○ 5.3 – Projet simple GPIO avec interruptions sous STM32CubeIDE : *« BP-LED-Interruption »*

○

Projet :	GPIO sous interruptions : ***« BP-LED-Interruption ».***
Cahier des charges :	Un appui sur le bouton poussoir RESET B2 de la carte Nucleo-L152RE initialise le programme. **La LED LD2 change d'état lorsque le bouton poussoir utilisateur B1 est relâché ou si un front descendant apparait sur PA8 ou PA9.**
Choix de la carte :	NUCLEO64-STM32L152RE
Initialisation :	Utilisation de STM32CubeMX (configurateur graphique) pour générer le code d'initialisation du programme.
EDI :	Mise au point du programme avec STM32CubeIDE.

Test sur la carte NUCLEO-L152RE

Les GPIO ont **16** lignes d'interruption :

Toutes les broches avec le même numéro sont connectées à la ligne du même numéro.
Ils sont multiplexés sur une ligne. Chaque ligne peut déclencher une interruption en **cas de** front **montant** ou **descendant**.
IMPORTANT : vous ne pouvez pas utiliser deux broches sur une même ligne simultanément.

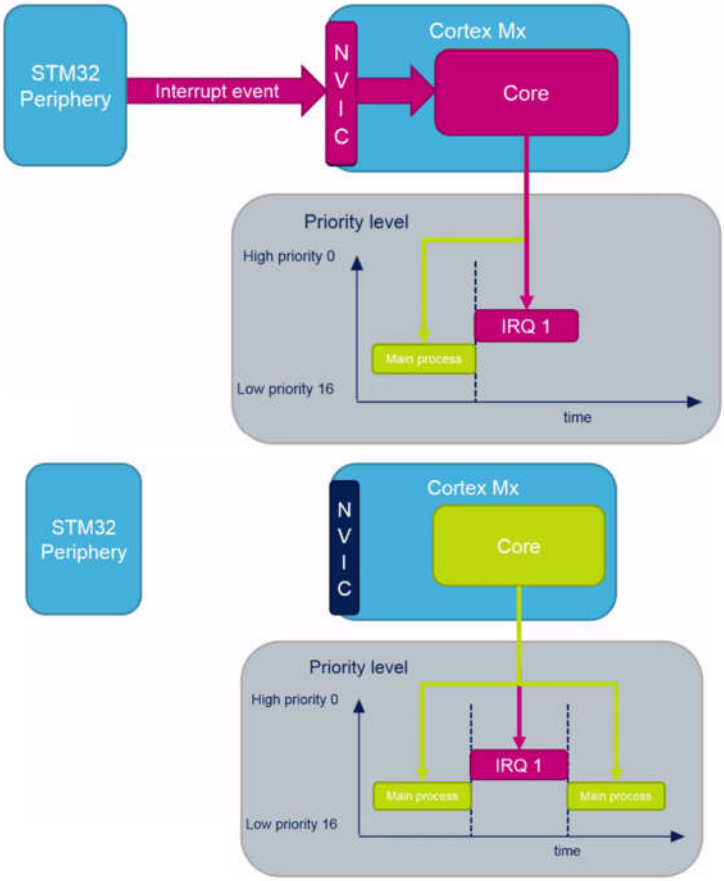
Port Pins	Port Pins	Port Pins	Port Pins	Lignes d'interruption
PA0	PB0	… ..	Px0	Line_0
PA1	PB1	… ..	Px1	Line_1
PA2	PB2	… ..	Px2	Line_2
PA3	PB3	… ..	Px3	Line_3
PA4	PB4	… ..	Px4	Line_4
PA5	PB5	… ..	Px5	Line_5
PA6	PB6	… ..	Px6	Line_6
PA7	PB7	… ..	Px7	Line_7
PA8	PB8	… ..	Px8	Line_8
PA9	PB9	… ..	Px9	Line_9
PA10	PB10	… ..	Px10	Line_10
PA11	PB11	… ..	Px11	Line_11
PA12	PB12	… ..	Px12	Line_12
PA13	PB13	… ..	Px13	Line_13
PA14	PB14	… ..	Px14	Line_14
PA15	PB15	… ..	Px15	Line_15

ATTENTION : en réalité, dans le Cortex-Mx (STM32), nous n'avons pas 16 lignes d'interruption externes, normalement, il y en a beaucoup moins.
Consultez le manuel de référence de la famille STM32 que vous devez utiliser.

