Tema 7
RESUMEN

En el **tema 7** (***Interrupciones y EXTI***) se enseña cómo utilizar el bloque EXTI y cómo funcionan las interrupciones del microcontrolador, dividido en varios puntos:

En el primer punto se habla de los conceptos básicos para la gestión de interrupciones de manera que se tiene lo siguiente.

- Las interrupciones que reciba la CPU van a poder clasificarse en dos tipos:
  - No Enmascarables (NMI): Aquellas que, cada vez que ocurren, provocan la atención de la CPU.
  - Enmascarables (IRQ): Aquellas que la CPU puede inhibir en determinados momentos, según el interés del programador.
- Cada fuente de interrupción puede, a su vez, tener varios eventos que la provoquen.
- Las IRQs se pueden enmascarar de forma global (por la propia CPU), o de forma local (configurando el periférico implicado en dicha IRQ). Si el enmascaramiento es local, la configuración del periférico puede enmascarar (inhabilitar) o desenmascarar (habilitar) cada una de los distintos eventos que pueden provocar la IRQ.
  - En caso de que el mecanismo de habilitación de la fuente esté habilitada, pero no el de la CPU, entonces no se habla de interrupciones, sino de eventos.
  - Los eventos no se gestionan por Rutinas de Atención a la Interrupción (RAI), sino por espera activa**.**
- Por último, se indican los pasos que da la CPU cuando ocurre una petición de interrupción, qué pasa cuando se termina la RAI y unos conceptos básicos de Interrupciones.

A continuación, se describe el Controlador de Interrupciones Vectorizadas (NVIC) que tiene el microcontrolador del curso. También se muestra la tabla resumen con las interrupciones que van a ser utilizadas en el cuso, con su nombre, descripción, acrónimo, prioridad y posición en el vector. Por último, se describen los registros de 32 bits que se utilizan para controlar el NVIC (en el ámbito de este curso):

- **NVIC->ISER[x]** – Habilita la fuente de interrupción correspondiente al bit en el que se escriba un '1' del registro x
  - Los bits a 0 no modifican el estado de enmascaramiento
  - ISER[0] cubre los bits para las primeras 32 fuentes de interrupción

  - ISER[1] cubre los bits para las fuentes 32 a 63, correspondiendo el bit 0 a la fuente 32

- **NVIC->ICER[x]** – Enmascara la fuente de interrupción correspondiente al bit en el que se escriba un '1' del registro x
  - Los bits a 0 no modifican el estado de enmascaramiento
  - ICER[0] e ICER[1] análogo a ISER[x]

En el tercer punto se describe el **BLOQUE EXTI**, indicando primero sus características básicas y finalmente sus diferentes registros de control, datos y estado.

Los diferentes ***registros de control*** del bloque EXTI, que sirven para configurar su funcionalidad antes de utilizarlo, son los siguientes:

- **SYSCFG→EXTICR –** EXTI Control Register, que es un conjunto de registros de 16 bits (p. ej. SYSCFG->EXTICR[0]), en el que cada grupo de 4 bits indica el puerto asociado con la EXTI Line
  - 0000 – GPIOA
  - 0001 – GPIOB
  - 0010 – GPIOC
  - 0011 – GPIOD
  - De forma que cada registro son 4 pines:
    - EXTICR[0] – pin3 a pin0
    - EXTICR[1] – pin7 a pin4
    - EXTICR[2] – pin11 a pin8
    - EXTICR[3] – pin15 a pin12
- **EXTI→IMR –** Interrupt Mask Register, que es un registro de 32 bits (EXTI->IMR), con sólo 16 útiles, uno por cada pin (bit0 = EXTI0, bit15 = EXTI15):
  - Un '1' habilita esa línea de EXTI
  - Un '0' enmascara esa línea de EXTI
- **EXTI→RTSR –** Rising Edge Trigger Selection Register, que es un registro de 32 bits (EXTI->RTSR), con sólo 16 útiles, uno por cada pin (bit0 = EXTI0, bit15 = EXTI15):
  - Un '1' habilita el evento por flanco de subida
  - Un '0' inhabilita el evento por flanco de subida
- **EXTI→FTSR –** Falling Edge Trigger Selection Register, que es un registro de 32 bits (EXTI->FTSR), con sólo 16 útiles, uno por cada pin (bit0 = EXTI0, bit15 = EXTI15):
  - Un '1' habilita el evento por flanco de bajada
  - Un '0' inhabilita el evento por flanco de bajada

El bloque EXTI no tiene ***registro de datos.***

El único ***registro de estado*** del bloque EXTI, que sirve para comprobar y consultar la ejecución de las actuaciones sobre los pines del bloque, es el siguiente:

- **EXTI→PR –** Pending Register, que es un registro de 32 bits (EXTI->PR), con sólo 16 útiles, uno por cada pin (bit0 = EXTI0, bit15 = EXTI15):
  - Un '1' indica que ha saltado ese evento
  - Un '0' indica que no ha saltado
  - Para limpiar el flag que se encuentre a '1' hay que escribir un '1' en dicho bit. Escribir '0' no afecta al estado del bit

Finalmente se muestran dos ejemplos prácticos (uno por eventos y otro por IRQ) para probar por el alumno, con su programa correspondiente (inicialización y funcionamiento continuo para el ejemplo por eventos y variables globales/RAI, inicialización y funcionamiento continuo para el ejemplo por IRQ) -> El ejemplo modifica los estados de encendido/apagado del LED verde y del LED azul de la placa DISCOVERY al pulsar el botón USER, pero gestionándolo con interrupciones.