CURSO DE STM32

25 ENERO 2021

003 GPIO External Interrupt

Creado por: Ing. Christian Salazar

SECCION 3

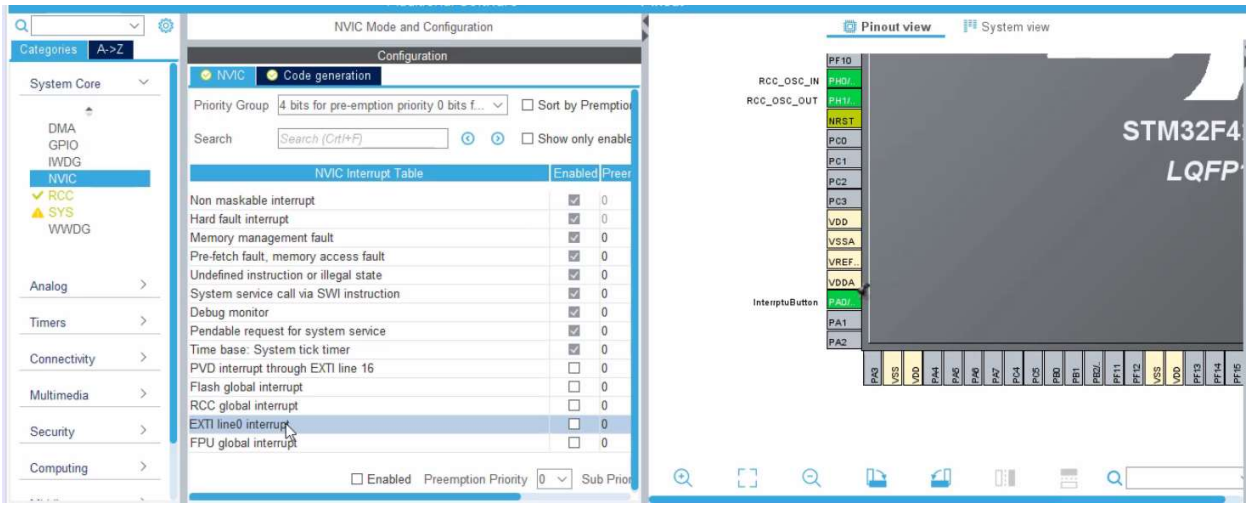
003 GPIO External Interrupt

Que aprenderemos?

En este video programaremos la entrada PA0 de la tarjeta STM32F429 Discovery como entrada de Interrupción, la cual tiene un Botón Azul (User Button) para poder generar en terminos matemáticos un escalón unitario que no es mas que un pulso, debemos tomar en cuenta lo antes aclarado en el video anterior sobre el ruido o rebote de los elementos de acción mecánica, también manipularemos el vector de prioridades NVIC al cual responde CMSIS en donde cambiaremos la prioridad de nuestra interrupción.

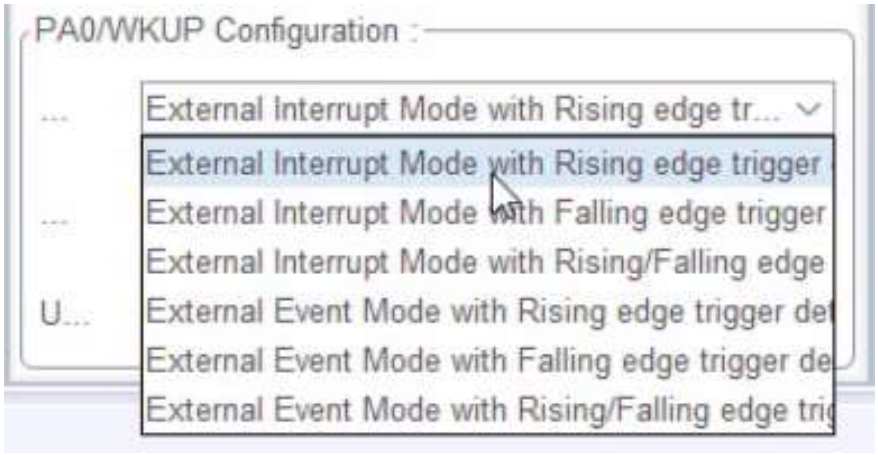
> ***"Usaremos HAL Drivers, lo cual nos ayudará en gran medida a portar y reciclar rutinas de código de un procesador de una Familia a otro de otra Familia".***

