Ficha de proposta de projeto

Nome do Aluno: Camila De Araújo Bastos

Polo: Feira De Santana

Data: 26/05/2025

Sistema inteligente de monitoramento de Ocupação para Bibliotecas

Objetivo Geral

O projeto tem como objetivo desenvolver um sistema interativo de controle de acesso para bibliotecas, visando promover organização, segurança e acessibilidade no ambiente. O sistema limita a entrada de usuários com base em uma capacidade máxima de 10 pessoas, fornecendo sinais visuais e sonoros em tempo real para indicar a ocupação atual, a disponibilidade de vagas, a condição de lotação e a reinicialização do sistema. Com isso, busca-se melhorar a gestão do fluxo de pessoas em espaços físicos de uso coletivo.

Descrição Funcional

A interação do usuário com o sistema é feita por três botões físicos: entrada, saída e reset. Esses botões geram interrupções que acionam uma função de tratamento, responsável por identificar qual botão foi pressionado. Para os botões de entrada e saída, é usada uma flag global chamada *evento_botao*, que indica o tipo de ação (1 para entrada, 2 para saída). As tarefas *vEntradaTask* e *vSaidaTask* verificam essa flag e executam a lógica correspondente. Já o botão de reset aciona diretamente a tarefa *vTaskReset*. A seguir, a Tabela 1 apresenta as tarefas do sistema e suas respectivas funcionalidades.

Tabela 1: Tarefas e suas funcionalidades

Tarefa	Funcionalidade
vEntradaTask	Gerencia a entrada de usuários: decrementa o contador de vagas, atualiza o display OLED com o número de vagas e ocupação, aciona o buzzer se lotado, e atualiza os LEDs conforme a ocupação.
vSaidaTask	Gerencia a saída de usuários: incrementa o contador de vagas, atualiza o display OLED com o número de vagas e ocupação, e atualiza os LEDs conforme a ocupação.

vTaskReset	Realiza o reset do sistema: reinicializa o contador de vagas para o máximo, atualiza o display OLED com mensagem de reset, emite dois beeps no buzzer.

Fonte: Próprio autor

Para garantir a sincronização entre as tarefas e evitar condições de corrida, o sistema utiliza diferentes tipos de semáforos fornecidos pelo FreeRTOS. Um semáforo do tipo *counting* (*xCounterSemaphore*) é utilizado para controlar o número de vagas disponíveis, permitindo que as tarefas de entrada e saída ajustem corretamente o contador de ocupação. Um semáforo binário (*xResetSemaphore*) é usado para sinalizar a tarefa de reset sempre que o botão correspondente for pressionado, garantindo que o reset ocorra de forma segura e sincronizada. Além disso, um *mutex* (*xDisplayMutex*) é empregado para proteger o acesso ao display OLED, assegurando que apenas uma tarefa por vez possa escrever na tela, evitando sobreposição ou corrupção de dados visuais durante a operação do sistema.

No código desenvolvido, o semáforo de contagem (*xCounterSemaphore*) é utilizado como mecanismo principal para controlar a quantidade de vagas disponíveis no sistema. Ele é criado com um valor máximo igual ao número total de vagas (*MAX_VAGAS*) e já inicia com todas as unidades disponíveis, representando que todas as vagas estão inicialmente livres. A cada vez que o botão de entrada é pressionado, indicando que uma pessoa está entrando, a tarefa associada tenta ocupar uma vaga utilizando a função *xSemaphoreTake*. Essa chamada decrementa o contador do semáforo, simbolizando que uma vaga foi ocupada. Caso não existam mais vagas disponíveis, a função bloqueia a tarefa até que uma vaga seja liberada. Já quando o botão de saída é pressionado, indicando que uma pessoa está saindo, a tarefa correspondente chama *xSemaphoreGive*, que incrementa o contador do semáforo, liberando assim uma vaga no sistema.

Além disso, o botão de reset permite restaurar o sistema ao estado inicial, e para isso, o código verifica quantas vagas estão ocupadas usando *uxSemaphoreGetCount* e, em seguida, libera vagas até que o semáforo retorne ao valor máximo, garantindo que todas as vagas estejam novamente disponíveis. Dessa forma, o semáforo de contagem garante que o número de pessoas dentro do sistema nunca ultrapasse a capacidade definida, mantendo a lógica de controle de vagas segura e sincronizada entre as tarefas concorrentes.

Para completar, o mutex (*xMutex*) é utilizado como mecanismo de exclusão mútua, garantindo que apenas uma tarefa por vez acesse e modifique recursos compartilhados, como a escrita no display OLED. Como múltiplas tarefas podem tentar acessar o display simultaneamente, o uso do mutex evita condições de corrida e possíveis inconsistências na exibição das informações. Antes de realizar qualquer operação crítica, como escrever no display, a tarefa correspondente adquire o mutex com *xSemaphoreTake*, bloqueando outras tarefas de fazerem o mesmo até que a operação seja concluída. Após terminar a operação, o mutex é liberado com xSemaphoreGive, permitindo que outras tarefas possam acessar o recurso.

Uso dos Periféricos da BitDogLab

Para melhor compreensão da utilização dos periféricos no sistema de controle de vagas com a BitDogLab e FreeRTOS, a Tabela 2 apresenta os detalhes dos componentes empregados, os respectivos pinos GPIO utilizados e suas funções no sistema.

Tabela 2 – Mapeamento dos periféricos utilizados no sistema de controle de vagas

Periféricos	Pinos	Função no sistema
Display OLED	SDA: GPIO 14 SCL: GPIO 15	Exibe mensagens e informações sobre vagas ocupadas e disponíveis
LED RGB	Vermelho: GPIO 13 Verde: GPIO 11 Azul: GPIO 12	Indica o estado da biblioteca: • Azul: vazio • Verde: livre • Verde + Vermelho: quase cheio • Vermelho: cheio
Buzzer	GPIO 21	Emite um beep em caso de lotação e dois beeps em caso de reset
Botão de Entrada (A)	GPIO 5	Sinaliza a entrada de uma pessoa, reduzindo a contagem de vagas
Botão de Saída (B)	GPIO 6	Sinaliza a saída de uma pessoa, aumentando a contagem de vagas
Botão de Reset (SW)	GPIO 22	Restaura a contagem de vagas ao valor máximo (reset do sistema)
FreeRTOS	—	Gerencia tarefas concorrentes e semáforos para controle dos eventos do sistema

Fonte: Próprio autor

Links para acesso ao código e ao vídeo.

Caamilab/Biblioteca

https://youtu.be/1sUmn0zY7PI