Project_5.0_Timer_TimerMode

In this project, we will use a timer to generate an interrupt signal in a second and toggle the LED. We will use a general purpose timer that will work in timer mode.

We will use the " **T_{OUT} =(PSC x Preload) / F_{CLK}** " equation to set the overflow time to 1 second. An ISR will be created for the overflow event so that we can toggle the LED..

Since the MCU I am using supports a maximum of 48 MHz, I will set FCLK to 48 MHz. To achieve a Tout of 1 second, I will set the Prescaler (PSC) value to 1000 and the Counter Period to 48000.

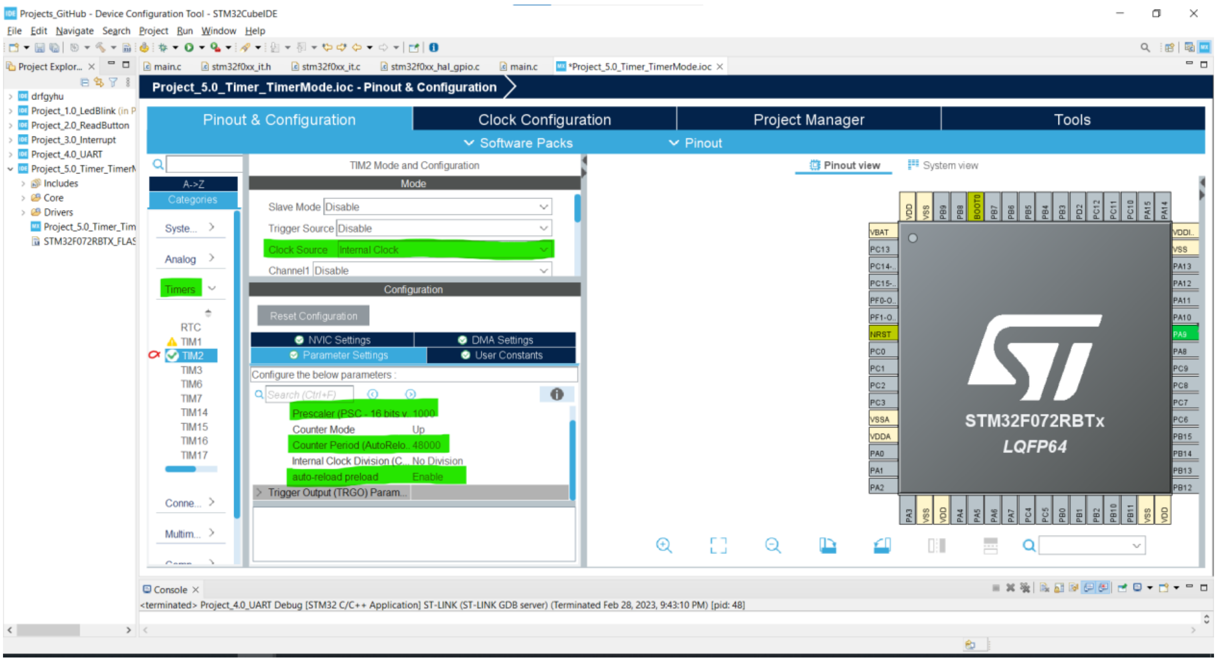
STEP 1 : Open STM32CubeIDE and Create New Project.

STEP 2 : Select Target microcontroller and double Click. My MCU is STM32F072RBT6.

STEP3 : Enter the project name and finish .

STEP 4 : Click the pin you want to set output and select GPIO_output. For example, in this project, I choose the A9 pin for output.

