Blinking LED with STM32CubeMX and HAL

This tutorial will demonstrate how to utilize STM32CubeMX tool to initialize peripherals, build and generate C code using HAL libraries.

After this tutorial you will be able to:

- Create and configure STM32CubeMX project and generate initialization code
- Program and use HAL functions to blink 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:

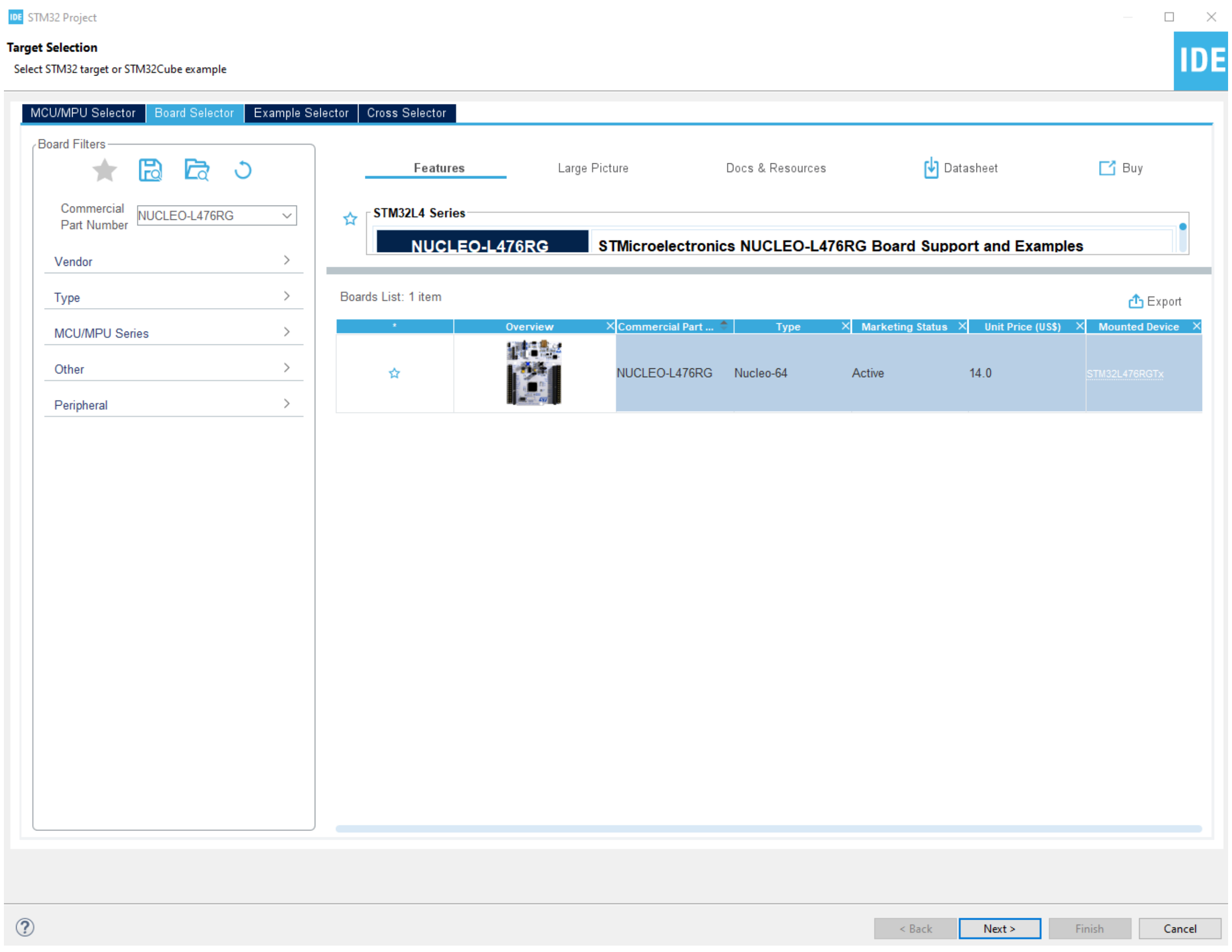
- STM32L476xx Datasheet
- UM1724 User manual STM32 Nucleo-64 boards
- UM1884 Description of STM32L4/L4+ HAL and low-layer drivers
- UM1718 User manual STM32CubeMX for STM32 configuration and initialization C code generation

Stages

- 1: Create New Project with STM32CubeMX
- 2: Pinout Configuration
- 3: Clock Configuration
- 4: GPIO Configuration
- 5: Configure project and Generate Source Code
- 6: Edit main.c to blink LED
- 7: Build Project
- 8: 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_Blinky” and click Finish.
- Answer “Yes” to “Initialize all peripherals with their default mode?” popup.



