Relatório Técnico do Projeto hid\_composite no Raspberry Pi Pico W

# 1. Introdução

O projeto hid\_composite foi desenvolvido para o microcontrolador Raspberry Pi Pico (RP2040) com o objetivo de implementar uma interface USB HID (Human Interface Device) composta, suportando múltiplas funcionalidades simultaneamente: teclado, mouse, controle de consumidor (Consumer Control) e gamepad.  
Esse relatório analisa a estrutura do projeto, seu funcionamento, e como ele pode ser reutilizado ou customizado para novas implementações HID.

# 2. Estrutura de Diretórios

O projeto apresenta a seguinte estrutura de diretórios e arquivos:  
  
hid\_composite/  
├── CMakeLists.txt  
├── hid\_composite.c  
├── usb\_descriptors.c  
├── usb\_descriptors.h  
├── tusb\_config.h  
└── build/ (diretório gerado após compilação)

# 3. Explicação dos Arquivos Principais

## 3.1 CMakeLists.txt

Arquivo de configuração de build usado pelo CMake. Define o nome do projeto, inclui os diretórios da SDK do Pico, TinyUSB e as dependências necessárias.

## 3.2 hid\_composite.c

Arquivo principal em C. Contém a função main() e as tarefas que controlam os dispositivos HID. Inicializa o hardware, configura os pinos GPIO, inicializa a pilha USB TinyUSB e gerencia o envio dos relatórios HID.

## 3.3 usb\_descriptors.c e usb\_descriptors.h

Contêm as descrições HID USB necessárias para o dispositivo ser reconhecido como um dispositivo HID composto. Especificam os relatórios HID para cada função (teclado, mouse, consumer control, gamepad).

## 3.4 tusb\_config.h

Arquivo de configuração da pilha TinyUSB, onde são definidas as classes HID ativas e outras configurações como tamanho de buffer e número de interfaces USB.

# 4. Funcionamento do Projeto

Ao carregar o firmware no Raspberry Pi Pico, o dispositivo é reconhecido pelo sistema operacional como um dispositivo HID composto. Cada uma das funções HID pode ser ativada via eventos no firmware (exemplo: pressionamento de botões).  
No projeto original, a tecla BOOTSEL é utilizada para gerar eventos, simulando a ação de diferentes dispositivos HID.

# 5. Customização e Reutilização

Este projeto pode ser facilmente customizado para implementar apenas uma ou algumas das interfaces HID suportadas. Para isso, basta modificar os descritores USB no arquivo usb\_descriptors.c e os relatórios enviados no hid\_composite.c.  
Além disso, novos botões ou eventos podem ser mapeados adicionando código para leitura de outros GPIOs e envio dos códigos HID correspondentes.

# 6. Conclusão

O projeto hid\_composite demonstra de forma completa o uso da biblioteca TinyUSB no Raspberry Pi Pico para criar dispositivos HID compostos. Ele serve como um excelente ponto de partida para desenvolvedores que desejam criar periféricos USB personalizados baseados no RP2040.  
Sua modularidade e clareza permitem que partes específicas sejam isoladas ou expandidas para novos projetos HID.