Step 5 : Configure Timer2 , enable Timer Interrupt Signaling in NVIC Tab and set the System Clock to 48 MHz



TIM2 Mode and Configuration

Mode

Slave Mode: Disable

Trigger Source: Disable

Clock Source: Internal Clock

Channel1: Disable

Configuration

Reset Configuration

NVIC Settings | DMA Settings | Parameter Settings | User Constants

Configure the below parameters :

Prescaler (PSC - 16 bits v.. 1000

Counter Mode Up

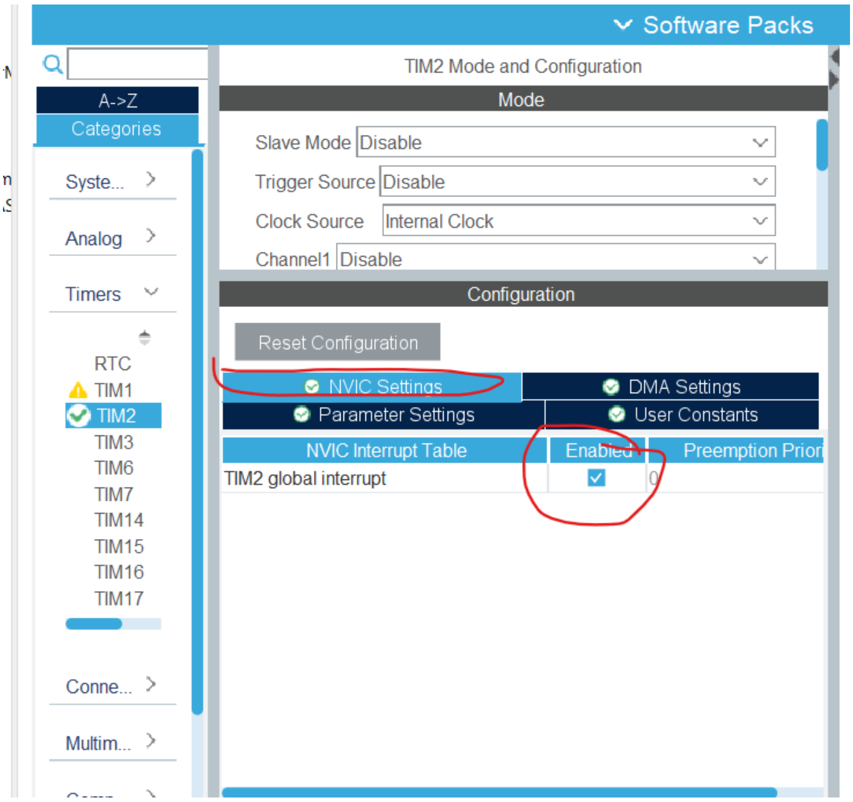
Counter Period (AutoRelo.. 48000

Internal Clock Division (C... No Division

auto-reload preload Enable

Trigger Output (TRGO) Param...

```
<terminated> Project_4.0_UART Debug [STM32 C/C++ Application] ST-LINK (ST-LINK GDB server) (Terminated Feb 28, 2023, 9:43:10 PM) [pid: 48]
```



