Escuela Técnica Superior de Ingeniería y Sistemas de Telecomunicación

UNIVERISDAD POLITÉCNICA DE MADRID

INGENIERÍA DE SISTEMAS ELECTRÓNICOS

Servidor WEB compacto embebido (III)
Modos de bajo consumo y Watchdog

Departamento de Ingeniería Telemática y Electrónica

Introducción

En esta práctica el estudiante analizará los modos de bajo consumo del microcontrolador STM32F429 y añadirá al servidor web compacto, desarrollado en las prácticas 1 y 2, el código necesario para poder poner el sistema en alguno de dichos modos de bajo consumo. Además, se realizarán las medidas correspondientes para caracterizar el ahorro energético que esto supone.

La duración de esta práctica es de 1 semana y a su finalización deberá justificar a su profesor que ha resuelto cada uno de los apartados propuestos.

Material necesario

Para la realización de esta práctica es necesario haber concluido en su totalidad las prácticas 1 y 2. Será necesario realizar una lectura de las páginas relativas a los modos de bajo consumo del manual de usuario del microcontrolador, y se recomienda encarecidamente la lectura del libro *Mastering STM32* de Carmine Noviello. Como código de ejemplo de los modos de bajo consumo en la tarjeta Nucleo-144 STM32F429 se utilizará el ejemplo "*PWR_CurrentConsumption*" de STM32CubeMX.

Se incluye finalmente un apartado opcional de manejo del Watchdog del sistema.

A continuación, se referencian todos los elementos necesarios:

- Servidor Web compacto desarrollado en las prácticas 1 y 2, con los componentes de red v7.15
- Capítulo 5 "*Power controller (PWR)*" (y específicamente el apartado 5.3 "*Low-power modes*") del documento *RM0090. Reference manual STM32F429/439 advanced Arm®-based 32-bit MCUs*.
- Capítulo 19 "*Power Management*" (y específicamente el apartado 19.3 "*Power Management in STM32F Microcontrollers*") del libro *Mastering STM32, A step-by-step guide to the most complete ARM Cortex-M platform, using a free and powerful development environment based on Eclipse and GCC*.

- Ejemplo de uso de los modos de bajo consumo de STM32CubeMX "*PWR_CurrentConsumption*" (disponible también en la página Web de la asignatura (Moodle).
- Capítulos 21 "*Independent watchdog (IWDG)*" y 22 "*Window watchdog (WWDG)"* del libro *Mastering STM32, A step-by-step guide to the most complete ARM Cortex-M platform, using a free and powerful development environment based on Eclipse and GCC*.

APARTADO 1. Modos de bajo consumo

En este apartado se pretende analizar los diferentes modos de bajo consumo en los que se puede hacer trabajar al microcontrolador STM32F429.

Para ello, se hará uso del código proporcionado en los ejemplos de STM32CubeMX, para lo que se hará uso concretamente del ejemplo denominado *PWR_CurrentConsumption*, específicamente codificado para su uso en la tarjeta Nucleo-144 STM32F429.

Este código permite introducir al sistema en los diferentes modos de bajo consumo, así como hacerlos salir de los mismos bajo diferentes circunstancias.

Descargue y analice el código del ejemplo mencionado.

Identifique las modificaciones que debe realizar en dicho código para hacer entrar al sistema en cada uno de los modos de bajo consumo para los que ha sido preparado.

Ejecute el código ejemplo para cada uno de los modos y observe su funcionamiento (tenga en cuenta que no podrá utilizar las herramientas de depuración mientras el sistema está en cualquiera de los modos de bajo consumo).

Mida el consumo del microcontrolador (utilizando el JP5 disponible en la placa Nucleo-144) para cada uno de los modos de bajo consumo.

