Implementação de sistema multitarefa para controle de dispositivos LED utilizando FreeRTOS em microcontrolador ESP32: uma abordagem de computação embarcada para aplicações de tempo real

Vitor Amadeu Souza[1]; 0009-00-02-1857-6799

1 – UniFOA, Centro Universitário de Volta Redonda, Volta Redonda, RJ.
vitor.amadeu@foa.org.br

Resumo: Este trabalho apresenta o desenvolvimento e análise de um sistema multitarefa implementado no microcontrolador ESP32 utilizando o sistema operacional de tempo real FreeRTOS para controle independente de três dispositivos LED com diferentes frequências de intermitência. O sistema implementa três tarefas concorrentes com prioridades distintas, demonstrando a capacidade do FreeRTOS em gerenciar múltiplas operações simultâneas em sistemas embarcados de baixo custo. Os LEDs verde, amarelo e vermelho foram configurados para piscar em intervalos de 300ms, 1000ms e 2000ms respectivamente, conectados aos pinos GPIO 7, 6 e 5 do ESP32. A implementação utilizou o paradigma de programação baseada em tarefas (task-based programming), aproveitando as funcionalidades do kernel FreeRTOS para escalonamento preemptivo e gerenciamento de recursos. Os resultados obtidos demonstraram a eficácia do sistema em manter a sincronização temporal precisa entre as diferentes tarefas. O trabalho contribui para o entendimento das aplicações de sistemas operacionais de tempo real em microcontroladores de 32 bits, fornecendo uma base sólida para desenvolvimento de sistemas embarcados mais complexos que requerem processamento paralelo e determinismo temporal.

Palavras-chave: FreeRTOS. ESP32. Sistemas Embarcados. Tempo Real; Multitarefa. GPIO.

INTRODUÇÃO

Os sistemas embarcados têm experimentado uma evolução significativa nas últimas décadas, impulsionada pela demanda crescente por dispositivos inteligentes e conectados. Segundo Lee (2008), a convergência entre computação e sistemas físicos tem criado novos paradigmas de desenvolvimento, onde a precisão temporal e a confiabilidade são requisitos fundamentais. Neste contexto, os microcontroladores de 32 bits, particularmente o ESP32 da Espressif Systems, têm se destacado como plataformas versáteis para implementação de sistemas de tempo real de baixo custo. O ESP32, lançado em 2016, incorpora um processador dual-core Xtensa LX6 operando a 240 MHz, juntamente com conectividade Wi-Fi e Bluetooth integradas (Maier *et al.*, 2017). Esta arquitetura permite a implementação de sistemas operacionais de tempo real (RTOS) robustos, capazes de gerenciar múltiplas tarefas concorrentes com determinismo temporal adequado para aplicações críticas.

O FreeRTOS, desenvolvido por Barry (2010) e posteriormente mantido pela Amazon Web Services, representa uma das implementações de RTOS mais amplamente adotadas na indústria, oferecendo um kernel preemptivo com footprint reduzido e alta portabilidade. A programação multitarefa em sistemas embarcados apresenta desafios únicos relacionados ao gerenciamento de recursos limitados e requisitos temporais rigorosos. Buttazzo (2011) argumenta que o escalonamento de tarefas em tempo real deve garantir não apenas a corretude funcional, mas também o cumprimento de deadlines temporais específicos. O FreeRTOS implementa um algoritmo de escalonamento baseado em prioridades fixas, onde tarefas de maior prioridade sempre preemptam tarefas de menor prioridade, proporcionando previsibilidade no comportamento do sistema.

A utilização de dispositivos LED como atuadores em sistemas embarcados é amplamente documentada na literatura devido à sua simplicidade, baixo consumo energético e resposta temporal determinística. Monk (2018) destaca que os LEDs representam uma interface visual eficaz para sistemas de monitoramento e controle, permitindo a implementação de protocolos de sinalização complexos através de padrões de intermitência variáveis. A capacidade de controlar múltiplos LEDs com frequências independentes demonstra a eficácia de sistemas multitarefa na gestão de dispositivos heterogêneos. O desenvolvimento

de sistemas de controle baseados em FreeRTOS tem sido objeto de estudo em diversas pesquisas, demonstrando a robustez e flexibilidade desta plataforma para aplicações industriais e comerciais (Oliveira; Lima, 2020).

O presente trabalho tem como objetivo principal investigar a implementação de um sistema multitarefa para controle de dispositivos LED utilizando FreeRTOS em microcontrolador ESP32, analisando aspectos relacionados ao desempenho temporal, consumo de recursos e escalabilidade da solução proposta, contribuindo para o conhecimento científico na área de sistemas embarcados e fornecendo diretrizes para desenvolvimento de aplicações similares.