Par exemple le **STM32F4** dispose de 7 gestionnaires d'interruptions pour les broches GPIO.
Ils sont dans le tableau ci-dessous :

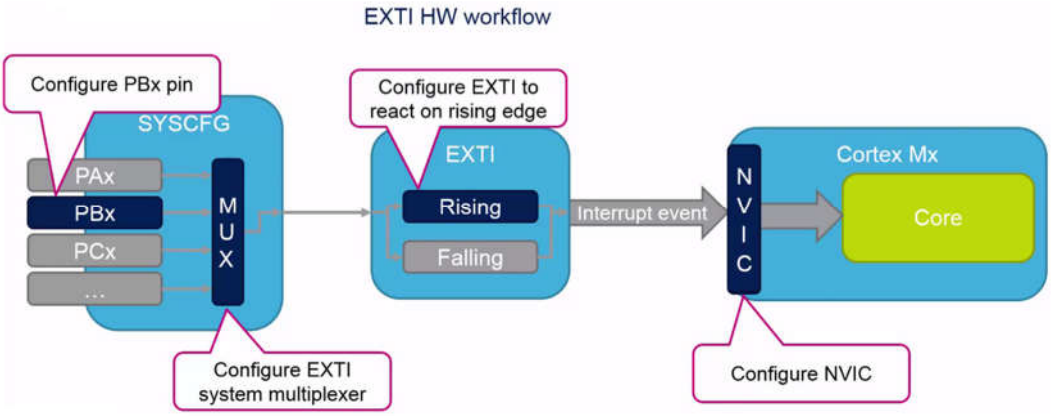
Irq	Gestionnaire	La description
EXTI **0**_IRQn	EXTI **0** _IRQHandler	Gestionnaire pour les broches connectées à la ligne 0
EXTI **1**_IRQn	EXTI **1** _IRQHandler	Gestionnaire pour les broches connectées à la ligne 1
EXTI **2**_IRQn	EXTI **2** _IRQHandler	Gestionnaire pour les broches connectées à la ligne 2
EXTI **3**_IRQn	EXTI **3** _IRQHandler	Gestionnaire pour les broches connectées à la ligne 3
EXTI **4**_IRQn	EXTI **4** _IRQHandler	Gestionnaire pour les broches connectées à la ligne 4
EXTI **9_5** _IRQn	EXTI **9_5**_IRQHandler	Gestionnaire pour les broches connectées aux lignes 5 à 9
EXTI **15_10**_IRQn	EXTI **15_10**_IRQHandler	Gestionnaire pour les broches connectées aux lignes 10 à 15

Les **interruptions externes** sont gérées par le périphérique **NVIC** (contrôleur d'interruption vectorisée imbriqué).
Si une interruption survient, NVIC le gère et le noyau Cortex-Mx suspend le PROCESS PRINCIPAL et répond à l'IRQ1.
À la fin de la réponse à IRQ1, le noyau retourne au PROCESS PRINCIPAL à la position exacte où il a été suspendu auparavant.
Voir les images ci-dessous.



NVIC est l'arbitre qui décide de l'exécution de l'interruption en fonction de la priorité et de la sous-priorité de l'interruption.

L'image ci-dessous représente la configuration que vous devez effectuer pour utiliser EXTI (entrée d'**I**nterruption **EXT**erne).



Nous allons maintenant montrer comment configurer deux GPIO en mode ***Entrée en interruption***.
Pour ce test, nous utilisons la carte NUCLEO-L152RE.

Configuration du GPIO dans STM32CubeMX (intégré dans STM32CubeIDE)

- **Lancement de STM32CubeIDE**
- **Création d'un nouveau projet sur STM32CubeIDE**

  File → New → STM32 Project
- **Choix de la carte NUCLEO utilisée (NUCLEO-L152RE)**

