

External Interrupts

This tutorial will demonstrate how to configure the GPIO that is connected to the user button as External Interrupt (EXTI) with falling edge trigger.

After this tutorial you will be able to:

- Create and configure STM32CubeMX project and generate initialization code
- Program and use interrupt to toggle LED on Nucleo-L476RG board

Hardware:

- Nucleo-L476RG board(64-pin),available at: www.st.com/en/evaluation-tools/nucleo-l476rg.html
- Standard-A -to- Mini USB cable

Literature:

- [STM32L476xx Datasheet](#)
- [UM1724](#) User manual STM32 Nucleo-64 boards
- [UM1884](#) Description of STM32L4/L4+ HAL and low-layer drivers
- [UM1718](#) User manual STM32CubeMX for STM32 configuration and initialization C code generation

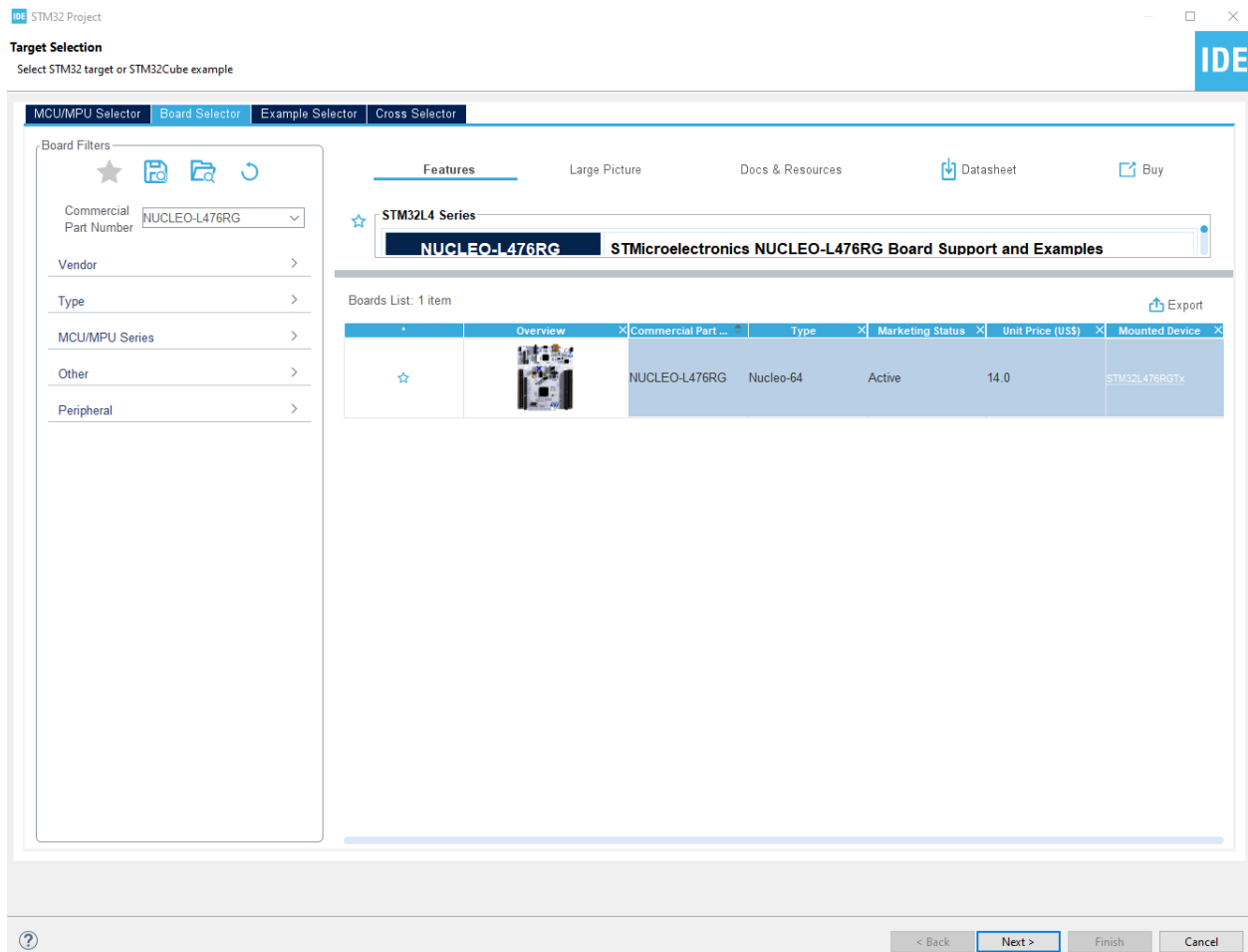
Stages

- 1: Create New Project with STM32CubeMX
- 2: Pinout Configuration
- 3: Clock Configuration
- 4: Configure project and Generate Source Code
- 5: Edit main.c
- 6: Build Project
- 7: Debug the Project