STM32 Project

Target Selection

Select STM32 target or STM32Cube example

MCU/MPU Selector | Board Selector | Example Selector | Cross Selector

Board Filters

Commercial Part Number: NUCLEO-L476RG

Vendor
Type
MCU/MPU Series
Other
Peripheral

Features | Large Picture | Docs & Resources | Datasheet | Buy

STM32L4 Series

NUCLEO-L476RG — STMicroelectronics NUCLEO-L476RG Board Support and Examples

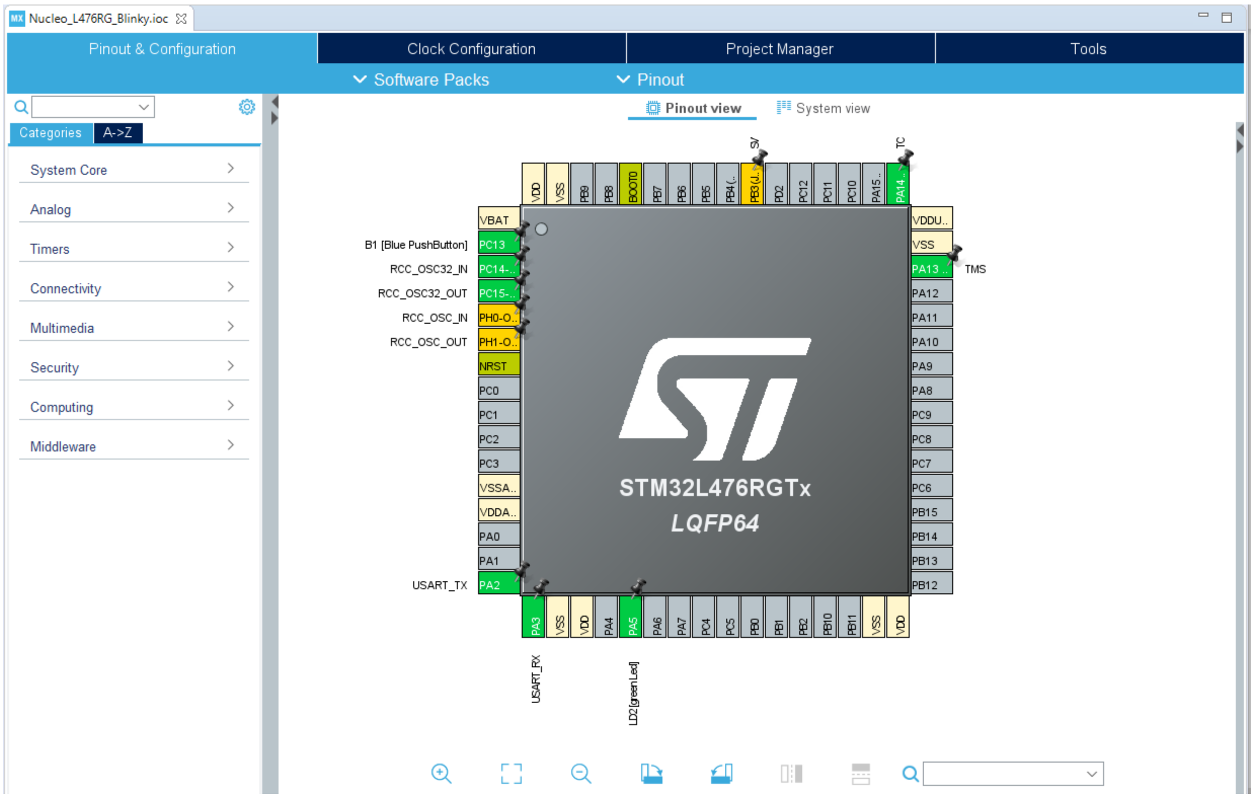
Boards List: 1 item

Export

*	Overview	Commercial Part ...	Type	Marketing Status	Unit Price (US$)	Mounted Device
☆		NUCLEO-L476RG	Nucleo-64	Active	14.0	STM32L476RGTx