Software Packs

TIM2 Mode and Configuration

Mode

Slave Mode: Disable

Trigger Source: Disable

Clock Source: Internal Clock

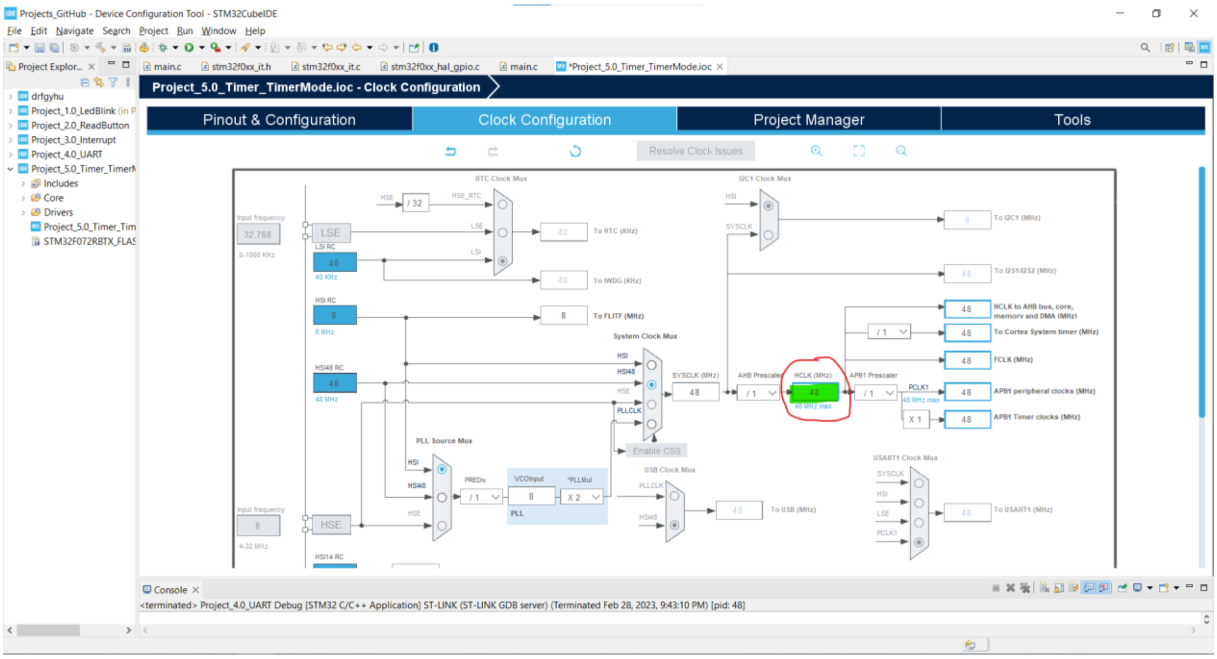
Channel1: Disable

Configuration

Reset Configuration

NVIC Settings | DMA Settings | Parameter Settings | User Constants

NVIC Interrupt Table	Enabled	Preemption Prior
TIM2 global interrupt	☑	0



Projects_GitHub - Device Configuration Tool - STM32CubeIDE

Project_5.0_Timer_TimerMode.ioc - Clock Configuration

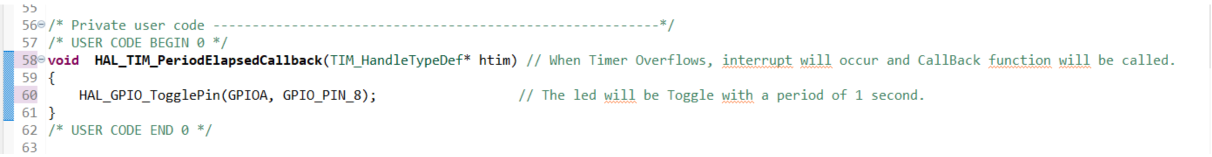
Pinout & Configuration | Clock Configuration | Project Manager | Tools

<terminated> Project_4.0_UART Debug [STM32 C/C++ Application] ST-LINK (ST-LINK GDB server) (Terminated Feb 28, 2023, 9:43:10 PM) [pid: 48]

