F.- Gestión de colas en FreeRTOS

- **Objetivo:**
  - **Uso del IDE (edición, compilación y depuración de programas)**
  - **Uso de GPIO & FreeRTOS (manejo de Salidas y de Entradas Digitales en Aplicaciones)**
  - **Modificar/Documentar lo que se solicita en c/ítems**

- **Referencias:**
  - **FreeRTOS**, **API Reference**, **FreeRTOS Documentation**: **Mastering the FreeRTOS Real Time Kernel - a Hands On Tutorial Guide**, **FreeRTOS V10.0.0 Reference Manual**, **Book companion source code**
  - **CMSIS-RTOS Documentation, CMSIS-RTOS API Version 1**
  - **FreeRTOS on STM32 CMSIS_OS API (CMSIS_V1)Archivo**

F.1.- Tome del Campus los proyectos correspondientes a los siguientes ejemplos de Mastering the FreeRTOS Real Time Kernel - a Hands On Tutorial Guide:

freertos_book_Example020 - Re-writing vPrintString() to use a semaphore
freertos_book_Example030 - Synchronizing tasks

Se solicita:
- **Documentar** mediante un diagrama temporal la distribución del tiempo de CPU entre tareas, Kernel, Interrupciones (buscar imagen en: Mastering the FreeRTOS Real Time Kernel - a Hands On Tutorial Guide), detallar qué ocurre en cada cambio de contexto
- **Documentar** observaciones
- **Documentar** el valor de **time slice** de **FreeRTOS**, dónde y cómo modificarlo (**FreeRTOSConfig.h**)
- **Documentar** el efecto de modificar **time slice** sobre la ejecución de **tareas** (probar con 1000mS/100mS/10mS/1mS)
- **Documentar** el criterio a aplicar para la elección del valor de **time slice** para una aplicación

F.2.- Tome del Campus el proyecto:

freertos_app_Example6_6 - Práctica Obligatoria (6 de 6)

Se solicita:
- **Documentar** mediante un diagrama temporal la distribución del tiempo de CPU entre tareas, Kernel, Interrupciones, detallar qué ocurre en cada cambio de contexto
- **Documentar** observaciones

F.3.- Tome del Campus el proyecto:

freertos_app_Example6_6 - **Práctica Obligatoria (6 de 6)**

Se solicita:
- **Modificar app.c**, **task_Button.c**, **task_Led.c** y **app_Resources.h** para que:
  - la función **vTaskButton** que usa un semáforo binario para controlar la función **vTaskLed** (**vSemaphoreCreateBinary**() / **xSemaphoreTake**() / **xSemaphoreGive**()), utilice una variable global de blinking (**ledBlinkingFlag**), protegiendo el acceso a la misma con un semáforo mutex (**xSemaphoreCreateMutex**() / **xSemaphoreTake**() / **xSemaphoreGive**())
  - dicha variable impacta sobre la variable de blinking del led (**ledFlag**), para que las tareas (**TaskButton** y **TaskLed**) sigan haciendo exáctamente lo mismo que hacen en el proyecto original
- **Documentar** observaciones

- **Subir al Campus:** La carpeta **App comprimida** (archivo del tipo "**.zip** o **.rar**"), **nombrar**: **RTOS I - PO 6_6 - Apellidos_Nombres.zip**