< Back | Next > | Finish | Cancel

2: PINOUT CONFIGURATION



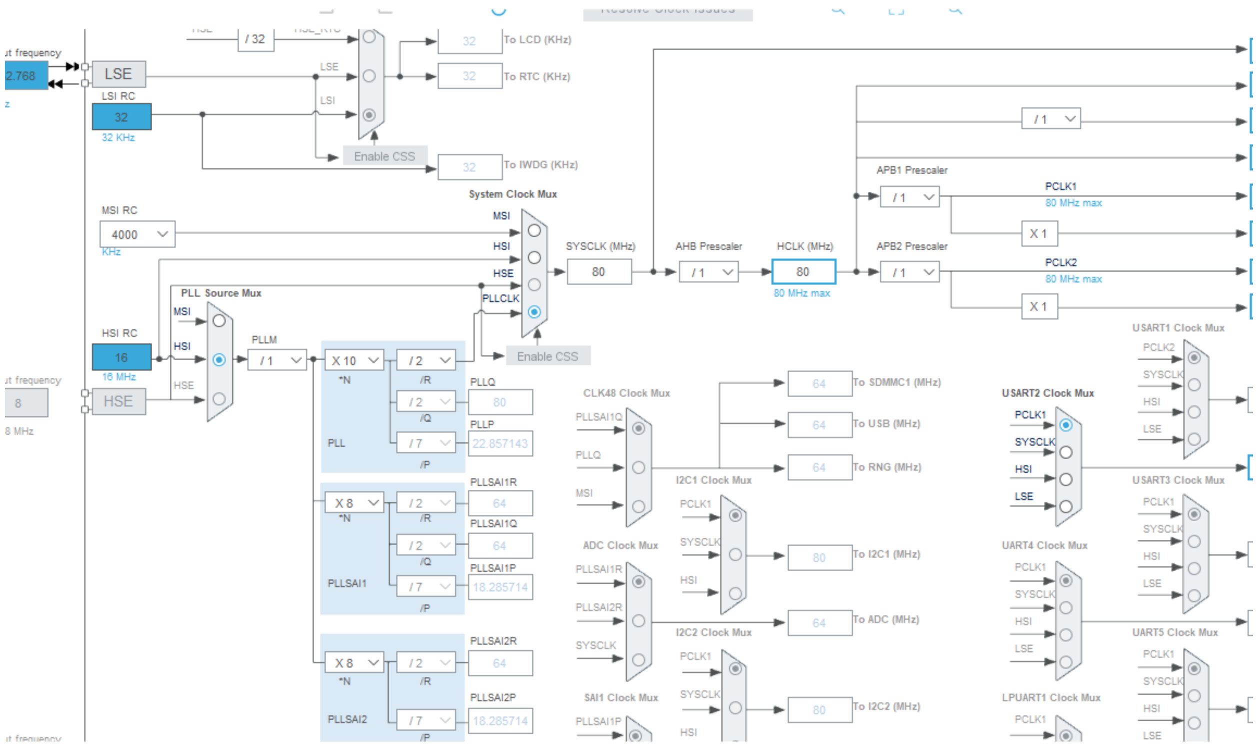
After a board is selected, STM32CubeMX allows you to select pinout settings for the board for communication interfaces, pin assignments, LEDs, and other functions.

This example shows the use of the green LED pin LD2 as a GPIO_output. This can be verified by hovering over the PA5 pin.

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 HIS is selected as the PLL source and the PLLCLK is selected in the system clock mux.

