



Step 2:



Blinking LED with STM32CubeMX and HAL

Target description

This tutorial shows how to use STM32CubeMX tool to initialize the peripherals, build and generate your starting projects with initialization C code using HAL libraries.

After this tutorial, you should be able to:

- create and configure STM32CubeMX project and generate initialization code.
- program and use HAL functions to blink a LED on the NUCLEO-L476RG board

Prerequisites

- Previous Tutorial:
- Step1: Tools Installation and First test.

Hardware

The hardware requirements to start the application are the following:

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

Literature

- STM32L476xx Datasheet
- UM1724 User manual STM32 Nucleo-64 boards
- <u>UM1884</u> Description of STM32L4/L4+ HAL and low-layer drivers
- <u>UM1718</u> User manual STM32CubeMX for STM32 configuration and initialization C code generation
- Video: How to build a "Blink LED" project from STM32CubeMX for ST/Atollic TrueSTUDIO® for STM32

Stages

- 1: Create New Project using STM32CubeMX (page 2)
- 2: Pinout Configuration (page 3)
- 3: Clock Configuration (page 5)
- 4: GPIO Configuration (page 6)
- 5: Configure project and Generate Source Code (page 8)
- 6: Edit main.c to Toggle the LED (page 9)
- 7: Build the Project (page 11)
- <u>8: Debug the Project</u> (page 11)





Blinking LED with STM32CubeMx and HAL

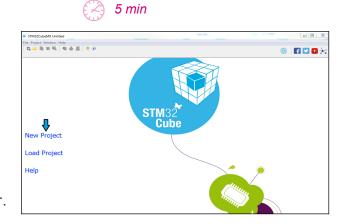
In this tutorial, we will explain step-by-step how to blink a LED on the NUCLEO-L476RG board, using STM32CubeMX tool, HAL, and TrueSTUDIO IDE. Steps to follow:

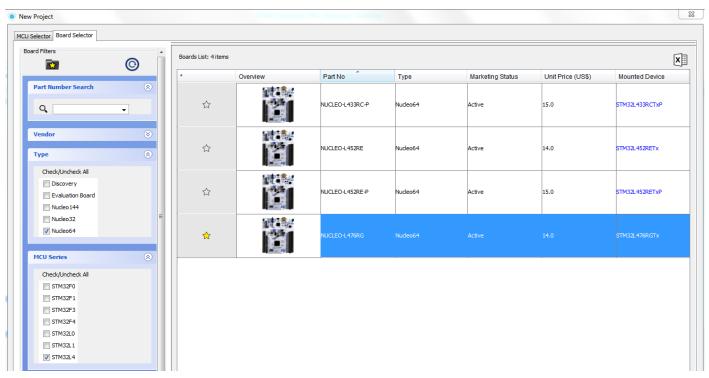
1: CREATE NEW PROJECT USING STM32CUBEMX

- Run STM32CubeMX tool.
- Click New Project or Menu -> File -> New Project.
- From Board Selector, filter to select and use

NUCLEO-L476RG board:

- Check *Nucleo64* type.
- Check STM32L4 on MCU Series.
- Select NUCLEO-L476RG board using Board selector.
- Click Start Project to continue.
- Answer Yes "Initialize all peripherals with their default Mode?" popup.





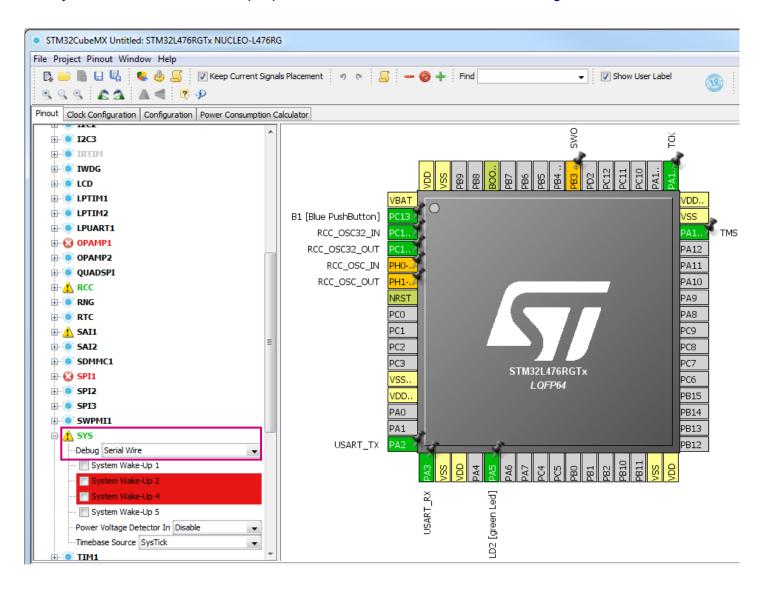




2: PINOUT CONFIGURATION



Verify in "Pinout" tab, under SYS peripheral, that Serial Wire is selected as Debug interface.



