Presentación Proyecto Final

Laboratorio de Sistemas Embebidos

Fausto Alvarez Mollo
Prácticas Profesionalizantes
E.E.S.T. N°7 "T.R.Q."
Quilmes, Buenos Aires, Argentina
faustoalvarezmollo@gmail.com

Martín Alejandro Cabrera
Prácticas Profesionalizantes
E.E.S.T. N°7 "T.R.Q."
Quilmes, Buenos Aires, Argentina
martincabreracirco@gmail.com

***Resumen*—En este trabajo se realizó una aplicación en FreeRTOS integrando varios de los periféricos del microcontrolador LPC845.**

***Index Terms*—Tareas, Sistema Operativo en Tiempo Real, Microcontrolador, Semáforos, Colas**

I. Introducción

El fin de este documento es explicar el funcionamiento del proyecto integrador del curso Sistemas Embebidos realizado por ambos estudiantes, a fin de demostrar los conocimientos aprendidos en el transcurso del mismo, y aplicarlos como muestra de las practicas realizadas previamente.

II. Resumen

El código funciona como una aplicación en C para un sistema embebido utilizando FreeRTOS y periféricos para controlar un sistema de iluminación en función de la intensidad de luz medida, junto a funcionalidades de visualización y control. Este sistema esta diseñado para ejecutarse en un LPC845 en MCUXpresso tiene el soporte de librerías para periféricos y comunicación.

III. Funcionamiento

III-A. Inicialización del sistema y periféricos

BOARD_BootClockFRO30M() establece el reloj del sistema a 30 MHz.

project_init() es una función que inicializa los periféricos esenciales: I2C, el ADC, el PWM y el display de 7 segmentos.

Esta configuración define el entorno de hardware y la velocidad de reloj con la que trabajan las tareas y periféricos.

III-B. Definición de Tareas y Semáforos en FreeRTOS

Cada tarea tiene una función específica y se ejecuta en paralelo, permitiendo que el sistema realice varias acciones simultáneamente:

- task_button: Detecta la pulsación de un botón y alterna entre mostrar el "setpoint"(referencia de temperatura) o la intensidad de luz en el display de 7 segmentos. Utiliza (queue_show_setpoint) para almacenar el valor de si se debe mostrar el setpoint o la intensidad de luz.
- task_display: Controla la visualización en el display de 7 segmentos. Alterna la visualización entre el setpoint y la intensidad de luz dependiendo del valor de show_setpoint.
- task_measure_light: Lee los datos de un sensor de luz conectado mediante I2C para medir la intensidad de luz en lumenes. Los datos se almacenan en queue_light_intensity para después ser usados en otras tareas.
- task_change_setpoint: Cambia el "setpoint" del sistema al detectar las pulsaciones de los botones: BTN_1 y BTN_2.
  Este valor se usa para ajustar el sistema de iluminación. Se utiliza un semáforo (semphr) para contar las pulsaciones y generar un "setpoint" que se guarda en queue_setpoint.
- task_print_information: Muestra por consola la información sobre el tiempo, la intensidad de luz medida, el setpoint y la intensidad del LED. Es la tarea encargada de monitorear el estado del sistema.
- task_change_blue_led_intensity: Ajusta la intensidad de un LED azul en función del potenciómetro conectado al puerto de ADC. La intensidad del LED se controla mediante PWM y se almacena el valor de ciclo de trabajo en queue_duty_cycle.

III-C. Funciones de inicialización de Periféricos

- i2c_init(): Configura la comunicación I2C y el sensor de luz BH1750. Prepara el sensor para realizar lecturas de la intensidad de la luz del LED.
- adc_init(): Configura el puerto ADC para leer el valor del potenciómetro, Valor usado para ajustar la intensidad del LED mediante PWM.
- pwm_init(): Configura el temporizador y el PWM para controlar la intensidad del LED. El temporizador se inicializa con una frecuencia de 1 kHz y permite que varie la intensidad del LED.
- display_init(): Configura los pines para el display de 7 segmentos. Inicializa los pines como salidas y asegura que estén apagados inicialmente.
- buttons_init(): Configura los botones como entradas y crea colas y semáforos para almacenar el estado de los

botones y el “setpoint”.