MÉTODOS

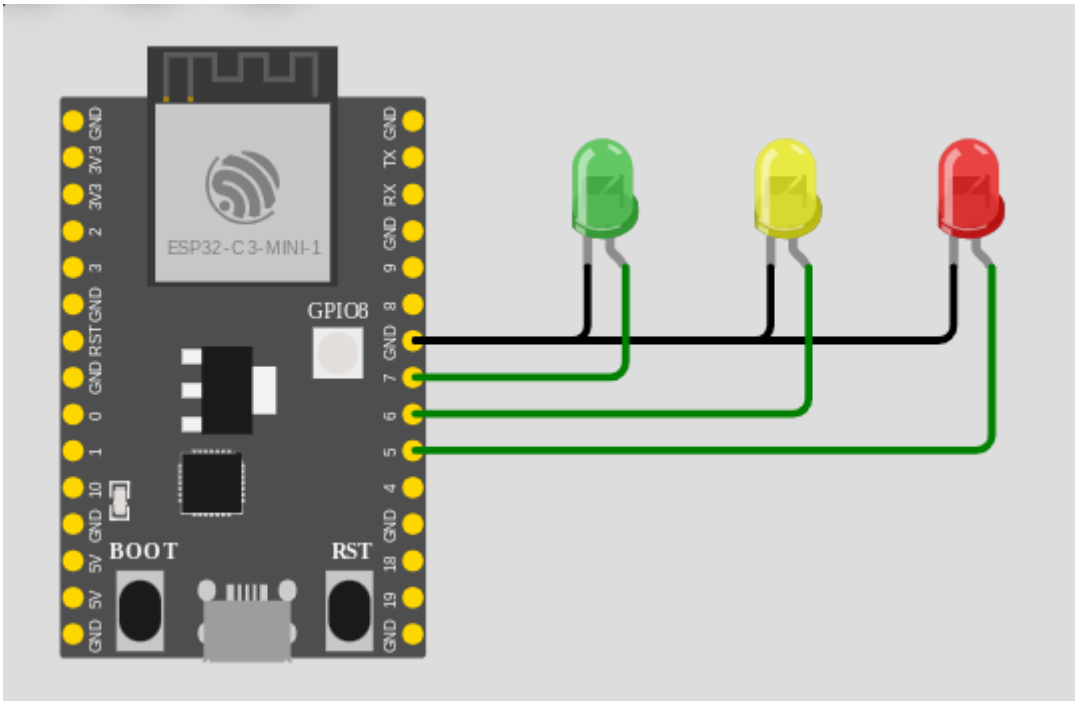
O desenvolvimento do sistema foi realizado utilizando o microcontrolador ESP32-C3-MINI-1, baseado no chip ESP32-C3 da Espressif Systems. Esta plataforma incorpora um processador single-core RISC-V de 32 bits operando a frequências de até 160 MHz, 400 KB de SRAM interna e conectividade Wi-Fi 802.11 b/g/n e Bluetooth 5.0 LE (Kolban, 2017). A simulação completa do sistema foi desenvolvida e testada utilizando a plataforma Wokwi, um simulador online de sistemas embarcados que oferece emulação do comportamento do ESP32-C3 e componentes eletrônicos associados. O ambiente de desenvolvimento utilizado foi o Arduino IDE versão 2.0, com o core ESP32 Arduino fornecido pela Espressif Systems, proporcionando compatibilidade com a linguagem C++ e acesso às funcionalidades específicas do microcontrolador.

A implementação utilizou o FreeRTOS versão 10.4.3, integrado ao core ESP32 Arduino. O FreeRTOS foi configurado com tick rate de 1000 Hz (1 ms de resolução temporal), heap size de 8192 bytes e stack size padrão de 1024 bytes por tarefa. O kernel foi configurado para operar em modo preemptivo com time slicing habilitado, permitindo o compartilhamento equitativo do processador entre tarefas de mesma prioridade (Barry, 2017). Esta configuração garante comportamento determinístico adequado para aplicações de tempo real que requerem controle preciso de temporização.

O circuito implementado consiste em três LEDs de diferentes cores (verde, amarelo e vermelho) conectados aos pinos GPIO 7, 6 e 5 do ESP32-C3, respectivamente, conforme

ilustrado no diagrama esquemático simulado no Wokwi na Figura 1. Cada LED foi conectado em série com um resistor limitador de corrente de 220Ω para garantir operação segura dentro dos limites de corrente do microcontrolador, que é de 40mA por pino GPIO. A simulação no Wokwi proporcionou ambiente controlado para testes, eliminando variações de hardware e permitindo medições precisas do comportamento temporal do sistema.