The corresponding pins PA13, PA14 are assigned and configured automatically.

When a board is selected, STM32CubeMX allows automatically the pinout setting for the board with the pin assignments for the communication interfaces, LEDs, and other functions.

(To configure LED pins, check in the <u>STM32 Nucleo-64 boards</u> User Manual and <u>STM32L476xx Datasheet</u> which LED pins to use).

TIPS & TRICKS

To see alternate pins for a signal, drag and drop the signal to a pin while keeping the Ctrl key pressed.





This example shows the use of the green LED pin LD2 present on the NUCLEO-L476RG board as GPIO_Output.

To verify that LD2 is set to **GPIO_Output** mode:

Type "LED" in the *Find* field and check that LD2(green Led) is enabled to PA5 pin as **GPIO_Output.** When found, the pin that matches the search criteria blinks on the Chip view. Click on Chip view to stop the blinking. Signals can be set directly from the pinout view.



TIPS & TRICKS

Pinout search field allows the user to search for a pin name, signal name, or signal label in the Pinout view. When found, the pin or set of pins that matches the search criteria blinks on the Chip view. Click the Chip view, for it to stop blinking.



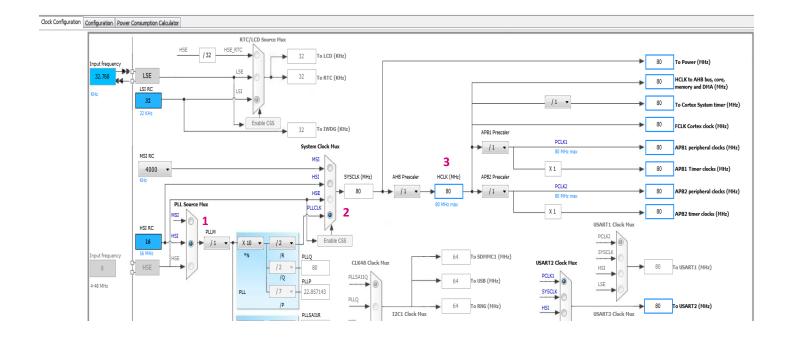


3: CLOCK CONFIGURATION



In the *Clock Configuration* tab, check that STM32CubeMX automatically configures the internal oscillator in the clock system with PLL @80MHz and proposes the PLL configuration as follows:

- 1. HSI selected in PLL Source Mux (HSI High Speed Internal clock)
- 2. PLLCLK selected in the System Clock Mux
- 3. HCLK set to 80



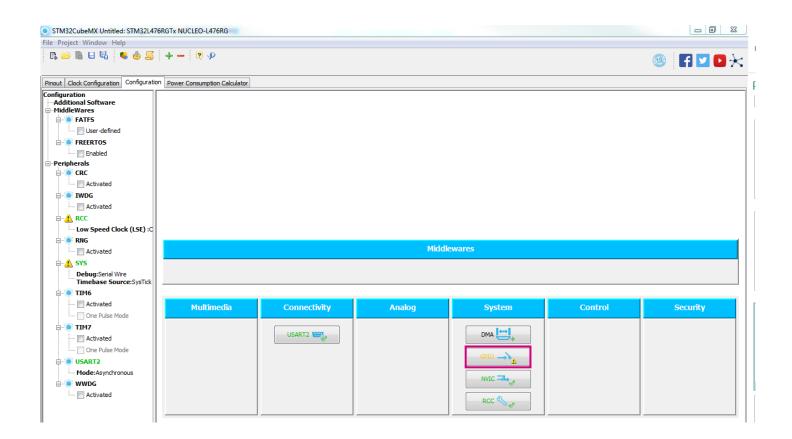




4: GPIO CONFIGURATION



To configure the GPIOs, click the GPIO button in the *Configuration* Tab to open the GPIO Configuration window.