APARTADO 2. Integración en la aplicación modo *sleep*

Modifique la aplicación del servidor Web compacto desarrollada en las prácticas 1 y 2 para que el sistema entre en el modo *sleep* después de 15 segundos de ejecución. El sistema debe salir del modo *sleep* tras una pulsación del pulsador azul de la placa Nucleo-144. Para identificar que el sistema entra y sale del modo *sleep*, añada un *thread* a la aplicación que haga parpadear el led verde de la placa cada 100ms. Antes de entrar en el modo *sleep*, el sistema debe encender el led rojo de la placa y apagarlo nada más salir de dicho modo.

Compruebe que las peticiones desde el navegador Web no son atendidas por el sistema mientras este se encuentra en el modo sleep.

Mida el consumo del microcontrolador en modo *run* y en modo *sleep* y cuantifique el ahorro energético que supone este último respecto al primero.

APARTADO 3. Análisis de otros modos (OPCIONAL)

Modifique la aplicación del apartado anterior para hacer entrar al sistema en los otros modos de bajo consumo e identifique las ventajas e inconvenientes de cada uno de ellos. El desarrollo de este apartado le será de utilidad para la realización del proyecto del bloque 2 de la asignatura donde deberá minimizar el consumo del sistema al máximo.

APARTADO 4. Watchdog (OPCIONAL)

En este apartado se pretende que el estudiante se familiarice con el manejo básico del Watchdog. Como ya se explicó, el Watchdog es un Timer especial, que permite detectar, en determinadas circunstancias, un mal funcionamiento de la aplicación. El primer paso es el análisis detallado del funcionamiento del Watchdog, leyendo con atención los capítulos 21 y 22 del manual de referencia del STM32F429. Como ya sabe, el Watchdog puede configurarse para

generar una interrupción o un reset del sistema si no se "alimenta" en el tiempo predeterminado en su configuración.

En este apartado se configurará el Watchdog para generar un reset del sistema si su estado de cuenta alcanza el 0.

Las especificaciones de la aplicación que debe desarrollar son las siguientes:

- Se creará un nuevo proyecto de Keil desde cero, sin sistema operativo.
- Se configurará el Watchdog para que genere un reset si no es "alimentado" en un tiempo de 5 segundos.
- La aplicación deberá iniciarse encendiendo el LED verde durante aproximadamente cuatro segundos. Posteriormente deberá pasar a un estado en el que el LED verde y el LED azul se enciendan alternativamente cada medio segundo, aproximadamente.
- Se deberá "alimentar" periódicamente el Watchdog en el programa principal para que, en circunstancias normales, no se llegue a producir el reset del sistema provocado por el Watchdog.

Ejecute esta aplicación y compruebe que no se produce nunca un reset provocado por el Watchdog.

A continuación, se incluirá el siguiente código, correspondiente a la rutina de atención a interrupciones externas, de forma que, cuando se produzca una pulsación del pulsador azul de la tarjeta, se conmute el estado del LED rojo. Este comportamiento será así para las 10 primeras pulsaciones, mientras que las siguientes se ignorarán (tenga en cuenta que no están eliminados los rebotes del pulsador en este punto, ni hace falta eliminarlos). Deberá configurar correctamente el resto de los recursos del sistema para el uso del pulsador.

```
void EXTI15_10_IRQHandler(void) {
      static uint32_t cuenta;
      if (cuenta++ < 10) {
            HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_14);
            __HAL_GPIO_EXTI_CLEAR_IT(GPIO_PIN_13);
}
```

Ejecute la aplicación y observe cuál es el nuevo comportamiento.

Identifique si se produce en algún caso la interrupción del Watchdog y explique cuál es la causa. Corrija el problema detectado y vuelva a ejecutar la aplicación para determinar si el funcionamiento es correcto.

Además, en el arranque del sistema, se deberá identificar cuál ha sido la última la causa de reset. Deberá diferenciarse si se ha producido un reset normal (por conexión de la alimentación o pulsación del botón de reset) o provocado por el Watchdog. Deberá indicarse, durante aproximadamente los 3 primeros segundos de funcionamiento de la aplicación, dicha causa, encendiendo el LED azul si el reset ha sido por alimentación o el LED rojo si ha sido provocado por el Watchdog.