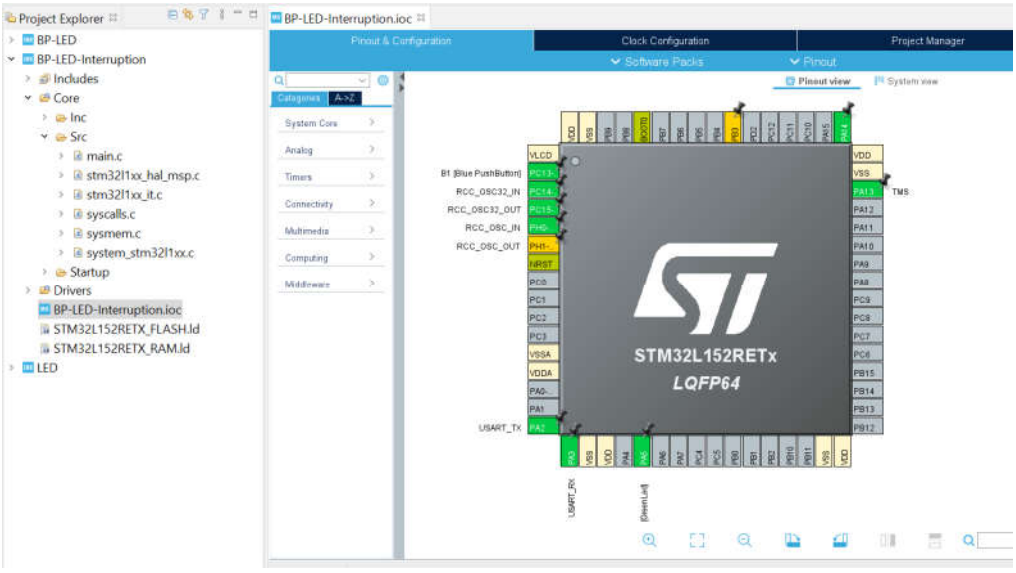
  Cliquer sur Next et donner un nom au projet ici « BP-LED-Interruption »

  Cliquer sur Next puis sur Finish

  Cliquer sur Yes à la question

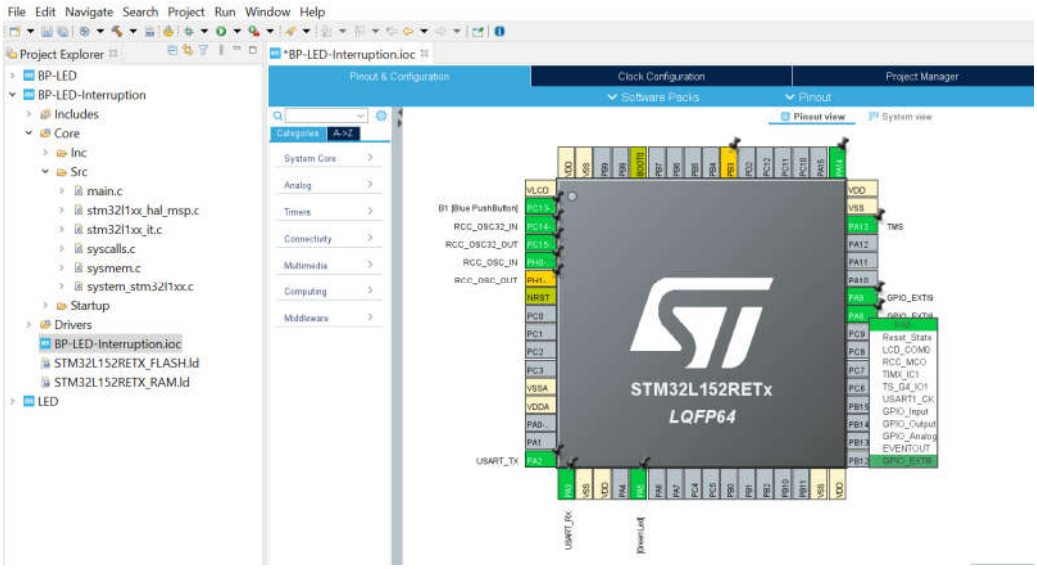
  « **Initialize all peripherals with their default Mode ?** »

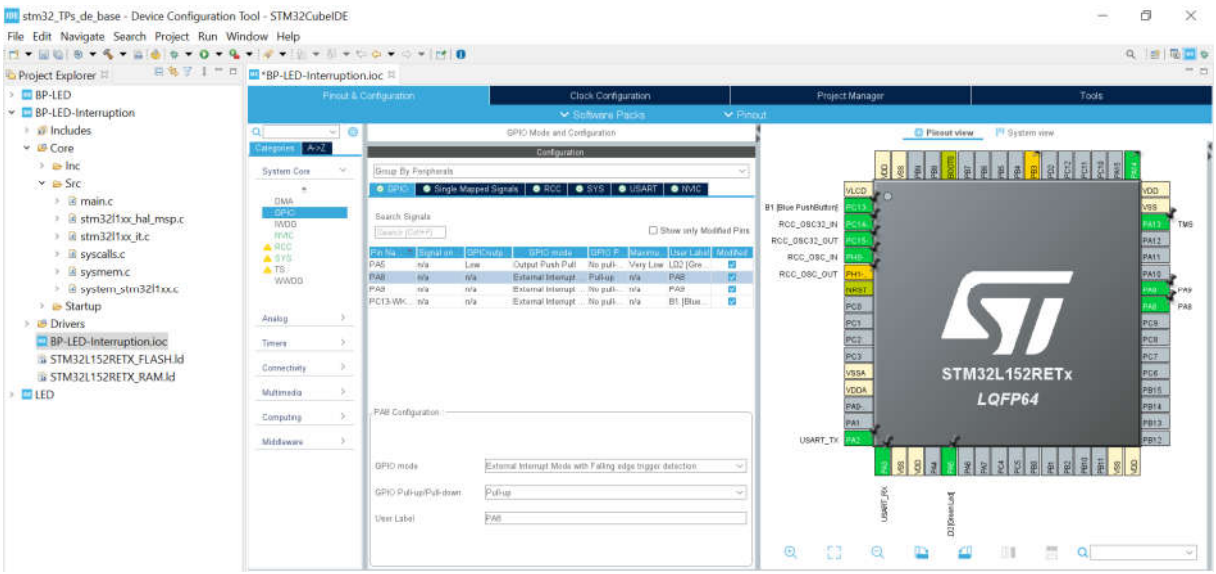
  Cliquer sur Yes
- **Création du projet « BP-LED-Interruption » sur STM32CubeIDE et ouverture du fichier LED.ioc (fichier STM32CubeMX)**



