

Mikroprocesory i mikrokontrolery Laboratorium nr 8	Temat: Timery i przerwania
Grupa: 21b	Michał Lechowicz

Cel ćwiczenia:

Celem ćwiczenia było zapoznanie się z działaniem i obsługą timerów w mikrokontrolerze **STM32F0**. Timery służą do odmierzania czasu oraz mogą być użyteczne w regulacji PWM. Do konfiguracji kontrolera posłużył program CubeMX oraz IDE IAR.

Zadanie na ocenę 4.0:

1. Opis zadania:

W zadaniu należało napisać program, w którym wykorzystamy Timer trzeci do wygenerowania przerwania o okresie 3000 cykli zegarowych. W przerwaniu ma on inkrementować zmienną **czas**.

2. Opis programu:

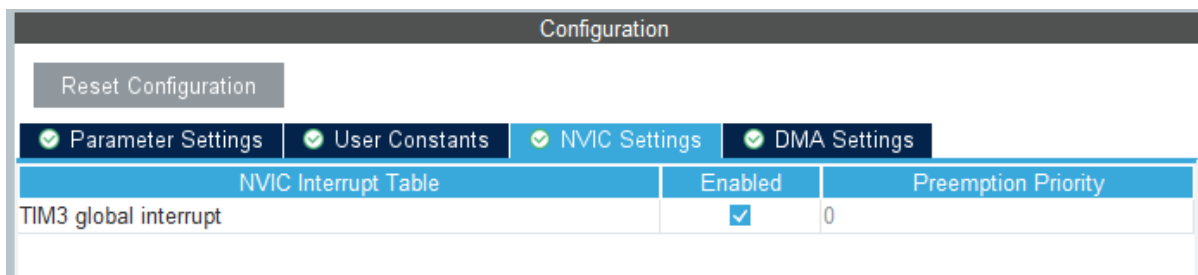
1. Konfiguracja programu STM32CubeMX;

The screenshot displays the STM32CubeMX configuration interface. On the left, the 'Pinout & Configuration' tab is active, showing a tree view of system components. 'Timers' is expanded, and 'TIM3' is selected. The right pane shows the 'TIM3 Mode and Configuration' settings. Under the 'Mode' section, 'Slave Mode' is set to 'Disable', 'Trigger Source' to 'Disable', 'Clock Source' to 'Internal Clock', and all channels (Channel1, Channel2, Channel3, Channel4, and Combined Channels) are set to 'Disable'. Below this, 'Use ETR as Clearing Source', 'XOR activation', and 'One Pulse Mode' are all unchecked. The 'Configuration' section at the bottom has tabs for 'Parameter Settings', 'User Constants', 'NVIC Settings', and 'DMA Settings'. The 'Parameter Settings' tab is active, showing a list of parameters to configure. The 'Counter Settings' section includes 'Prescaler (PSC - 16 bits value)' set to 64, 'Counter Mode' set to 'Up', and 'Counter Period (AutoReload Register - 16 bits val...)' set to 2999. Other parameters like 'Internal Clock Division (CKD)', 'auto-reload preload', 'Trigger Output (TRGO) Parameters', 'Clear Input', and 'Output Compare No Output Channel 1' are also visible with their respective settings.

Section	Parameter	Value
Mode	Slave Mode	Disable
	Trigger Source	Disable
	Clock Source	Internal Clock
	Channel1	Output Compare No Output
	Channel2	Disable
	Channel3	Disable
	Channel4	Disable
	Combined Channels	Disable
	Use ETR as Clearing Source	<input type="checkbox"/>
	XOR activation	<input type="checkbox"/>
One Pulse Mode	<input type="checkbox"/>	
Configuration - Parameter Settings	Prescaler (PSC - 16 bits value)	64
	Counter Mode	Up
	Counter Period (AutoReload Register - 16 bits val...)	2999
	Internal Clock Division (CKD)	No Division
	auto-reload preload	Disable
	Trigger Output (TRGO) Parameters	
	Master/Slave Mode (MSM bit)	Disable (Trigger input effect not delayed)
	Trigger Event Selection	Reset (UG bit from TIMx_EGR)
	Clear Input	
	Clear Input Source	Disable
	Output Compare No Output Channel 1	
	Mode	Frozen (used for Timing base)
	Pulse (16 bits value)	0
	Output compare preload	Disable
	CH Polarity	High

Rys. 1 Konfiguracja timera

Ze względu na to, iż ustawienia timera liczone są od 0, a nie od 1, by uzyskać przerwanie w czasie 3000 cykli zegara, należy ustawić 2999 zamiast 3000.



