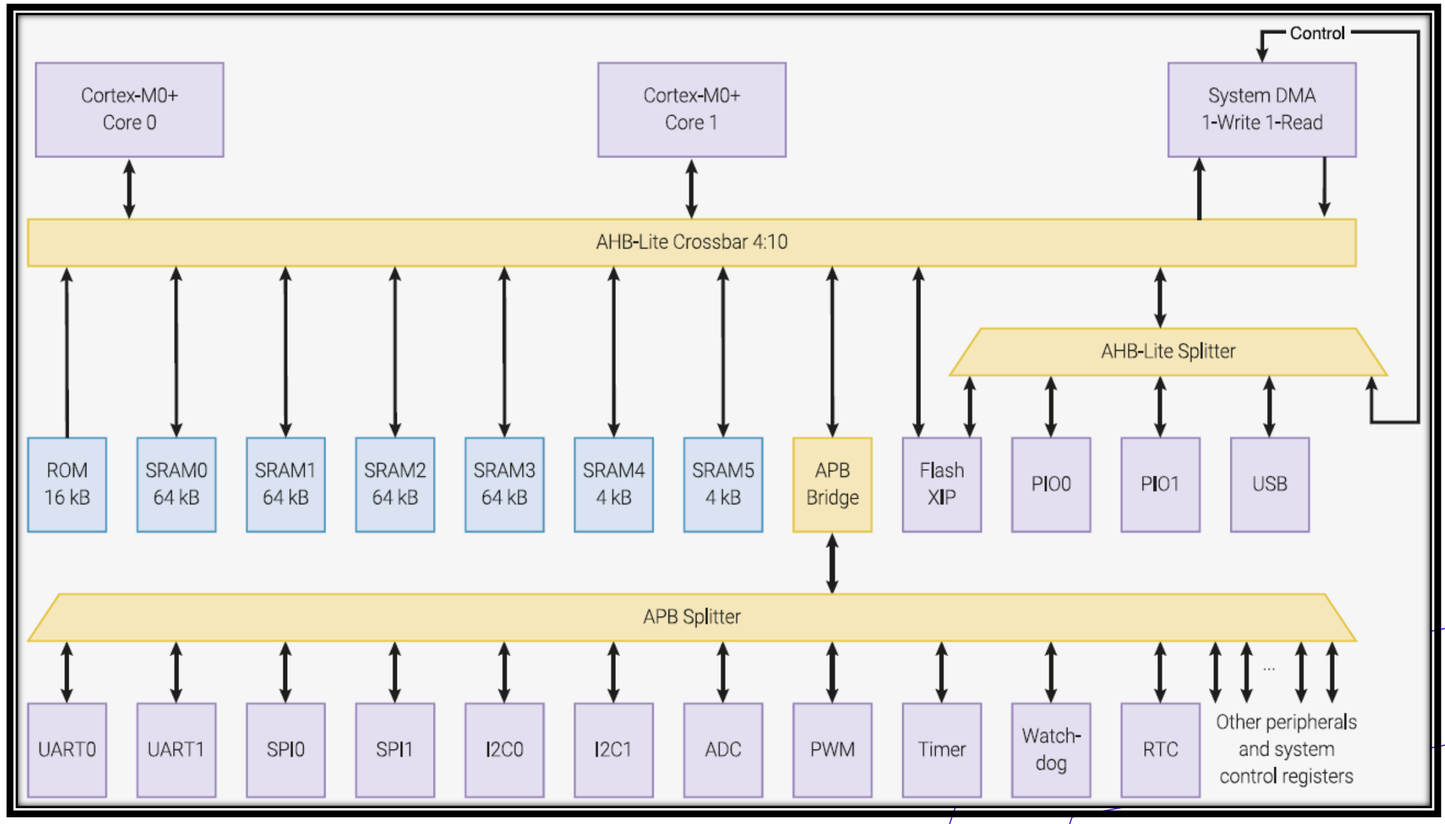
- Dual Cortex M0+ processors, up to 133MHz
- 264kB of embedded SRAM in 6 banks (4x64 + 2x4)
- Support for up to 16MB of off-chip Flash memory via dedicated QSPI bus
- 30 GPIO pins, 4 of which can be used as analogue inputs
- 6 dedicated I/O for SPI flash (supporting XIP)
- Dedicated hardware for commonly used peripherals
- Programmable I/O for extended peripheral support
- 4 channel ADC with internal temperature sensor, 500ksps, 12-bit conversion
- Peripherals
  - 2 UARTs
  - 2 SPI controllers
  - 2 I2C controllers
  - 16 PWM channels
  - USB 1.1 controller and PHY, with host and device support
  - 8 PIO state machines



Microcontrolador RP2040



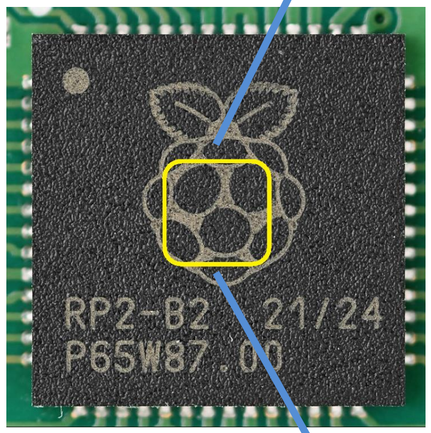


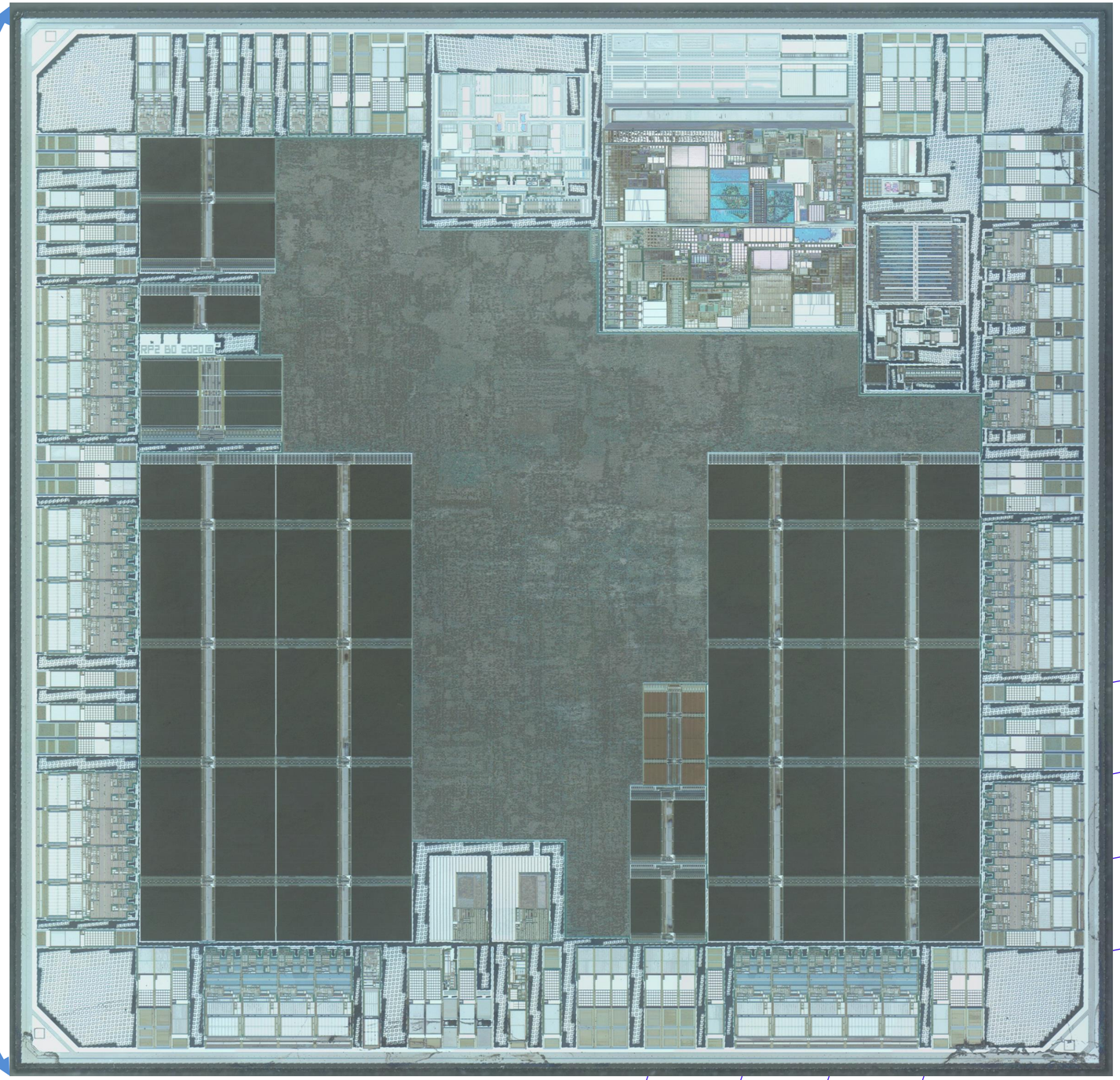


Representações esquemáticas da estrutura interna do RP2040

Die do RP2040

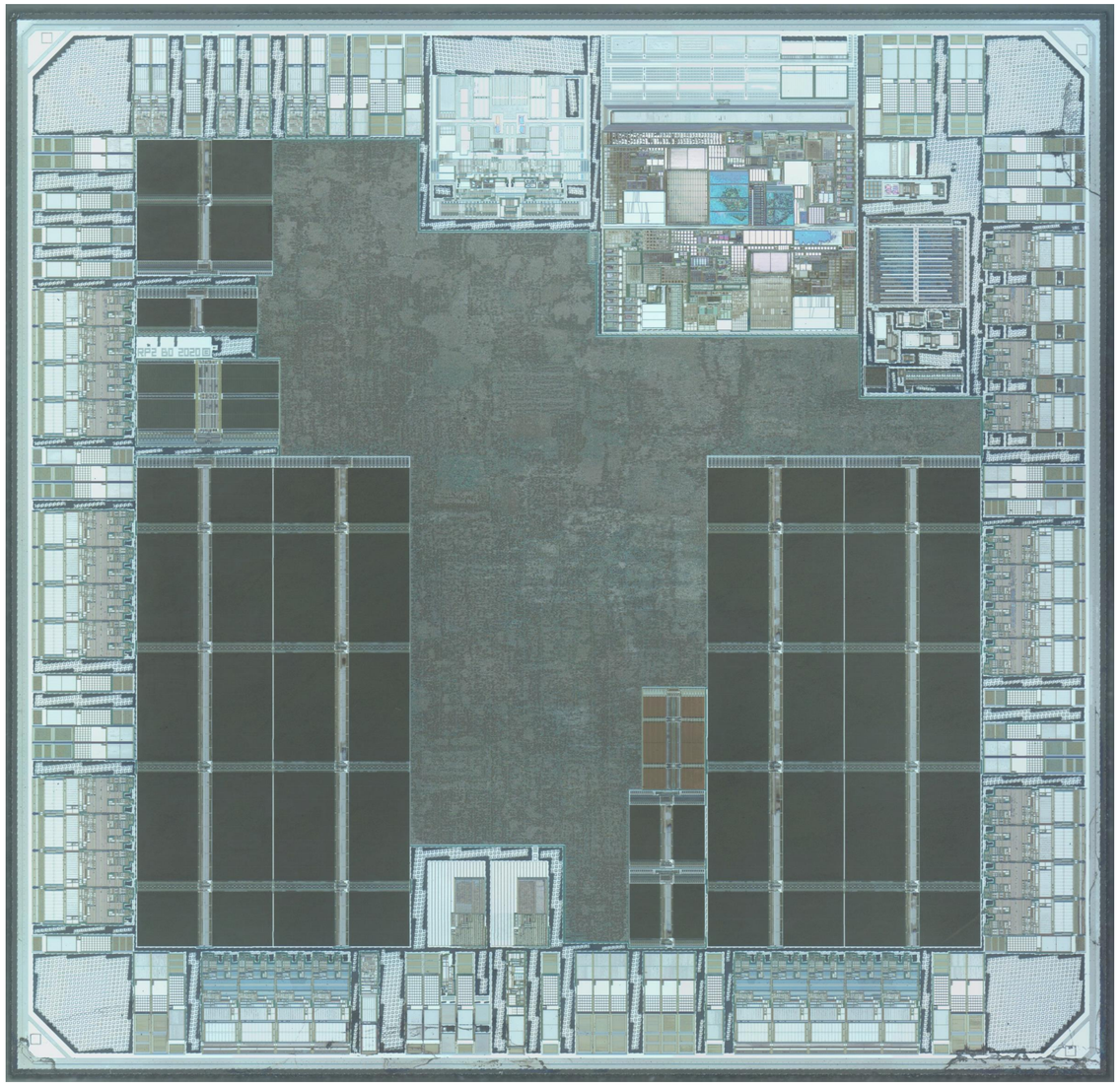
Embora o **die** seja a parte funcional principal, ele é protegido por um encapsulamento de plástico ou cerâmica que o conecta ao mundo exterior através de pinos ou pads (as GPIOs, por exemplo).