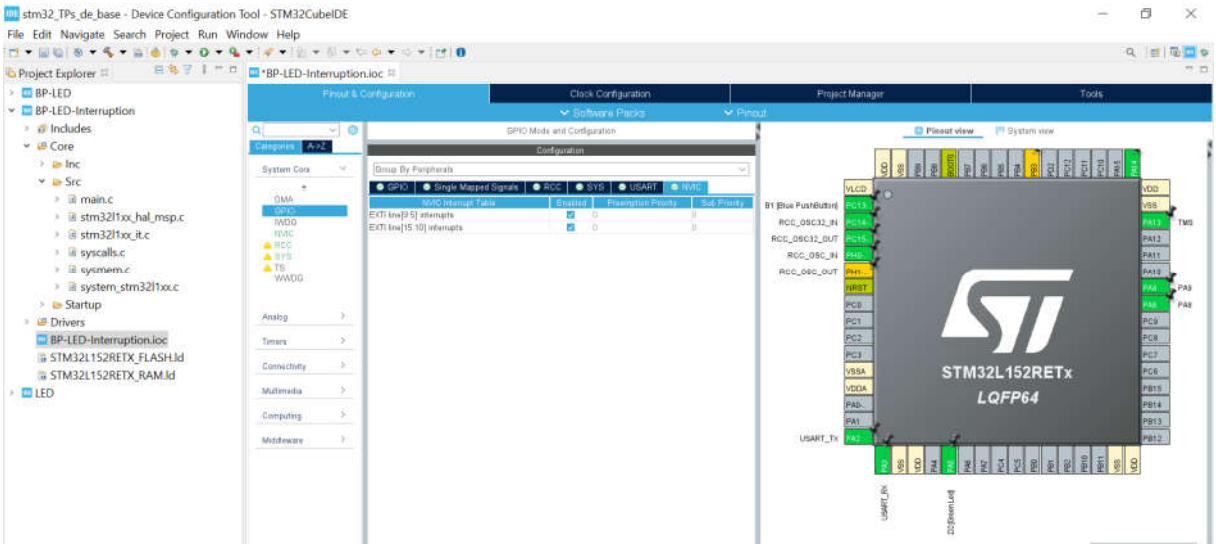
Le fait d'avoir coché la case **« Initialize all IP with their default mode »** conduit STM32CubeMX à initialiser toutes les ressources disponibles sur la carte (broches colorées en vert), dont celle qui va nous intéresser dans ce chapitre, LD2, qui est une LED de couleur verte connectée au GPIO du port PA5. (Et plus tard le bouton poussoir user PC13)
Hypothèse : l'onglet Clock Configuration est par défaut correctement configuré pour la carte sélectionnée.

Configurez maintenant les GPIO PA8 et PA9 comme indiqué ci-dessous.
Pull-Up et mode d'interruption externe avec détection de déclenchement par front descendant.