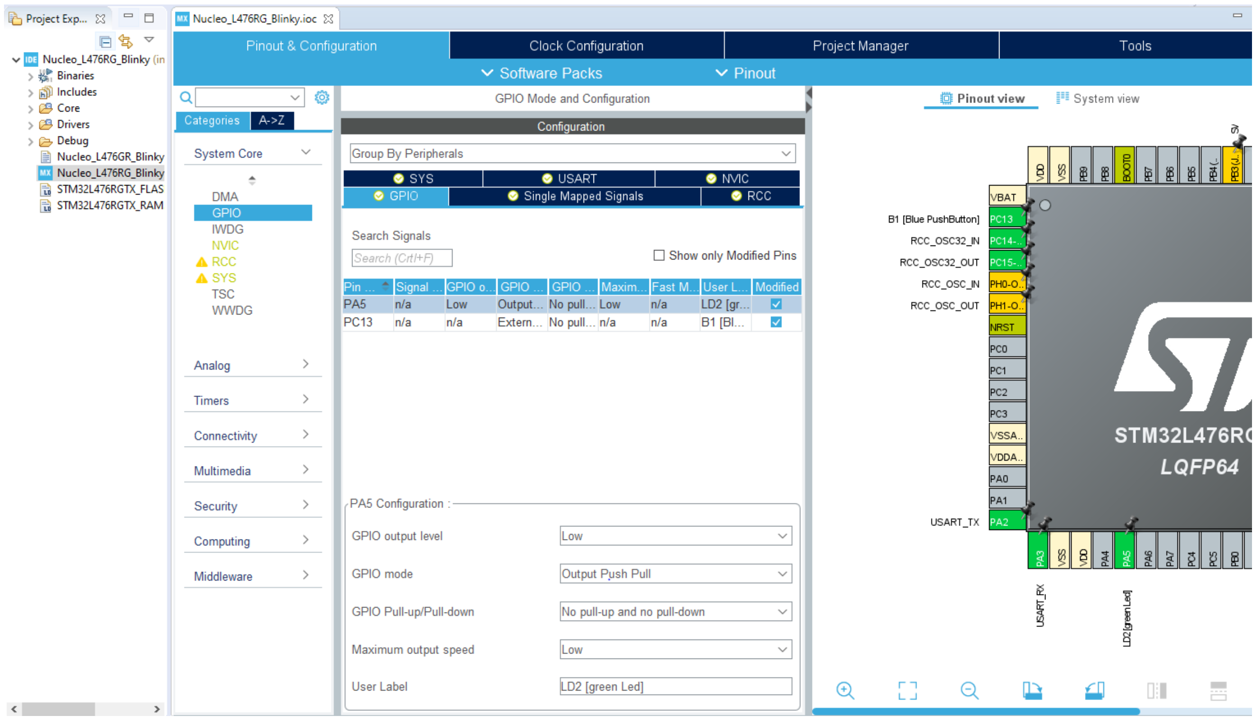
HCLK is set to 80 MHz.



4. GPIO CONFIGURATION

To configure GPIOs, click the GPIO under "System Core" on the left menu. Here we can view parameters and configurations.

- GPIO Output Level set to Low
- GPIO mode automatically configures pins with the relevant alternate function and GPIOs into Output Push Pull mode.
- GPIO Pull-up/Pull-down is set to no pull up and no pull down
- GPIO Maximum output speed set to Low by default.
- The User Label is a name assigned to the GPIO and can be changed here.



Pin ...	Signal ...	GPIO o...	GPIO ...	GPIO ...	Maxim...	Fast M...	User L...	Modified
PA5	n/a	Low	Output...	No pull...	Low	n/a	LD2 [gr...	✓
PC13	n/a	n/a	Extern...	No pull...	n/a	n/a	B1 [Bl...	✓

PA5 Configuration :