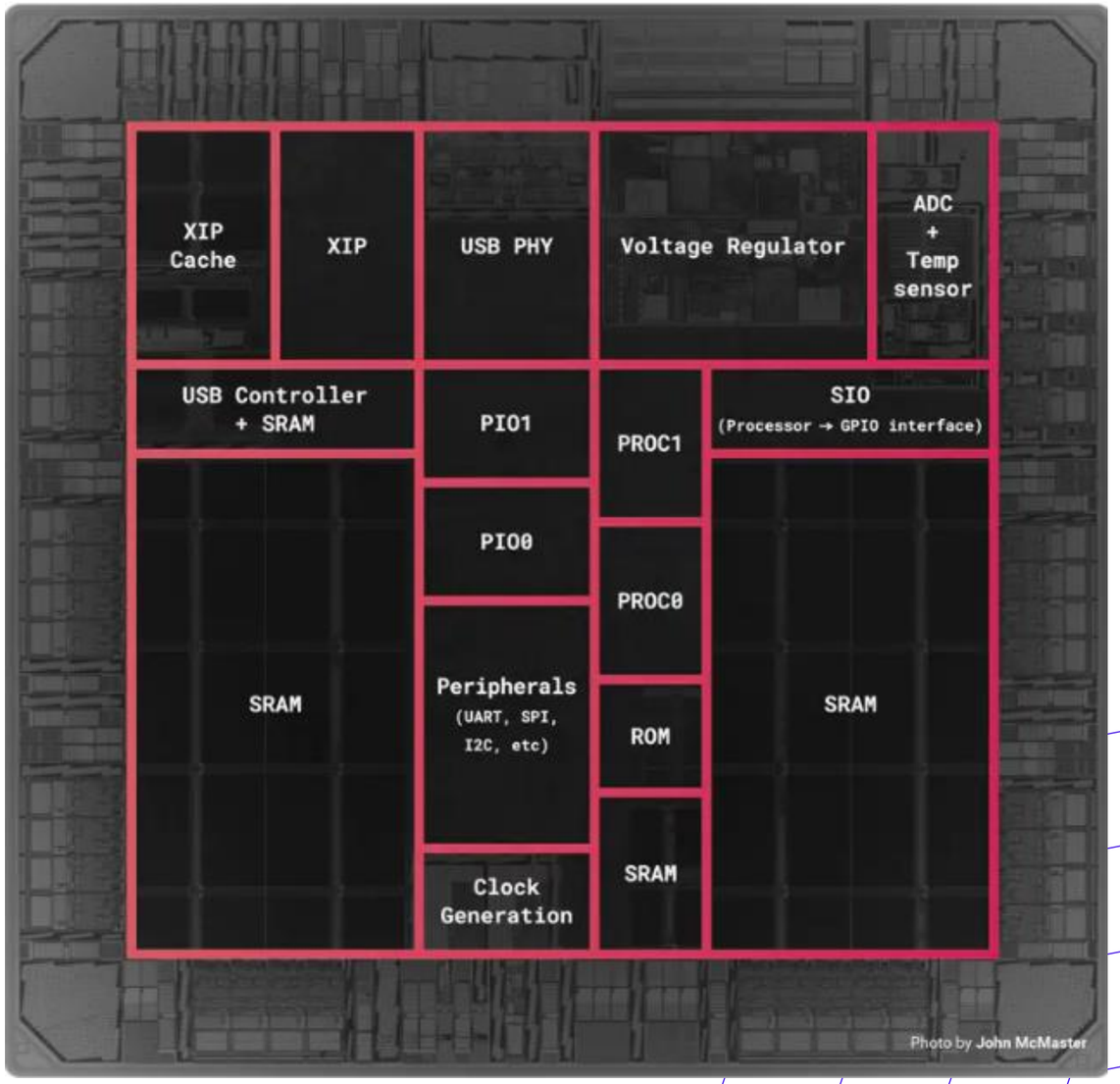


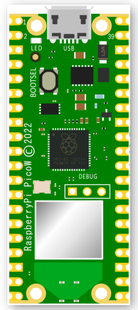


- **O die é o pedaço de silício semicondutor onde os circuitos eletrônicos são fabricados.**
- **Construção à 40nm**
- **Estimativa de 264.000 transistores**

RP2040



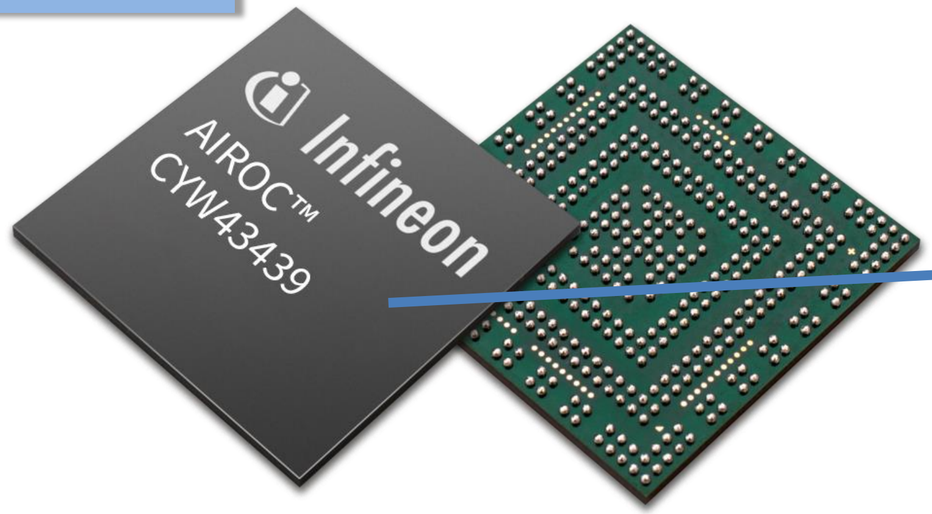

XIP Cache
XIP
USB PHY
Voltage Regulator
ADC + Temp sensor
USB Controller + SRAM
PIO1
PROC1
SIO
(Processor → GPIO interface)
PIO0
PROC0
SRAM
Peripherals
(UART, SPI, I2C, etc)
ROM
SRAM
SRAM
Clock Generation
Photo by John McMaster

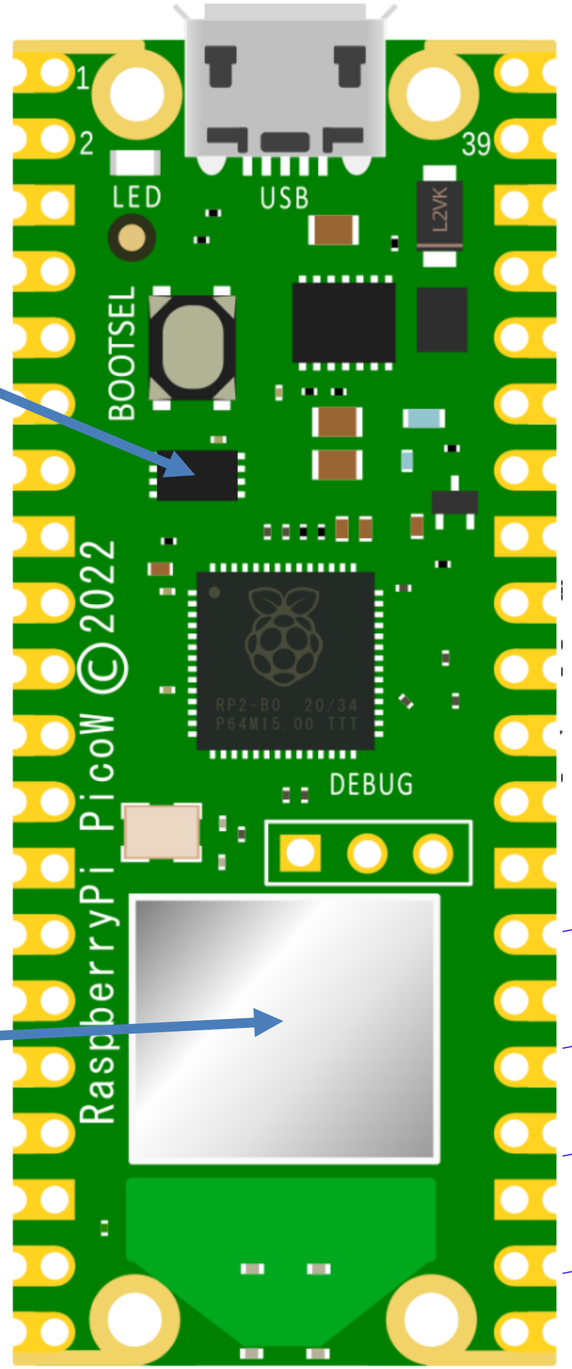


Placa de desenvolvimento Raspberry Pi Pico W

- Memória Flash
  Original com 2MB
  Máximo de 16MB

- Chip Infineon CYW43439 responsável por 802.11b/g/n Wi-Fi 4 e Bluetooth 5.2



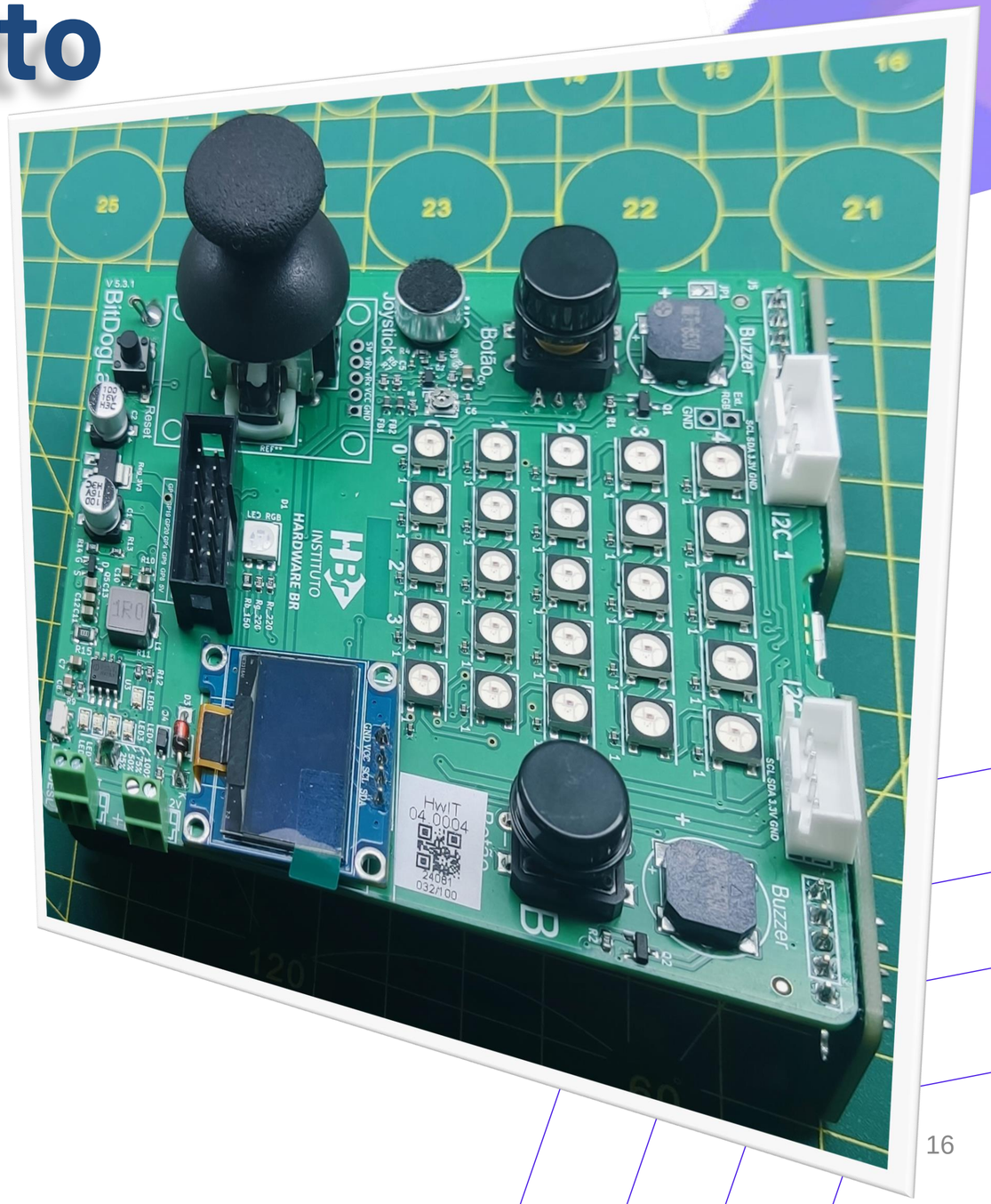


Kit de Desenvolvimento BitDogLab

- A BitDogLab é uma ferramenta educacional voltada para o ensino de sistemas embarcados, desenvolvida no contexto do projeto Escola 4.0 da Universidade de Campinas

- Equipada com microcontrolador Raspberry Pi Pico W, o qual conta com Wi-Fi e Bluetooth.

- A BitDogLab com uma variedade de dispositivos eletrônicos integrados, e diversas portas para expansão e conexão de sensores e atuadores.



