

Blink LED with Timer Interrupt

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 create timer interrupt to blink LED

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

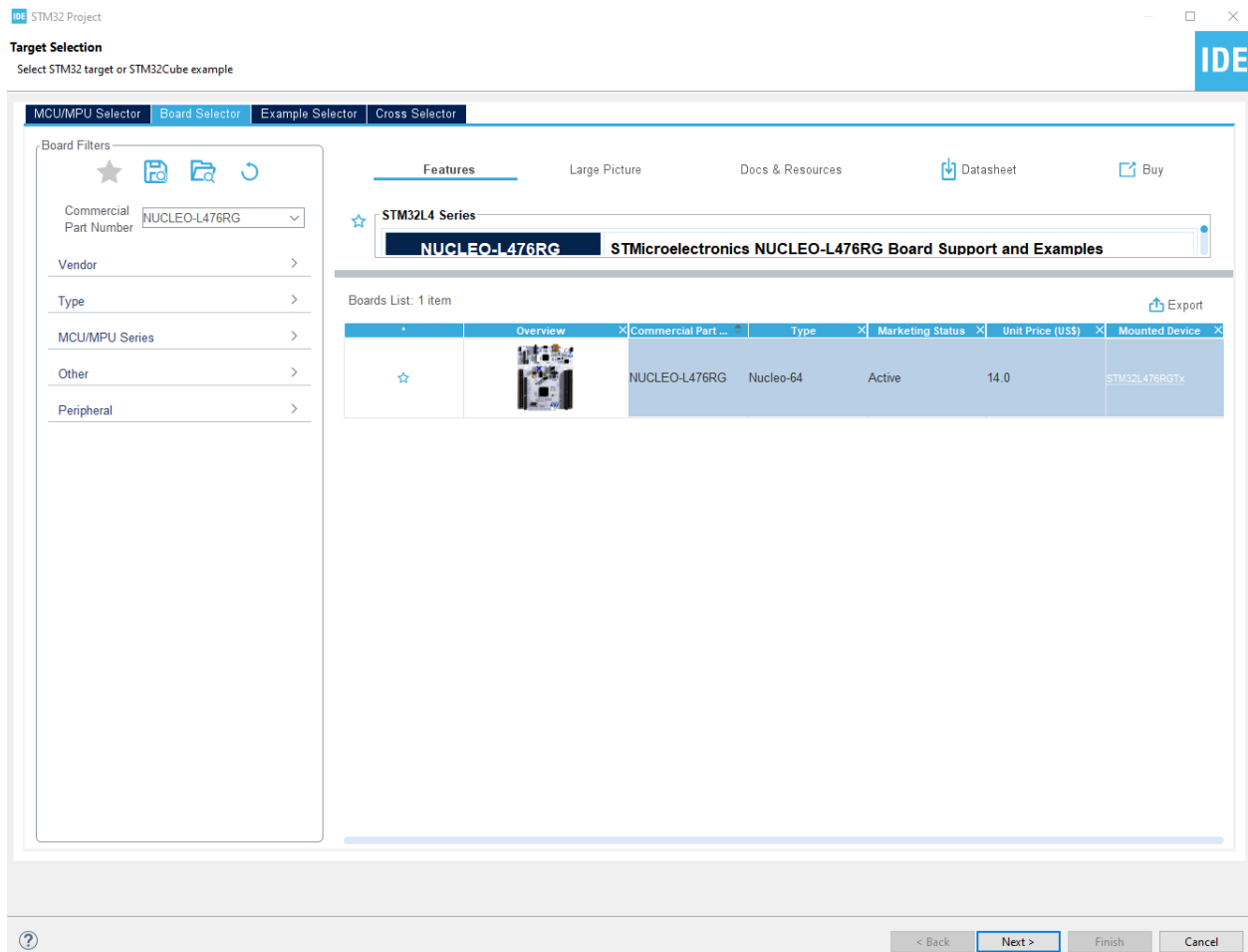
Stages

- 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



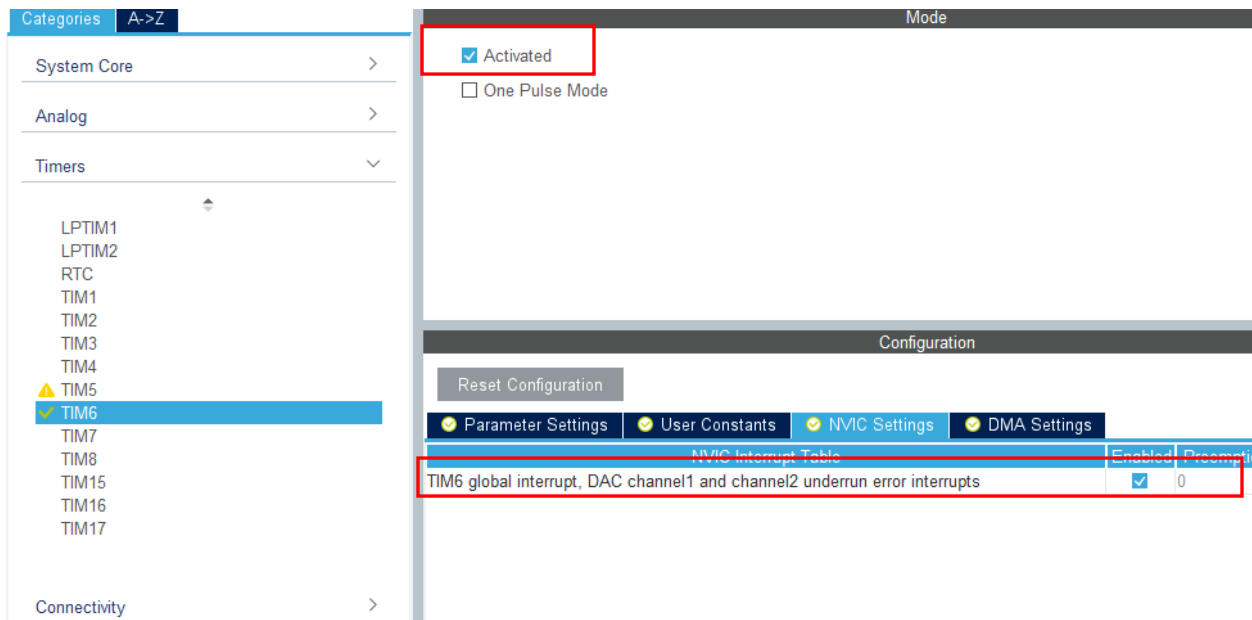
1: CREATE NEW PROJECT USING STM32CUBEMX:

- 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_Timer” and click Finish.
- Answer “Yes” to “Initialize all peripherals with their default mode?” popup.



2: Pinout Configuration

Under *Timers*->*TIM6* check “Activated”. Next, Under the NVIC Settings tab check “TIM6 global interrupt,DAC channel1 and channel2 underrun error interrupts.”



