INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ

RESIDÊNCIA TECNOLÓGICA EM SISTEMAS EMBARCADOS

Sistema de Monitoramento Simples

OTÁVIO BRUNO SOUSA MARTINS

Professor-Orientador:

Msc Antônio Santos de Sousa

Parnaíba - PI

2025

RESUMO

Neste trabalho, foi desenvolvido um sistema de monitoramento simples utilizando o FreeRTOS em um microcontrolador Raspberry Pi Pico. O sistema simula o funcionamento de um botão e um LED, com três tarefas cooperando para realizar as funções de leitura, processamento e controle. A comunicação entre as tarefas foi implementada utilizando uma fila para compartilhar o estado do botão e um semáforo para sincronizar o controle do LED.

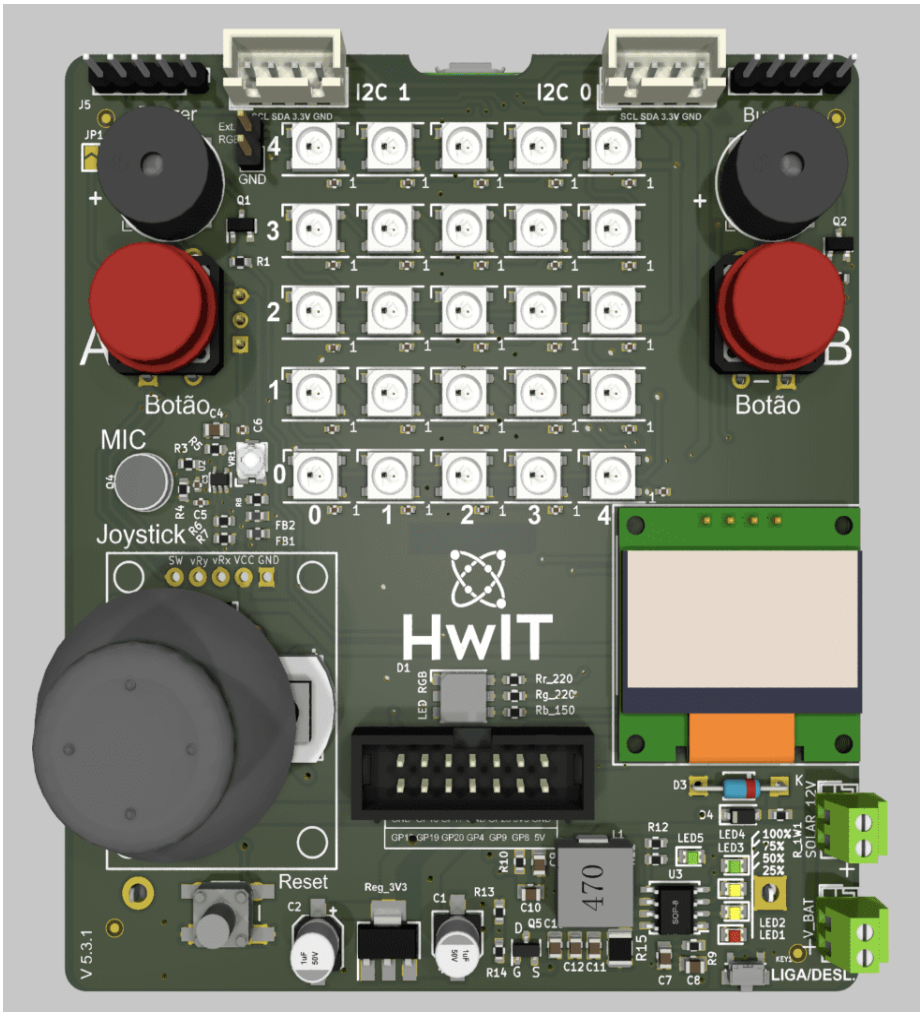
LISTA DE FIGURAS

SUMÁRIO

1. INTRODUÇÃO

Este trabalho apresenta o desenvolvimento de um sistema de monitoramento simples utilizando o FreeRTOS em um microcontrolador Raspberry Pi Pico. O objetivo é simular o funcionamento de um botão e um LED, com três tarefas cooperando para realizar as funções de leitura, processamento e controle. A comunicação entre as tarefas é feita utilizando filas e semáforos, garantindo a sincronização e o funcionamento correto do sistema.

Figura 1 - BitDogLab



Fonte: Hardware Innovation Technologies (2024)

2. DESCRIÇÃO DO SISTEMA

O sistema é composto por três tarefas principais:

Tarefa 1: Leitura do Botão

- Simula a leitura do estado de um botão.
- Alterna o estado do botão (pressionado ou não pressionado) a cada 100ms.
- Envia o estado do botão para uma fila compartilhada.

Tarefa 2: Processamento do Botão

- Recebe o estado do botão da fila.
- Caso o botão esteja pressionado, aciona a próxima tarefa (controle do LED) utilizando um semáforo.
- Caso contrário, aguarda o próximo ciclo de leitura.

Tarefa 3: Controle do LED

- Aguarda o semáforo liberado pela Tarefa 2.
- Alterna o estado do LED (aceso ou apagado).
- Exibe o estado do LED no Serial Monitor.

