Sistemas Basados en Microprocesador

Bloque 2 – Práctica 5

Sistema operativo en tiempo real. Introducción a CMSIS-RTOS2

Departamento de Ingeniería Telemática y Electrónica
Universidad Politécnica de Madrid

Introducción y objetivos de la práctica.

El objetivo de esta práctica es desarrollar una aplicación que haga uso de una de las funcionalidades básicas del sistema operativo CMSIS-RTOS2, como es la gestión de *threads* y la temporización, en un microcontrolador Cortex-M.

El uso de un RTOS representa un enfoque de diseño de aplicaciones más sofisticado, que fomenta el desarrollo de código estructurado mediante el uso de la API del mismo y ayuda a obtener un comportamiento más predictivo del sistema. La utilización de un sistema operativo en tiempo real proporciona soporte multithread y mayores garantías en la consecución de *deadlines* en aplicaciones de tiempo real, incluso en pequeños microcontroladores como el que se está utilizando.

Antes de comenzar a desarrollar la práctica es de suma importancia que lea la documentación que se indica en apartados posteriores.

Documentación

Además de los documentos empleados en las prácticas anteriores debe disponer de los siguientes:

- CMSIS-RTOS2 Documentation:
  https://www.keil.com/pack/doc/CMSIS/RTOS2/html/index.html

Ejemplo 1.

Cree un proyecto nuevo (P5_1) en Keil uVision para la tarjeta de desarrollo, utilizando CMSIS-RTOS. Añada dos ficheros fuente, `main.c` y `Thread.c`, haciendo uso de los *"User Code Template"*.

Cree el código necesario para tener un único *thread* (denominado Thled1) en la aplicación. Dicho *thread* debe encargarse de encender y apagar periódicamente el LED1 de la tarjeta, de forma que este permanezca encendido 200 ms y apagado 800 ms en cada ciclo.

Utilice el analizador lógico para comprobar que los tiempos han sido generados correctamente.

Utilizando el "wizard" de configuración del fichero RTX_Config.h cambie el valor del tick del sistema para que los tiempos antes indicados sean la mitad**. El código fuente no debe ser modificado**.

Ejemplo 2.

Cree un proyecto (P5_2) a partir del P5_1 en el que se añadan dos nuevos ficheros fuente denominados Thled2.c y Thled3.c. Cada fichero fuente se encargará de manejar un nuevo

thread denominados Thled2 y Thled3 respectivamente. Asegúrese que el tick base del sistema vuelve a ser de 1 ms.

El *thread* Thled2 debe encargarse de encender y apagar periódicamente el LED2 de la tarjeta, de forma que este permanezca encendido 137 ms y apagado 137 ms.

El *thread* Thled3 debe encargarse de encender y apagar periódicamente el LED3 de la tarjeta, de forma que este permanezca encendido 287 ms y apagado 287 ms.

Todos los *threads* deben estar activos desde el inicio de la aplicación de manera que los tres se ejecuten de manera concurrente.

Utilice el analizador lógico para comprobar que los tiempos han sido generados correctamente.

En el *thread* main, una vez creados los tres *threads* coloque un punto de ruptura y utilice la herramienta de visualización del sistema operativo (View->Watch Windows->RTX RTOS) para analizar el comportamiento de la aplicación. Rellene la siguiente tabla:

Número de Threads de la aplicación:		
Nombre del Thread	**Estado del mismo**	**Prioridad del mismo**

Cambie el punto de ruptura y sitúelo en el comienzo del *thread* Thled2. Vuelva a rellenar la siguiente tabla.

Nombre del Thread	**Estado del mismo**	**Prioridad del mismo**

Pruebe a cambiar el punto de ruptura entre los diferentes *threads* y verifique cómo evoluciona el estado de los *threads* mediante la herramienta de análisis.

Ejemplo 3.

Cree un proyecto (P5_3) a partir del P5_2. En este último ejemplo de esta práctica se pretende sincronizar la ejecución de los tres *threads* definidos anteriormente mediante el uso de flags (Event Flags de CMSIS-RTOS2).

La secuencia en la que deben ejecutarse los *threads* es la definida a continuación:

1. Cuando la aplicación arranca, los LED 2 y 3 permanecen apagados inicialmente. El LED 1 comenzará a parpadear según lo indicado en el Ejemplo 1 y se mantendrá así indefinidamente.
2. Después de los primeros cinco ciclos del LED1, el LED2 deberá comenzar su parpadeo durante 20 ciclos, con la cadencia indicada en el Ejemplo 2. Pasados esos 20 ciclos se detendrá.
3. Después de 15 ciclos del LED2 el LED3 deberá comenzar su parpadeo durante 30 ciclos, con la cadencia indicada en el Ejemplo 2. Pasados esos 30 ciclos se detendrá hasta una nueva activación tras 15 nuevos ciclos del LED2.
4. Después de 25 ciclos del LED3 el LED2 deberá comenzar nuevamente su parpadeo durante otros 20 ciclos.
5. Los puntos 3 y 4 se repetirán indefinidamente.

La figura 1 representa esquemáticamente la secuencia temporal que indica los instantes en que cada uno de los tres LEDs deben estar parpadeando.

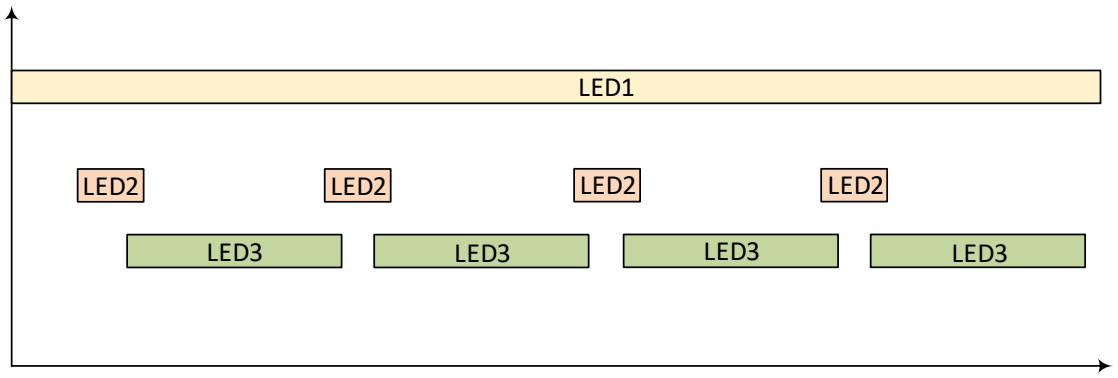

Figura 1

Ejecute el código y verifique su funcionamiento. Haga uso de la herramienta *"System and Thread Viewer"* para verificar el estado en que se encuentran los *threads* de su aplicación.