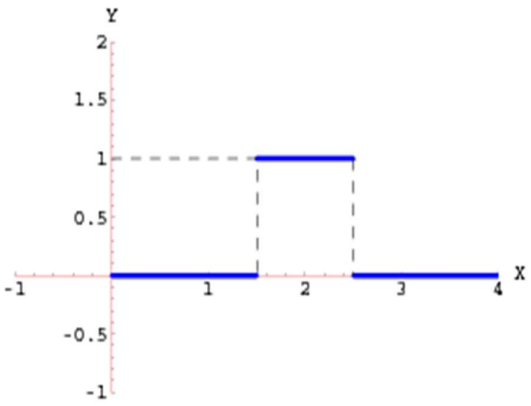
NVIC

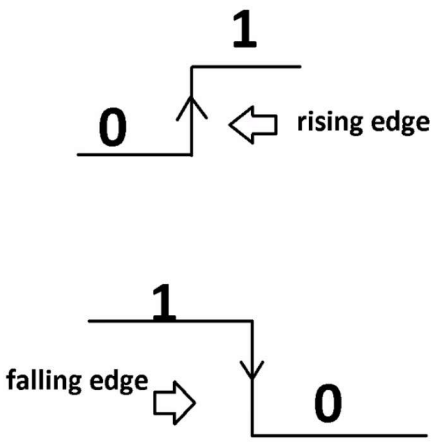


El NVIC es un vector de prioridades al cual se remite CMSIS para tratar las interrupciones habilitadas, estas pueden ser de un periférico USART, I2C, SPI, TIMER, etc, en nuestro caso la interrupción de EXTI es decir una interrupción externa, este NVIC tambien es afectado y usado cuando se usa RTOS para gestionar nuestras interrupciones, tener en cuenta que un número bajo es decir 0 (Cero) es la prioridad mas alta y seguir sumando es seguir bajando la prioridad, este vector no ayuda cuando en un instante dado se presenten dos o mas interrupciones al mismo tiempo, entonces se atenderá la interrupción desde la mas alta prioridad a la mas baja, si tenemos dos interrupciones de misma prioridad, tenemos la opción de la subprioridad o si a su vez incluso coinciden dos o mas interrupciones con la misma prioridad y subprioridad lo que hace CMSIS es atender la interrupción que tenga primero en su lista de interrupciones, estos criterios de prioridades puede llegar a ser deterministico si llegaramos a usar varias interrupciones en algun proyecto.

FLANCO ASCENDENTE, DESCENDENTE O AMBOS

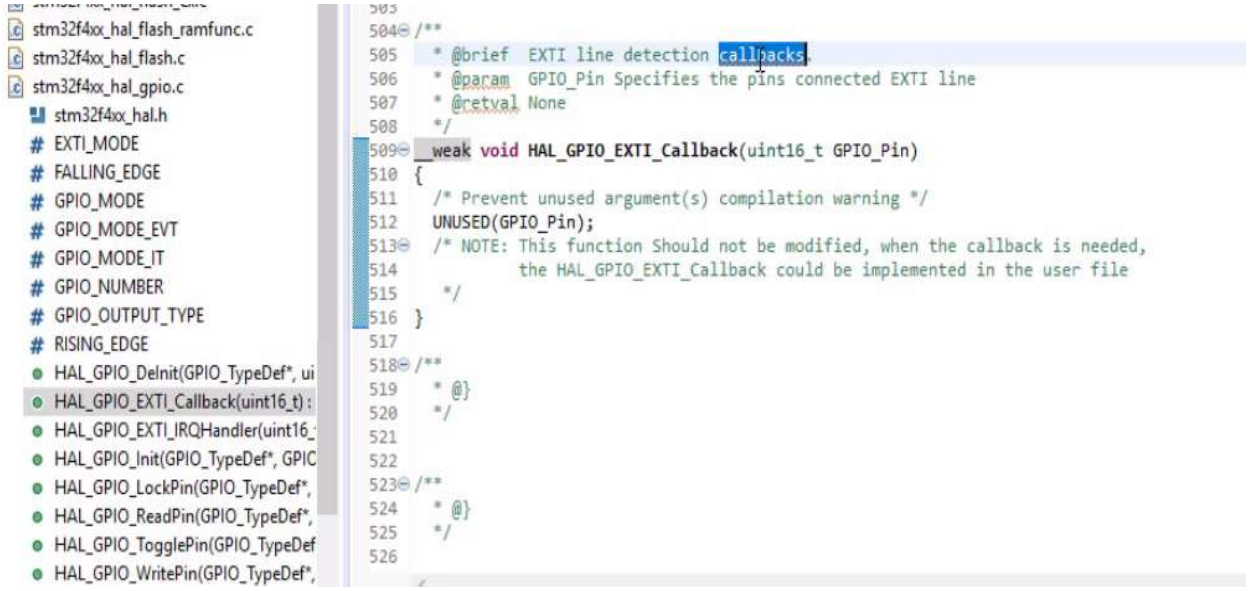






La acción de presionar y soltar un botón puede ser considerado como un escalón unitario en condiciones ideales, pero en condiciones reales debemos considerar el ruido o rebote que hay al momento de la conexión y desconexion, como podemos observar en la imagen tenemos la opción de generar una interrupción del programa principal (while infinito) de tres modos con los EXTI, estos son con el flanco ascendente que es el cambio de 0 a 1 que es cuando se producirá la interrupción, también con el flanco descendente que es el cambio de 1 a 0, recordemos que normalmente tendremos ambos estados en un ciclo completo y además podemos generar la interrupción con ambos flancos a la vez es decir interrumpe con el cambio de estado de la entrada.

FUNCION CALLBACK DE INTERRUPCION



Una vez que se presenta la interrupción y CMSIS ha determinado atenderla, la secuencia de hacerlo consiste en primer lugar modificar los registros adecuados (Registros de interrupción EXTI) para reiniciar la señal de interrupción disparada, esto se lo hace en nuestro caso en el archivo stm32f4xxit.c que está en la carpeta /src donde se encuentra main.c, una vez atendida la interrupción por el CMSIS podemos agregar una función o código a ejecutarse cuando se produzca la interrupción, esto se lo hace con la función que muestra la imagen anterior que es la que se ejecuta o levanta inmediatamente después de atender los registros y banderas antes mencionadas, cabe aclarar que a esta función void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin) la llamaran todos los EXTI habilitados (EXTI0 – EXTI15) con un argumento (uint16_t GPIO_Pin) el cual puede ser de desde el GPIO_PIN_0 al GPIO_PIN_15 para poder determinar quién disparó la función, es decir podemos tener hasta 16 interrupciones bien diferenciadas de este tipo porque los pines PA0, PB0, PC0, etc estan multiplexados para la interrupción EXTI0 , asi que podemos disparar

EXTI0 con cualquira de estos pines asi mismo para PA1, PB1, PC1 para EXTI1 y asi sucesivamente, en el video podremos apreciar como podemos separar códigos para cada EXTI con un simple bucle IF().