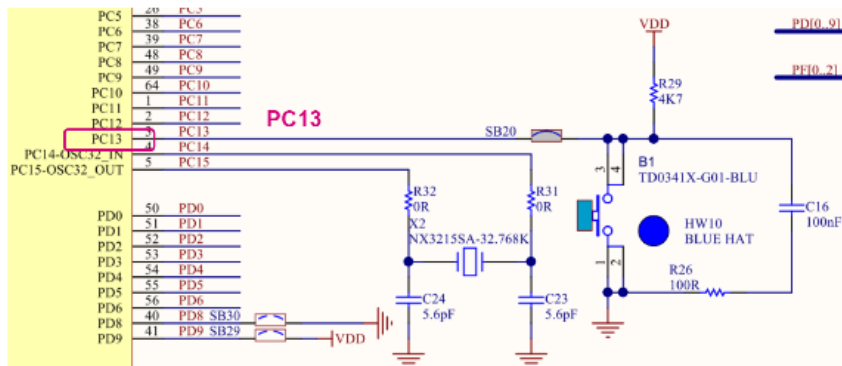


1: CREATE NEW PROJECT USING STM32CUBEMX:

- Open STM32CubeIDE
- Click *File -> New -> STM32 Project*. A target selection window will open.
- From Board Selector type *Nucleo-L476RG*. Select the board and click next.
- Name your project “Nucleo_L476RG_Button_Interrupt” and click Finish.
- Answer “Yes” to “Initialize all peripherals with their default mode?” popup.

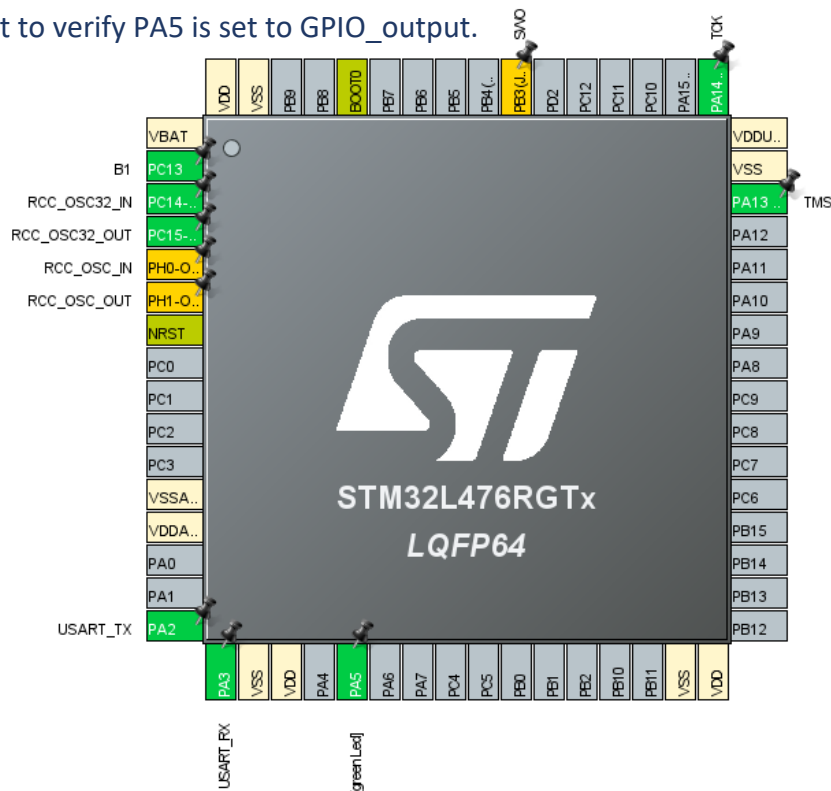


2: PINOUT CONFIGURATION



From the image above we can see user button B1 is connected to PC13. In STM32CubeMX right click PC13 and select GPIO_EXTI13. This will configure it as an external interrupt.

We also want to verify PA5 is set to GPIO_output.