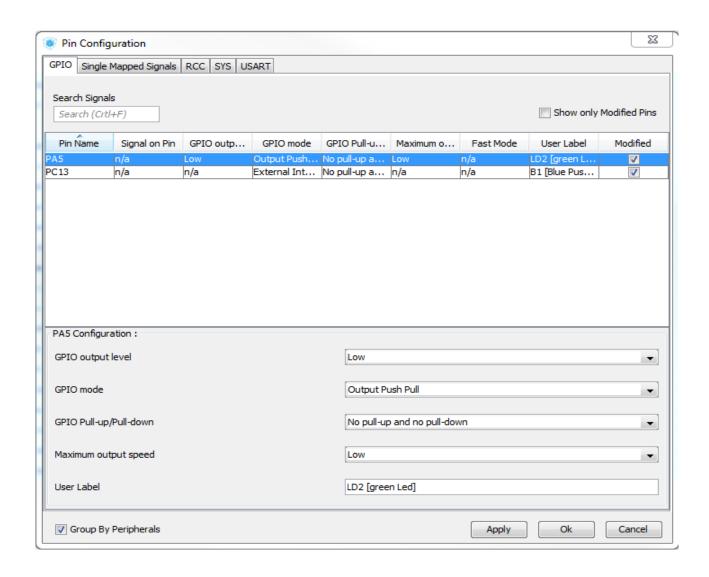






In the *GPIO* Tab, select Pin Name column *PA5* to display the corresponding GPIO parameters and configuration to drive the NUCLEO-L476RG LED:

- GPIO Output level: it is set to Low by default and can be changed to High.
- GPIO mode automatically configures the pins with the relevant alternate function and GPIOs into
 Output Push Pull mode.
- *GPIO Pull-up/Pull-down* set to *No pull-up and no pull-down* by default can be configured when other choices are allowed.
- GPIO Maximum output speed set to Low by default for power consumption optimization can be changed to a higher frequency to fit application requirements.
- User Label is a name assigned to a GPIO. The GPIO can be found under this name via the *Find* menu.
 Click *Apply* then *OK* to close the window.



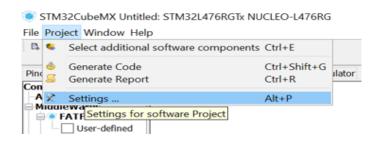




5: CONFIGURE PROJECT AND GENERATE SOURCE CODE

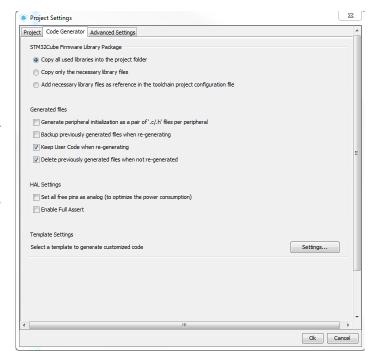


To configure the project, click on **Settings** in the menu.



In the *Code Generator* tab, ensure that the following options are checked:

- In STM32Cube Firmware Library Package section:
 Copy all used libraries into the project folder.
 This option is checked by default, for STM32CubeMX to copy driver (HAL, CMSIS) and middleware libraries to the user project folder, as relevant to the user configuration such as FatFS or USB.
- In Generated files section:
 Keep user code when regenerating the C code option, which only applies to user sections within STM32CubeMX generated files.

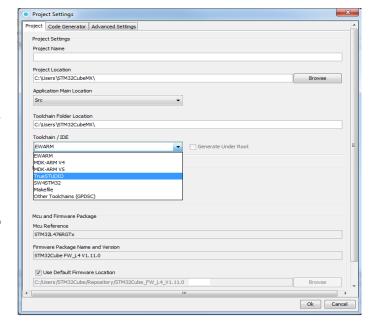


Switch to the *Project* tab and fill *Project Name* and *Project Location* fields, then generate C initialization code for *TrueSTUDIO* toolchain:

- Open Menu > Project > Settings (Alt + P)
- Under Project tab, set the Project Name and the Project Location.

Set Toolchain/IDE to TrueSTUDIO.

- Click on OK.
- Click on Project > Generate Code
 (Ctrl + Shift + G) or on the licon
- Click on *Open Project* to open the project with TrueSTUDIO.