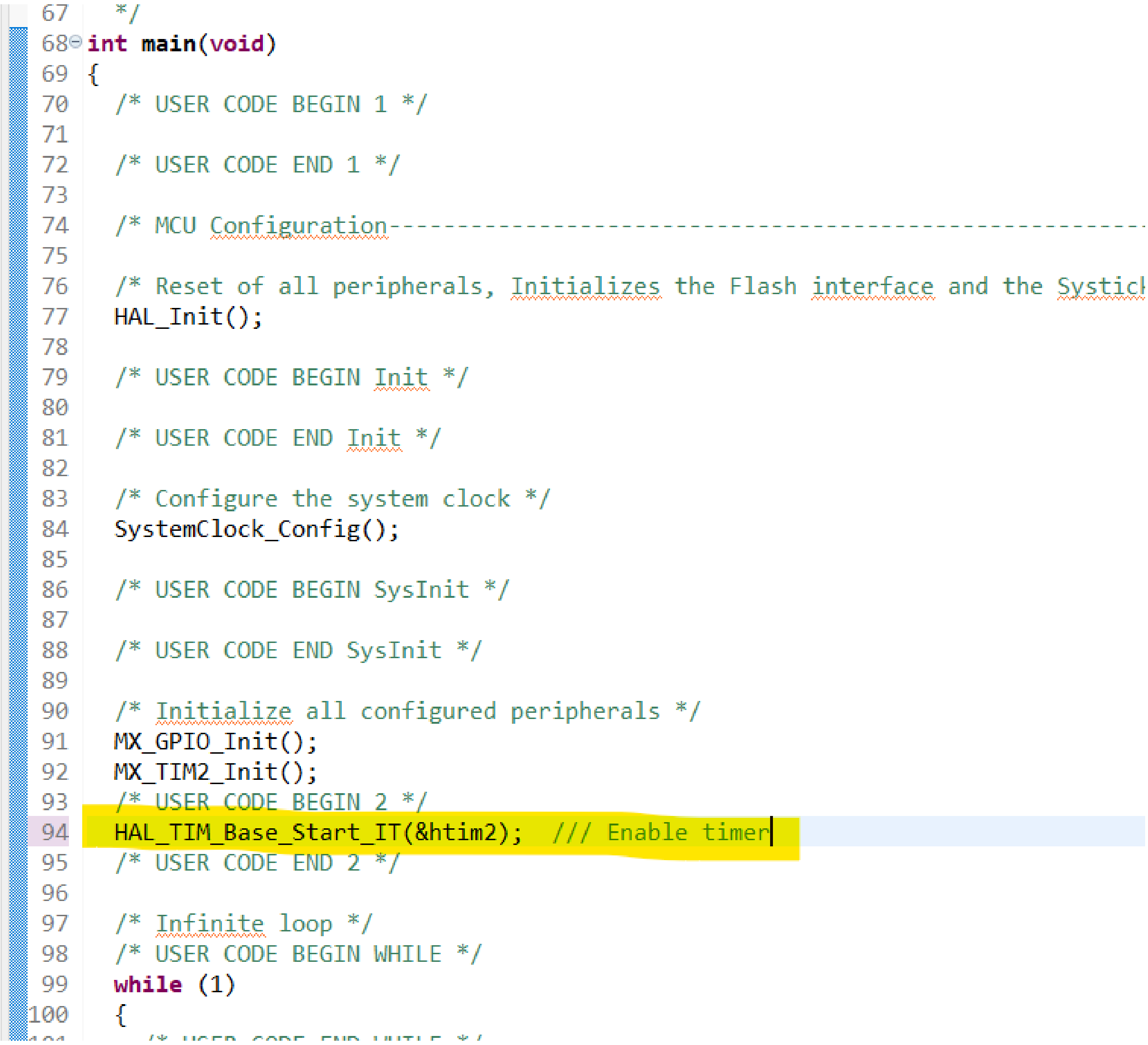
STEP 6 : Set the RCC External Clock Source and then CTRL + S to generate the project code. And we open our main.c file in the project files.

STEP 7 : Find CallBack Function.

STEP 8 : Code :



```c
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
void  HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef* htim) // When Timer Overflows, interrupt will occur and CallBack function will be called.
{
    HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_8);                  // The led will be Toggle with a period of 1 second.
}
/* USER CODE END 0 */
```



```c
  */
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------

  /* Reset of all peripherals, Initializes the Flash interface and the Systick
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_TIM2_Init();
  /* USER CODE BEGIN 2 */
  HAL_TIM_Base_Start_IT(&htim2);  /// Enable timer
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */
```

STEP 7 : We press RUN to compile the code and upload it to the board.

STEP 8 : Now it's time to connect the led to the board.

As soon as we load the code, the timer will start running and an overflow will occur every second. When an overflow occurs, an interrupt will occur and this interrupt will call the CallBack function. The CallBack function will toggle the GPIO pin we specified.