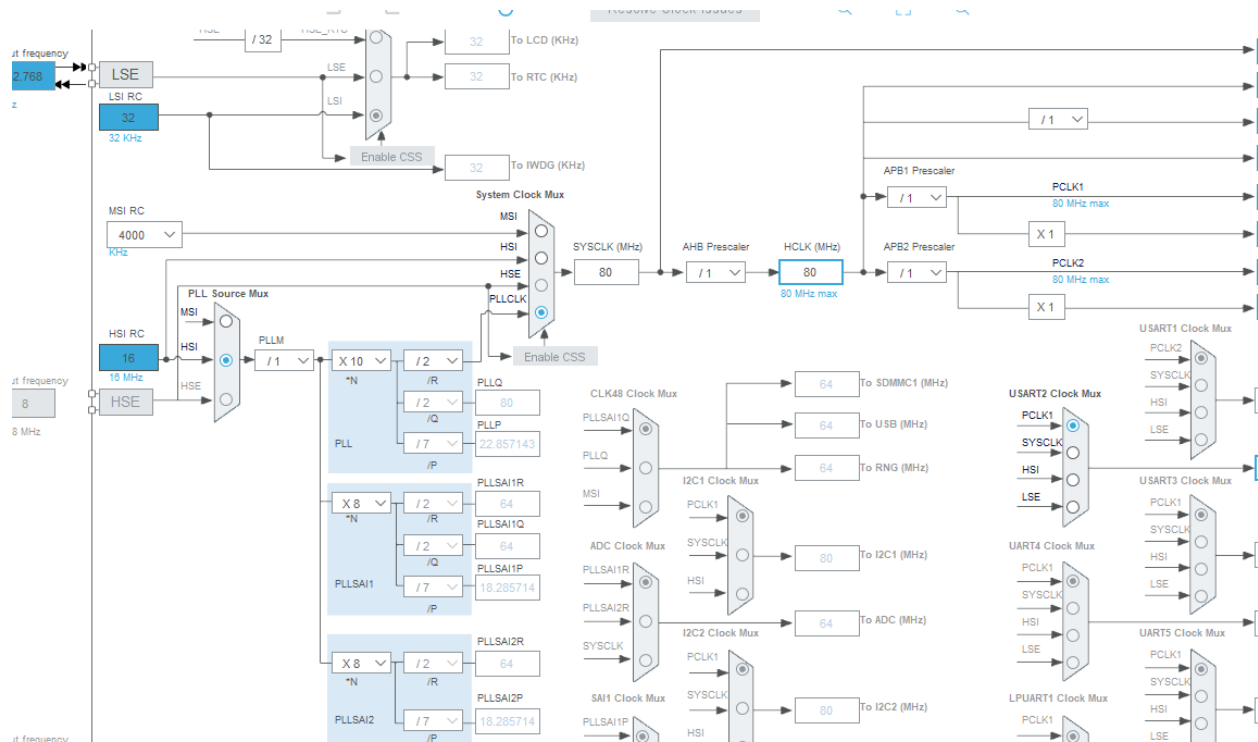


Under System Core-> GPIO, select PC13. Change the GPIO mode to “External Interrupt Mode with Falling edge trigger detection.” Then under the NVIC tab, check enable for “EXTI line[15:10] interrupts.”

3. CLOCK CONFIGURATION

In the clock configuration tab you can see that STM32CubeMX automatically configures the internal oscillator in the clock system with PLL @80MHz. The HSI is selected as the PLL source and the PLLCLK is selected in the system clock mux.

HCLK is set to 80 MHz.