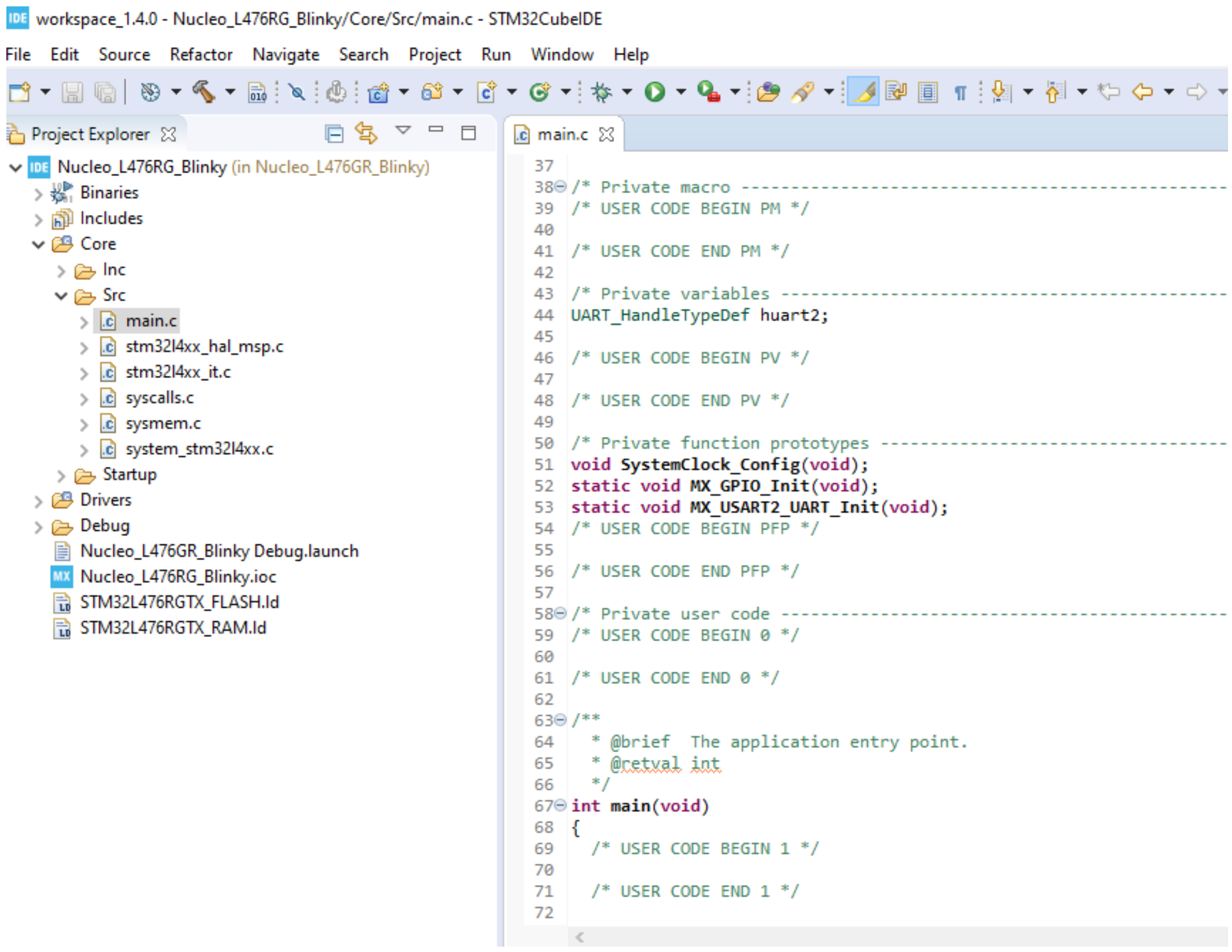
GPIO output level	Low
GPIO mode	Output Push Pull
GPIO Pull-up/Pull-down	No pull-up and no pull-down
Maximum output speed	Low
User Label	LD2 [green Led]

5: CONFIGURE PROJECT AND GENERATE CODE

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

Under the project explorer go *to Core->Src->main.c*. This is where we will write code to blink the LED.

