# Anexo A4-2 – Exclusão Mútua com Spinlock no RP2040

Este experimento demonstra como utilizar mecanismos de exclusão mútua baseados em spinlocks no RP2040 para proteger seções críticas do código em ambientes concorrentes.

## Objetivo

Utilizar o recurso de spinlocks do RP2040 para proteger o acesso à variável global 'contador' de forma segura entre interrupções (ISR) e o código principal (loop).

## Descrição do Funcionamento

Neste experimento, o contador é incrementado tanto dentro da função principal quanto dentro da ISR acionada por um botão (GPIO 5). Para garantir que não ocorram condições de corrida (race conditions), usamos um spinlock exclusivo da arquitetura RP2040: SPINLOCK\_ID\_USER0.

## Código Fonte

#include <stdio.h>  
#include "pico/stdlib.h"  
#include "hardware/sync.h"  
#include "hardware/structs/sync.h" // Para SPINLOCK\_ID\_USER0  
  
#define BOTAO 5  
  
volatile int contador = 0;  
spin\_lock\_t \*lock;  
  
void gpio\_callback(uint gpio, uint32\_t events) {  
 if (gpio == BOTAO && (events & GPIO\_IRQ\_EDGE\_FALL)) {  
 uint32\_t flags = spin\_lock\_blocking(lock);  
 contador++;  
 spin\_unlock(lock, flags);  
 }  
}  
  
int main() {  
 stdio\_init\_all();  
 sleep\_ms(2000);  
  
 lock = spin\_lock\_instance(SPINLOCK\_ID\_USER0);  
  
 gpio\_init(BOTAO);  
 gpio\_set\_dir(BOTAO, GPIO\_IN);  
 gpio\_pull\_up(BOTAO);  
 gpio\_set\_irq\_enabled\_with\_callback(BOTAO, GPIO\_IRQ\_EDGE\_FALL, true, &gpio\_callback);  
  
 while (true) {  
 uint32\_t flags = spin\_lock\_blocking(lock);  
 contador++;  
 printf("Contador: %d\n", contador);  
 spin\_unlock(lock, flags);  
 sleep\_ms(200);  
 }  
}

## Como Testar

1. Conecte um botão ao pino GPIO 5 do RP2040, com resistor de pull-down externo, se necessário.  
2. Compile o código e grave no RP2040.  
3. Abra o terminal serial (USB).  
4. Observe o contador sendo incrementado periodicamente no terminal.  
5. Pressione o botão: o contador será incrementado também dentro da ISR.

## Conclusão

Este experimento demonstrou o uso de spinlocks no RP2040 como ferramenta eficiente de exclusão mútua, especialmente em situações em que uma interrupção pode modificar a mesma variável usada no fluxo principal do programa.