Proyecto Final
Curso de Sistemas Operativos y Laboratorio

FreeRTOS para Riego Automatizado en Microcontrolador

Sebastian Aristizabal Castañeda

Luis Mateo Ochoa Agudelo

Alejandro Arias Ortiz

Resumen

Este proyecto propone el desarrollo de un sistema de riego inteligente utilizando el microcontrolador ESP32 y el sistema operativo en tiempo real FreeRTOS. El objetivo principal es automatizar el riego en función de los niveles de humedad del suelo, abordando aspectos clave como la concurrencia en la ejecución de tareas y la persistencia de los datos. Además, se evaluará el rendimiento de FreeRTOS en la gestión de recursos del ESP32, midiendo el uso de memoria, los tiempos de respuesta y la eficiencia en el manejo de tareas concurrentes. El proyecto incluye pruebas experimentales para validar el correcto funcionamiento del sistema y su desempeño bajo diferentes escenarios.

Introducción

En la actualidad, el Internet de las Cosas (IoT) ha emergido como una tecnología clave en la automatización de procesos en múltiples áreas, desde hogares inteligentes hasta la industria agrícola. Uno de los elementos fundamentales en un sistema IoT es el microcontrolador, un dispositivo pequeño y eficiente que actúa como el cerebro del sistema, conectando sensores y

actuadores. En este contexto, los sistemas operativos para microcontroladores, como **FreeRTOS**, permiten gestionar tareas concurrentes y recursos limitados de manera eficiente, facilitando el desarrollo de aplicaciones robustas y escalables.

El sistema desarrollado abordará dos aspectos clave en el diseño de sistemas embebidos: la concurrencia y la persistencia. Por un lado, el uso de FreeRTOS permitirá manejar varias tareas simultáneamente, como la lectura de sensores de humedad, la activación de la bomba de riego y el almacenamiento de datos históricos. Por otro lado, se implementarán mecanismos de persistencia para registrar el historial de los niveles de humedad y las acciones del sistema, asegurando que la información se mantenga disponible incluso tras reinicios o fallos del sistema.

Este proyecto busca, además, evaluar el desempeño de FreeRTOS en la gestión de recursos del ESP32, comparando su eficiencia y capacidad para manejar aplicaciones IoT de forma eficiente.

Antecedentes o marco teórico

Para el desarrollo de aplicaciones en el contexto del Internet de las Cosas (IoT), es fundamental el uso de microcontroladores debido a su capacidad para ejecutar tareas sencillas con bajo consumo energético y a un costo reducido. En este proyecto, se utilizará el ESP32, un microcontrolador ampliamente utilizado en aplicaciones IoT debido a su capacidad de procesamiento, conectividad Wi-Fi y Bluetooth, así como su versatilidad para manejar tareas en tiempo real. Estas características hacen del ESP32 una opción ideal para proyectos de automatización, como el sistema de riego inteligente propuesto.

Uno de los desafíos clave en el desarrollo de aplicaciones IoT es la gestión eficiente de los recursos limitados del microcontrolador, como la memoria y el procesamiento. Es en este contexto donde los sistemas operativos en tiempo real (RTOS) juegan un papel esencial. Un sistema operativo en tiempo real (RTOS), como FreeRTOS, proporciona una plataforma que facilita la gestión de múltiples tareas concurrentes de forma predecible y eficiente, asegurando que las tareas críticas se ejecuten en tiempos específicos. FreeRTOS ha ganado popularidad en aplicaciones IoT debido a su bajo peso, flexibilidad y su capacidad para adaptarse a microcontroladores de recursos limitados como el ESP32.

Además, conceptos como la concurrencia y la persistencia son esenciales en sistemas embebidos modernos. La concurrencia permite que el microcontrolador ejecute múltiples tareas de forma simultánea, por ejemplo, leyendo los sensores de humedad mientras controla la activación de la bomba de riego. Por otro lado, la persistencia asegura que los datos relevantes, como el historial de humedad del suelo y las acciones del sistema, se mantengan disponibles después de un reinicio o un fallo del sistema, lo cual es crucial para garantizar un funcionamiento continuo y confiable en sistemas IoT.

Objetivos (principal y específicos)

Objetivo principal

- Desarrollar e implementar un sistema de riego inteligente basado en el microcontrolador ESP32 y el sistema operativo FreeRTOS, capaz de automatizar el riego en función de los niveles de humedad del suelo, abordando los conceptos de concurrencia y persistencia.