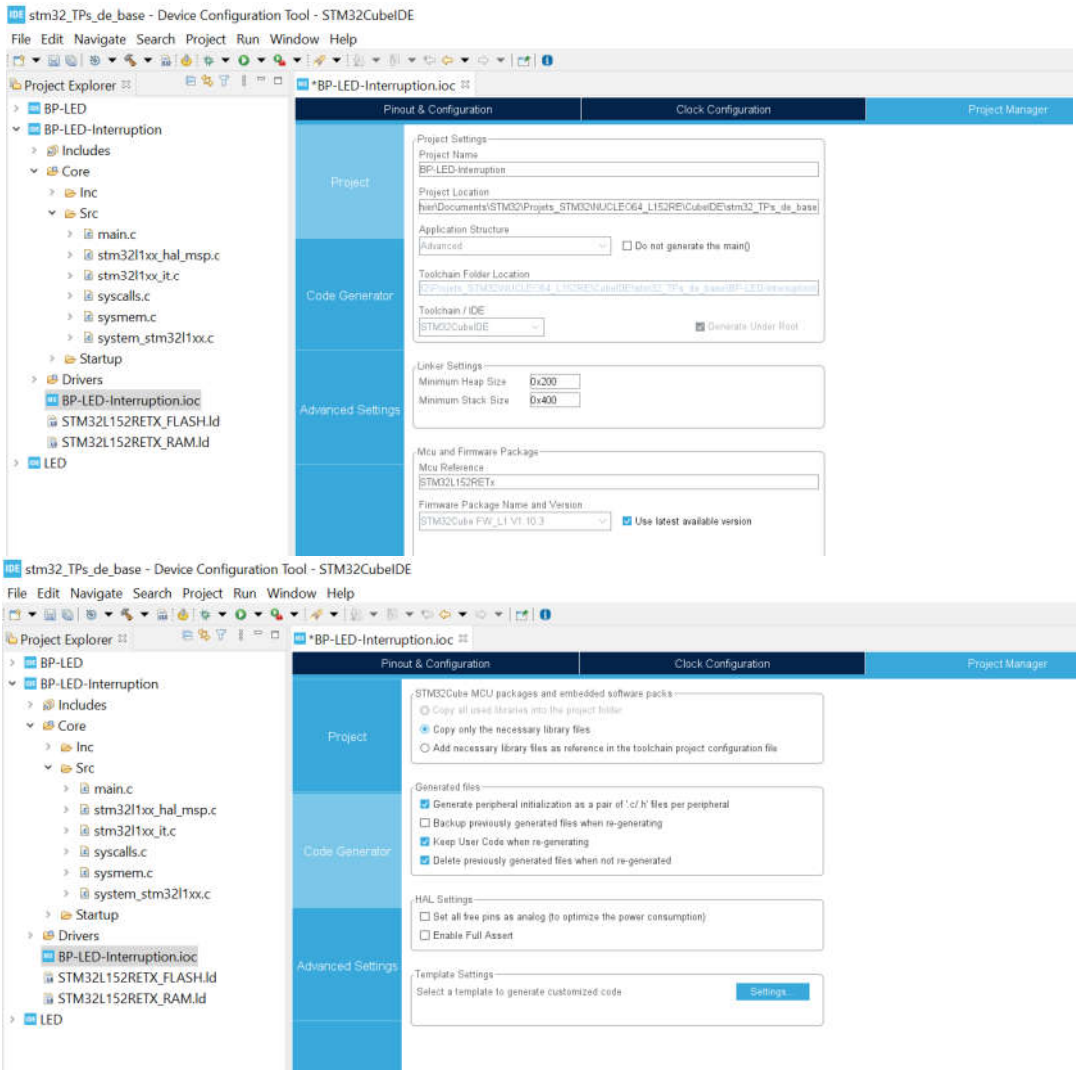




Activez maintenant l'INTERRUPTION sur PA8 et PA9.