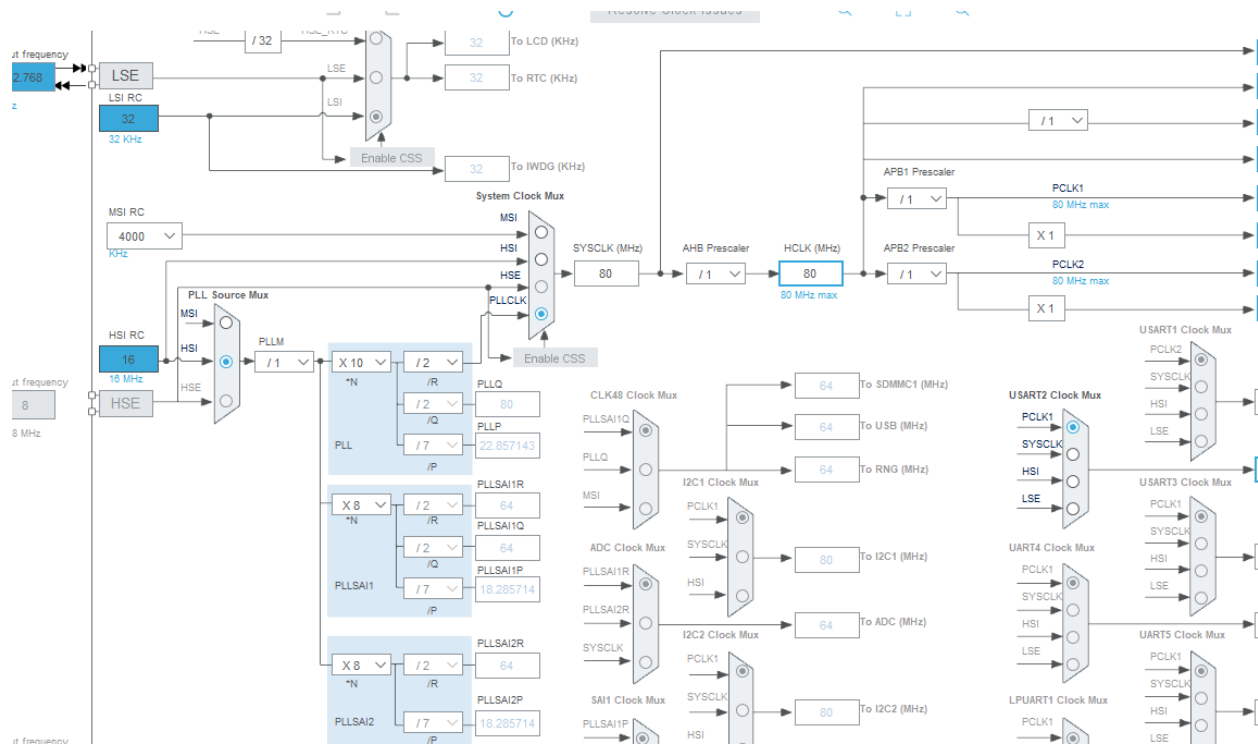
To make our LED blink every 0.5 seconds, we need to set the prescaler and counter period under the Parameter Settings tab. Set the values to match the image below:

Counter Settings		
Prescaler (PSC - 16 bits value)		39999
Counter Mode		Up
Counter Period (AutoReload Register - 16 bits value)		999
auto-reload preload		Disable
Trigger Output (TRGO) Parameters		
Trigger Event Selection		Reset (UG bit from TIMx_EGR)

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*.

5: EDIT main.c

In int main() we can use the following functions found in the UM1884 manual.

HAL_TIM_Base_Init

Function name

HAL_StatusTypeDef HAL_TIM_Base_Init (TIM_HandleTypeDef * htim)

Function description

Initializes the TIM Time base Unit according to the specified parameters in the TIM_HandleTypeDef and initialize the associated handle.

Parameters

- **htim**: TIM Base handle

Return values

- **HAL**: status

Notes

HAL_TIM_Base_Start_IT

Function name

HAL_StatusTypeDef HAL_TIM_Base_Start_IT (TIM_HandleTypeDef * htim)

Function description

Starts the TIM Base generation in interrupt mode.

Parameters

- **htim**: TIM Base handle

Return values

- **HAL**: status

```
91  /* Initialize all configured peripherals */
92  MX_GPIO_Init();
93  MX_USART2_UART_Init();
94  MX_TIM6_Init();
95  /* USER CODE BEGIN 2 */
96  HAL_TIM_Base_Init(&htim6);
97  HAL_TIM_Base_Start_IT(&htim6);
98
99  /* USER CODE END 2 */
00
01  /* Infinite loop */
02  /* USER CODE BEGIN WHILE */
03  while (1)
04  {
05      /* USER CODE END WHILE */
```

Add these functions under int main():


```
112 void TIM6_DAC_IRQHandler(void) {  
113     HAL_TIM_IRQHandler(&htim6);  
114 }  
115  
116 void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim) {  
117     if(htim->Instance == TIM6)  
118         HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin);  
119 }  
120
```

NOTE: The function TIM6_DAC_IRQHandler 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 every 0.5 seconds.