Kit BitDogLab

1a e 1b - Push-Butons

2 - Led RGB

3a e 3b – Buzzer

4 – Microfone

5 – Joystick

6 – Display (128 x 64)

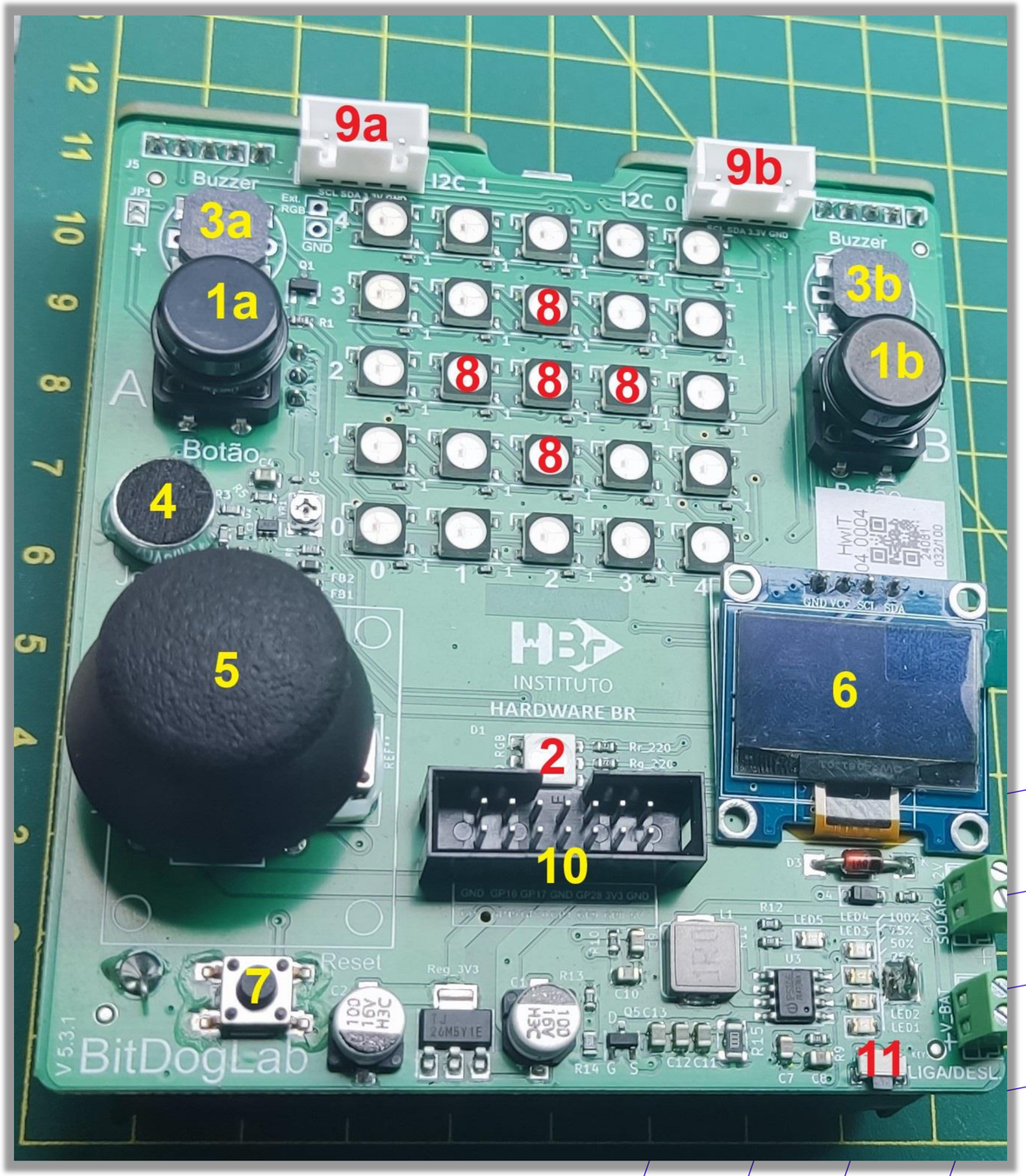
7 – Chave de Reset

8 – Matriz Led (5 x 5)

9a e 9b – Conectores de expansão

10 – Conector de expansão

11 – Conector liga e desliga

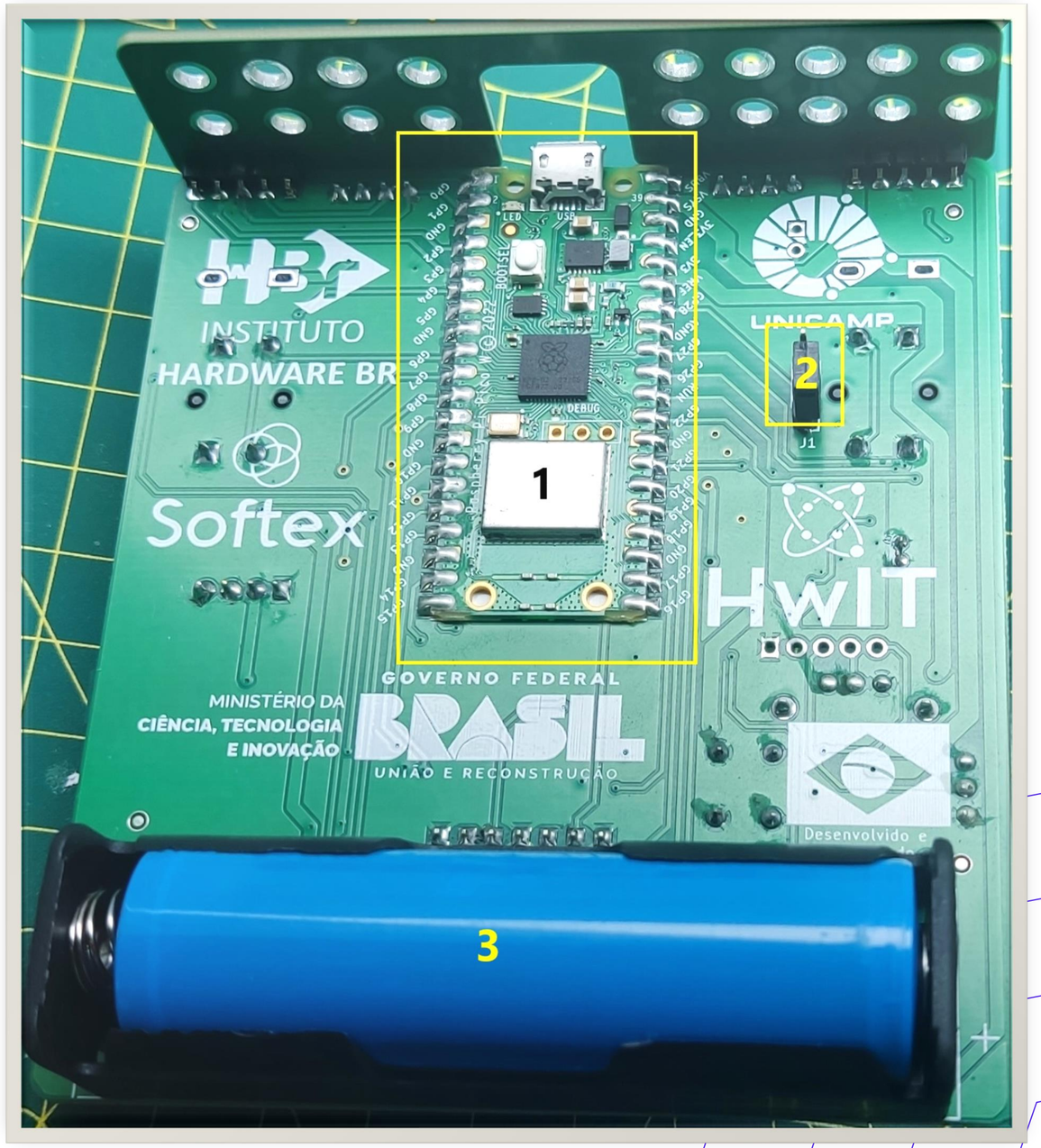


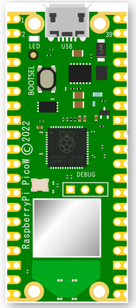
Kit BitDogLab

1 – Raspberry Pi Pico W

2 – Jumper p/ configuração.