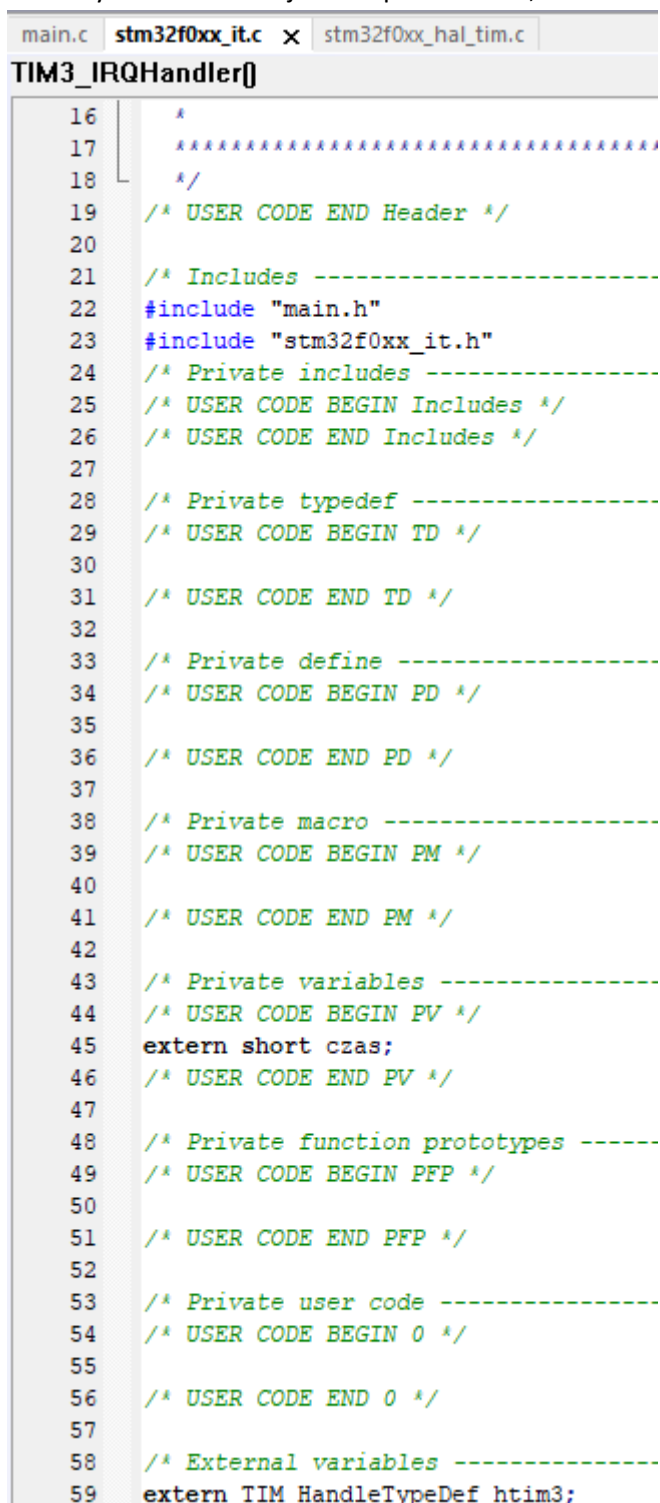
Rys. 2 Aktywacja przerwania

2. Konfiguracja wykonana przez Cube oraz ustawienie zmiennej **czas**;

```
main.c x stm32f0xx_it.c stm32f0xx_hal_tim.c
42  /* Private variables -----
43  TIM_HandleTypeDef htim3;
44
45  /* USER CODE BEGIN PV */
46  short czas = 0; // usatwienie zmiennej czas
47  /* USER CODE END PV */
48
49  /* Private function prototypes -----
50  void SystemClock_Config(void);
51  static void MX_TIM3_Init(void);
52  /* USER CODE BEGIN PFP */
53
54  /* USER CODE END PFP */
55
56  /* Private user code -----
57  /* USER CODE BEGIN 0 */
58
59  /* USER CODE END 0 */
60
61  /**
62   * @brief The application entry point.
63   * @retval int
64   */
65  int main(void)
```

