Project_5.1_Timer_CounterMode

In this project we will run a timer in counter mode.

First we will set TIM2 in counter mode. Then we will set the ARR value to 15 so that the counter overflows after 15 clicks and generates an interrupt.

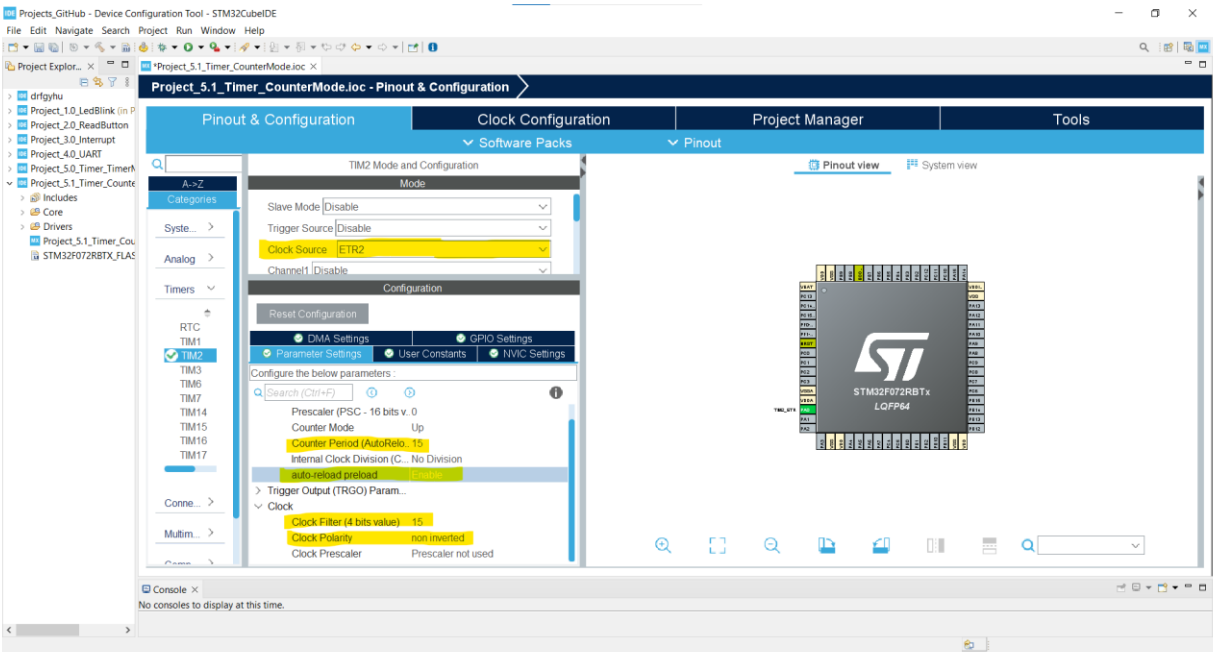
In the while loop, we will instantly read the counter value and print the value we read with uart. And we will write the interrupt function and print a message with uart when the interrupt occurs.

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 5 : Configure Timer2 , enable Timer Interrupt Signaling in NVIC Tab. And we will use a button as the counter input, we make the clock filter : 15 and clock polarity : non inverted in the parameter settings to reduce the noise of the button.



