Autor: Leonardo Bonifácio Vieira Santos

Passos necessários:

- 1. Ter o vs code instalado e todas as extensões necessárias para o projeto(extensões de C/C++, cmake, raspberry pico, wokwi , etc)

- 2. Instalar e configurar o gcc(que é o compilador de C/C++ no windows, já que o gcc ja vem instalado no linux e mac), colocando ele no caminho das variáveis de ambiente para que seja acessado por todo o computador.
  Link para download: https://sourceforge.net/projects/gcc-win64/

- 3. Instalar e configurar uma variável de ambiente para o compilador Arm gcc.
  Link para download:
  https://developer.arm.com/-/media/Files/downloads/gnu/13.3.rel1/binrel/arm-gnu-toolchain-13.3.rel1-mingw-w64-i686-arm-none-eabi.exe

- 4. Configurar as variáveis de ambiente do sdk do pico após instalar o sdk mais recente(2.1.0) pelo vs code ao inicializar o projeto com a extensão do raspberry pico(new project c/c++ project, escolher versão mais recente e escolher a raspberry pico w, já que é a placa microcontroladora presente na bitdoglab).

- 5. Desenvolver o código necessário para criar um sinal de SOS em código morse utilizando 1 led na GPIO13 da Raspberry pico W.

- 6. Configurar sua licença gratuita de 30 dias do wokwi pelo vs code.

- 7. Criar os arquivos necessários para usar o wokwi(.toml e .json)

- 8. Configurar os dois arquivos wokwi.toml(colocando a versão e caminhos para os arquivos uf2 e elf após já ter compilado o arquivo.c e gerado os mesmos) e diagram.json(colocando a versão,autor,editor,quais partes você irá utilizar e que conexões serão necessárias)
- 9. Caso queira ‘Rodar’ o projeto na BitDogLab basta plugar ela no pc colocar em modo bootsel e arrastar o arquivo uf2 no explorador de arquivos em cima do nome da bitDogLab que aparecerá como um flash drive ou apenas clicar em run se você tiver instalado e configurado o zadig

Imagens abaixo

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

// Definições de tempos em milissegundos
#define TEMPO_DOS_PONTOS 200      // Duração do ponto (.) em milissegundos
#define TEMPO_DOS_TRACOS 800     // Duração do traço (-) em milissegundos
#define TEMPO_ENTRE_PONTOS_E_TRACOS 125      // Intervalo entre pontos/traços no mesmo caractere
#define TEMPO_ENTRE_LETRAS 250 // Intervalo entre letras
#define TEMPO_ENTRE_PALAVRAS 3000  // Intervalo entre palavras

// Pino do LED
#define LED_PIN 13

// Função para piscar o LED por um tempo específico
void blink(int duracao) {
    gpio_put(LED_PIN, 1); // Liga o LED
    sleep_ms(duracao);   // Aguarda o tempo especificado
    gpio_put(LED_PIN, 0); // Desliga o LED
    sleep_ms(TEMPO_ENTRE_PONTOS_E_TRACOS); // Aguarda o intervalo entre sinais
}

// Função para emitir o sinal SOS em código Morse
void sos() {
    // S (3 pontos)
    for (int i = 0; i < 3; i++) {
        blink(TEMPO_DOS_PONTOS);
    }
    sleep_ms(TEMPO_ENTRE_LETRAS);

    // O (3 traços)
    for (int i = 0; i < 3; i++) {
        blink(TEMPO_DOS_TRACOS);
    }
    sleep_ms(TEMPO_ENTRE_LETRAS);

    // S (3 pontos)
    for (int i = 0; i < 3; i++) {
        blink(TEMPO_DOS_PONTOS);
    }
    sleep_ms(TEMPO_ENTRE_PALAVRAS);
}

// Função principal
int main() {
    // Inicializa o SDK e configura o pino do LED como saída
    stdio_init_all();
    gpio_init(LED_PIN);
    gpio_set_dir(LED_PIN, GPIO_OUT);

    while (1) {
        sos(); // Chama a função para emitir o padrão SOS
    }

    return 0;
}
```

```json
{
    "version": 1,
    "author": "Leonardo Bonifácio Vieira Santos",
    "editor": "wokwi",
    "parts": [
      { "type": "board-pi-pico-w", "id": "pico", "top": -41.65, "left": -63.65, "attrs": {} },
      { "type": "wokwi-led", "id": "led1", "top": -90, "left": -149.8, "attrs": { "color": "red" } },
      {
        "type": "wokwi-resistor",
        "id": "r1",
        "top": 128.75,
        "left": -201.6,
        "attrs": { "value": "1000" }
      }
    ],
    "connections": [
      [ "pico:GP0", "$serialMonitor:RX", "", [] ],
      [ "pico:GP1", "$serialMonitor:TX", "", [] ],
      [ "pico:GP13", "led1:A", "green", [ "h0" ] ],
      [ "r1:1", "led1:C", "black", [ "v-48", "h57.6" ] ],
      [ "r1:2", "pico:GND.4", "black", [ "h46.8", "v-28.8" ] ]
    ],
    "dependencies": {}
  }
```

```
[wokwi]
version = 1
firmware = 'build/codigo_morse.uf2'
elf = 'build/codigo_morse.elf'
```