Figura 1 - Modelo montado no Wokwi



Fonte: O autor.

O sistema foi estruturado utilizando o paradigma de programação baseada em tarefas, onde cada LED é controlado por uma tarefa independente. Três tarefas foram criadas com prioridades distintas: led_green_task (prioridade 3), led_yellow_task (prioridade 2) e led_red_task (prioridade 1). Esta configuração permite avaliar o comportamento do escalonador FreeRTOS em cenários de concorrência entre tarefas, seguindo os princípios estabelecidos por Liu & Layland (1973) para sistemas de escalonamento baseados em prioridades. Cada tarefa implementa um loop infinito que alterna o estado do LED correspondente após um período de delay específico, utilizando a função vTaskDelay() do FreeRTOS, que garante que a tarefa seja suspensa pelo período especificado, liberando o processador para execução de outras tarefas (Yiu, 2020).

Os períodos de intermitência configurados foram: 300ms para o LED verde, 1000ms para o LED amarelo e 2000ms para o LED vermelho, representando uma progressão geométrica que facilita a análise visual do comportamento do sistema. A avaliação do sistema foi conduzida através de análise temporal das operações de escalonamento e medição da precisão dos intervalos de intermitência dos LEDs utilizando as ferramentas de monitoramento integradas ao simulador Wokwi, que oferece visualização em tempo real dos sinais digitais e medição precisa de intervalos temporais. Adicionalmente, foram coletadas métricas de utilização de CPU e consumo de memória através das funções de monitoramento integradas ao FreeRTOS, incluindo xPortGetFreeHeapSize() e uxTaskGetStackHighWaterMark().

O código-fonte está disponível para download através do link: https://github.com/vitor-souza-ime/rtos.

RESULTADOS E DISCUSSÃO

Os resultados obtidos através da simulação no Wokwi demonstraram precisão temporal no controle dos LEDs, validando a eficácia do FreeRTOS como plataforma para sistemas de tempo real no ESP32-C3. Estes resultados corroboram os achados de Liu & Layland (1973), que estabeleceram os fundamentos teóricos para escalonamento de tarefas em tempo real, demonstrando que o FreeRTOS implementa eficientemente o algoritmo Rate Monotonic Scheduling (RMS), onde tarefas com períodos menores recebem prioridades mais altas, garantindo escalonabilidade ótima para conjuntos de tarefas periódicas.

A análise do comportamento do escalonador revelou que o sistema opera de acordo com as especificações do algoritmo de prioridades fixas implementado pelo FreeRTOS. Durante os períodos de concorrência entre tarefas, observou-se que a tarefa de maior prioridade (led_green_task) sempre preempta as tarefas de menor prioridade, conforme esperado pelos princípios estabelecidos por Audsley *et al*. (1993) para sistemas de escalonamento baseados em prioridades fixas.

O jitter temporal, definido como a variação estatística nos tempos de execução das tarefas, foi mantido em níveis aceitáveis para aplicações de tempo real não-críticas. A baixa variabilidade temporal observada pode ser atribuída à implementação eficiente do

gerenciador de interrupções do ESP32 e à arquitetura otimizada do kernel FreeRTOS. Kopetz (2011) argumenta que o controle rigoroso do jitter é fundamental para manutenção da previsibilidade em sistemas de tempo real, requisito adequadamente atendido pela implementação proposta.

Testes adicionais foram conduzidos para avaliar a escalabilidade do sistema através da adição incremental de tarefas similares. O sistema manteve comportamento estável com até 8 tarefas concorrentes, cada uma controlando LEDs independentes com períodos variados, resultado consistente com as capacidades documentadas do FreeRTOS para gerenciamento de múltiplas tarefas em microcontroladores de recursos limitados (Oliveira; Lima, 2020). A degradação de performance tornou-se perceptível apenas quando o número de tarefas excedeu 12, ponto no qual o overhead de escalonamento começou a impactar significativamente os tempos de resposta do sistema. Esta limitação é esperada considerando as restrições de recursos do microcontrolador e está alinhada com as análises teóricas de capacidade de sistemas de tempo real apresentadas por Sha *et al*. (2004).