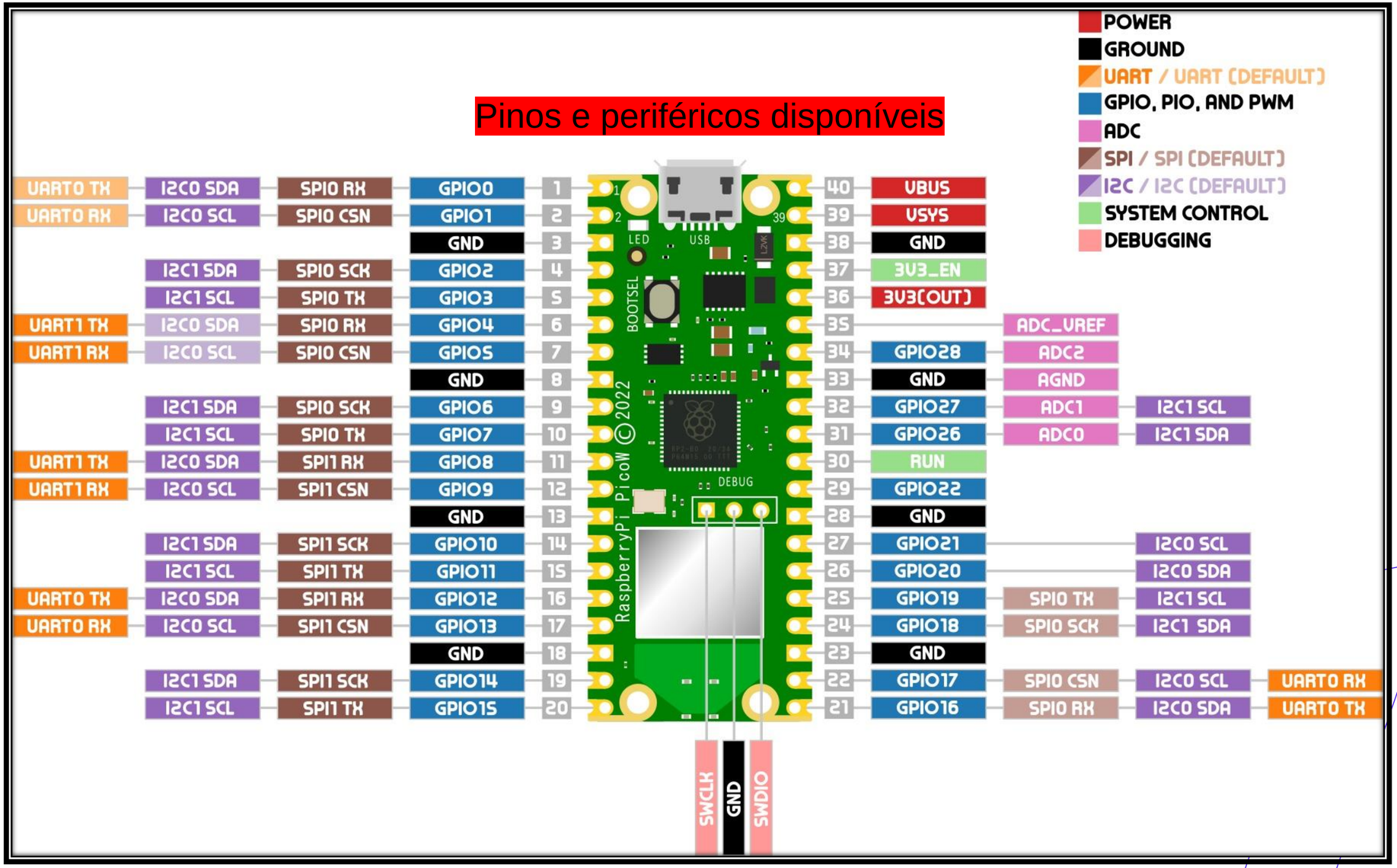
3 – Bateria 18650

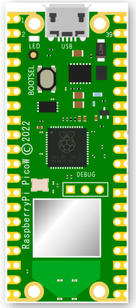




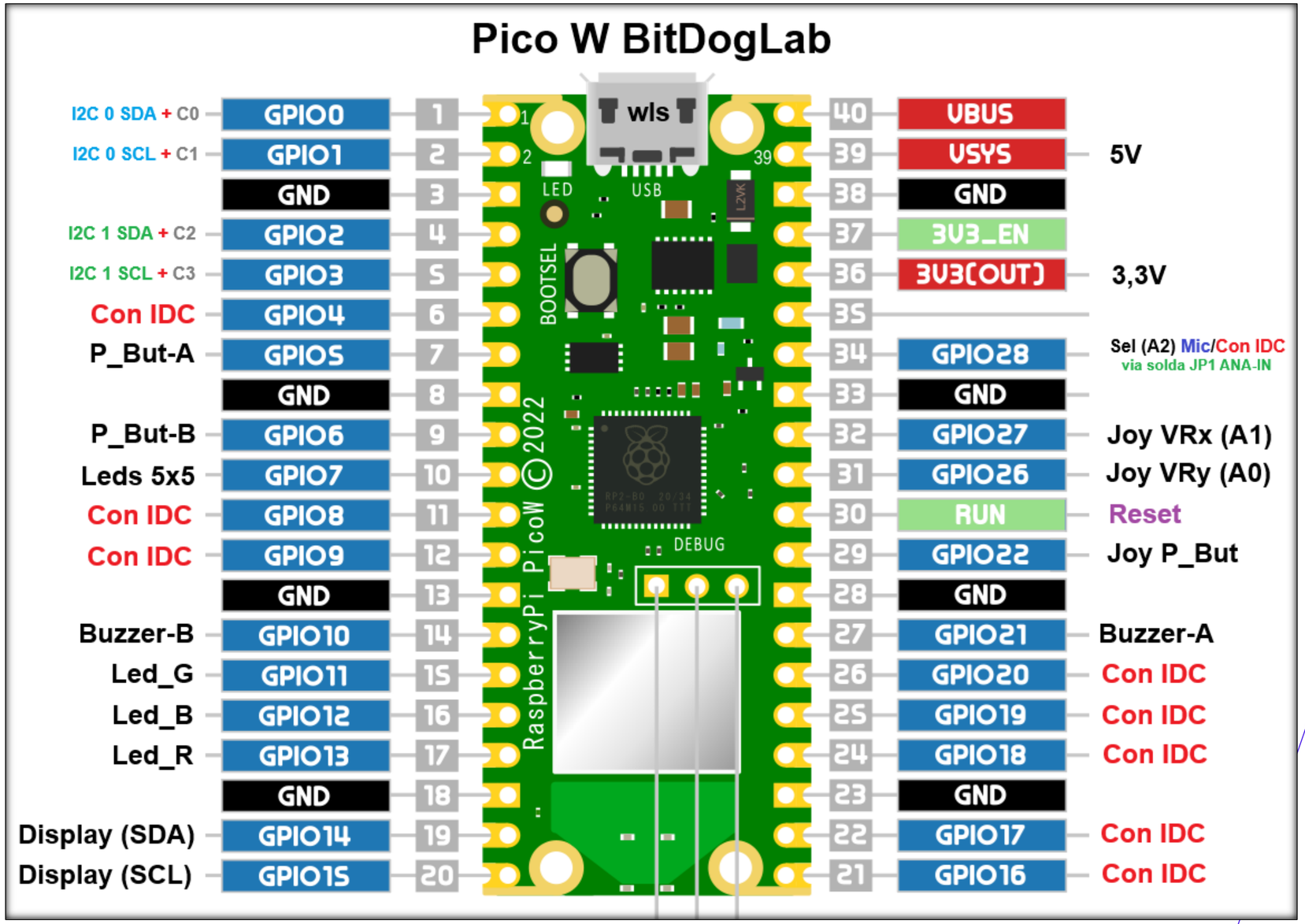
Raspberry Pi Pico W

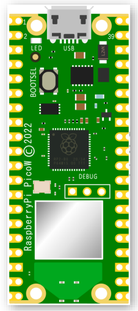
Pinos e periféricos disponíveis



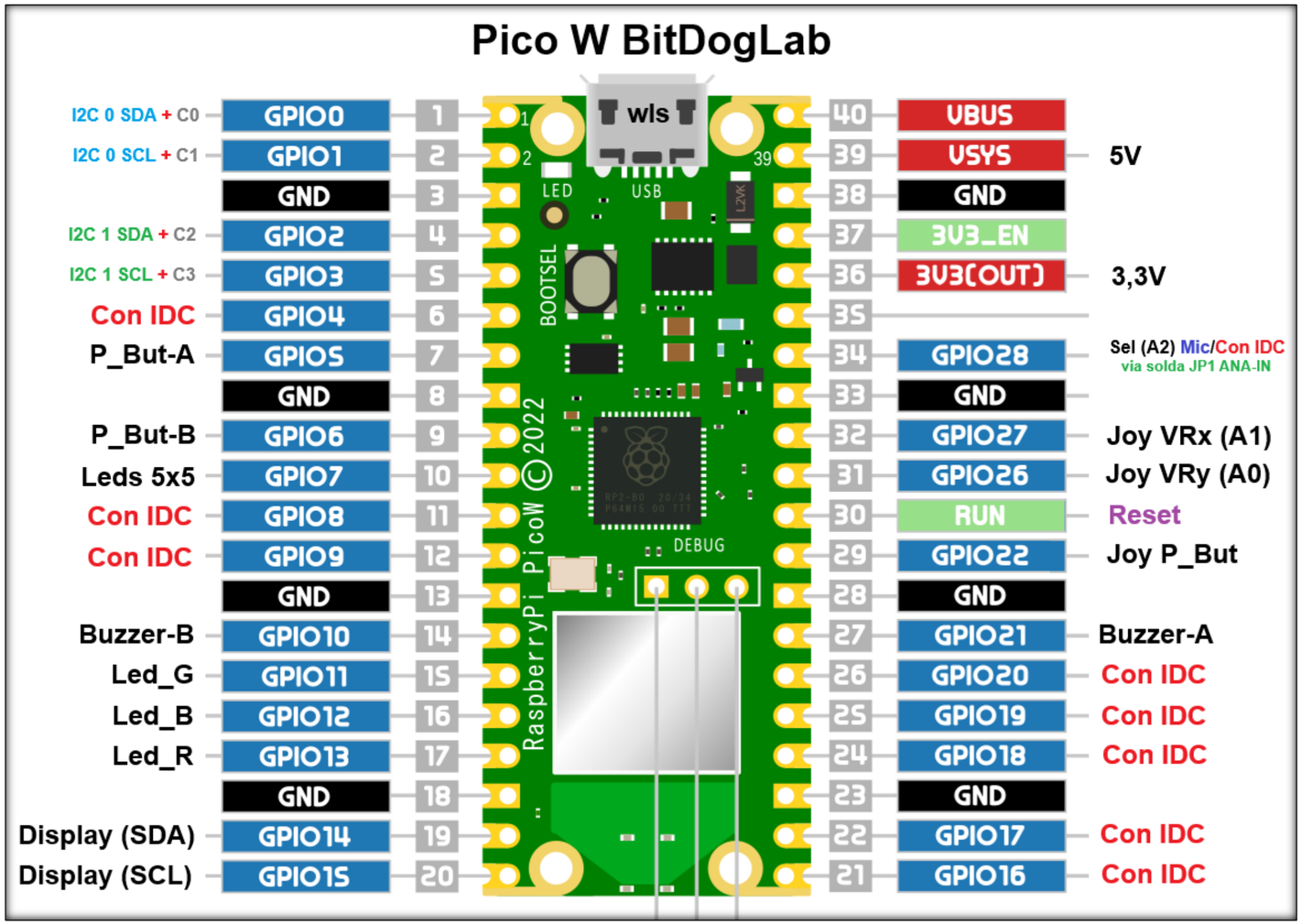


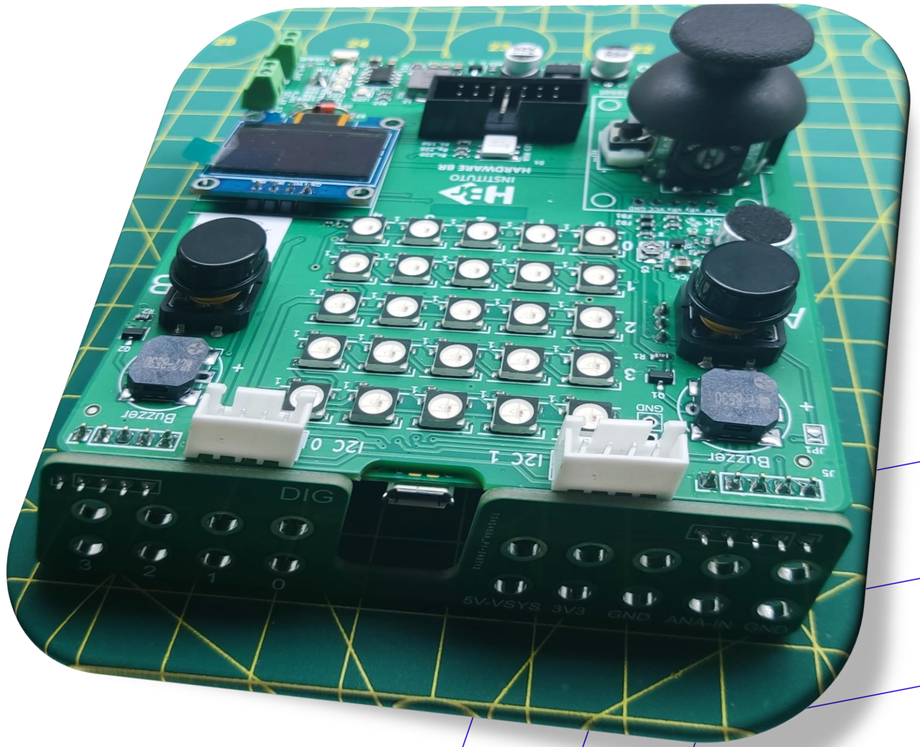
Kit BitDogLab-Raspberry Pi Pico W

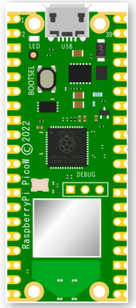




Kit BitDogLab-Raspberry Pi Pico W







Kit BitDogLab na prática

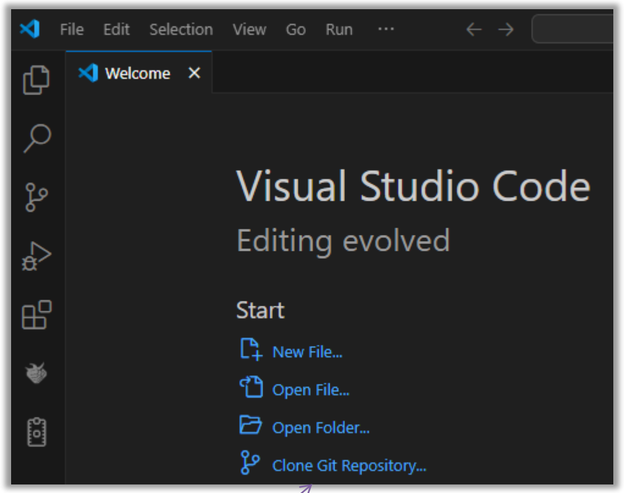
Colocar o RP2040 no modo BootSel e carregar alguns arquivos UF2 para demonstração.

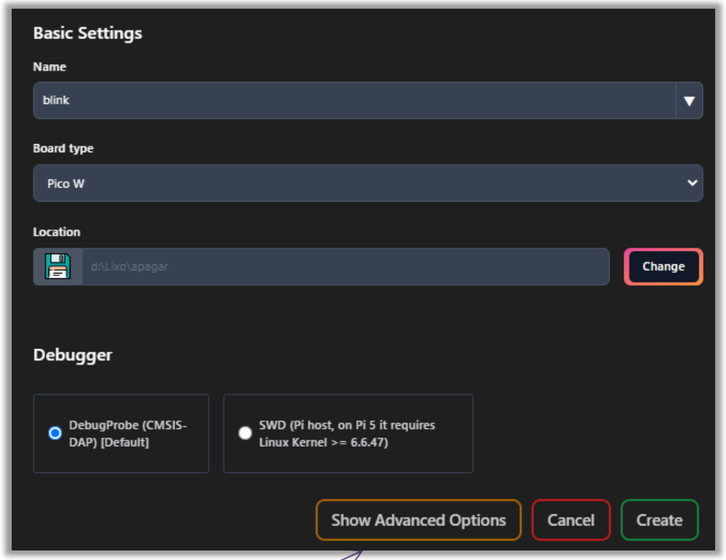
Vá no: **github.com/wiltonlacerda**

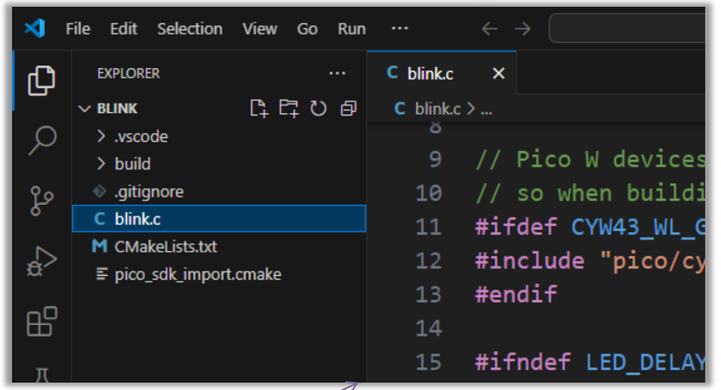
git clone https://github.com/wiltonlacerda/EmbarcaTechU4C1.git

Carregamento de firmwares ...

Kit BitDogLab na prática “Como criar novo Projeto de um Exemplo?

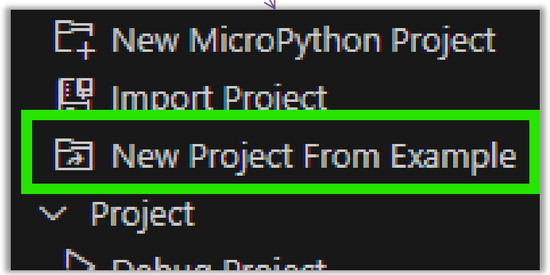


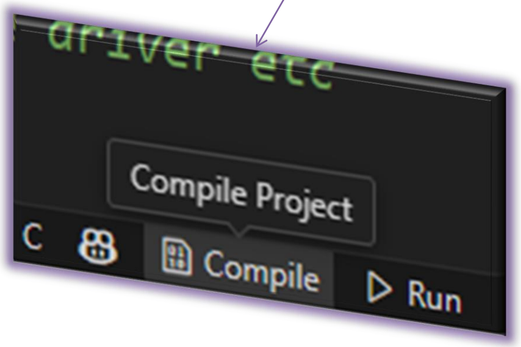




Carregar o arquivo UF2 para o RP2.
Para tanto é necessário colocar o RP2 no modo BOOTSEL.

