

# Introdução ao Desenvolvimento de Software Embarcado com Microcontroladores

Unidade 4 | Capítulo 1

Executores:



Coordenação:



Iniciativa:



# Sumário

- Objetivos
- Introdução
- Kit de Desenvolvimento
- RaspberryPi Pico W
- RP2040
- Organização do SDK
- Principais Pontos

# Objetivos

Entender as características do desenvolvimento de software em microcontroladores.

Conhecer e caracterizar a placa de desenvolvimento BitDogLabe seus componentes principais: CPU, memória, interfaces de entrada/saída.

Compreender a organização das bibliotecas de esenvolvimento do RaspberryPi Pico.

# Introdução



## Desenvolvimento em Microcontroladores

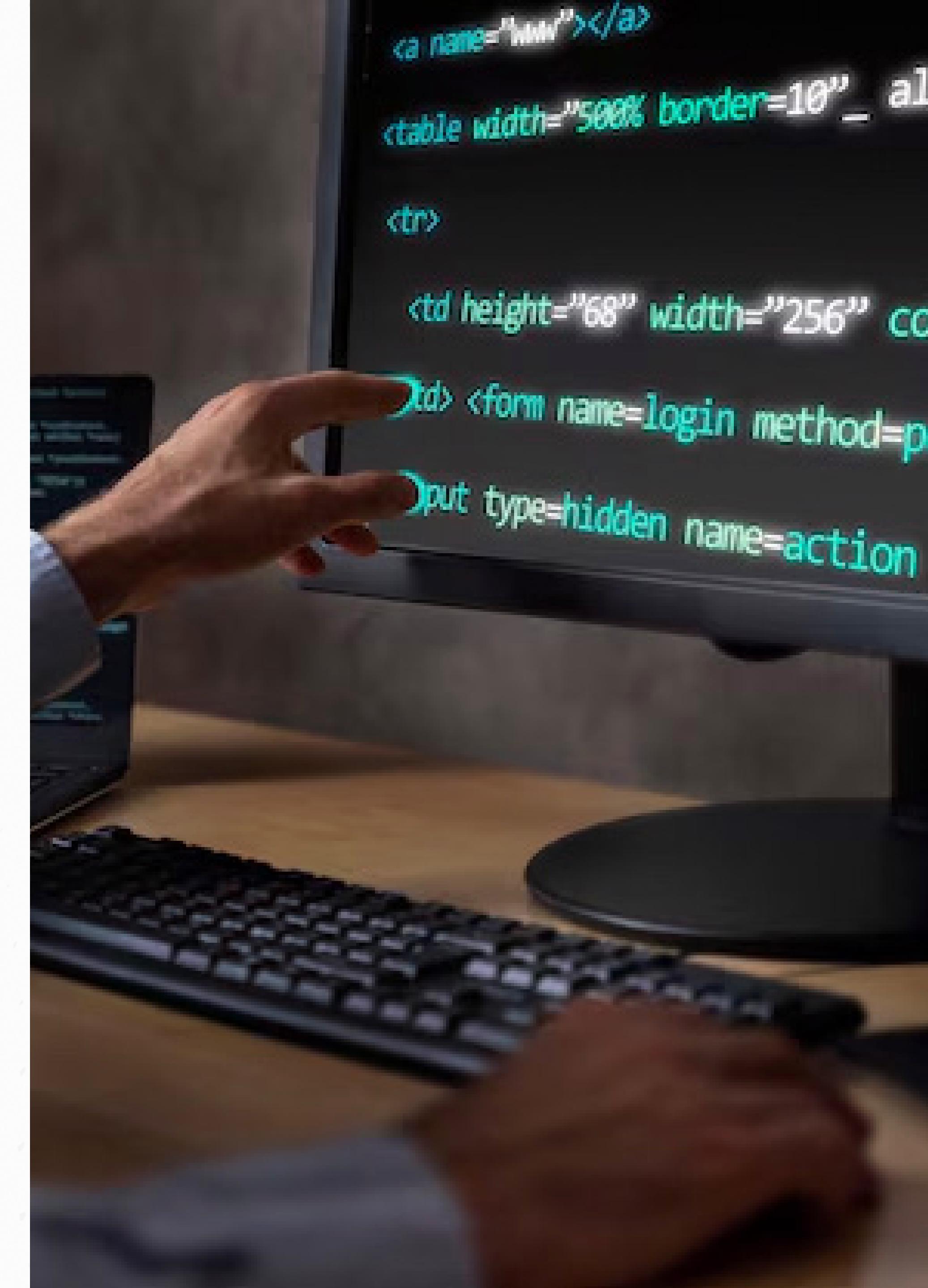
- Recursos Limitados
- Programação em Baixo Nível
- Controle Preciso de Tempo
  - » Requisitos de Tempo-Real
- Interação com o mundo físico