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

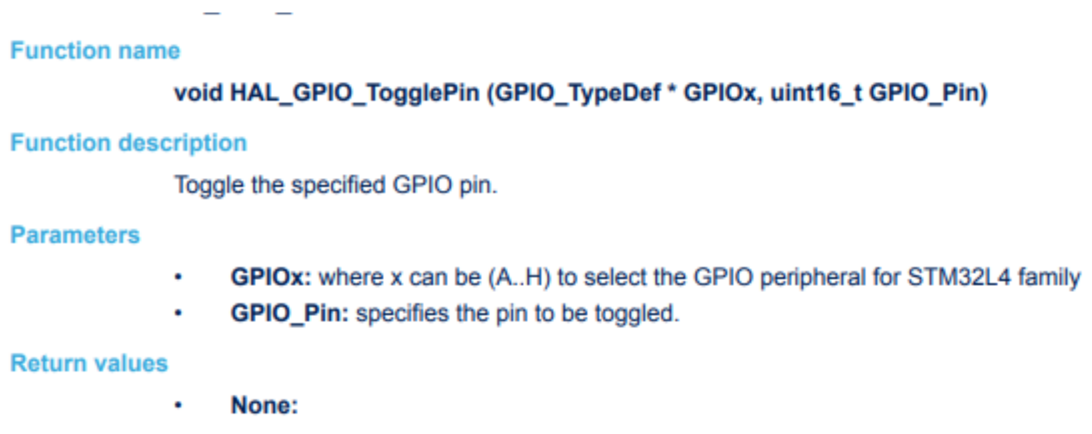

workspace_1.4.0 - Nucleo_L476RG_Blinky/Core/Src/main.c - STM32CubeIDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
main.c

```
Nucleo_L476RG_Blinky (in Nucleo_L476GR_Blinky)
  Binaries
  Includes
  Core
    Inc
    Src
      main.c
      stm32l4xx_hal_msp.c
      stm32l4xx_it.c
      syscalls.c
      sysmem.c
      system_stm32l4xx.c
    Startup
  Drivers
  Debug
  Nucleo_L476GR_Blinky Debug.launch
  Nucleo_L476RG_Blinky.ioc
  STM32L476RGTX_FLASH.ld
  STM32L476RGTX_RAM.ld
```

```c

/* Private macro -----------------------------------------------
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables -------------------------------------------
UART_HandleTypeDef huart2;

/* USER CODE BEGIN PV */

/* USER CODE END PV */

/* Private function prototypes ---------------------------------
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_USART2_UART_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code -------------------------------------------
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

```

6: EDIT MAIN.C to toggle the LED.

If you refer to the UM1884 “Description of STM32L4/L4+ HAL and low-layer drivers” user manual we can look at functions to use. Below is the function we want to utilize.



Function name

void HAL_GPIO_TogglePin (GPIO_TypeDef * GPIOx, uint16_t GPIO_Pin)

Function description

Toggle the specified GPIO pin.

Parameters

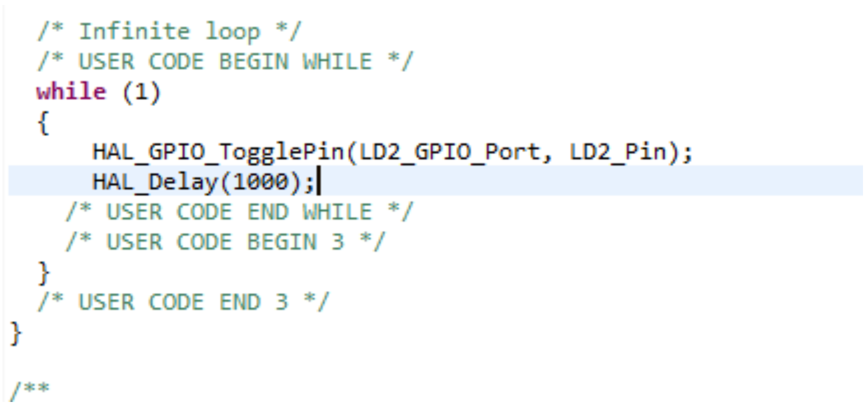
- **GPIOx:** where x can be (A..H) to select the GPIO peripheral for STM32L4 family
- **GPIO_Pin:** specifies the pin to be toggled.

Return values

- **None:**

We can add code to the main.c file inside the while(1) loop within int main() between `/* USER CODE BEGIN WHILE */` and `/* USER CODE END WHILE */`

It is important to write your code between these lines. If code is ever generated again from STM32CubeMX, any code written outside of these boundaries will be deleted.



```
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
      HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin);
      HAL_Delay(1000);
    /* USER CODE END WHILE */
    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

/**
```

7: 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.

8: 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.