Objetivos específicos

- **Configurar y programar el microcontrolador ESP32** utilizando FreeRTOS para gestionar tareas concurrentes, como la lectura de sensores de humedad y el control de la bomba de riego.
- **Implementar un sistema de monitoreo de humedad** que active de manera automática la bomba de riego cuando los niveles de humedad caigan por debajo de un umbral determinado.
- **Desarrollar un mecanismo de persistencia** que registre los niveles de humedad y el historial de activación del sistema de riego, para garantizar la continuidad de los datos incluso en caso de reinicios del sistema.
- **Evaluar el rendimiento del sistema operativo FreeRTOS** en la gestión de los recursos del ESP32, analizando el uso de memoria, tiempos de respuesta y manejo de múltiples tareas concurrentes.
- **Realizar pruebas experimentales** para validar la funcionalidad y el desempeño del sistema de riego, utilizando un conjunto de criterios predefinidos como la eficiencia en el uso de recursos y la fiabilidad de la automatización.

Metodología

El desarrollo del sistema de riego inteligente se llevará a cabo en varias fases, utilizando el microcontrolador ESP32 y el sistema operativo en tiempo real FreeRTOS. La metodología sigue una estructura lógica para asegurar la correcta implementación y validación del proyecto.

Fase 1: Configuración del entorno de desarrollo

- Se utilizará Visual Studio Code con el plugin PlatformIO, que proporciona compatibilidad con el ESP32 y FreeRTOS, permitiendo la programación del microcontrolador utilizando el framework de Arduino junto con FreeRTOS.

Fase 2: Implementación de la lectura de sensores y control de la bomba

- Se integrarán sensores de humedad del suelo conectados al ESP32 para obtener mediciones en tiempo real.
- Activar o desactivar la bomba de riego en función de los niveles de humedad predefinidos.

Fase 3: Persistencia de datos

- Se implementará un mecanismo de persistencia que permita almacenar localmente el historial de las mediciones de humedad y el estado del sistema de riego. Este almacenamiento garantizará que los datos persistan tras reinicios o cortes de energía.

Fase 4: Pruebas y validación del sistema

- Se llevarán a cabo pruebas de rendimiento para evaluar la eficiencia del sistema en cuanto al uso de memoria, tiempos de respuesta, y la capacidad de FreeRTOS para manejar tareas concurrentes.

Fase 5: Análisis de resultados

- Se realizará un informe final con los datos de desempeño y las conclusiones derivadas del experimento.

Diseño de Experimentos

Para evaluar la funcionalidad y el rendimiento del sistema de riego inteligente, se diseñarán experimentos que permitirán analizar la eficiencia de FreeRTOS en la gestión de las tareas y la persistencia de los datos. Los experimentos se enfocarán en los siguientes aspectos:

1. Validación de la concurrencia

Experimento: Configurar dos tareas concurrentes en FreeRTOS: una para la lectura del sensor de humedad y otra para el control de la bomba de riego. Se observará cómo FreeRTOS gestiona la ejecución de ambas tareas sin bloqueos o interferencias.

Métricas:

- Tiempos de respuesta de cada tarea (ejemplo: tiempo entre la lectura de humedad y la activación de la bomba).
- Uso de CPU mientras se ejecutan ambas tareas.

2. Validación de la persistencia

Experimento: Registrar los niveles de humedad del suelo en la memoria no volátil (NVS) del ESP32. Luego, se simulará un reinicio del sistema para verificar que los datos persisten y están disponibles tras el reinicio.

Métricas:

- Verificación de los datos almacenados antes y después del reinicio.
- Tiempos de acceso a la memoria no volátil.

3. Evaluación del uso de recursos

Experimento: Monitorear el uso de recursos del ESP32, incluyendo memoria y CPU, mientras el sistema de riego está en funcionamiento.

Métricas:

- Cantidad de memoria utilizada por FreeRTOS y las tareas del sistema.
- Consumo de energía del ESP32 bajo condiciones normales y de estrés.

Cronograma

Tarea	Semana 1 (21 - 27 Oct)	Semana 2 (28 - 03 Nov)	Semana 3 (4 - 10 Nov))	Semana 4 (11 - 17 Nov)
Fase 1: Configuración del entorno de desarrollo				
Fase 2: Implementación de la lectura de sensores y control de la bomba				
Fase 3: Persistencia de datos				
Fase 4: Pruebas y validación del sistema				
Fase 5: Análisis de resultados				

Referencias

FreeRTOS Documentation. FreeRTOS API Reference. Disponible en: https://www.freertos.org/Documentation

Espressif Systems. ESP32 Technical Reference Manual. Disponible en: https://www.espressif.com/en/products/hardware/esp32/overview

PlatformIO. PlatformIO IDE for VSCode. Disponible en: https://platformio.org/install/ide?install=vscode

Arduino. ESP32 Arduino Core. Disponible en: https://github.com/espressif/arduino-esp32

Caranha, C. (2021). Experiment Design for Computer Science. Disponible en: https://caranha.github.io/ExperimentDesignCS/syllabus.html