➢ Configuration générale du projet



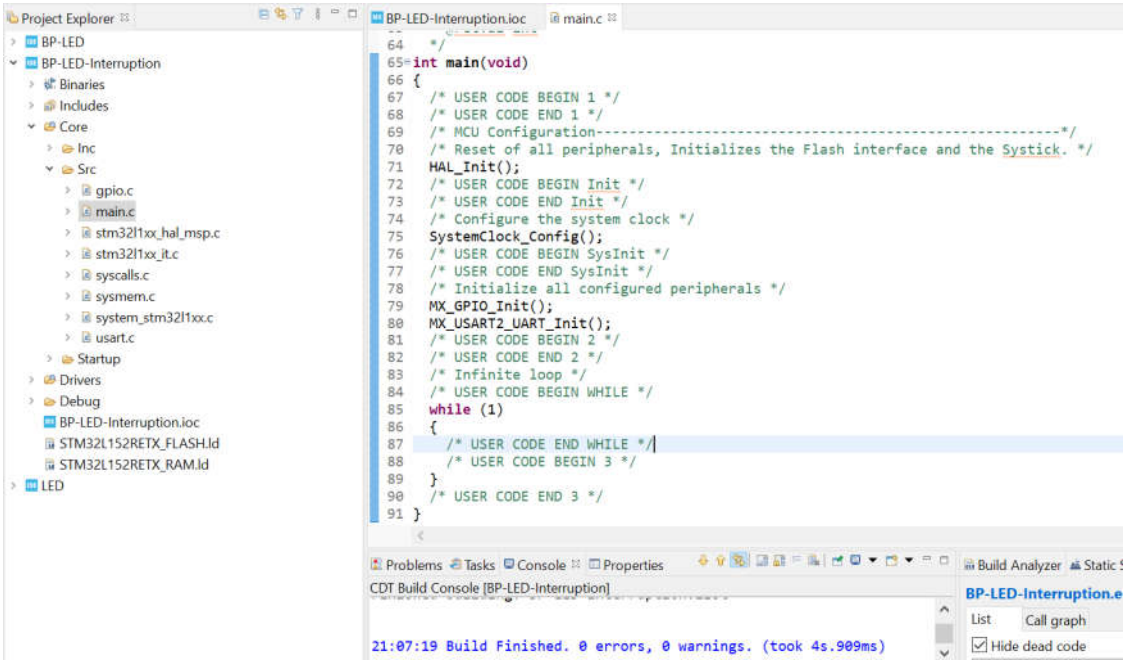
➢ Génération du code C d'initialisation

Merci de sauver le projet « BP-LED-Interruption »

Une fenêtre s'ouvre pour savoir si vous voulez générer le code C « cliquer sur yes »

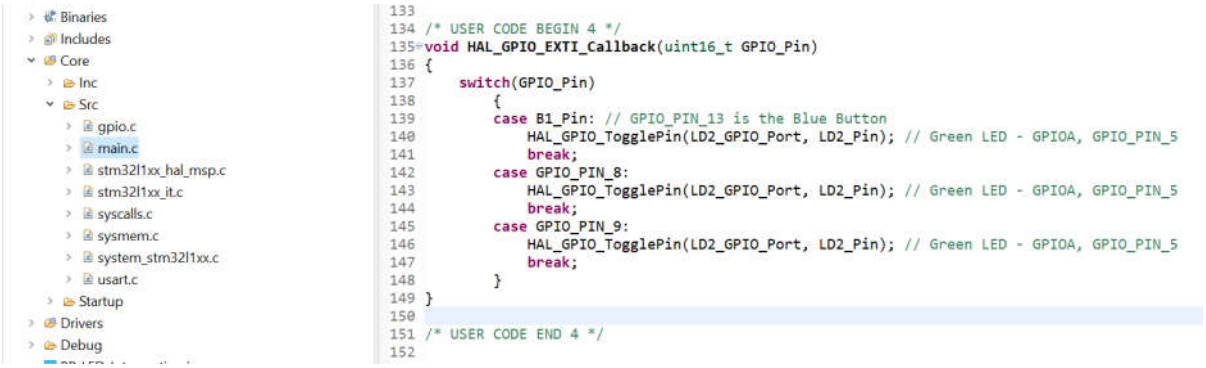
Ou alors aller dans l'onglet « Projet » et « Generate Code »

- **Compilation du projet « BP-LED-Interruption » :**



Maintenant, ouvrez le fichier **main.c** et insérer entre les balises utilisateur début et fin 4 :

```
/* USER CODE BEGIN 4 */
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
	switch(GPIO_Pin)
		{
		case B1_Pin: // GPIO_PIN_13 is the Blue Button
			HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin); // Green LED - GPIOA, GPIO_PIN_5
			break;
		case GPIO_PIN_8:
			HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin); // Green LED - GPIOA, GPIO_PIN_5
			break;
		case GPIO_PIN_9:
			HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin); // Green LED - GPIOA, GPIO_PIN_5
			break;
		}
}
/* USER CODE END 4 */
```



RE Compilation du projet « BP-LED-Interruption » : Onglet « Project » → « Build Project »

Vérifier si O Errors et 0 Warnings

➢ Programmation de la carte NUCLEO

La LED LD1 clignote alternativement vert et rouge pendant la programmation, puis se stabilise au rouge une fois la programmation terminée.
Un appui sur le bouton-poussoir RESET B2 de la carte Nucleo-L152RE initialise le programme.

➢ Vérification sur la carte NUCLEO

Test sur la carte NUCLE-L152RE du respect du cahier des charges :
Maintenant, si vous appuyez sur le bouton bleu ou que vous vous connectez à la masse PA8 ou PA9, la DEL verte change d'état.

