CSE433

HW 1

STM32 Timer Module Routines

Lecturer: Alp Arslan Bayrakçı

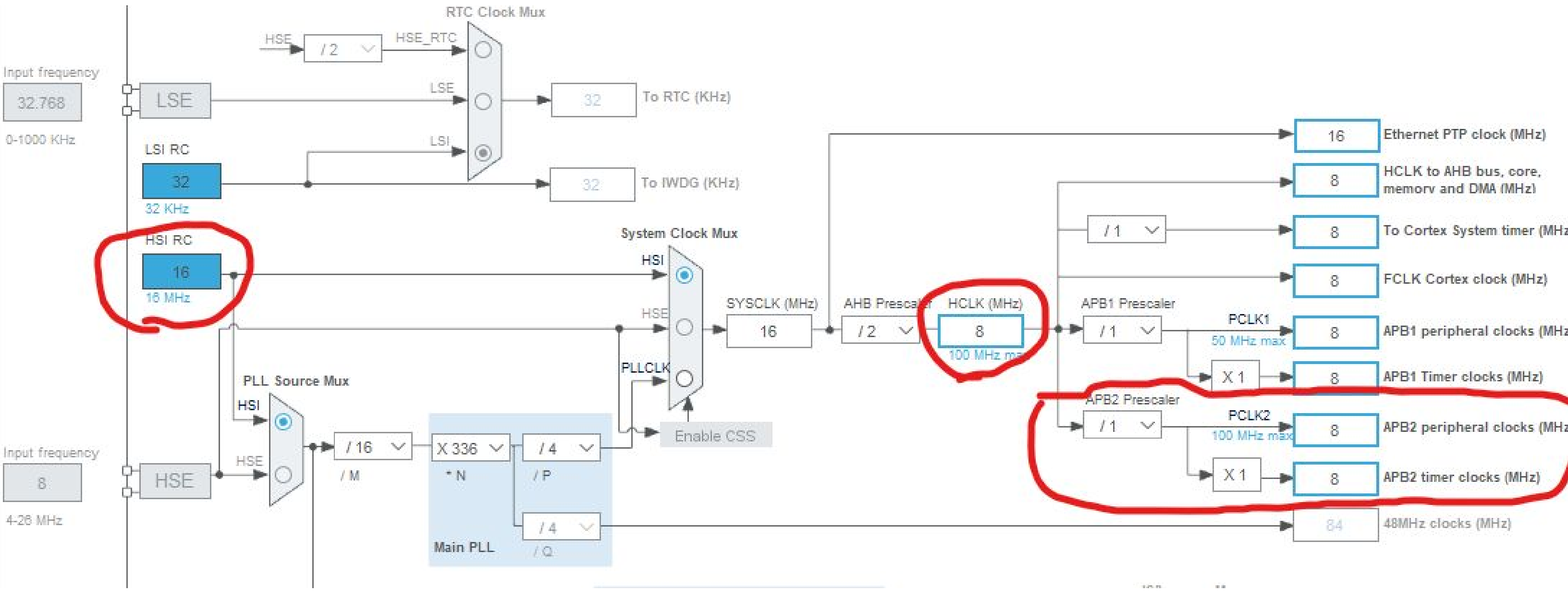
Assistant: Fatma Nur Esirci

Muhammed Okumuş
151044017

Part 1

Question: Using Down Count property and prescalar settings, change the overflow period to 500ms.