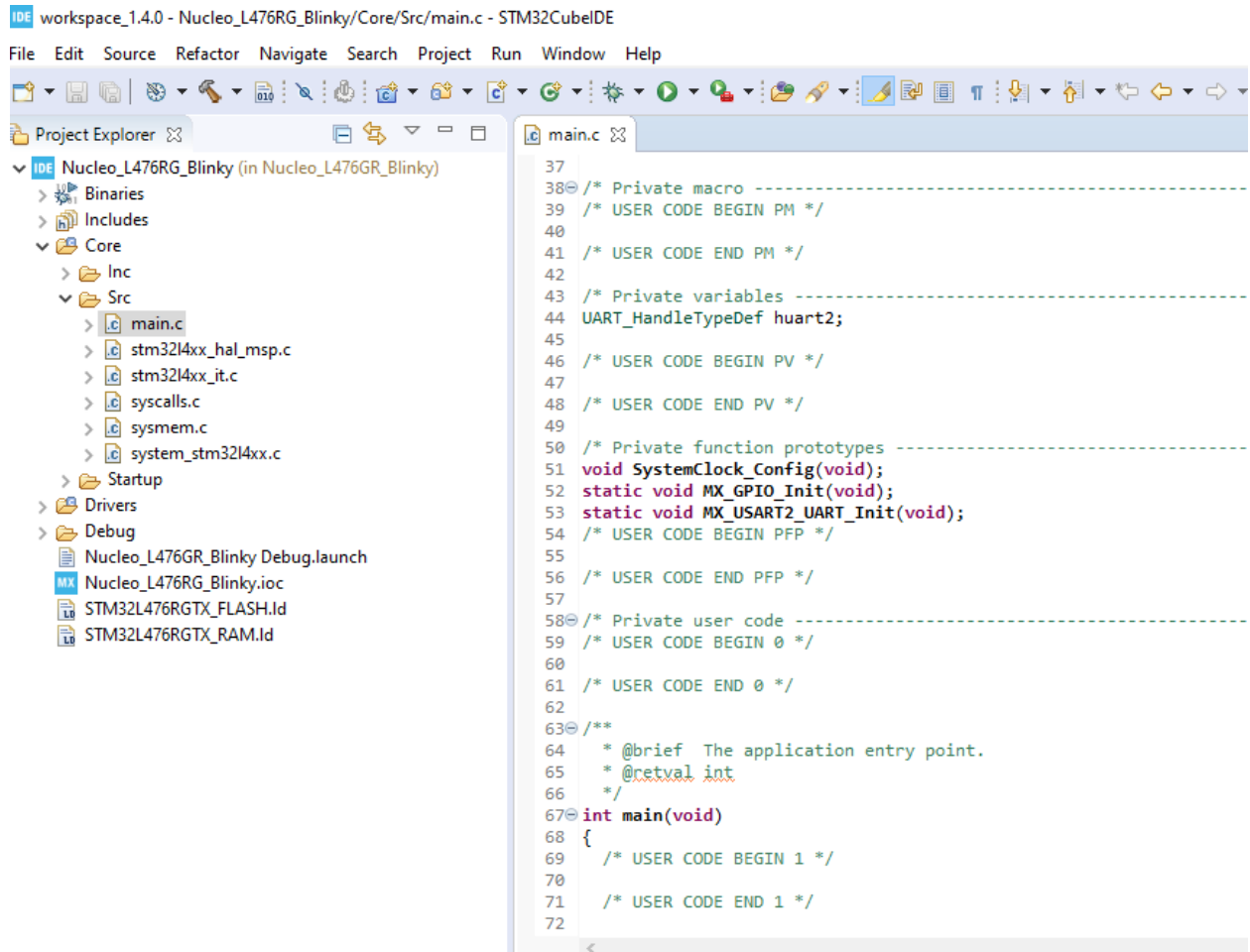


4: GENERATE CODE

We can now generate code. Click File->Save. You will be asked to generate code, press yes.

Under the project explorer navigate to *Core->Src->main.c*.

We can see that there was code automatically generated for us using STM32CubeMX.



The screenshot shows the STM32CubeIDE interface. The title bar indicates the workspace is 'workspace_1.4.0 - Nucleo_L476RG_Blinky/Core/Src/main.c - STM32CubeIDE'. The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar contains various icons for file operations, editing, and running. The Project Explorer on the left shows the project structure for 'Nucleo_L476RG_Blinky (in Nucleo_L476GR_Blinky)'. It includes folders for Binaries, Includes, Core, and Startup. Under the Core folder, the Src sub-folder is expanded, showing files like main.c, stm32l4xx_hal_msp.c, stm32l4xx_it.c, syscalls.c, systemem.c, and system_stm32l4xx.c. The main.c file is selected. The main editor displays the code for main.c, which includes private macros, variables, function prototypes, and the main function. The code is as follows:

```
37
38 /* Private macro -----
39 /* USER CODE BEGIN PM */
40
41 /* USER CODE END PM */
42
43 /* Private variables -----
44 UART_HandleTypeDef huart2;
45
46 /* USER CODE BEGIN PV */
47
48 /* USER CODE END PV */
49
50 /* Private function prototypes -----
51 void SystemClock_Config(void);
52 static void MX_GPIO_Init(void);
53 static void MX_USART2_UART_Init(void);
54 /* USER CODE BEGIN PFP */
55
56 /* USER CODE END PFP */
57
58 /* Private user code -----
59 /* USER CODE BEGIN 0 */
60
61 /* USER CODE END 0 */
62
63 /**
64  * @brief The application entry point.
65  * @retval int
66  */
67 int main(void)
68 {
69     /* USER CODE BEGIN 1 */
70
71     /* USER CODE END 1 */
72
```

5: EDIT main.c

We can add code between the “User code begin 4” and “User code end 4” lines. When the button is pushed, the LED will toggle.


```
228
229 }
230
231 /* USER CODE BEGIN 4 */
232 void EXTI15_10_IRQHandler(void) {
233     __HAL_GPIO_EXTI_CLEAR_IT(B1_Pin);
234     HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin);
235 }
236 /* USER CODE END 4 */
```

NOTE: This function may already be defined in *Core->Src->stm32l4xx_it.c*. If so, during compilation an error will be thrown. The code within the function may either be defined in *main.c*, or in *stm32l4xx_it.c* where the function was generated for you. To avoid the error, make sure to define this function only once.

6: BUILD THE PROJECT

Connect your USB cable from the computer to your Nucleo Board. Right click the project from the project explorer and click “Build project” to compile the project.

7: DEBUG THE PROJECT

Click on the Debug toolbar icon to start the debug session. Another way to debug is to *Run->Debug* . 

Click the Resume icon to continue the execution.

We can now see the green LED(LD2) toggling on the Nucleo-L476RG board when pushing the blue user button.