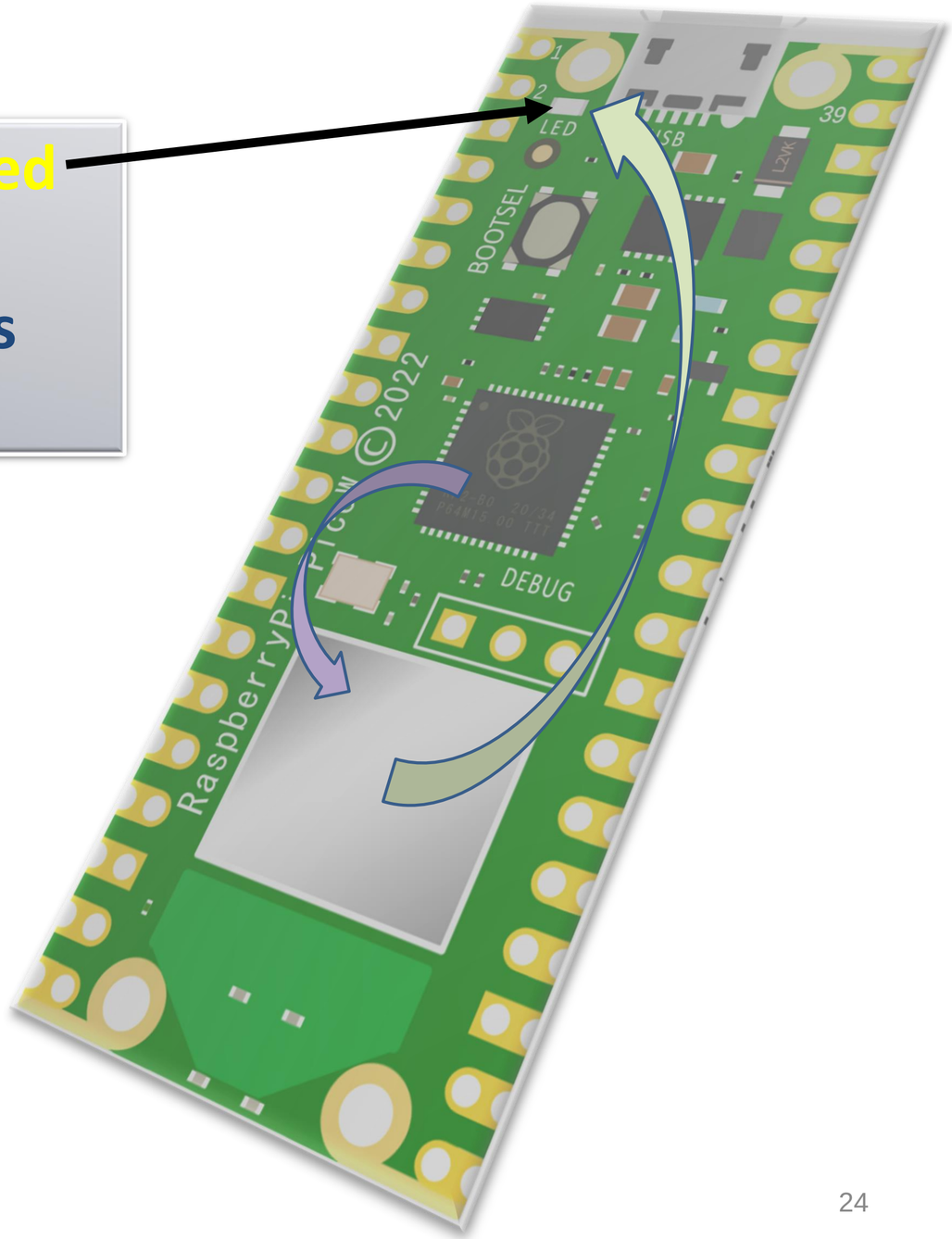
Abra o VS Code → **Vá em Raspberry Pi Pico Project** → **Escolha:** New Project From Example → **Name** : Selecione “blink” **Board type**: Escolha Pico W **Location**: Selecione uma pasta para seu projeto → **Create** → **Compilar o Projeto**. → **Run** Carregar o arquivo UF2 para o RP2 → **Led Piscando**





Kit BitDogLab “Blink Led”

- Criação de um novo projeto para piscar o Led on board do RP2 W.
- Apresentar a estrutura de arquivos e pastas produzidas.



Obs.
Lembre-se que o Chip Infineon CYW43439 é quem realmente vai piscar o Led.

```
#include "pico/cyw43_arch.h"
```

Veja na prática...

VSCode: Estrutura de projeto

Projeto blink

Enxugamento do código para melhor entendimento.

File: pico_sdk_import.cmake

pico_sdk_import.cmake

```
5
6  if (DEFINED ENV{PICO_SDK_PATH} AND (NOT PICO_SDK
7      set(PICO_SDK_PATH $ENV{PICO_SDK_PATH})
8      message("Using PICO_SDK_PATH from environmen
9  endif ()
10
11 if (DEFINED ENV{PICO_SDK_FETCH_FROM_GIT} AND (NO
12     set(PICO_SDK_FETCH_FROM_GIT $ENV{PICO_SDK_FE
13     message("Using PICO_SDK_FETCH_FROM_GIT from
14 endif ()
15
```

File: blink.c

blink.c

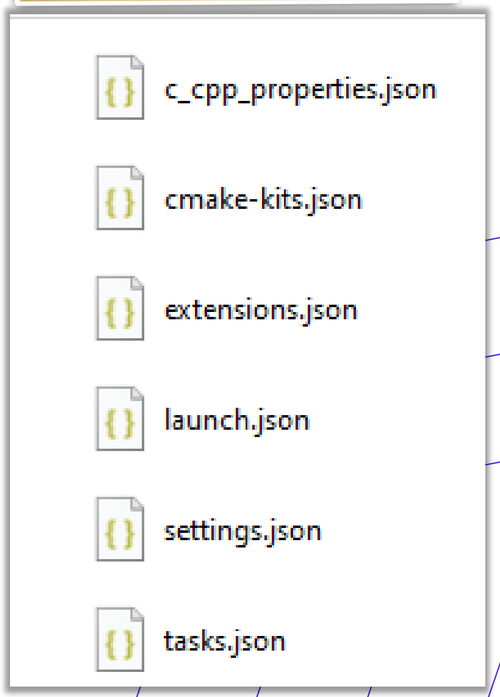
```
1  #include "pico/stdlib.h"
2
3  #define led_pin_red 13
4
5  int main() {
6      gpio_init(led_pin_red);
7      gpio_set_dir(led_pin_red, GPIO_OUT);
8
9      while (true) {
10         gpio_put(led_pin_red, true);
11         sleep_ms(1000);
12         gpio_put(led_pin_red, false);
13         sleep_ms(1000);
14     }
15 }
```

File: CMakeLists.txt

CMakeLists.txt

```
21 set(picoVscode ${USERHOME}/.pico-sdk/cmake/pic
22 if (EXISTS ${picoVscode})
23     include(${picoVscode})
24 endif()
25 # ============================================
26 set(PICO_BOARD pico CACHE STRING "Board type")
27
28 include(pico_sdk_import.cmake)
29 project(blink C CXX ASM)
30 pico_sdk_init()
31 add_executable(blink blink.c)
32 target_link_libraries(blink pico_stdlib)
33 pico_add_extra_outputs(blink)
```

Pasta: .vscode



VSCode: Estrutura de projeto

Trechos de código da biblioteca Pico SDK, utilizada para o Raspberry Pi Pico

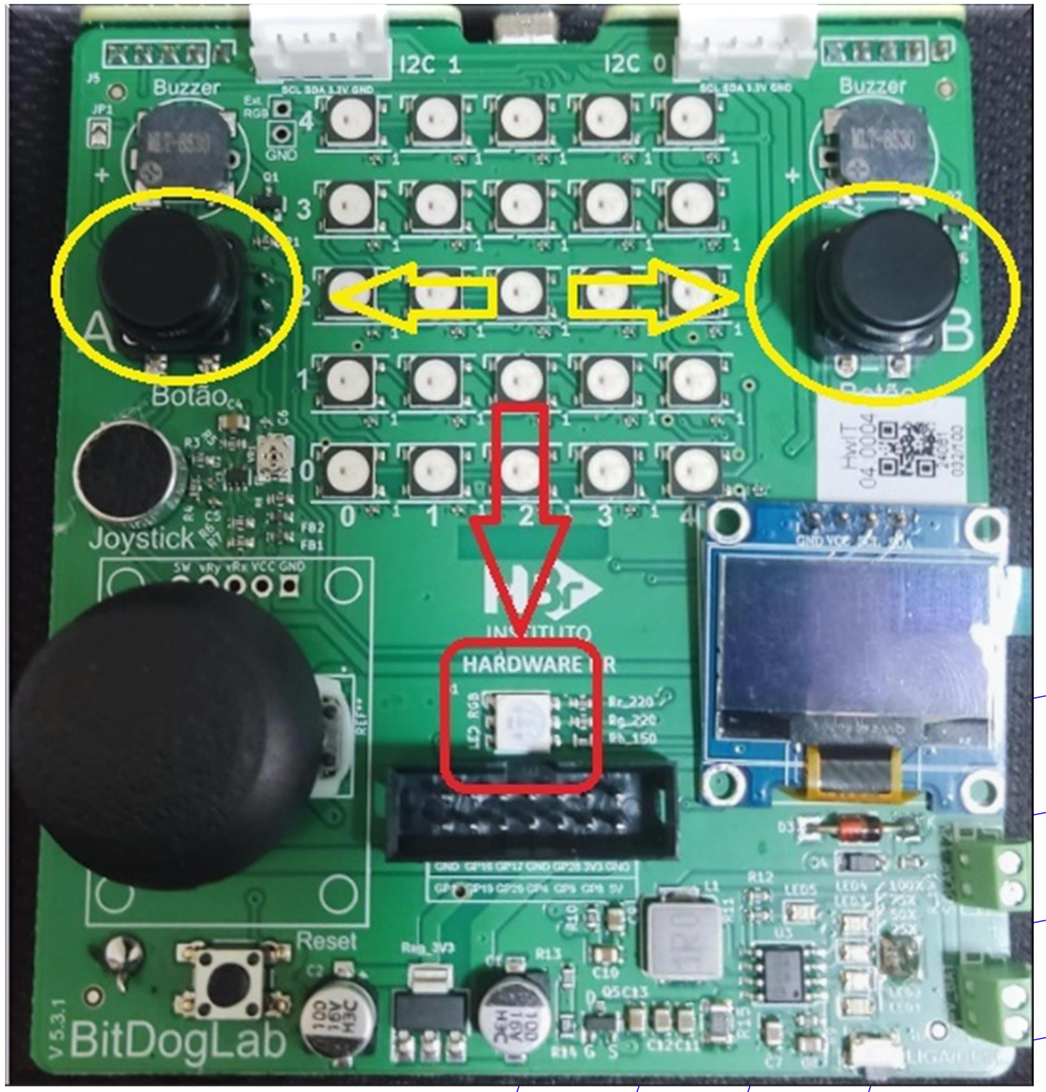
```
#include <stdio.h> // Inclui a biblioteca padrão de entrada e saída
stdio_init_all(); // Inicializa a biblioteca de entrada e saída padrão para uso de print
printf(“Ola mundo! \n"); // Imprime na tela

#include "pico/stdlib.h" //  Inclui a biblioteca de funções padrão
gpio_init(led_pin); // Inicializa o pino do led_pin
gpio_set_dir(led_pin, GPIO_OUT); // Define o pino do led_pin como saída
gpio_put(led_pin, true); // Liga o led
gpio_put(led_pin, 1); // Liga o led
gpio_put(led_pin, false); // Desliga o led
gpio_put(led_pin, 0); // Desliga o led
sleep_ms(duracao); // Aguarda um tempo "duracao" em milissegundos
```

VSCode: Estrutura de arquivos e pastas

Projeto blink (RGB)
Enxugamento do código para melhor entendimento.

O que acontece caso pastas e/ou arquivos forem apagados?
Em que condições será possível recuperar este projeto?

Vamos dar uma olhada de perto...