Rys. 3. Ustawienie zmiennej **czas** w pliku main.c

3. Przekazywanie zmiennej **czas** z pliku **main.c**;



```
main.c  stm32f0xx_it.c  x  stm32f0xx_hal_tim.c
TIM3_IRQHandler()
16      *
17      ****
18      */
19      /* USER CODE END Header */
20
21      /* Includes -----
22      #include "main.h"
23      #include "stm32f0xx_it.h"
24      /* Private includes -----
25      /* USER CODE BEGIN Includes */
26      /* USER CODE END Includes */
27
28      /* Private typedef -----
29      /* USER CODE BEGIN TD */
30
31      /* USER CODE END TD */
32
33      /* Private define -----
34      /* USER CODE BEGIN PD */
35
36      /* USER CODE END PD */
37
38      /* Private macro -----
39      /* USER CODE BEGIN PM */
40
41      /* USER CODE END PM */
42
43      /* Private variables -----
44      /* USER CODE BEGIN PV */
45      extern short czas;
46      /* USER CODE END PV */
47
48      /* Private function prototypes -----
49      /* USER CODE BEGIN PFP */
50
51      /* USER CODE END PFP */
52
53      /* Private user code -----
54      /* USER CODE BEGIN 0 */
55
56      /* USER CODE END 0 */
57
58      /* External variables -----
59      extern TIM_HandleTypeDef htim3;
```

Rys. 3. Ustawienie zmiennej **czas** w pliku **stm32f0xx_it.c** jako extern

4. Inkrementacja zmiennej **czas** w obsłudze przerwania (przepełnienia timera);

```
147 void TIM3_IRQHandler(void)
148 {
149     /* USER CODE BEGIN TIM3_IRQn 0 */
150
151     /* USER CODE END TIM3_IRQn 0 */
152     HAL_TIM_IRQHandler(&htim3);
153     /* USER CODE BEGIN TIM3_IRQn 1 */
154     czas++;
155     /* USER CODE END TIM3_IRQn 1 */
156 }
157
```

Rys. 4. Inkrementacja zmiennej **czas** w handlerze timera 3.

5. Podgląd rejestru TIM3;

Registers 2	
Find: <input type="text"/>	Group: TIM3
Name	Value
+ CR1	0x0000'0000
+ CR2	0x0000'0000
+ SMCR	0x0000'0000
+ DIER	0x0000'0001
+ SR	0x0000'0000
+ EGR	WWWWWWW
+ CCMR1_Output	0x0000'0000
+ CCMR1_Input	0x0000'0000
+ CCMR2_Output	0x0000'0000
+ CCMR2_Input	0x0000'0000
+ CCER	0x0000'0000
- CNT	0x0000'0000
CNT_H	0x0000
CNT_L	0x0000
+ PSC	0x0000'0040
+ ARR	0x0000'0bb7
+ CCR1	0x0000'0000
+ CCR2	0x0000'0000
+ CCR3	0x0000'0000
+ CCR4	0x0000'0000
+ DCR	0x0000'0000
+ DMAR	0x0000'0000

Rys. 5. Rejestr przed uruchomieniem

Registers 2	
Find: <input type="text"/>	Group: TIM3
Name	Value
+ CR1	0x0000'0001
+ CR2	0x0000'0000
+ SMCR	0x0000'0000
+ DIER	0x0000'0001
+ SR	0x0000'001f
+ EGR	WWWWWWW
+ CCMR1_Output	0x0000'0000
+ CCMR1_Input	0x0000'0000
+ CCMR2_Output	0x0000'0000
+ CCMR2_Input	0x0000'0000
+ CCER	0x0000'0000
- CNT	0x0000'045e
CNT_H	0x0000
CNT_L	0x045e
+ PSC	0x0000'0040
+ ARR	0x0000'0bb7
+ CCR1	0x0000'0000
+ CCR2	0x0000'0000
+ CCR3	0x0000'0000
+ CCR4	0x0000'0000
+ DCR	0x0000'0000
+ DMAR	0x0000'0001

Rys. 6. Rejestr po uruchomieniu

6. Podgląd zmiennej czas, po przepelnieniu licznika.

```

145  * @brief This function handles TIM3 global interrupt.
146  */
147  void TIM3_IRQHandler(void)
148  {
149      /* USER CODE BEGIN TIM3_IRQn 0 */
150
151      /* USER CODE END TIM3_IRQn 0 */
152      HAL_TIM_IRQHandler(&htim3);
153      /* USER CODE BEGIN TIM3_IRQn 1 */
154      czas++;
155      /* USER CODE END TIM3_IRQn 1 */
156  }
157
158  /* USER CODE BEGIN 1 */
159
160  /* USER CODE END 1 */
161  /***** (C) COPYRIGHT STMicroelectronics *****/
162

```

Expression	Value
czas	2
<click to add>	

Rys. 7. Podgląd na zmienną czas.

4. Podsumowanie i wnioski:

Obsługa timerów w ARM jest bardziej złożona niż w INTEL 8051. Dają one jednak znacznie większe możliwości konfiguracyjne oraz wachlarz użycia.