Fig. 1

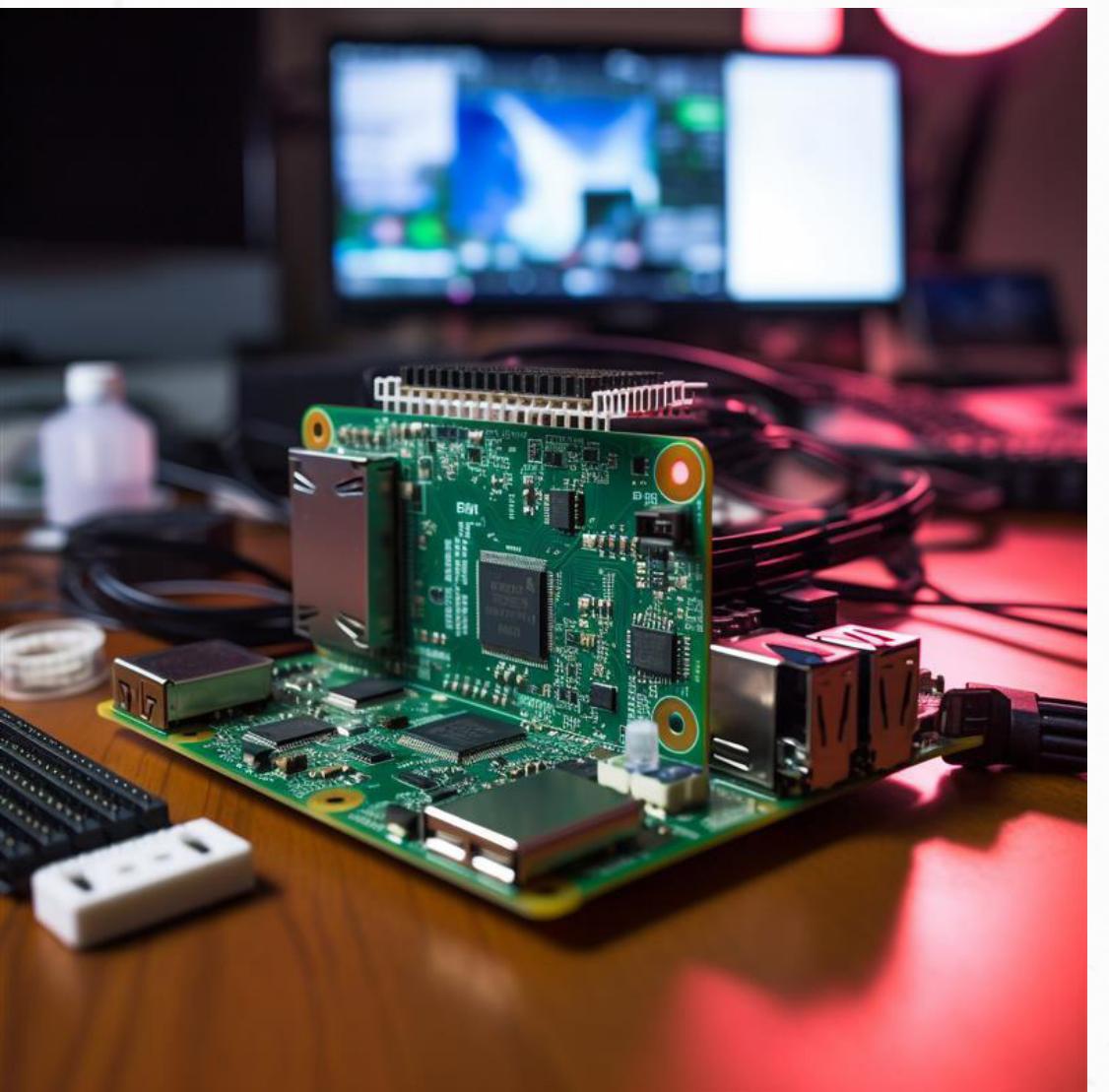
A imagem mostra um relógio de areia de vidro com areia dourada fluindo através do gargalo estreito. O relógio de areia está inclinado em um ângulo de 45 graus e a areia está caindo pelo lado do vidro. O relógio de areia está apoiado em um fundo cinza claro.

Fig. 2

A imagem mostra um programador sentado em frente a um computador em uma mesa de trabalho. O programador é homem e sua mão direita está estendida em direção à tela, como se estivesse interagindo com o código na tela. O computador está ligado e a tela mostra código escrito em uma linguagem de programação. A cor do texto é verde. O teclado e o mouse estão posicionados na mesa à frente do programador. O fundo da imagem é escuro e desfocado, mostrando que a sala está escura.



# Introdução



## Desenvolvimento em Microcontroladores

- Gerenciamento de Energia
- Otimização de Código
- Depuração e Testes

**Fig. 3**

A imagem mostra uma placa de circuito impresso verde, com diversos componentes eletrônicos, sobre uma mesa de madeira. A placa está conectada a vários cabos e fios, além de um pequeno circuito branco. Ao fundo da imagem, pode-se ver um monitor de computador ligado.

**Fig. 4**

A imagem mostra uma mão segurando uma lâmpada incandescente acesa. A lâmpada está brilhando intensamente e a luz dela ilumina a mão, que aparece em silhueta. O fundo da imagem é desfocado e mostra um céu com nuvens, com tons de laranja e amarelo, como um pôr do sol. A imagem transmite a sensação de esperança, criatividade e inspiração, como se a lâmpada representasse uma ideia brilhante.



# Kit de Desenvolvimento

## BitDogLab

- Iniciativa do projeto escola 4.0
  - » Unicamp
- Ferramenta Educacional
  - » Sistemas Embarcados
  - » Eletrônica
- RaspberryPi Pico W