Remarques sur la gestion des interruptions :

Maintenant, si vous regardez dans le fichier : **stm32f4xx_it.c** vous devez voir les deux fonctions qui gèrent les interruptions externes, voir ci-dessous.

```
 / **
* @brief Cette fonction gère les interruptions de ligne EXTI [9:5].
* /
void EXTI9_5_IRQHandler (void)
{
/ * CODE UTILISATEUR DEBUT EXTI9_5_IRQn 0 * /

/ * USER CODE END EXTI9_5_IRQn 0 * /
HAL_GPIO_EXTI_IRQHandler ( GPIO_PIN_8 );  // Réinitialiser l'interruption PIN8
HAL_GPIO_EXTI_IRQHandler ( GPIO_PIN_9 );  // Réinitialiser l'interruption PIN9
/ * CODE UTILISATEUR DEBUT EXTI9_5_IRQn 1 * /

/ * USER CODE END EXTI9_5_IRQn 1 * /
}

/ **
* @brief Cette fonction gère les interruptions de ligne EXTI [15:10].
* /
void EXTI15_10_IRQHandler (void)
{
/ * CODE UTILISATEUR DEBUT EXTI15_10_IRQn 0 * /

/ * USER CODE END EXTI15_10_IRQn 0 * /
HAL_GPIO_EXTI_IRQHandler ( GPIO_PIN_13 );  // Réinitialiser l'interruption PIN13 - Bouton bleu
/ * CODE UTILISATEUR DEBUT EXTI15_10_IRQn 1 * /

/ * USER CODE END EXTI15_10_IRQn 1 * /
}
```

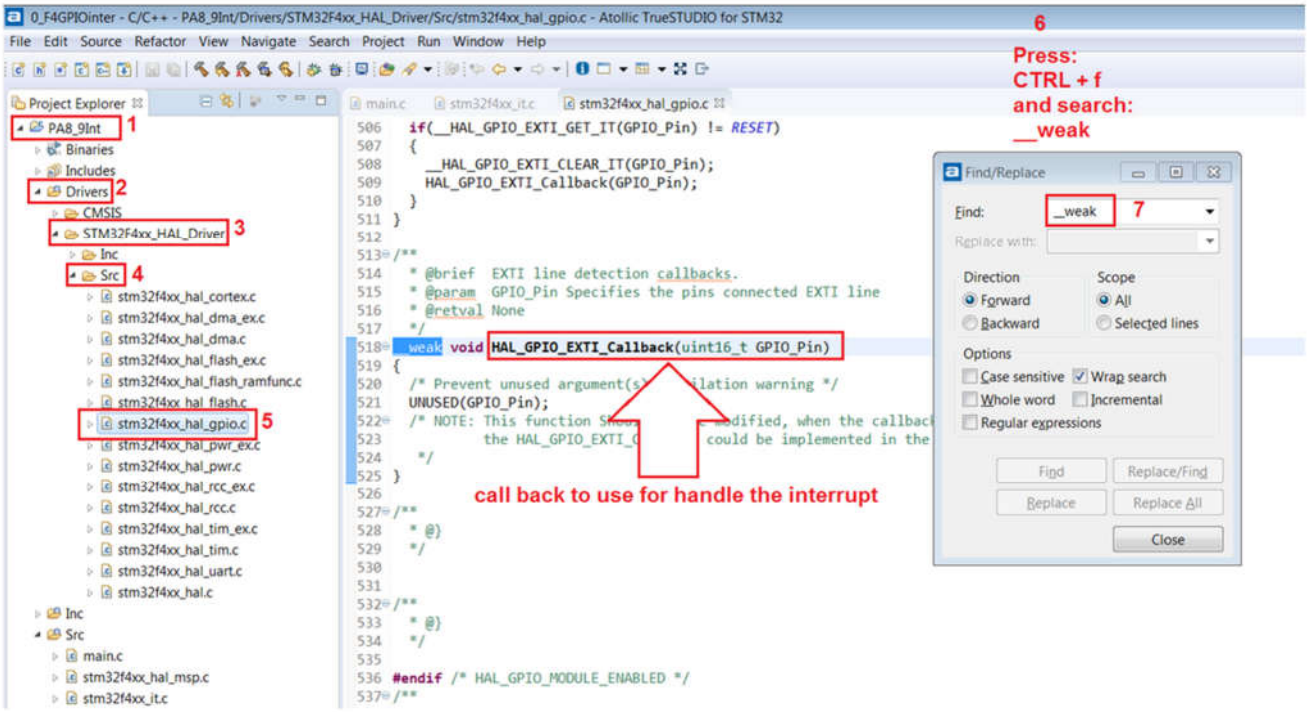
Notez également dans le fichier **gpio.c**, dans la fonction : **void statique MX_GPIO_Init (void)** il y a initialisation d'interruption de GPIO et également l'initialisation de NVIC.