III-D. Inicio de Scheduler en FreeRTOS

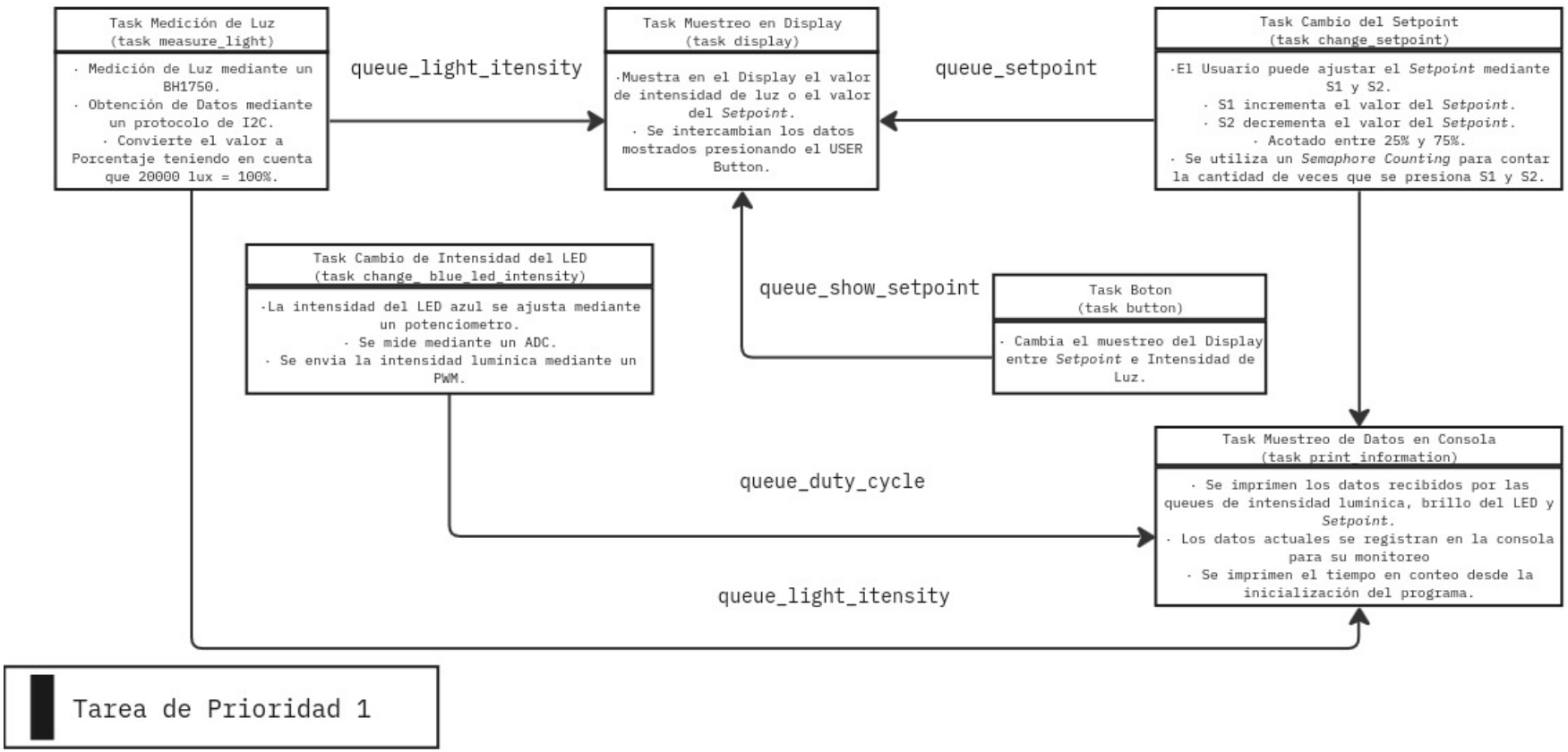
La función vTaskStartScheduler() inicia el scheduler de FreeRTOS, permitiendo que las tareas se ejecuten de acuerdo a sus prioridades y tiempos asignados.

III-E. Flujo de Trabajo

- Se mide la intensidad de luz usando el sensor BH1750 (task_measure_light).
- El usuario puede ajustar el “setpoint” usando botones (task_change_setpoint).
- Se visualiza en el display la intensidad de luz o el “setpoint” según lo indicado por el botón de usuario (task_button y task_display).
- La intensidad del LED azul se ajusta en función de la posición de un potenciómetro (task_change_blue_led_intensity).
- Los datos actuales se registran en la consola para su monitoreo (task_print_information).

IV. Conclusión

Este programa, diseñado para un LPC845, implementa un sistema de control de iluminación con retroalimentación y ajuste dinámico de intensidad. Utilizando FreeRTOS, el sistema realiza las siguientes funciones: medición de niveles de luz, ajuste de los puntos de referencia, control de la intensidad del LED y gestión de la visualización en un display de 7 segmentos.


Task Medición de Luz
(task measure_light)
· Medición de Luz mediante un BH1750.
· Obtención de Datos mediante un protocolo de I2C.
· Convierte el valor a Porcentaje teniendo en cuenta que 20000 lux = 100%.
queue_light_itensity
Task Muestreo en Display
(task display)
·Muestra en el Display el valor de intensidad de luz o el valor del Setpoint.
· Se intercambian los datos mostrados presionando el USER Button.
queue_setpoint
Task Cambio del Setpoint
(task change_setpoint)
·El Usuario puede ajustar el Setpoint mediante S1 y S2.
· S1 incrementa el valor del Setpoint.
· S2 decrementa el valor del Setpoint.
· Acotado entre 25% y 75%.
· Se utiliza un Semaphore Counting para contar la cantidad de veces que se presiona S1 y S2.
Task Cambio de Intensidad del LED
(task change_ blue_led_intensity)
·La intensidad del LED azul se ajusta mediante un potenciometro.
· Se mide mediante un ADC.
· Se envia la intensidad luminica mediante un PWM.
queue_show_setpoint
Task Boton
(task button)
· Cambia el muestreo del Display entre Setpoint e Intensidad de Luz.
queue_duty_cycle
Task Muestreo de Datos en Consola
(task print information)
· Se imprimen los datos recibidos por las queues de intensidad luminica, brillo del LED y Setpoint.
· Los datos actuales se registran en la consola para su monitoreo
· Se imprimen el tiempo en conteo desde la inicialización del programa.
queue_light_itensity
Tarea de Prioridad 1