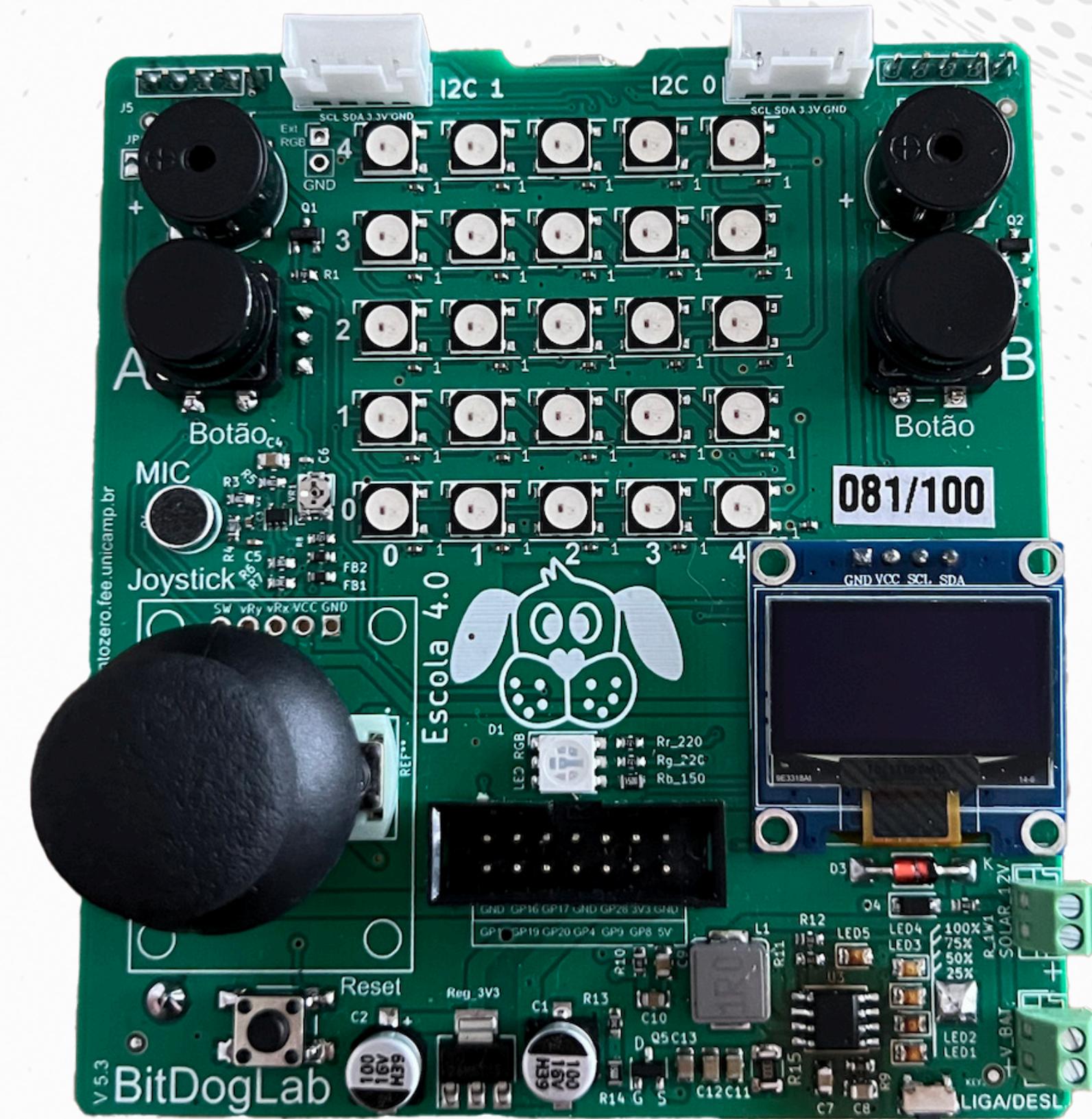


Fig. 5 - BitDogLab

A imagem mostra uma placa eletrônica verde com diversos componentes. Na parte superior da placa, há uma matriz de 4x4 botões pequenos, totalizando 16 botões. À esquerda da matriz, há um joystick, semelhante ao de controles de videogame. Logo abaixo do joystick, há alguns conectores e componentes menores.

Na parte direita da placa, ao lado da matriz de botões, há uma pequena tela (provavelmente uma tela OLED) e dois botões maiores acima dela. Próximo à parte inferior da placa, há uma série de conectores e componentes eletrônicos adicionais, incluindo um conector central.

A placa tem as marcas "BitDogLab" na parte inferior, e há outros textos e logotipos técnicos impressos na superfície.

Fonte: <https://github.com/BitDogLab/BitDogLab/raw/main/doc/illustrations/joystick-assembled.png>

# Kit de Desenvolvimento

## BitDogLab

- Componentes
  - » Bateria
  - » Display OLED 128x64 pixels
  - » Matriz de LEDs coloridos
  - » Microfone
  - » Joystick
  - » Botões
  - » Buzzer
  - » Conectores de expansão