Voir ci-dessous.

```
 / ** Configurer les pins en tant que
* Analogique
* Contribution
* Sortie
* EVENT_OUT
* EXTI
```

```
* /
void statique MX_GPIO_Init (void)
{

  GPIO_InitTypeDef GPIO_InitStruct;

 / * Activation de l'horloge des ports GPIO * /
  __HAL_RCC_GPIOC_CLK_ENABLE ();
  __HAL_RCC_GPIOH_CLK_ENABLE ();
  __HAL_RCC_GPIOA_CLK_ENABLE ();
  __HAL_RCC_GPIOB_CLK_ENABLE ();

  / * Configurer le niveau de sortie de la broche GPIO * /
  HAL_GPIO_WritePin (LD2_GPIO_Port, LD2_Pin, GPIO_PIN_RESET);

  / * Configurer la broche GPIO: B1_Pin - Bouton bleu * /
  GPIO_InitStruct.Pin = B1_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_IT_FALLING;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  HAL_GPIO_Init (B1_GPIO_Port, & GPIO_InitStruct);

  / * Configurer la broche GPIO: LD2_Pin * /
  GPIO_InitStruct.Pin = LD2_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init (LD2_GPIO_Port, & GPIO_InitStruct);

  / * Configurer les broches GPIO: PA8 PA9 * /
  GPIO_InitStruct.Pin = GPIO_PIN_8 | GPIO_PIN_9 ;
  GPIO_InitStruct.Mode = GPIO_MODE_IT_FALLING ;
  GPIO_InitStruct.Pull = GPIO_PULLUP ;
  HAL_GPIO_Init (GPIOA, & GPIO_InitStruct);

  / * EXTI interruption init * /
 HAL_NVIC_SetPriority (EXTI9_5_IRQn, 0, 0);
 HAL_NVIC_EnableIRQ (EXTI9_5_IRQn);

 HAL_NVIC_SetPriority (EXTI15_10_IRQn, 0, 0);
 HAL_NVIC_EnableIRQ (EXTI15_10_IRQn);

 }
```

Maintenant, pour gérer l'interruption, nous utilisons le **Callback** (**rappel)**.
Ouvrez le fichier : **stm32f4xx_hal_gpio.c** et recherchez (ctrl + f) le nom : **__weak**
Voir ci-dessous.
Pour gérer l'interruption, nous utilisons le rappel nommé : **HAL_GPIO_EXTI_Callback (uint16_t GPIO_Pin)**


0_F4GPIOinter - C/C++ - PA8_9Int/Drivers/STM32F4xx_HAL_Driver/Src/stm32f4xx_hal_gpio.c - Atollic TrueSTUDIO for STM32
File Edit Source Refactor View Navigate Search Project Run Window Help
Project Explorer
PA8_9Int 1
Binaries
Includes
Drivers 2
CMSIS
STM32F4xx_HAL_Driver 3
Inc
Src 4
stm32f4xx_hal_cortex.c
stm32f4xx_hal_dma_ex.c
stm32f4xx_hal_dma.c
stm32f4xx_hal_flash_ex.c
stm32f4xx_hal_flash_ramfunc.c
stm32f4xx_hal_flash.c
stm32f4xx_hal_gpio.c 5
stm32f4xx_hal_pwr_ex.c
stm32f4xx_hal_pwr.c
stm32f4xx_hal_rcc_ex.c
stm32f4xx_hal_rcc.c
stm32f4xx_hal_tim_ex.c
stm32f4xx_hal_tim.c
stm32f4xx_hal_uart.c
stm32f4xx_hal.c
Inc
Src
main.c
stm32f4xx_hal_msp.c
stm32f4xx_it.c
main.c
stm32f4xx_it.c
stm32f4xx_hal_gpio.c
if(__HAL_GPIO_EXTI_GET_IT(GPIO_Pin) != RESET)
__HAL_GPIO_EXTI_CLEAR_IT(GPIO_Pin);
HAL_GPIO_EXTI_Callback(GPIO_Pin);
* @brief EXTI line detection callbacks.
* @param GPIO_Pin Specifies the pins connected EXTI line
* @retval None
__weak void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
UNUSED(GPIO_Pin);
call back to use for handle the interrupt
#endif /* HAL_GPIO_MODULE_ENABLED */
6
Press:
CTRL + f
and search:
__weak
Find/Replace
Find: __weak 7
Replace with:
Direction
Forward
Backward
Scope
All
Selected lines
Options
Case sensitive
Wrap search
Whole word
Incremental
Regular expressions
Find
Replace/Find
Replace
Replace All
Close