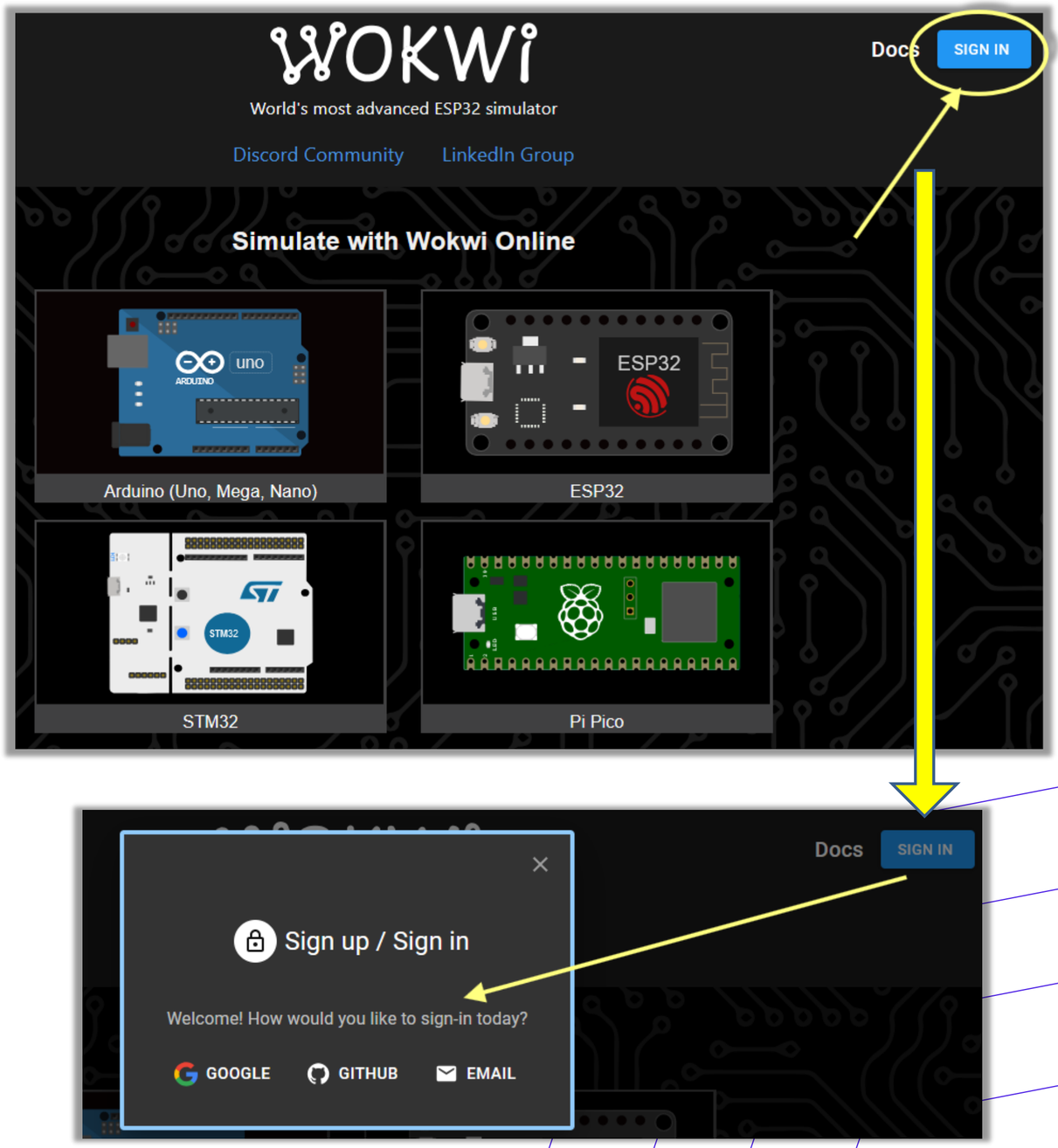


Simulador Wokwi

O **Wokwi** é uma plataforma online gratuita que permite criar, simular e testar projetos de eletrônica diretamente no navegador, sem a necessidade de hardware físico. Ele é amplamente utilizado para aprendizado, prototipagem e desenvolvimento de projetos com microcontroladores, sensores e outros componentes eletrônicos.

Abra sua conta no Wokwi

https://wokwi.com



Wokwi projetando e programando

WOKWI SAVE SHARE

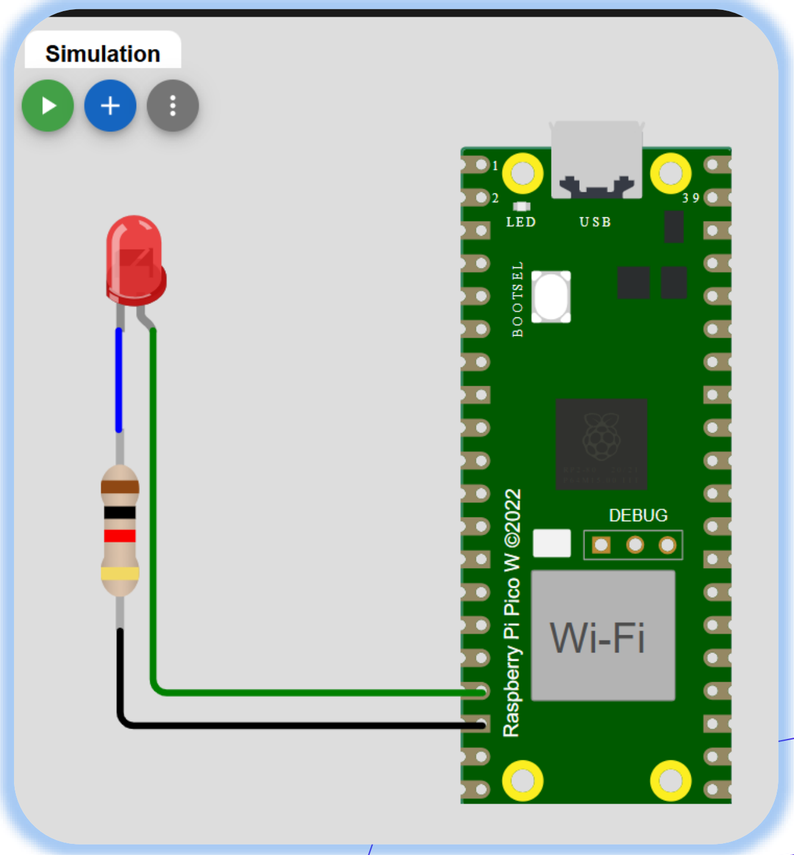
main.c | diagram.json •

```c
#include "pico/stdlib.h"

#define led_pin_red 13
int tempo = 1000;

int main() {
    gpio_init(led_pin_red);
    gpio_set_dir(led_pin_red, GPIO_OUT);

    while (true) {
        gpio_put(led_pin_red, true);
        sleep_ms(tempo);
        gpio_put(led_pin_red, false);
        sleep_ms(tempo);
    }
}
```

WOKWI SAVE SHARE BlinkLed13

main.c | diagram.json •

```json
{
  "version": 1,
  "author": "Wilton L. Silva",
  "editor": "wokwi",
  "parts": [
    {
      "type": "board-pi-pico-w",
      "id": "pico",
      "top": -80.05,
      "left": 13.15,
      "attrs": { "builder": "pico-sdk" }
    },
    {
      "type": "wokwi-led",
      "id": "led1",
      "top": -61.2,
      "left": -101.8,
      "attrs": { "color": "Red" }
    },
    {
      "type": "wokwi-resistor",
      "id": "r1",
      "top": 33.6,
      "left": -115.75,
      "rotate": 90,
      "attrs": { "value": "1000" }
    }
  ],
  "connections": [
    [ "led1:C", "r1:1", "blue", [ "v0" ] ],
    [ "pico:GND.4", "r1:2", "black", [ "h0" ] ],
    [ "pico:GP13", "led1:A", "green", [ "h0" ] ]
  ],
  "dependencies": {}
}
```



RP2 no Wokwi...

https://wokwi.com/projects/418071361765683201

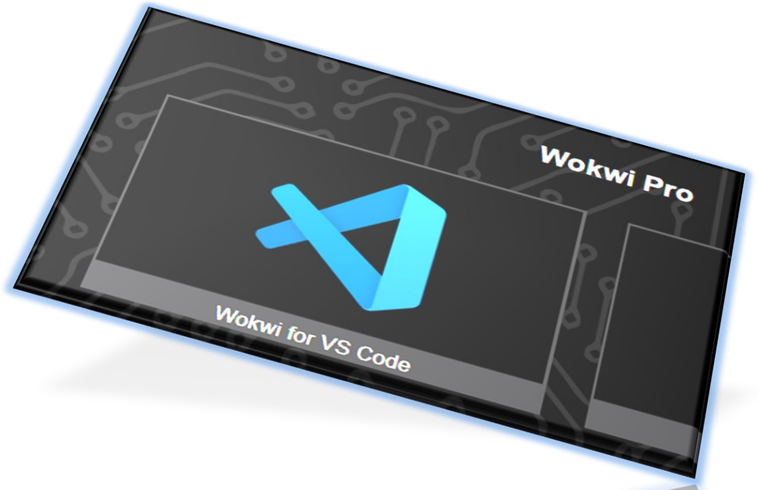
Simulador Wokwi para VSCode

Introdução ao Wokwi para VS Code

Wokwi for Visual Studio Code fornece uma solução de simulação para engenheiros de sistemas embarcados e IoT. A extensão se integra ao seu ambiente de desenvolvimento existente, permitindo que você simule seus projetos **diretamente do seu editor de código**.

Você pode usar Wokwi para VS Code com Zehpyr Project, PlatformIO, ESP-IDF, **Pi Pico SDK**, NuttX, Rust, Arduino CLI e outras estruturas e cadeias de ferramentas de desenvolvimento incorporadas.

É necessário conta no Wokwi



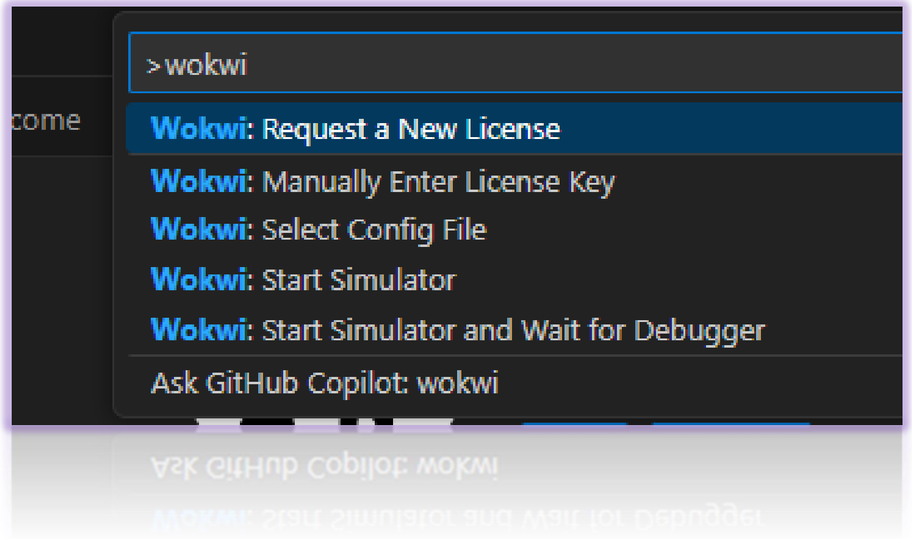
> **AVISO**
>
> Wokwi for VS Code está atualmente em beta público. Após a versão beta, alguns recursos estarão disponíveis apenas para usuários pagos.

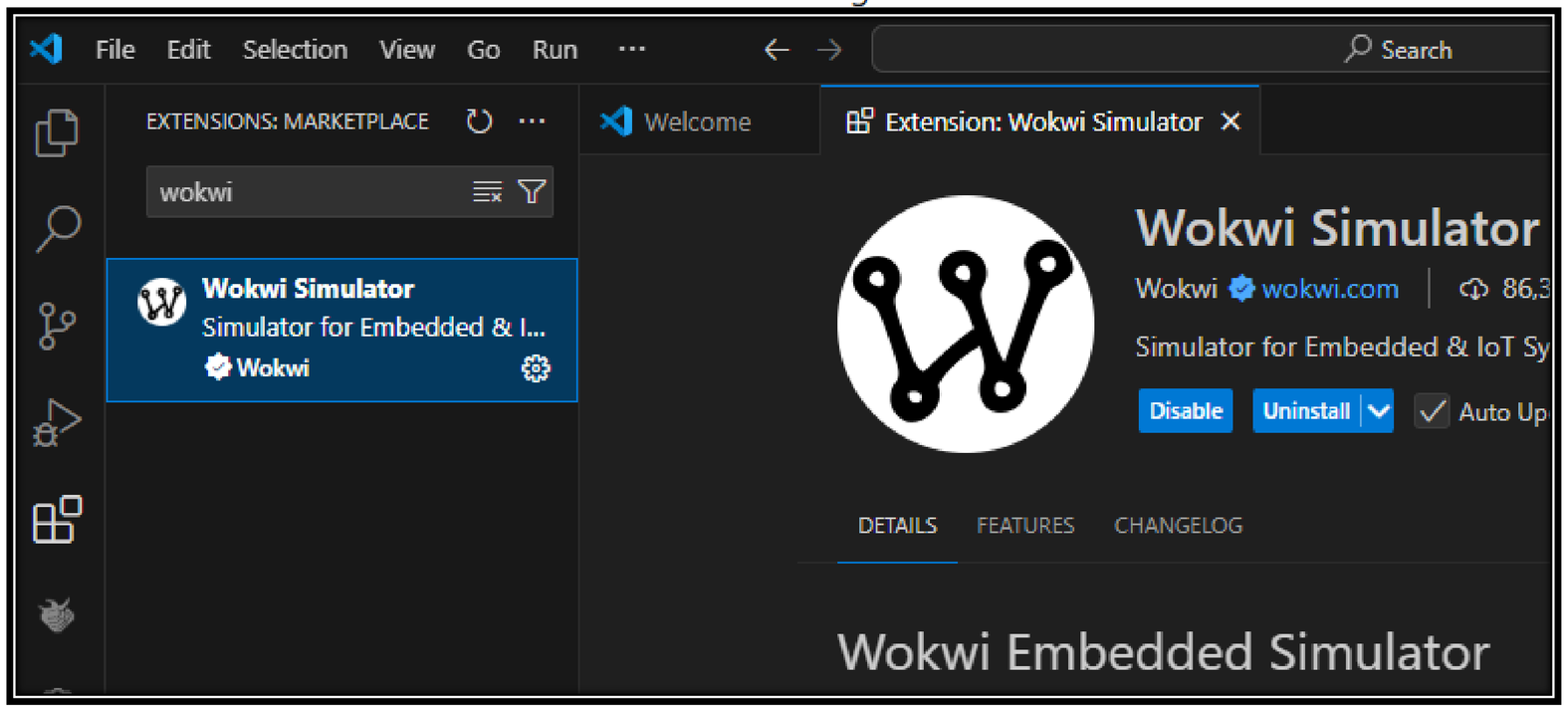
https://docs.wokwi.com/pt-BR/vscode/getting-started