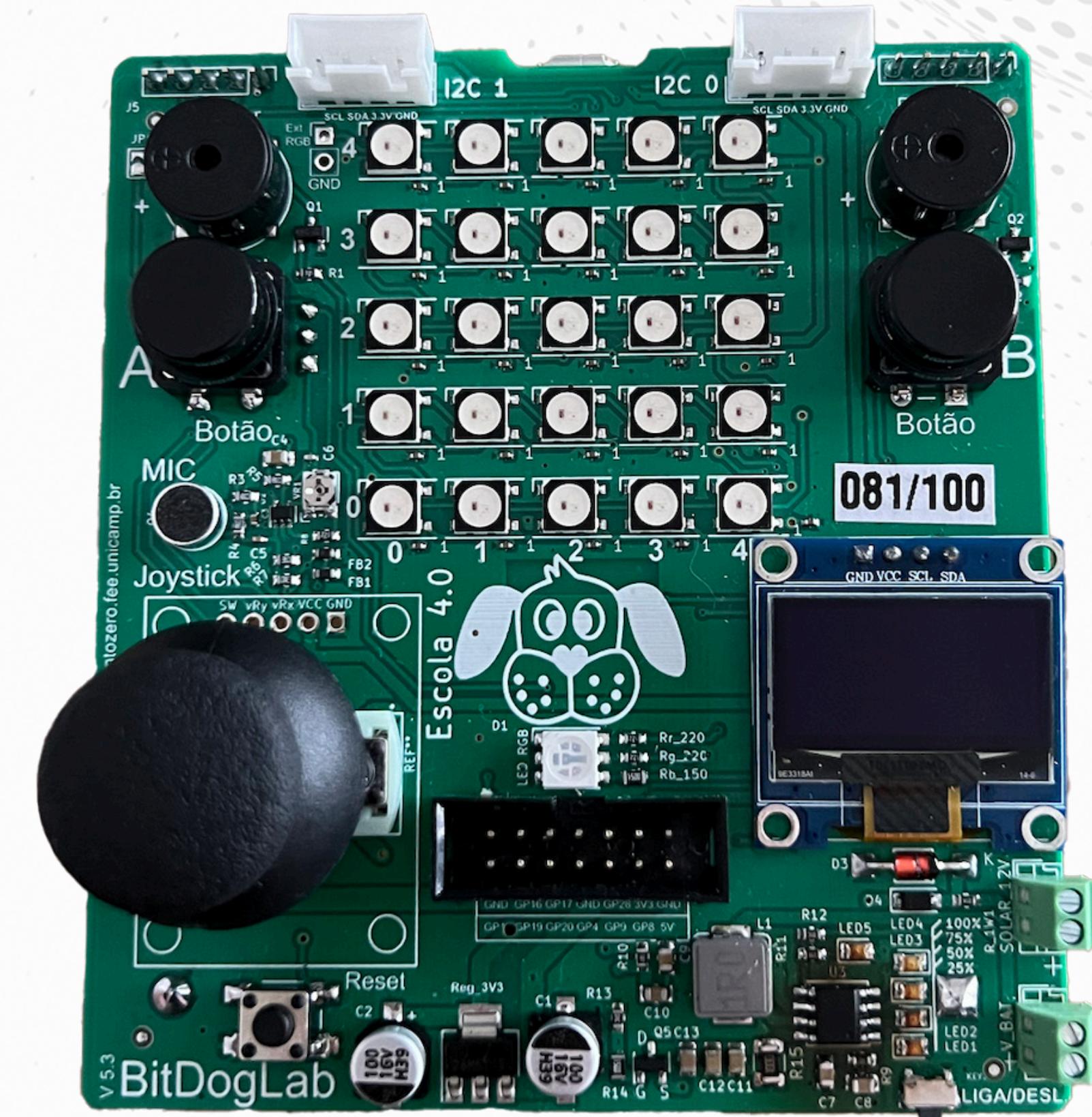


Fig. 6 - BitDogLab

A imagem mostra uma placa eletrônica verde com diversos componentes. Na parte superior da placa, há uma matriz de 4x4 botões pequenos, totalizando 16 botões. À esquerda da matriz, há um joystick, semelhante ao de controles de videogame. Logo abaixo do joystick, há alguns conectores e componentes menores.

Na parte direita da placa, ao lado da matriz de botões, há uma pequena tela (provavelmente uma tela OLED) e dois botões maiores acima dela. Próximo à parte inferior da placa, há uma série de conectores e componentes eletrônicos adicionais, incluindo um conector central.

A placa tem as marcas "BitDogLab" na parte inferior, e há outros textos e logotipos técnicos impressos na superfície.

Fonte: <https://github.com/BitDogLab/BitDogLab/raw/main/doc/illustrations/joystick-assembled.png>

# RaspberryPi Pico W

## Placa de microcontrolador

- Baseado no RP2040
- Baixo custo e flexível
- Interface sem-fiode 2.4GHz
  - » 802.11
  - » Bluetooth 5.2
- Porta micro-usbtipo B
  - » Alimentação e dados
- Porta de depuração SWD
- SDK simples de usar