3. DETALHAMENTO DA IMPLEMENTAÇÃO

Definições de Variáveis

- **buttonStateQueue**: Fila utilizada para compartilhar o estado do botão entre as tarefas de leitura e processamento.
- **ledControlSemaphore**: Semáforo utilizado para sincronizar as tarefas de processamento e controle do LED.
- **LEDState**: Variável booleana que armazena o estado atual do LED (aceso ou apagado).

Comunicação entre Tarefas

- A Tarefa 1 envia o estado do botão para a Tarefa 2 através da fila buttonStateQueue.
- A Tarefa 2 utiliza o semáforo ledControlSemaphore para acionar a Tarefa 3, garantindo a sincronização.

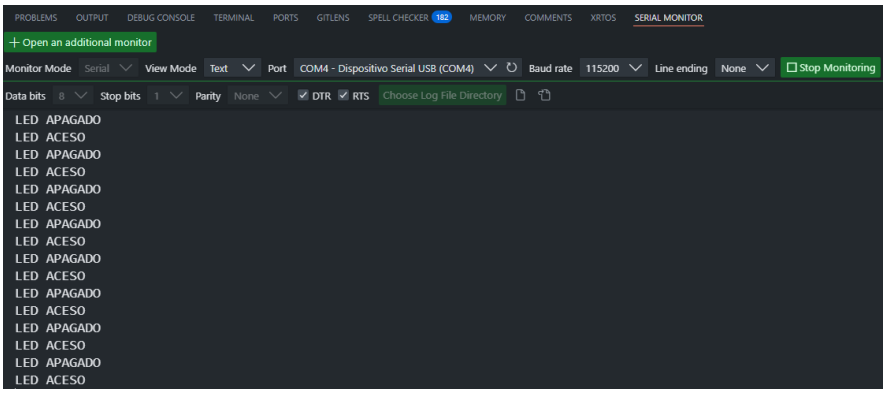
Criação das Tarefas no FreeRTOS

- **Tarefa 1**: Criada com prioridade baixa, executa a cada 100ms.
- **Tarefa 2**: Executa sempre que há um novo estado do botão na fila.
- **Tarefa 3**: Executa apenas quando acionada pelo semáforo.

4. RESULTADOS

O sistema foi implementado com sucesso, e o comportamento esperado foi validado através do Serial Monitor. As mensagens exibidas indicam o estado do LED, alternando entre "LED ACESO" e "LED APAGADO" conforme o botão simulado é pressionado.

Figura 2 - Saída no serial monitor



Fonte: Autoral (2025)

5. CONCLUSÃO

O projeto demonstrou a aplicação prática do FreeRTOS para multitarefa em sistemas embarcados. A utilização de filas e semáforos garantiu a comunicação eficiente entre as tarefas, permitindo a simulação

de um sistema de monitoramento funcional. Este trabalho pode ser expandido para incluir sensores reais ou outras funcionalidades, como controle de múltiplos LEDs ou integração com outros dispositivos.

6. REFERÊNCIAS

Modular Code and How to Structure an Embedded C Project. MicroForum, 2023. Disponível em: https://www.microforum.cc/blogs/entry/46-modular-code-and-how-to-structure-an-embedded-c-project/. Acesso em: 10 mar. 2025.

WETHERELL, Jack. C Project Structure. GitHub, 2023. Disponível em: https://github.com/JackWetherell/c-project-structure. Acesso em: 10 mar. 2025.

VALLINI, Luca. How to Structure C Projects: My Experience & Best Practices. Luca Vallini Blog, 2023. Disponível em: https://www.lucavall.in/blog/how-to-structure-c-projects-my-experience-best-practices. Acesso em: 10 mar. 2025.

A maneira mais fácil de usar a matriz de LEDs da BitDogLab. YouTube, 2023. Disponível em: https://www.youtube.com/watch?v=chQdNiFmVm0&t=262s. Acesso em: 22 jan. 2025.

Raspberry Pi. Documentation: Pico Series. Raspberry Pi, 2023. Disponível em: https://www.raspberrypi.com/documentation/microcontrollers/pico-series.html#picow-technical-specification. Acesso em: 10 jan. 2024.

Banco de Informações de Hardware (BIH). Google Docs, 2023. Disponível em: https://docs.google.com/document/d/13-68OqiU7ISE8U2KPRUXT2ISeBl3WPhXjGDFH52eWlU/edit?tab=t.0. Acesso em: 10 jan. 2025.

BitDogLab. Repositório da BitDogLab. GitHub, 2023. Disponível em: https://github.com/BitDogLab/BitDogLab. Acesso em: 10 jan. 2025.

BitDogLab. Repositório com códigos de exemplo. GitHub, 2023. Disponível em: https://github.com/BitDogLab/BitDogLab-C. Acesso em: 10 jan. 2025.

Barry, Richard. FreeRTOS Reference Manual V10.0.0. FreeRTOS, 2018. Disponível em: https://www.freertos.org/media/2018/FreeRTOS_Reference_Manual_V10.0.0.pdf. Acesso em: 24 mar. 2025.

Barry, Richard. Mastering the FreeRTOS Real-Time Kernel – A Hands-On Tutorial Guide. FreeRTOS, 2018. Disponível em: https://www.freertos.org/media/2018/161204_Mastering_the_FreeRTOS_Real_Time_Kernel-A_Hands-On_Tutorial_Guide.pdf. Acesso em: 24 mar. 2025.

7. ANEXOS

Código-Fonte

https://github.com/otaviossousa/SimpleMonitor-RTOS