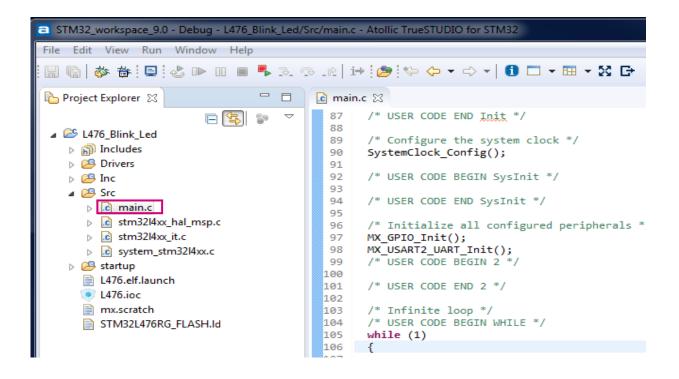




6: EDIT MAIN.C TO TOGGLE THE LED



In TrueSTUDIO, from the *Project Explorer* tab, open the main.c file, in *Src* folder and add the adequate functions for LED blinking, using HAL functions with STM32CubeL4 firmware package.



To get an idea about the usage of HAL functions, refer to the <u>UM1884</u> "Description of STM32L4/L4+ HAL and low-layer drivers" user manual, the common and generic functions to use.

HAL GPIO Generic Driver	UM1884
HAL_GPIO_Toggle	ePin .
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 (AH) to select the GPIO peripheral for STM32L4 family GPIO_Pin: specifies the pin to be toggled.
Return values	None:





User code can be added in the *main.c* file, inside the **while (1)** loop between /* USER CODE BEGIN 3 */ and / * USER CODE END 3 */ section (this will preserve your code after regeneration).

For LED toggling, use these functions:

```
HAL_GPIO_TogglePin (GPIOA, GPIO_PIN_5);
HAL Delay (100);  /* Insert delay 100 ms */
```

```
Project Explorer 🖂
                           _ _
                                   87
                                           /* USER CODE END Init */

■ L476_Blink_Led

                                     89
                                           /* Configure the system clock */
   ▶ M Includes
                                          SystemClock_Config();
                                     90
   Drivers
                                     91
                                          /* USER CODE BEGIN SysInit */
   Inc
                                     92
                                     93
   Src
                                     94
                                          /* USER CODE END SysInit */
     95
     /* Initialize all configured peripherals */
                                     96
     97
                                          MX_GPIO_Init();
                                          MX_GPIU_init();
MX_USART2_UART_Init();
                                     98
     99
                                          /* USER CODE BEGIN 2
   startup
                                    100
     L476.elf.launch
                                          /* USER CODE END 2 */
                                    101
    L476.ioc
                                    102
     mx.scratch
                                          /* Infinite loop */
                                    103
                                          /* USER CODE BEGIN WHILE */
     STM32L476RG_FLASH.Id
                                    104
                                    105
                                          while (1)
                                    106
                                    107
                                          /* USER CODE END WHILE */
                                    108
                                    109
                                    110
                                           /* USER CODE BEGIN 3 */
                                    111
                                          HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
                                    112
                                          HAL_Delay(100);
                                    113
                                          /* USER CODE END 3 */
                                    114
                                    115
                                    116
                                   117
```





7: BUILD THE PROJECT



To power the NUCLEO-L476RG, use its CN1 connector to connect in with a computer through a Standard-A -to- Mini-B USB cable.

Click on the project from the project explorer, then right click and select *Build Project*, to compile the project (or click on *Build* button on the toolbar).

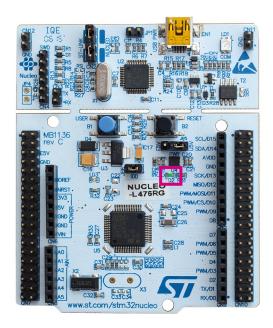
8: DEBUG THE PROJECT



2 min

Click on the **Debug** toolbar icon to start the debug session (or in the menu, select **Run > Debug**). Click on **Resume** icon to continue the execution.

Now watch the green LED (LD2) toggling on the Nucleo-L476RG board.



Now you are able to:

- create a new project using STM32CubeMX.
- configure project in STM32CubeMx and Generate initialization code.
- update project code in TrueSTUDIO IDE using HAL functions.
- execute a project in debug mode.
- make a LED blink.