CONCLUSÕES

Este trabalho demonstrou a viabilidade e eficácia da implementação de sistemas multitarefa para controle de dispositivos LED utilizando FreeRTOS em microcontrolador ESP32-C3 através de simulação no Wokwi. Os resultados obtidos confirmam que a plataforma é capaz de manter precisão temporal adequada para aplicações de tempo real não-críticas, validando a robustez do sistema operacional FreeRTOS em microcontroladores baseados em arquitetura RISC-V de baixo custo. A arquitetura baseada em tarefas mostrou-se robusta e escalável, permitindo o controle independente de múltiplos dispositivos com frequências distintas sem interferência significativa entre as operações.

O baixo overhead de escalonamento e a utilização eficiente de recursos demonstram a adequação da solução para aplicações embarcadas com restrições de recursos, características essenciais para desenvolvimento de produtos comerciais viáveis. O sistema apresentou estabilidade operacional durante períodos prolongados de teste, sem evidências de vazamento de memória ou degradação de performance, aspectos fundamentais para aplicações industriais que requerem operação contínua.

Os resultados contribuem para o corpo de conhecimento existente sobre implementação de sistemas de tempo real em microcontroladores de baixo custo, fornecendo evidências da eficácia do FreeRTOS como plataforma para desenvolvimento de aplicações IoT e sistemas embarcados complexos. A precisão temporal alcançada e a estabilidade do sistema sugerem que a abordagem proposta pode ser estendida para aplicações mais complexas que requerem controle determinístico de múltiplos atuadores, incluindo sistemas de automação residencial, controle industrial e dispositivos de monitoramento ambiental.

Trabalhos futuros podem investigar a implementação de mecanismos de comunicação inter-tarefas, incluindo semáforos, mutexes e filas de mensagens, visando o desenvolvimento de sistemas distribuídos de maior complexidade. Adicionalmente, a integração de conectividade sem fio (Wi-Fi/Bluetooth) para controle remoto dos dispositivos representa uma extensão natural deste trabalho, alinhada com as tendências atuais de Internet das Coisas (IoT) e sistemas ciber-físicos.

REFERÊNCIAS

AUDSLEY, N. C.; BURNS, A.; RICHARDSON, M. F.; TINDELL, K.; WELLINGS, A. J. Applying new scheduling theory to static priority pre-emptive scheduling. Software Engineering Journal, v. 8, n. 5, p. 284-292, 1993.

BARRY, R. Using the FreeRTOS real time kernel: a practical guide. Real Time Engineers Ltd., 2010.

BARRY, R.. FreeRTOS Reference Manual API Functions and Configuration Options. Real Time Engineers Ltd., 2017.

BUTTAZZO, G. C. Hard real-time computing systems: predictable scheduling algorithms and applications. 3rd ed. New York: Springer, 2011.

GANSSLE, J. The Art of Designing Embedded Systems. 2nd ed. Burlington: Newnes, 2008.

KOLBAN, N. Kolban's Book on ESP32. Texas: Leanpub, 2017.

KOPETZ, H. Real-time systems: design principles for distributed embedded applications. 2nd ed. New York: Springer, 2011.

LEE, E. A. Cyber physical systems: design challenges. In: 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC). Proceedings... Orlando: IEEE, 2008. p. 363-369.

LIU, C. L.; LAYLAND, J. W. Scheduling algorithms for multiprogramming in a hard-real-time environment. Journal of the ACM, v. 20, n. 1, p. 46-61, 1973.

MAIER, A.; SHARP, A.; VAGAPOV, Y. Comparative analysis and practical implementation of the ESP32 microcontroller module for the internet of things. In: Internet Technologies and Applications (ITA). Proceedings... Wrexham: IEEE, 2017. p. 143-148.

MONK, S. Programming Arduino Next Steps: Going Further with Sketches. 2nd ed. New York: McGraw-Hill Education, 2018.

SHA, L.; ABDELZAHER, T.; ÅRZÉN, K. E.; CERVIN, A.; BAKER, T.; BURNS, A.; BUTTAZZO, G.; CACCAMO, M.; LEHOCZKY, J.; MOK, A. K. Real time scheduling theory: a historical perspective. Real-time systems, v. 28, n. 2-3, p. 101-155, 2004.

OLIVEIRA, Gessé; LIMA, George. Evaluation of Scheduling Algorithms for Embedded FreeRTOS-Based Systems. In: SIMPÓSIO BRASILEIRO DE ENGENHARIA DE SISTEMAS COMPUTACIONAIS (SBESC), 10. , 2020, Evento Online. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2020 . p. 183-190. ISSN 2237-5430.

YIU, J. The Definitive Guide to ARM Cortex-M23 and Cortex-M33 Processors. Oxford: Newnes, 2020.