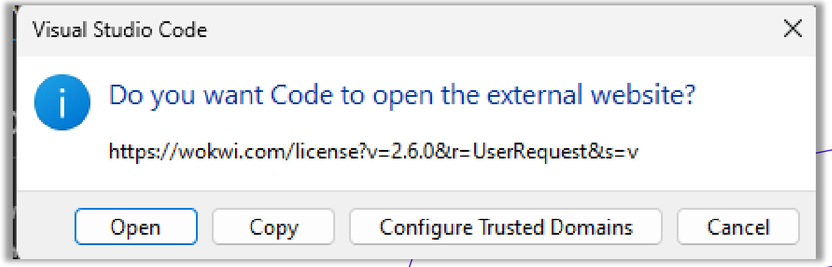
Wokwi para VSCode

Instalação

Primeiro, instale a extensão Wokwi for VS Code. Em seguida, pressione `F1` e selecione "Wokwi: Solicitar uma nova licença". O VS Code solicitará que você confirme a abertura do site Wokwi em seu navegador. Confirme clicando em "Abrir".

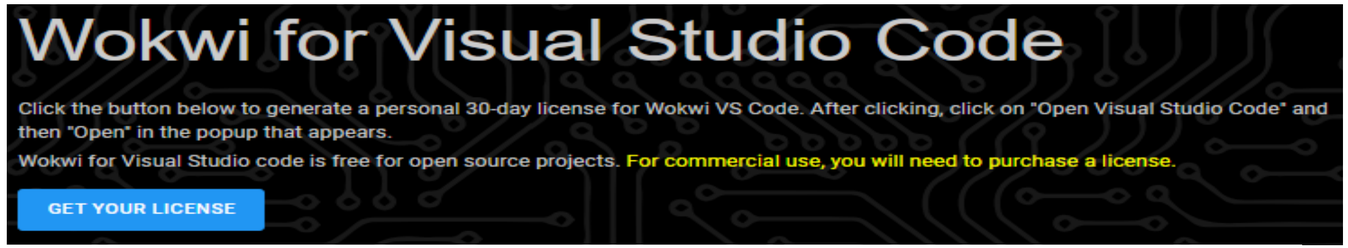




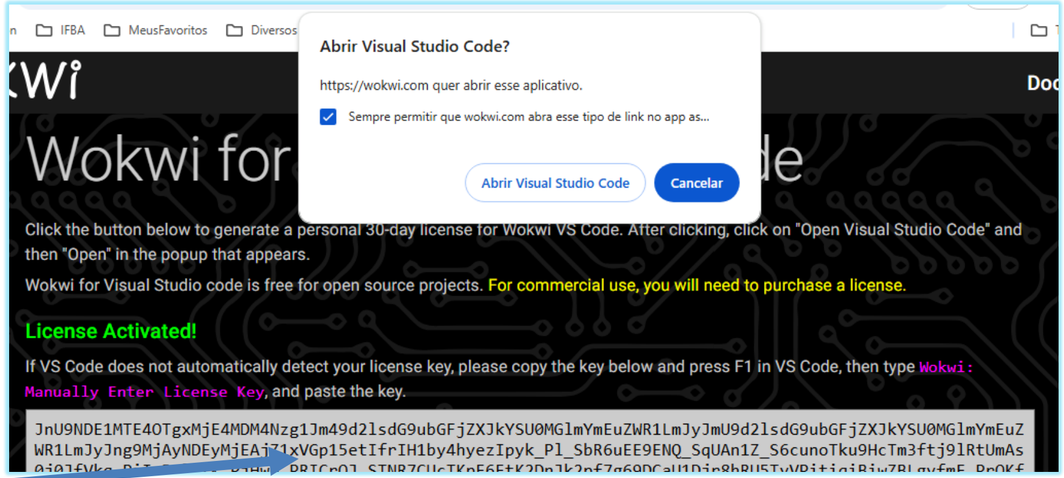


https://docs.wokwi.com/pt-BR/vscode/getting-started

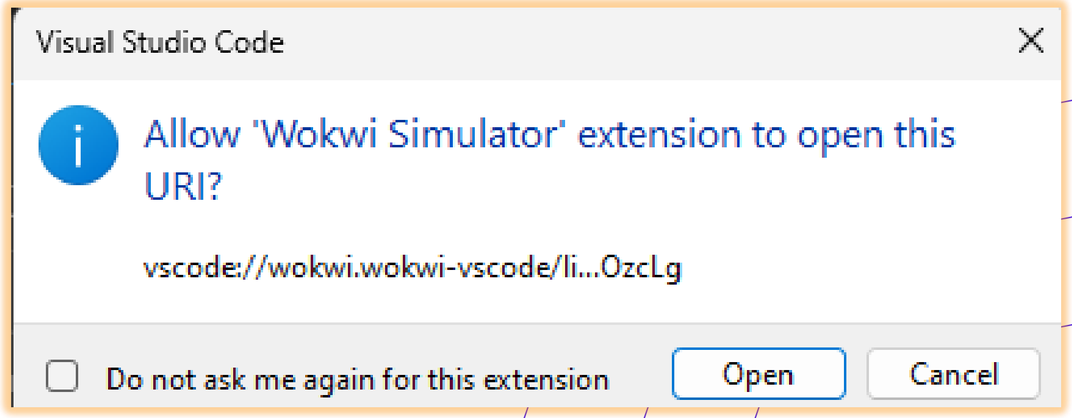
Wokwi para VSCode



Em seguida, clique no botão azul que diz "GET YOUR LICENSE". Você pode ser solicitado a entrar em sua conta Wokwi. Se você não tiver uma conta, poderá criar uma gratuitamente.



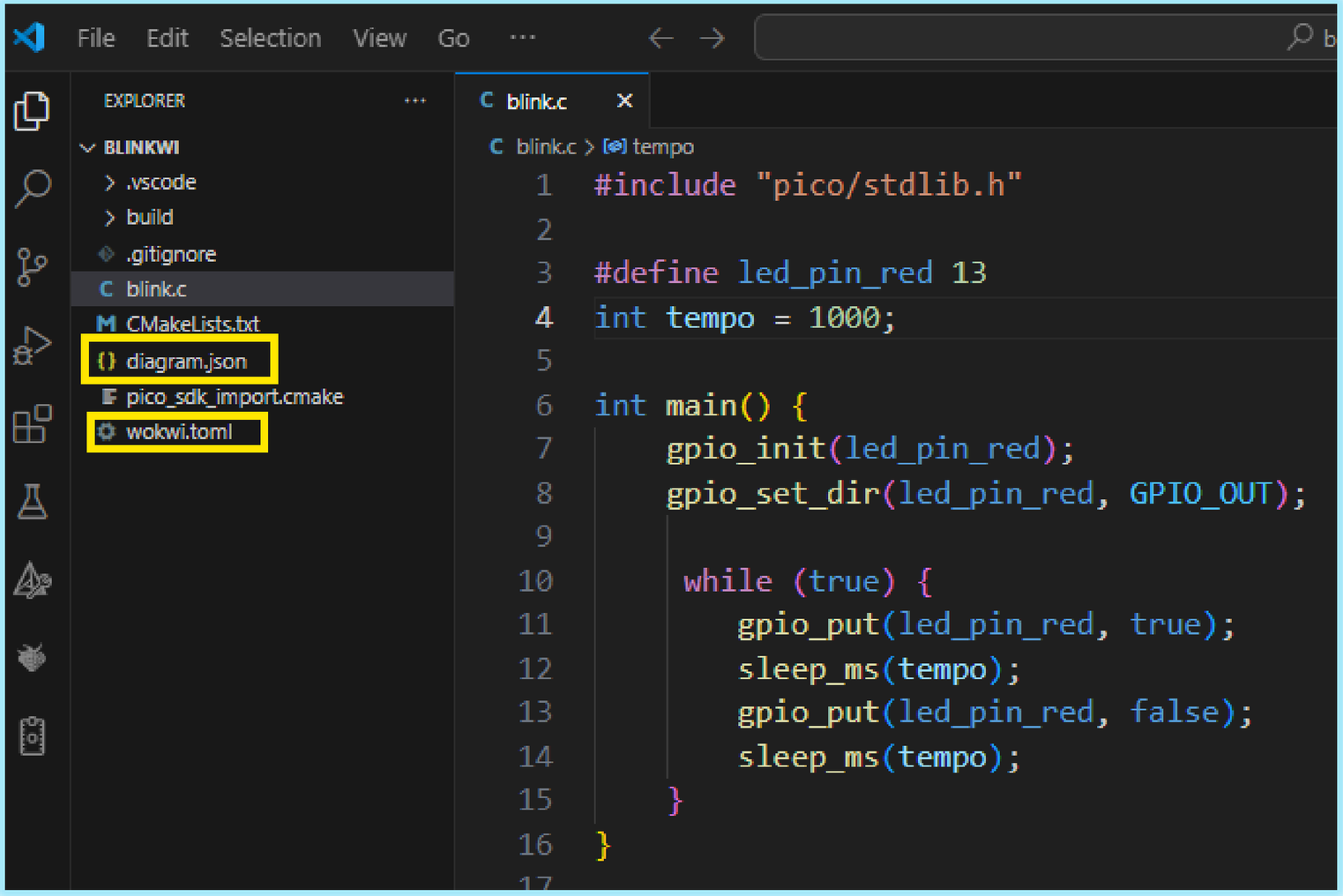
O navegador pedirá uma confirmação para enviar a licença para o VS Code. Confirme (talvez seja necessário confirmar duas vezes, uma vez no navegador e outra no VS Code). Você verá uma mensagem no VS Code que diz "License activated for [seu nome]". Parabéns!

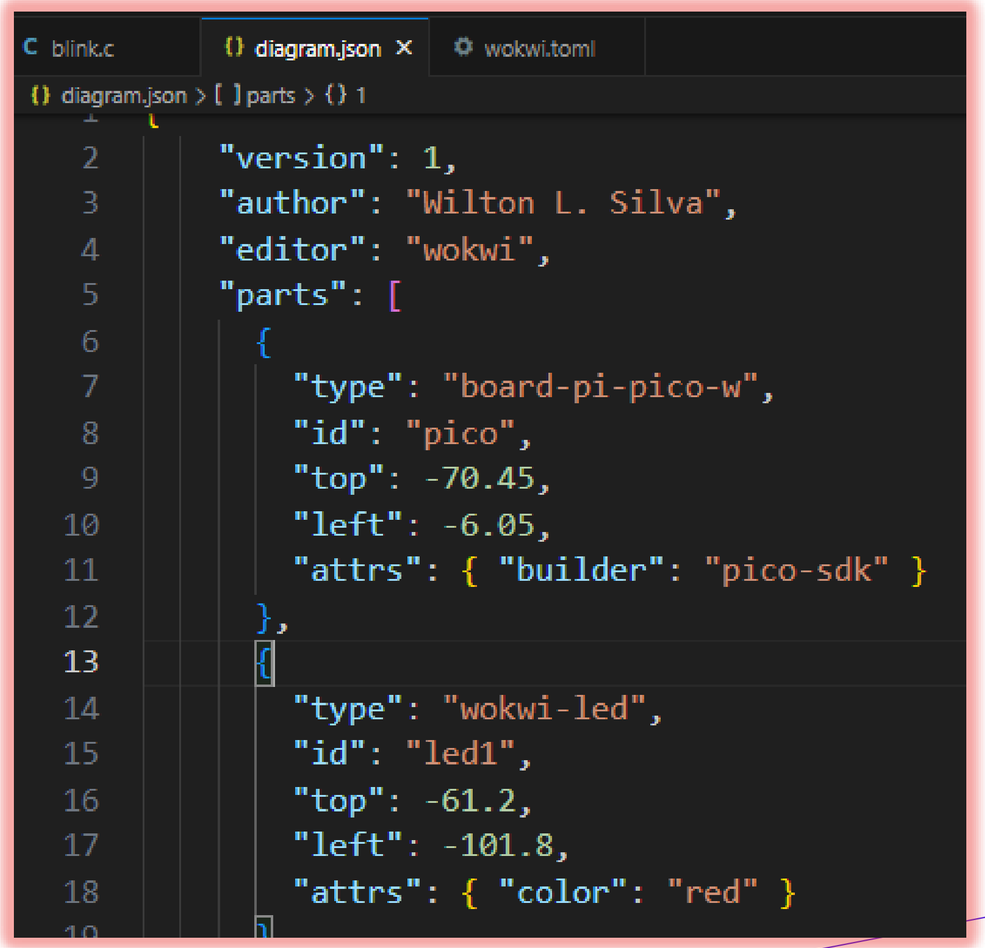


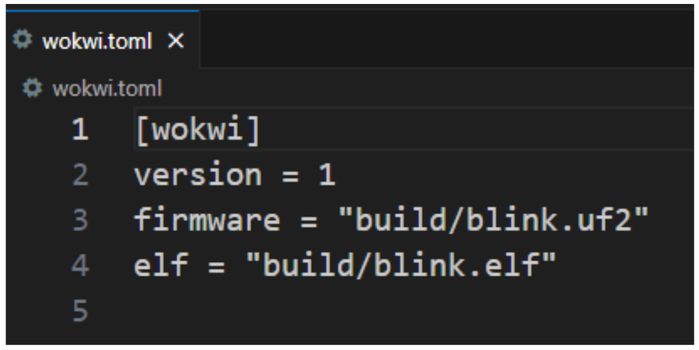
Extensão Wokwi no VS Code...