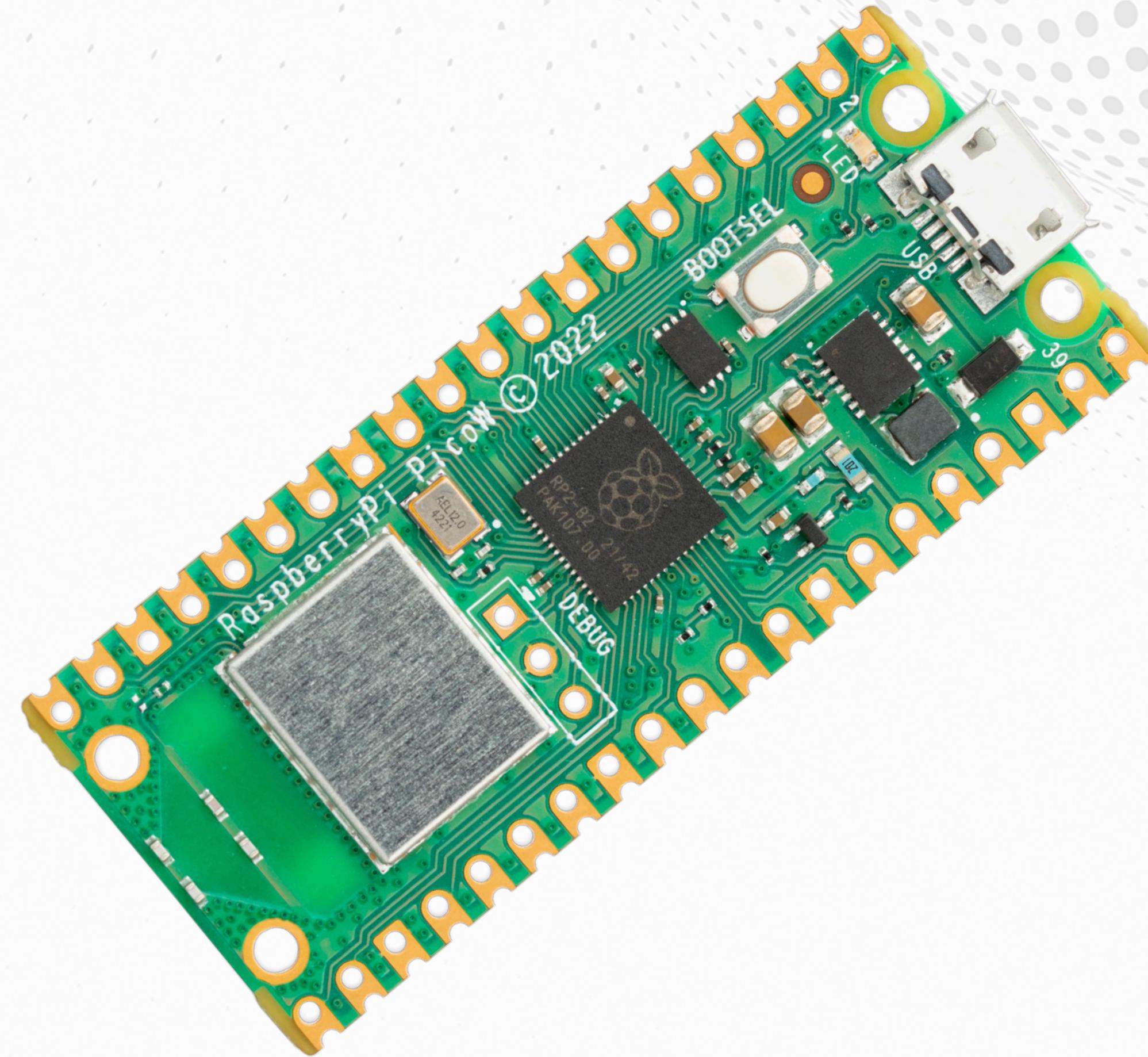
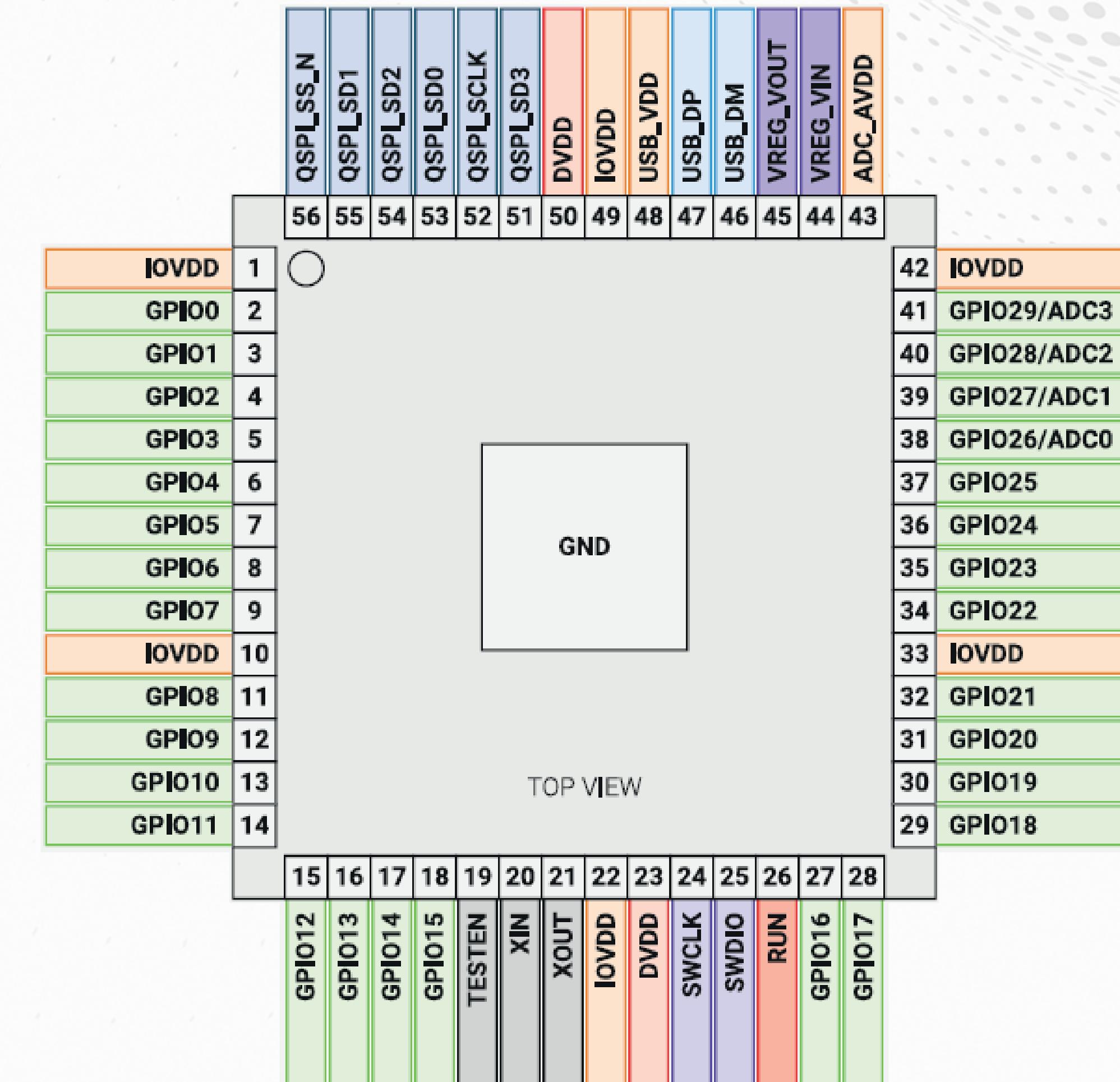


Fig. 7 - Placa de microcontrolador

A imagem mostra uma placa de desenvolvimento Raspberry Pi Pico W. Componentes Principais: No canto superior direito, há uma porta micro-USB, utilizada para alimentação e transferência de dados. Perto da porta USB, encontra-se um botão marcado como "BOOTSEL", utilizado para colocar o dispositivo em modo de carregamento de firmware. No centro da placa, há um chip microcontrolador RP2040 com o logotipo do Raspberry Pi. À esquerda do chip, há um módulo de metal, que abriga o chip Wi-Fi, uma das principais diferenças da versão "W" da placa. Vários outros componentes menores, como capacitores e resistores, estão distribuídos pela placa. Pinos de Conexão: A placa tem pinos de conexão (GPIO) em ambas as bordas, em um total de 20 pinos de cada lado. Esses pinos são utilizados para conectar a placa a outros componentes e sensores externos.