Project_5.1_Timer_CounterMode.ioc - Pinout & Configuration

TIM2 Mode and Configuration

Mode

Slave Mode: Disable
Trigger Source: Disable
Clock Source: ETR2
Channel1: Disable

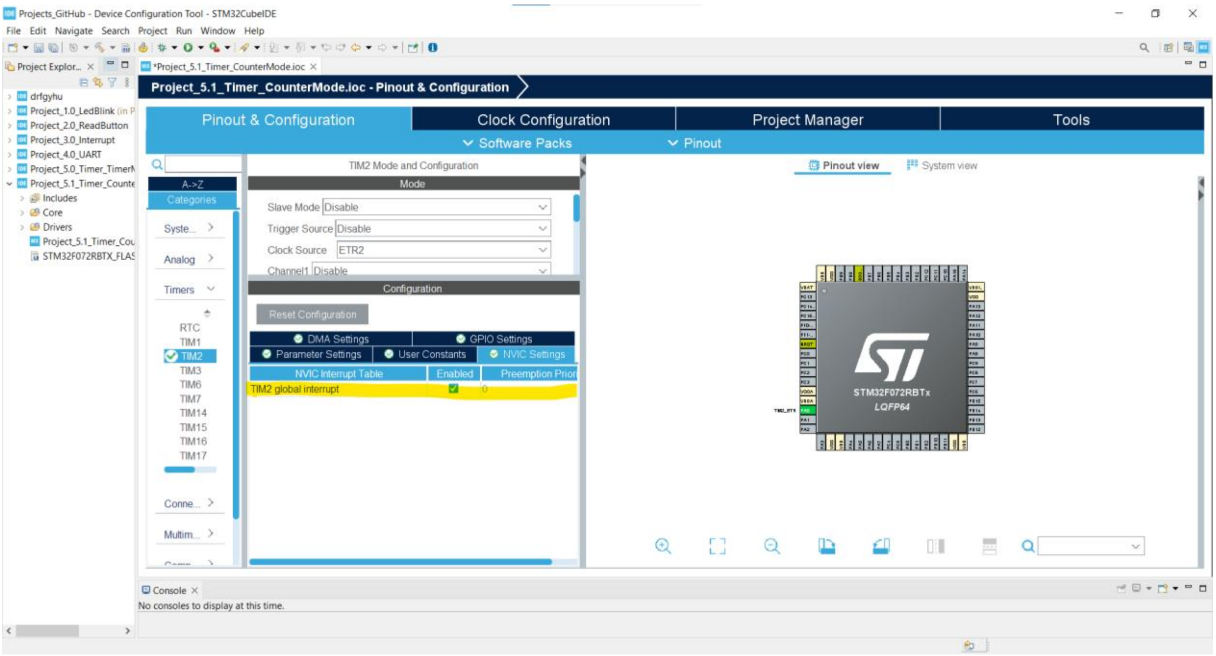
Configuration

Parameter Settings — Configure the below parameters :

Prescaler (PSC - 16 bits v.. 0
Counter Mode: Up
Counter Period (AutoRelo.. 15
Internal Clock Division (C... No Division
auto-reload preload: Enable

Clock
Clock Filter (4 bits value): 15
Clock Polarity: non inverted
Clock Prescaler: Prescaler not used

No consoles to display at this time.



Slave Mode Disable

Trigger Source Disable

Clock Source ETR2

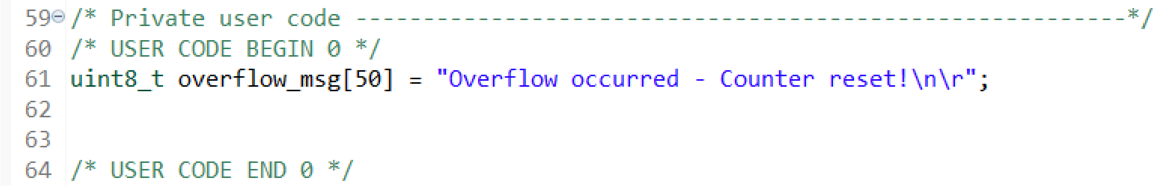
Channel1 Disable

NVIC Interrupt Table	Enabled	Preemption Prior
TIM2 global interrupt	☑	0

STEP 6: We configure the uart module.

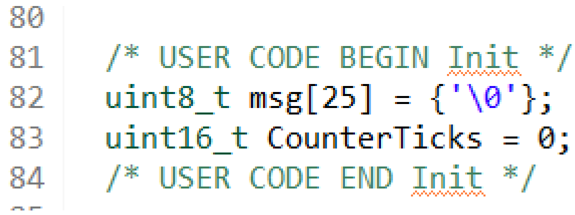
STEP 7 : Set the RCC External Clock Source and then CTRL + S to generate the project code. And we open our main.c file in the project files.

