Práctica 2
Programación con STM32Cube HAL

Resolver con un proyecto diferente para cada consigna (recuerde que si el proyecto está abierto puede “copiar y pegar” proyectos dentro del explorador de proyectos, cambiando luego a mano el nombre del archivo .ioc y borrando el archivo .launch), usando el asistente gráfico y la API de la HAL:

1. Imprimir “Hola mundo\r\n” en una terminal serie (UART) cada 10 segundos usando HAL_Delay.
2. Imprimir “Hola mundo\r\n” en una terminal serie (UART) cada 10 segundos usando un timer por polling.
3. Leer de un terminal serie un comando (terminado en ‘\r\n”)
   a. Si el comando es “LED ON” o “LED OFF” actuar en correspondencia con el LED on board
   b. Si el comando es “LED?” responder con un mensaje al terminal el estado del LED
   c. Si el comando es otro responder con un mensaje de error al terminal
4. Idem al ejercicio 3, pero se agrega que responda a un tercer comando “blink” que lleve a un estado de parpadeo 500ms encendido/500ms apagado. Se sugiere modelar la solución como máquina de estados.
5. Configurar un pin como entrada analógica usando el ADC1 en modo contínuo. Tomar una muestra cada 1 segundo y enviarla (en ASCII) por puerto serie.
6. Idem al ejercicio 5 pero en modo discontinuo disparado por software.
7. Idem al ejercicio 5 pero en modo discontinuo disparado por timer.
8. Tomar muestras del ADC cada 100ms y con las mismas controlar el brillo de un LED conectado a una salida PWM. Configure el timer correspondiente para que el período del PWM sea 1kHz.

---

9. Configure un pulsador conectado a un GPIO (entrada con pull-up). Asigne al pulsador una interrupción externa por flanco descendente. Implemente la siguiente función callback con un código que aumente el valor de un contador global cada vez que se ejecute. Analice lo que ocurre usando la característica de “live Expressions” en una sesión de depuración.

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
```

10. Configure el TIM3 y el GPIO del led. Configure el TIM3 para que desborde cada segundo y habilite la interrupción. Implemente la siguiente función callback con un código que invierta el estado del LED.

```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
```

11. Cree un proyecto con dos interrupciones para comprobar el funcionamiento de las prioridades

    a. Conecte un pulsador a un GPIO configurado como entrada que dispare una interrupción. En la interrupción, ponga en alto un segundo GPIO durante 10 ms (usando la función HAL_Delay). Es decir que cada vez que se presiona el pulsador, verá un pin en alto durante 10 ms. Compruebe lo realizado con el analizador lógico.
    b. Configure una interrupción de un timer con una prioridad menor a la del pulsador para dispararse cada 200 us. Cada vez que se dispara, invierta el estado de un tercer GPIO. Observe las salidas con el analizador lógico y verifique que cumple con sus expectativas en relación al disparo del timer. Obtenga una captura de pantalla que demuestre el funcionamiento, específicamente comprobando que se cumplan las prioridades.
    c. Invierta las prioridades y observe nuevamente las formas de onda. Obtenga una captura de pantalla que refleje la nueva situación.
    d. Si se quieren mantener las prioridades originales, pero quiere evitarse que el timer se vea relegado, ¿cómo podría mejorarse el código?