# RP2040

- Primeiro MCU do RaspberryPi
  - Alta performance
  - Baixo custo
  - Fácil de Usar
  - Recursos Avançados
    - » Dual-Core
    - » Rico Conjunto de Periféricos
    - » ProgrammableI/O



**Fig. 8 - RP2040**

A imagem mostra um diagrama de uma placa de circuito impresso, com a disposição dos pinos e conexões. O diagrama é retangular e mostra a parte superior da placa, com a inscrição "Top View" no centro. Ao longo das bordas da placa, existem vários pinos numerados de 1 a 42, com diferentes funções. Os pinos são organizados em linhas e colunas, e cada pino é identificado com uma abreviação, como "GPIO", "VDD", "GND", "SCL", etc. Além dos pinos, o diagrama mostra um retângulo no centro da placa, com a inscrição "GND", que representa a conexão à terra. O diagrama é colorido, com cada grupo de pinos recebendo uma cor diferente.

# RP2040

- Dual CortexM0+, 133 MHz
- 264kB SRAM
- Barramento
- 30 GPIO
- SPI Flash
- Vários periféricos
- 4 Canais ADC
  - » 500 ksps
  - » 12 bit
- USB 1.1

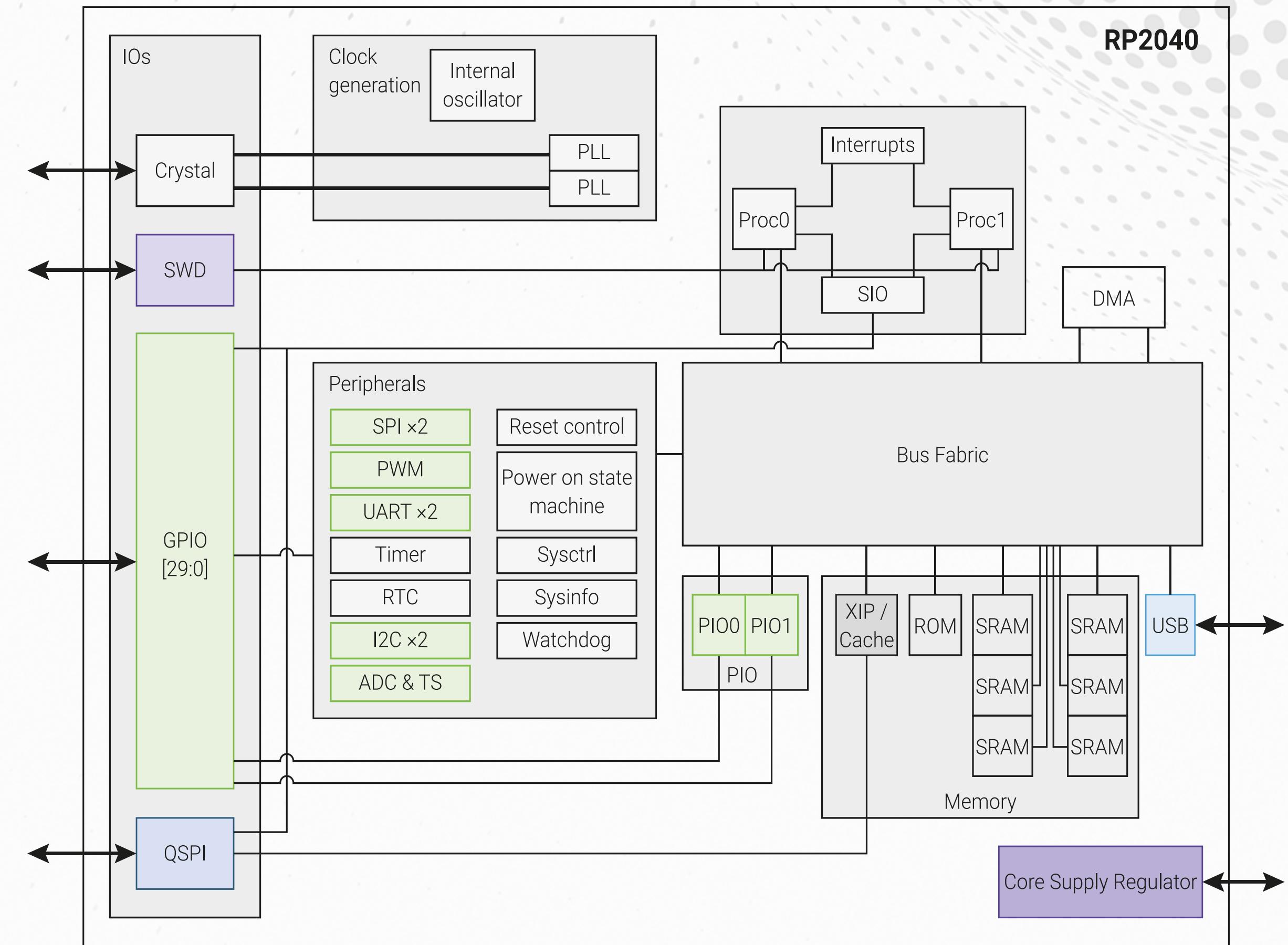


Fig. 9 - RP2040

Fonte: <https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf> página 11

A imagem é um diagrama de blocos que descreve a arquitetura de um microprocessador chamado RP2040. O diagrama mostra os principais componentes do microprocessador, como o núcleo do processador, a memória, os periféricos e as interfaces de comunicação.

O diagrama é organizado em uma estrutura hierárquica, com os componentes principais dispostos em blocos separados. No topo do diagrama, está o bloco "RP2040", que representa o microprocessador como um todo. Abaixo dele, estão os blocos "Clock generation", "Internal oscillator", "PLL", "Interrupts", "Proc0", "Proc1", "DMA", "Peripherals", "Reset control", "Bus Fabric", "PIO", "XIP/Cache", "ROM", "SRAM", "USB", "Memory" e "Core Supply Regulator".