Wokwi para VSCode

Ajustar o projeto **blink led red** para trabalhar o **Wokwi** no **VS Code**.
No projeto blink, criar os arquivos: **diagram.json** e o **wokwi.toml**

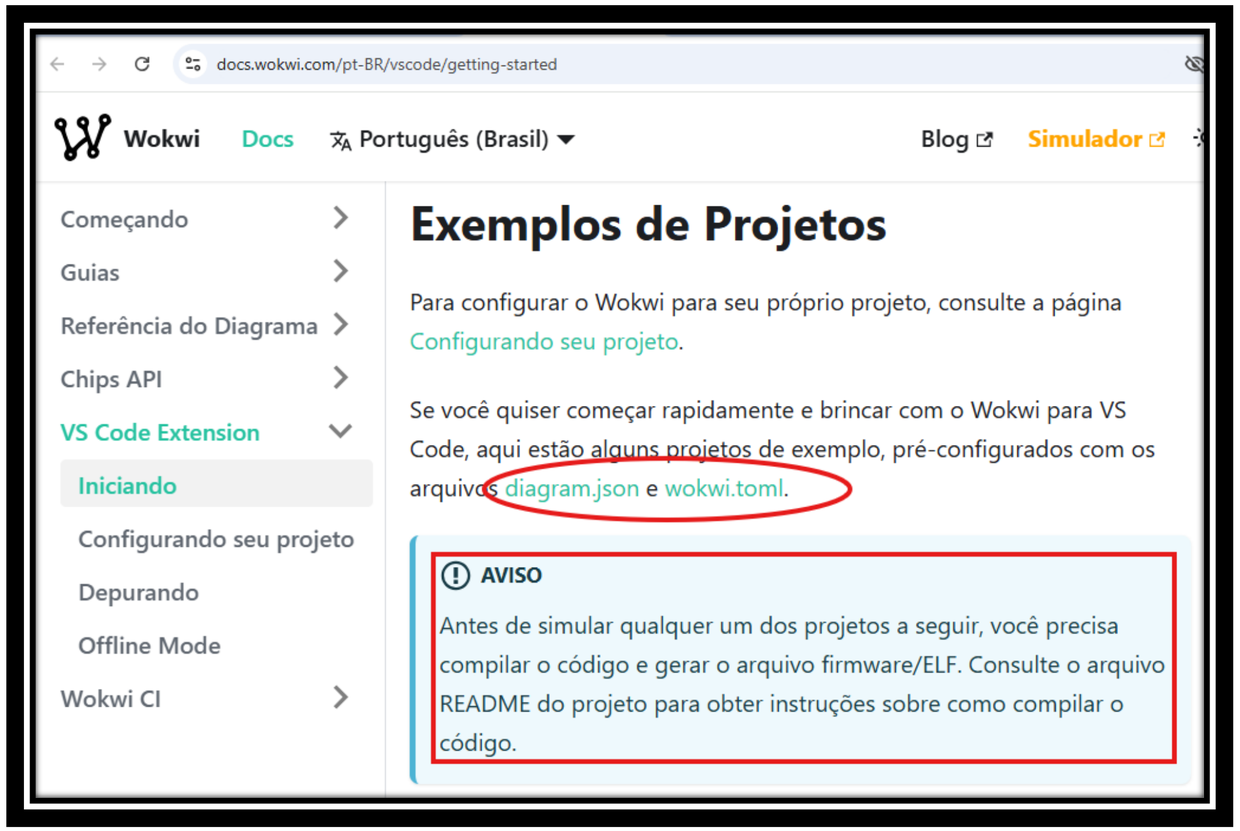


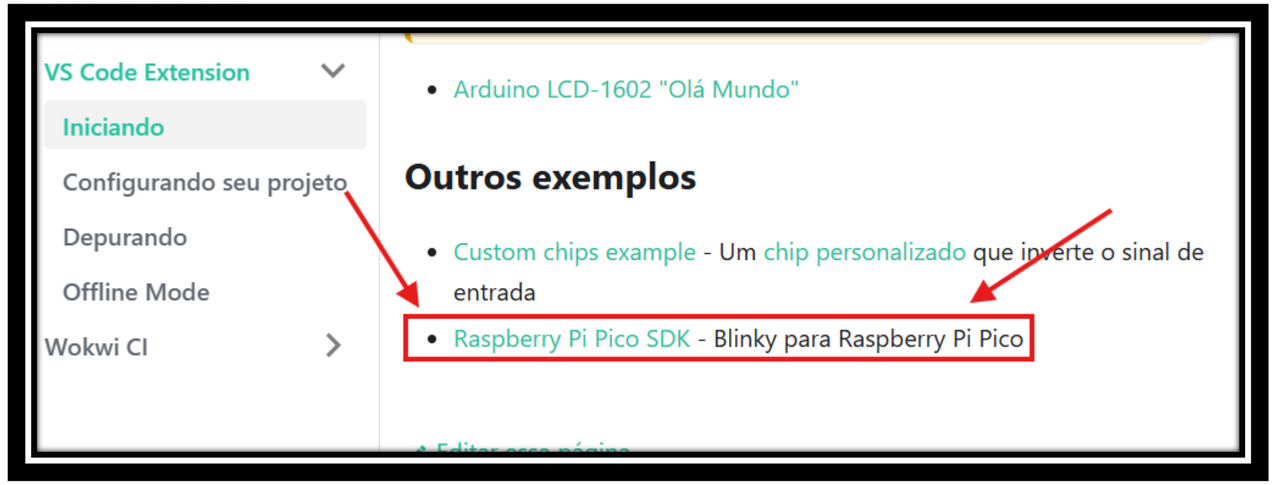


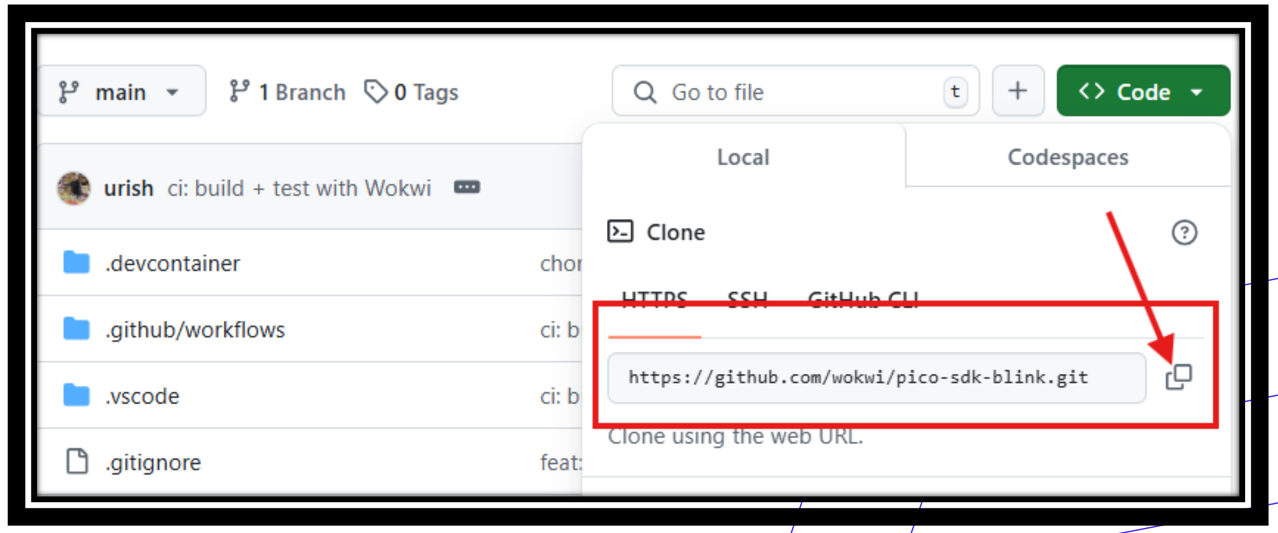


Blink Wokwi no VS Code...

Wokwi para VSCode







Fazer o clone no git para um pasta desejada.
git clone https://github.com/wokwi/pico-sdk-blink.git Ex_Wokwi

Clone Projeto...

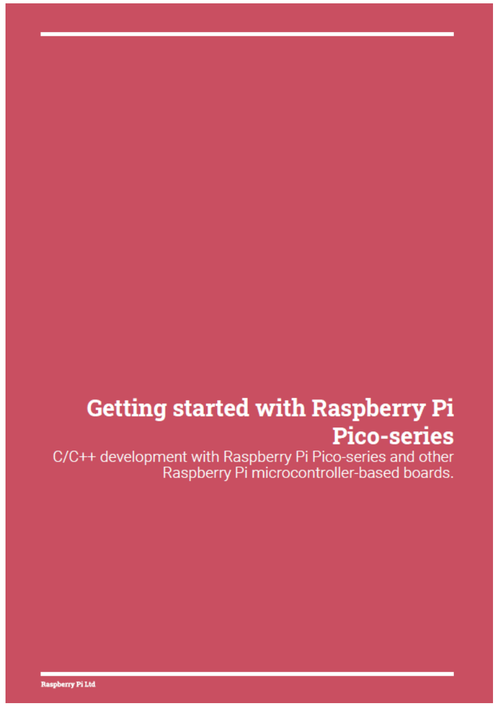
https://docs.wokwi.com/pt-BR/vscode/getting-started

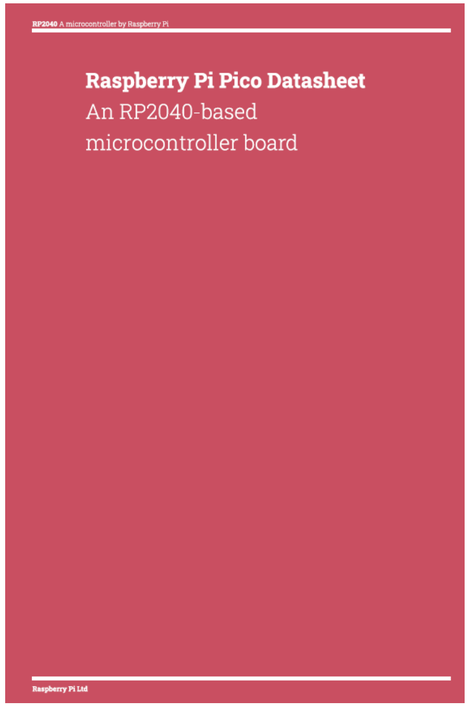
Finalizando: Introdução ao Desenvolvimento de Software Embarcado, Utilizando Linguagem C, com Microcontroladores

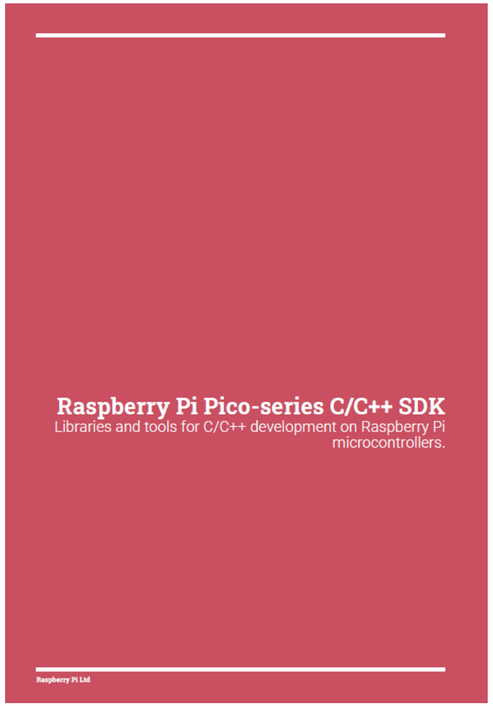
Pontos Principais

- Desenvolvimento em Microcontroladores
  - » BitDogLab
- Arquitetura e Recursos do RP2040
  - » Dual-Core CortexM0+, 256 kBSRAM, periféricos e PIO
- SDK do RaspberryPi Pico
- Arquivo importante: CMakeLists.txt
  - » Configuração do processo de build
- Utilização do simulador Wokwi

Referências







https://github.com/BitDogLab/BitDogLab/tree/main/doc
https://www.raspberrypi.com/documentation/microcontrollers/pico-series.html

Obrigado!