# Roteiro de Experimento: Leitura de ADC com Timer e ISR na BitDogLab

## 1. Título do Experimento

Leitura periódica do valor analógico de um joystick usando ADC e Timer com interrupção na BitDogLab

## 2. Objetivo

- Compreender a configuração e leitura do Conversor Analógico-Digital (ADC) do RP2040.  
- Utilizar um Timer periódico para acionar uma ISR que lê o valor do ADC.  
- Realizar ações de controle (acender/apagar LED) de acordo com o valor lido.

## 3. Materiais Necessários

- Placa BitDogLab.  
- Joystick analógico integrado à placa.  
- LED integrado (GPIO12).  
- Computador com VSCode e Pico SDK configurado.

## 4. Diagrama de Montagem

• Joystick eixo Y conectado ao GPIO27 (ADC1).  
• LED Vermelho conectado ao GPIO12 (ânodo comum).  
• (Toda a montagem já é interna à BitDogLab.)

## 5. Procedimento

### 5.1. Inicialização do projeto

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

### 5.2. Código Fonte Básico

#include <stdio.h>  
#include "pico/stdlib.h"  
#include "hardware/adc.h"  
#include "hardware/timer.h"  
  
#define JOY\_Y\_ADC\_CHANNEL 1  
#define LED\_VERMELHO 12  
#define INTERVALO\_MS 50  
#define LIMITE\_ACIONAMENTO 3500  
  
volatile bool acionado = false;  
  
bool leitura\_joystick\_callback(struct repeating\_timer \*t) {  
 adc\_select\_input(JOY\_Y\_ADC\_CHANNEL);  
 uint16\_t valor = adc\_read();  
  
 if (valor > LIMITE\_ACIONAMENTO) {  
 if (!acionado) {  
 acionado = true;  
 printf("Acionado! Valor do ADC: %d\n", valor);  
 gpio\_put(LED\_VERMELHO, 0);  
 }  
 } else {  
 if (acionado) {  
 acionado = false;  
 gpio\_put(LED\_VERMELHO, 1);  
 printf("Desativado. Valor do ADC: %d\n", valor);  
 }  
 }  
 return true;  
}  
  
int main() {  
 stdio\_init\_all();  
 sleep\_ms(2000);  
 printf(" Sistema iniciado. Aguarde...\n");  
  
 gpio\_init(LED\_VERMELHO);  
 gpio\_set\_dir(LED\_VERMELHO, GPIO\_OUT);  
 gpio\_put(LED\_VERMELHO, 1);  
  
 adc\_init();  
 adc\_gpio\_init(27);  
  
 struct repeating\_timer timer;  
 add\_repeating\_timer\_ms(INTERVALO\_MS, leitura\_joystick\_callback, NULL, &timer);  
  
 while (true) {  
 tight\_loop\_contents();  
 }  
}

## 6. Análises e Observações

- O LED acende apenas quando o joystick é empurrado além do limite definido.  
- O Timer periódico permite leituras sem bloquear o loop principal.  
- O programa imprime o valor do ADC no terminal.  
- Discuta:  
 • Vantagens de usar Timer + ISR.  
 • Consequências de reduzir demais o intervalo de amostragem.

## 7. Extensões e Desafios

- Ajustar o limite de acionamento.  
- Fazer o LED piscar ao invés de acender continuamente.  
- Usar outro eixo do joystick.  
- Exibir uma barra gráfica proporcional ao valor do ADC no terminal.  
- Implementar uma média móvel de 10 leituras.

## 8. Importante

- Inicializar corretamente o ADC e o Timer.  
- Evitar operações lentas dentro do callback.  
- O Timer garante execução periódica sem bloquear o main loop.