Cada bloco é conectado a outros blocos por linhas que representam as interfaces de comunicação entre eles. As interfaces de comunicação são identificadas por seus nomes, como "SWD", "QSPI", "GPIO", "SPI", "UART", etc. Os blocos também contêm texto que descreve suas funções, como "Clock generation", "Internal oscillator", "Peripherals", etc.

O diagrama representa a interconexão dos diferentes componentes do microprocessador, mostrando como eles interagem entre si e como o microprocessador funciona como um todo.

# RP2040

- Recursos

- » Clock
- » SWD
- » PWM
- » Timer
- » RTC
- » SPI
- » UART
- » I2C
- » ADC
- » PIO

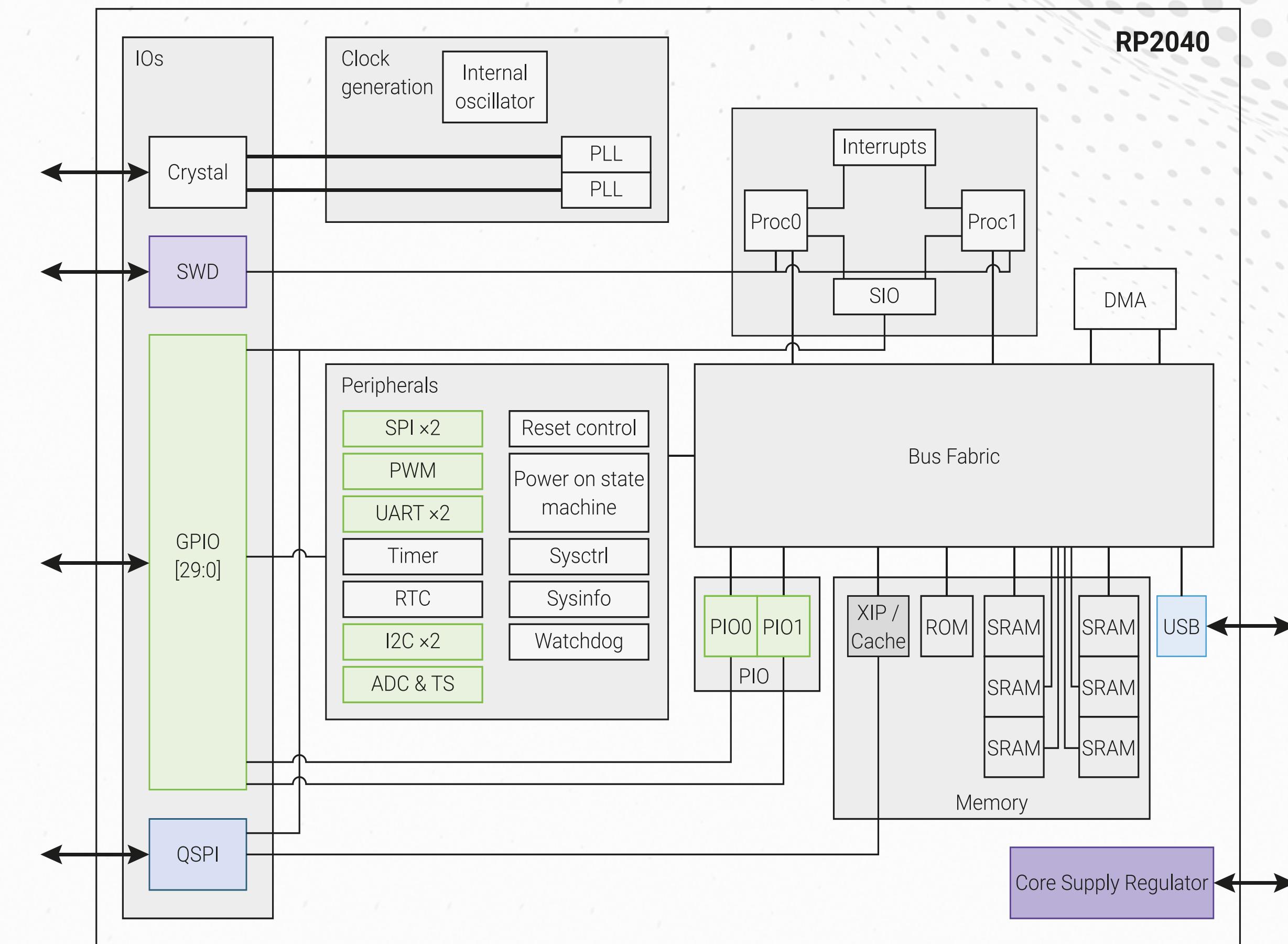


Fig. 10 - RP2040

A imagem é um diagrama de blocos que descreve a arquitetura de um microprocessador chamado RP2040. O diagrama mostra os principais componentes do microprocessador, como o núcleo do processador, a memória, os periféricos e as interfaces de comunicação.

O diagrama é organizado em uma estrutura hierárquica, com os componentes principais dispostos em blocos separados. No topo do diagrama, está o bloco "RP2040", que representa o microprocessador como um todo. Abaixo dele, estão os blocos "Clock generation", "Internal oscillator", "PLL", "Interrupts", "Proc0", "Proc1", "DMA", "Peripherals", "Reset control", "Bus Fabric", "PIO", "XIP/Cache", "ROM", "SRAM", "USB", "Memory" e "Core Supply Regulator".

Cada bloco é conectado a outros blocos por linhas que representam as interfaces de comunicação entre eles. As interfaces de comunicação são identificadas por seus nomes, como "SWD", "QSPI", "GPIO", "SPI", "UART", etc. Os blocos também contêm texto que descreve suas funções, como "Clock generation", "Internal oscillator", "Peripherals", etc.