STEP 8 : Code :



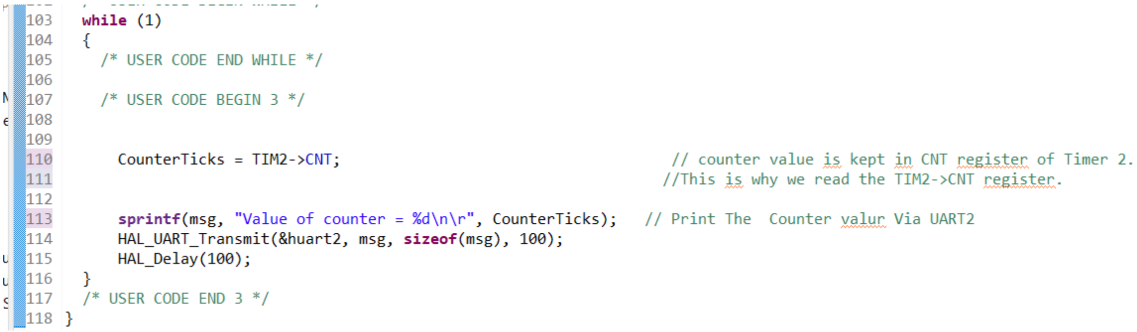
```
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
uint8_t overflow_msg[50] = "Overflow occurred - Counter reset!\n\r";


/* USER CODE END 0 */
```



```

  /* USER CODE BEGIN Init */
  uint8_t msg[25] = {'\0'};
  uint16_t CounterTicks = 0;
  /* USER CODE END Init */
```

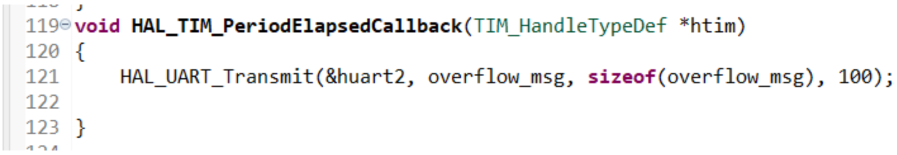


```
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */


      CounterTicks = TIM2->CNT;                              // counter value is kept in CNT register of Timer 2.
                                                            //This is why we read the TIM2->CNT register.

      sprintf(msg, "Value of counter = %d\n\r", CounterTicks);   // Print The  Counter valur Via UART2
      HAL_UART_Transmit(&huart2, msg, sizeof(msg), 100);
      HAL_Delay(100);
  }
  /* USER CODE END 3 */
}
```

