# Roteiro de Experimento + Exercícios: Interrupções (ISR) na BitDogLab

## Parte 1: Roteiro do Experimento

### 1. Título do Experimento

Controle de LED via Interrupções com Dois Botões na BitDogLab

### 2. Objetivo

- Compreender o uso de múltiplas interrupções externas.  
- Programar uma ISR capaz de identificar qual botão foi pressionado.  
- Acender ou apagar um LED de acordo com o botão pressionado.

### 3. Materiais Necessários

- Placa BitDogLab.  
- Jumpers (se necessário).  
- Computador com VSCode e Pico SDK configurado.

### 4. Diagrama de Montagem

• Usar botão A (GPIO5) e botão B (GPIO6) da BitDogLab.  
• Usar LED conectado ao GPIO11 (LED1 BDL).

### 5. Procedimento

#### 5.1. Inicialização do projeto

• Criar um novo projeto no VSCode com suporte ao Pico SDK.  
• Nome sugerido: isr\_led\_2botoes\_bitdoglab.

#### 5.2. Código Fonte Básico

#include "pico/stdlib.h"  
#include "hardware/gpio.h"  
  
#define BOTAO\_A 5  
#define BOTAO\_B 6  
#define LED 11  
  
void isr\_botoes(uint gpio, uint32\_t events) {  
 if (gpio == BOTAO\_A) {  
 gpio\_put(LED, 1); // Acende o LED  
 } else if (gpio == BOTAO\_B) {  
 gpio\_put(LED, 0); // Apaga o LED  
 }  
}  
  
int main() {  
 stdio\_init\_all();  
  
 gpio\_init(LED);  
 gpio\_set\_dir(LED, GPIO\_OUT);  
  
 gpio\_init(BOTAO\_A);  
 gpio\_set\_dir(BOTAO\_A, GPIO\_IN);  
 gpio\_pull\_up(BOTAO\_A);  
  
 gpio\_init(BOTAO\_B);  
 gpio\_set\_dir(BOTAO\_B, GPIO\_IN);  
 gpio\_pull\_up(BOTAO\_B);  
  
 gpio\_set\_irq\_enabled\_with\_callback(BOTAO\_A, GPIO\_IRQ\_EDGE\_FALL, true, &isr\_botoes);  
 gpio\_set\_irq\_enabled(BOTAO\_B, GPIO\_IRQ\_EDGE\_FALL, true);  
  
 while (true) {  
 tight\_loop\_contents();  
 }  
}

### 6. Análises e Observações

- Observar o comportamento do LED ao pressionar o botão A (GPIO5) e o botão B (GPIO6).  
- Analisar o funcionamento rápido da resposta.  
- Verificar se múltiplos apertos rápidos são detectados corretamente.  
- Discutir:  
 • Por que evitamos usar sleep\_ms() ou printf() dentro da ISR?  
 • Como o sistema diferencia qual botão foi pressionado?

### 7. Extensões e Desafios

- Adicionar contagem de quantas vezes o LED foi aceso ou apagado.  
- Implementar debounce via software.  
- Trocar o LED usado para GPIO12 ou GPIO13.

### 8. Importante

- A função gpio\_set\_irq\_enabled\_with\_callback() \*\*só precisa ser chamada uma vez\*\* para registrar o callback da ISR.  
- Para os outros GPIOs, usamos apenas gpio\_set\_irq\_enabled() para habilitar a interrupção.  
- ISR deve ser rápida e não pode usar funções bloqueantes como sleep ou printf.  
- Sempre proteger variáveis compartilhadas caso sejam usadas no main loop e na ISR.

## Parte 2: Exercícios de Modificação

### Introdução

Os exercícios a seguir visam aprofundar o entendimento sobre interrupções externas, manipulação de GPIOs e boas práticas de programação embarcada.

### Exercícios

#### 1. Inverter a lógica dos botões

- Botão A deve apagar o LED.  
- Botão B deve acender o LED.

#### 2. Acender LED1 no Botão A e LED2 no Botão B

- Use GPIO11 para LED1 e GPIO12 para LED2.  
- Cada botão controla seu próprio LED.

#### 3. Implementar contador de cliques

- Contar quantas vezes o LED foi aceso e quantas vezes foi apagado.  
- Exibir os valores utilizando printf() no main loop.

#### 4. Implementar debounce de botão

- Após detectar a interrupção, aplicar uma pequena espera (10–20 ms) no main loop para confirmar o estado do botão antes de agir.

#### 5. Fazer o LED piscar 3 vezes no Botão A

- Quando o Botão A for pressionado, o LED deve piscar rapidamente 3 vezes.

#### 6. Trocar interrupção para borda de subida

- Configurar a interrupção para detectar quando o botão é solto, e não quando é pressionado.

#### 7. Acionar múltiplos LEDs simultaneamente

- Botão A deve acender LED11 e LED12.  
- Botão B deve apagar ambos os LEDs.

#### 8. Diferenciar ações rápidas e longas

- Pressão rápida do botão: acende o LED.  
- Pressão longa (>2 segundos): piscar o LED enquanto o botão permanecer pressionado.

### Resumo dos Conceitos Trabalhados

Cada exercício proposto reforça conceitos importantes no desenvolvimento de sistemas embarcados:  
- Flexibilidade na lógica de controle.  
- Manipulação de múltiplos GPIOs.  
- Comunicação segura entre ISR e o loop principal.  
- Robustez contra ruídos mecânicos (debounce).  
- Controle de temporização e efeitos visuais.  
- Detecção de diferentes tipos de eventos em botões.