O diagrama representa a interconexão dos diferentes componentes do microprocessador, mostrando como eles interagem entre si e como o microprocessador funciona como um todo.

# Organização do SDK

O SDK provê recursos para criar aplicações C/C++ e Assembly

---

- Bibliotecas
  - » Padrão do C e C++
  - » API
    - Periféricos
    - Interrupções
    - DMA
    - Alto Nível
      - USB, Programação Multi-Core
- Build system

# Organização do SDK

## Exemplo de Aplicação

Fig. 11 - RP2040

A imagem exibe um trecho de código-fonte escrito em C para um microcontrolador, utilizando a biblioteca "pico/stdlib.h", que provavelmente é para o Raspberry Pi Pico. O código realiza a inicialização e controle de um LED conectado ao pino 12, fazendo-o piscar em intervalos de tempo.

Aqui está uma descrição detalhada do que o código faz:

1. Inclusão de bibliotecas:

- '#include <stdio.h>': Inclui a biblioteca padrão para entrada e saída de dados.
- '#include "pico/stdlib.h"': Inclui as funções necessárias para trabalhar com o Raspberry Pi Pico.

2. Definição de constante:

- `const uint led\_pin\_red=12;`: Define que o pino 12 será usado para controlar um LED.

3. Função principal ('main'):

- Define uma variável 'a' do tipo inteiro sem sinal ('uint a=100;').
- Inicia o pino ('gpio\_init(led\_pin\_red);'): Configura o pino 12 para ser utilizado.
- Define a direção do pino ('gpio\_set\_dir(led\_pin\_red, GPIO\_OUT);'): Configura o pino 12 como saída.
- Inicializa o sistema de entrada/saída padrão ('stdio\_init\_all();').

4. Loop infinito ('while(true)'):

- Dentro do loop, a variável 'a' é incrementada a cada iteração.
- Se o valor de 'a' for par ('if(a % 2)'), o código:
  - Exibe a mensagem "Blinking!" no console.
  - Liga o LED ('gpio\_put(led\_pin\_red, true);').
  - Espera 1 segundo ('sleep\_ms(1000);').
  - Desliga o LED ('gpio\_put(led\_pin\_red, false);').
  - Espera mais 1 segundo.

Este código faz o LED piscar continuamente, com uma pausa de 1 segundo entre cada estado (ligado e desligado).

```
#include <stdio.h>
#include "pico/stdlib.h"
const uint led_pin_red = 12;
int main(){
    uint a = 100;

    gpio_init(led_pin_red);
    gpio_set_dir(led_pin_red, GPIO_OUT);
    stdio_init_all();
    while (true){

        a++;
        if (a % 2)
            printf("Blinking!\r\n");
        gpio_put(led_pin_red, true);
        sleep_ms(1000);
        gpio_put(led_pin_red, false);
        sleep_ms(1000);
    }
}
```

# Organização do SDK

## Exemplo de Aplicação

- CMake

Fig. 12 - RP2040

A imagem exibe um arquivo de configuração do CMake, uma ferramenta usada para gerenciar a construção de projetos em C/C++, geralmente em sistemas embarcados como o Raspberry Pi Pico. O arquivo está configurado para compilar um projeto chamado "blink", que provavelmente faz um LED piscar.

Aqui está uma descrição do que cada linha faz:

1. `cmake\_minimum\_required(VERSION 3.21)`:
  - Especifica que a versão mínima do CMake para construir este projeto é 3.21.
2. `include(\$ENV{PICO\_SDK\_PATH}/external/pico\_sdk\_import.cmake)`:
  - Inclui o caminho do SDK (kit de desenvolvimento de software) do Raspberry Pi Pico. Ele usa a variável de ambiente 'PICO\_SDK\_PATH' para encontrar o SDK.
3. `project(pico-projects C CXX ASM)`:
  - Define o nome do projeto como "pico-projects" e informa que ele usa as linguagens C, C++ e Assembly.
4. `pico\_sdk\_init()`:
  - Inicializa o SDK do Raspberry Pi Pico.
5. `add\_executable(blink blink.c)`:
  - Cria um executável chamado "blink" a partir do arquivo de código-fonte "blink.c".
6. `target\_link\_libraries(blink pico\_stdl�)`:
  - Vincula o executável "blink" à biblioteca padrão do Pico ('pico\_stdl�'), que contém funções comuns para desenvolvimento.
7. `pico\_add\_extra\_outputs(blink)`:
  - Gera saídas adicionais para o projeto "blink", como arquivos binários e UF2, que são usados para carregar o programa no microcontrolador.

Essa configuração prepara o ambiente de desenvolvimento para compilar e gerar os arquivos necessários para o projeto "blink" no Raspberry Pi Pico

```
cmake_minimum_required(VERSION 3.21)
include($ENV{PICO_SDK_PATH}/external/pico_sdk_import.cmake)

project(pico-projects C CXX ASM)

pico_sdk_init()

add_executable(blink
    blink.c
)

target_link_libraries(blink pico_stdl�)

pico_add_extra_outputs(blink)
```

# Organização do SDK

## Estrutura das Bibliotecas

---

- CBibliotecas de Alto Nível
  - » pico\_XXX
  - » APIs que lidam com vários periféricos
    - pico\_time
- Suporte ao Ambiente de Execução
  - » Setup e inicialização
- Hardware
  - » hardware\_XXX
  - » Suporte
  - » Registradores

# Principais Pontos

- Desenvolvimento em Microcontroladores
  - » BitDogLab
- Arquitetura e Recursos do RP2040
  - » Dual-Core CortexM0+, 256 kB SRAM, periféricos e PIO
- SDK do RaspberryPi Pico
- Configuração do CMakeList
  - » Configuração do processo de build

# Introdução ao Desenvolvimento de Software Embarcado com Microcontroladores