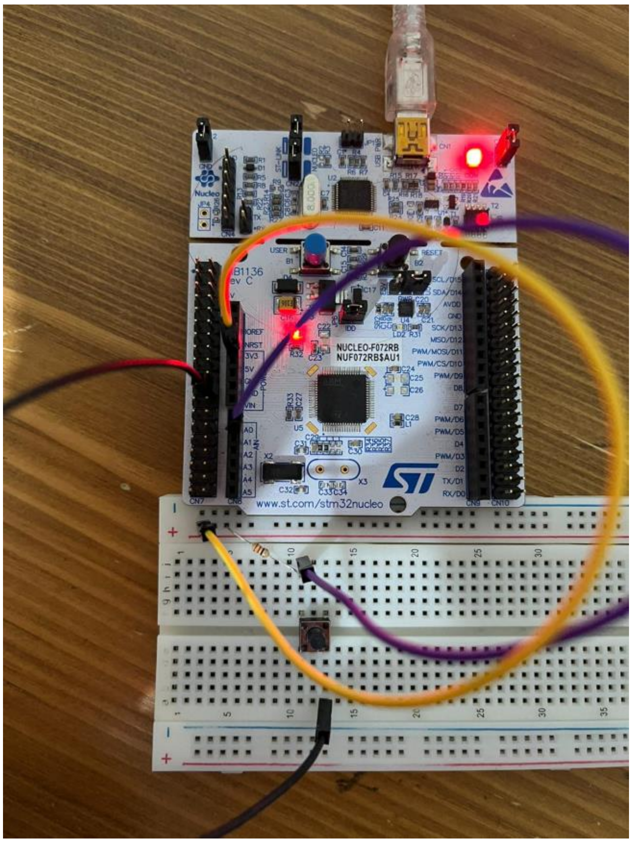


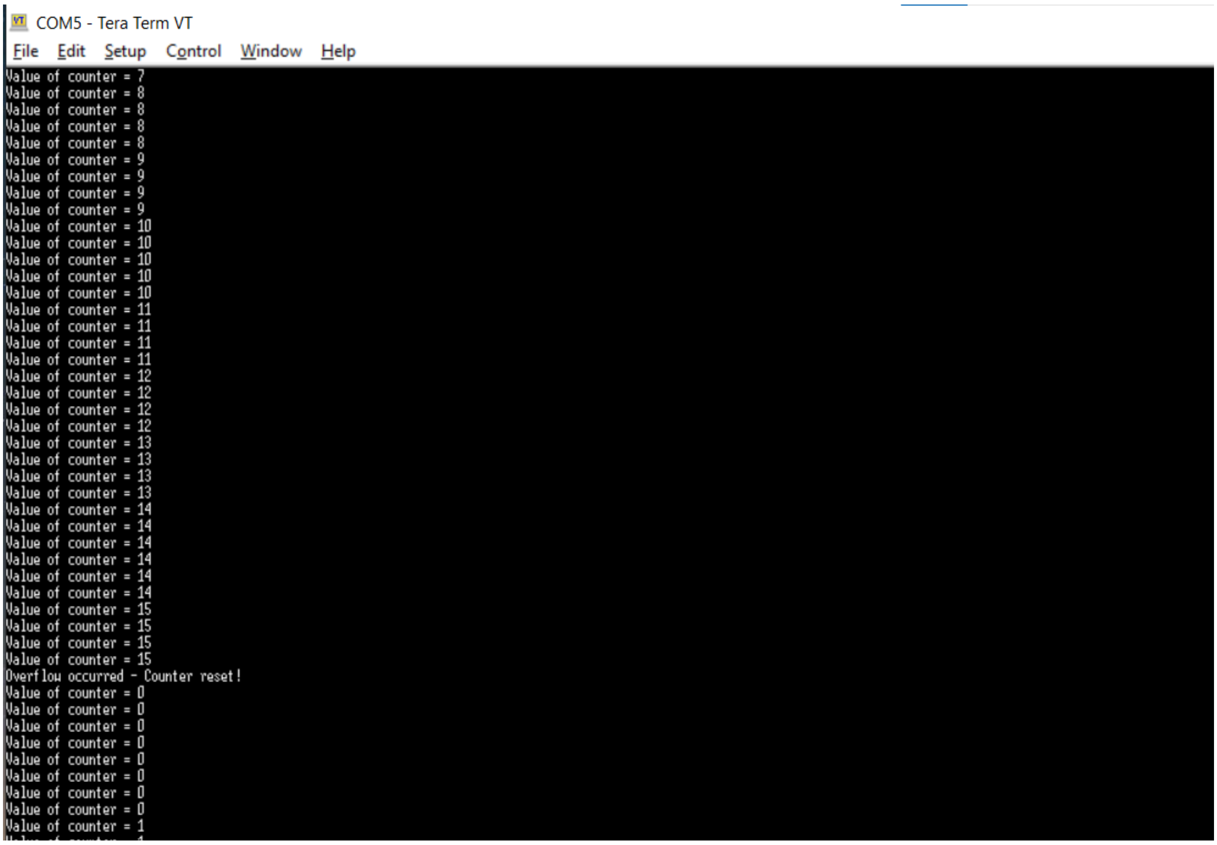
```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
    HAL_UART_Transmit(&huart2, overflow_msg, sizeof(overflow_msg), 100);

}
```

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

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





```
Value of counter = 7
Value of counter = 8
Value of counter = 8
Value of counter = 8
Value of counter = 8
Value of counter = 9
Value of counter = 9
Value of counter = 9
Value of counter = 9
Value of counter = 10
Value of counter = 10
Value of counter = 10
Value of counter = 10
Value of counter = 10
Value of counter = 11
Value of counter = 11
Value of counter = 11
Value of counter = 11
Value of counter = 12
Value of counter = 12
Value of counter = 12
Value of counter = 12
Value of counter = 13
Value of counter = 13
Value of counter = 13
Value of counter = 13
Value of counter = 14
Value of counter = 14
Value of counter = 14
Value of counter = 14
Value of counter = 14
Value of counter = 14
Value of counter = 15
Value of counter = 15
Value of counter = 15
Value of counter = 15
Overflow occurred - Counter reset!
Value of counter = 0
Value of counter = 0
Value of counter = 0
Value of counter = 0
Value of counter = 0
Value of counter = 0
Value of counter = 0
Value of counter = 0
Value of counter = 1
```