Step 1 : Clock configuration



Step 2 : Calculate prescaler value

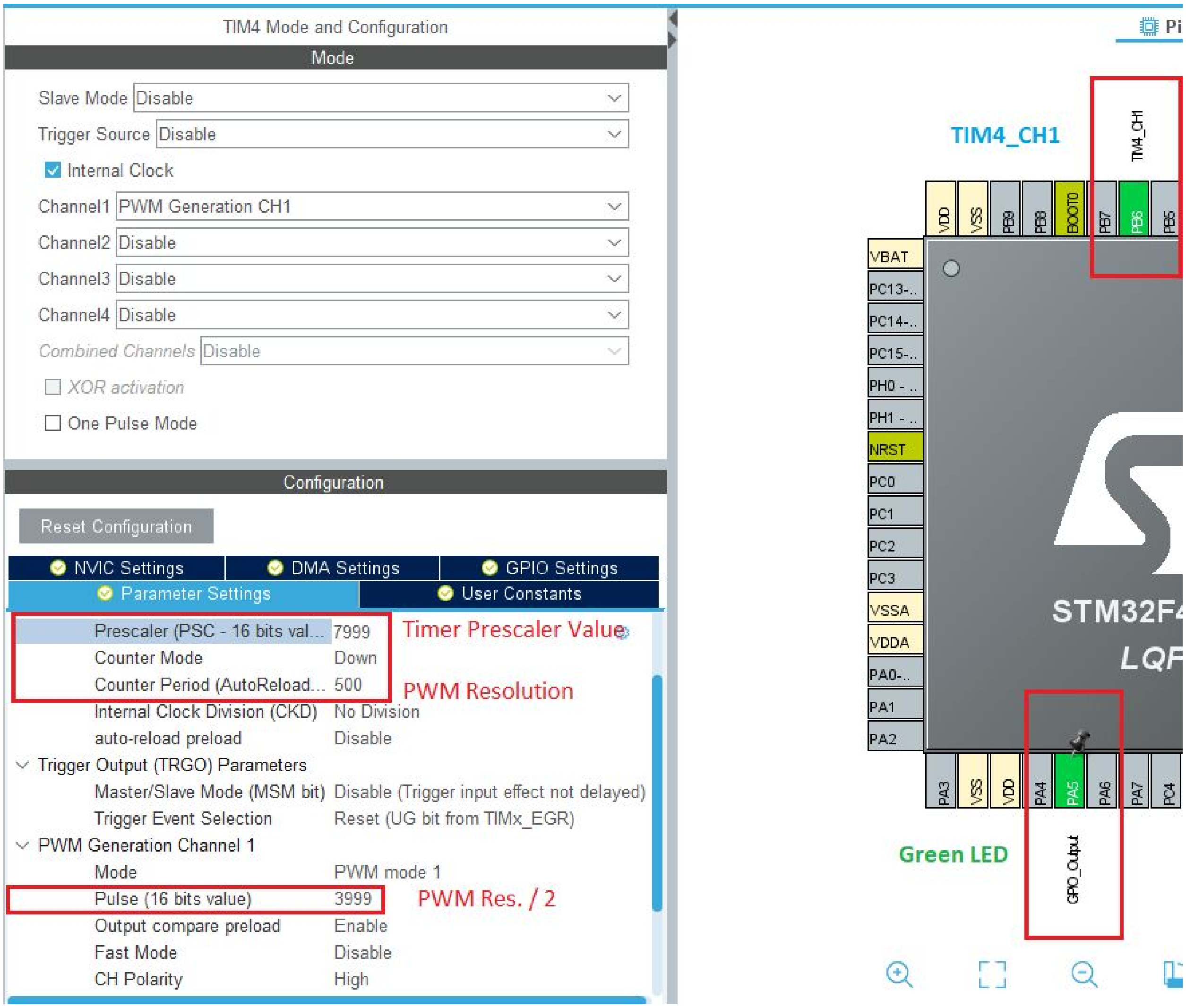
Timer Prescaler Value= (Timer Input Clock / (Frequency Required * PWM Resolution(Steps))) -1

Timer Input Clock = 8 Mhz = 8.000.000 Hz
Frequency Required = 500ms = 2 Hz
PWM Resolution = 500
Timer Prescaler Value = 7999

Note: PWM resolution is an arbitrary number between 0-65535 since this value is stored in 16-bit register and maximum 16-bit count is 65535.

Muhammed Okumuş
151044017

Step 3 : Further clock configuration using the values from step 2



TIM4 Mode and Configuration

Mode

Slave Mode: Disable

Trigger Source: Disable

☑ Internal Clock

Channel1: PWM Generation CH1

Channel2: Disable

Channel3: Disable

Channel4: Disable

Combined Channels: Disable

□ XOR activation

□ One Pulse Mode

Configuration

Prescaler (PSC - 16 bits val...	7999
Counter Mode	Down
Counter Period (AutoReload...	500
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)
PWM Generation Channel 1	
Mode	PWM mode 1
Pulse (16 bits value)	3999
Output compare preload	Enable
Fast Mode	Disable
CH Polarity	High

Timer Prescaler Value

PWM Resolution

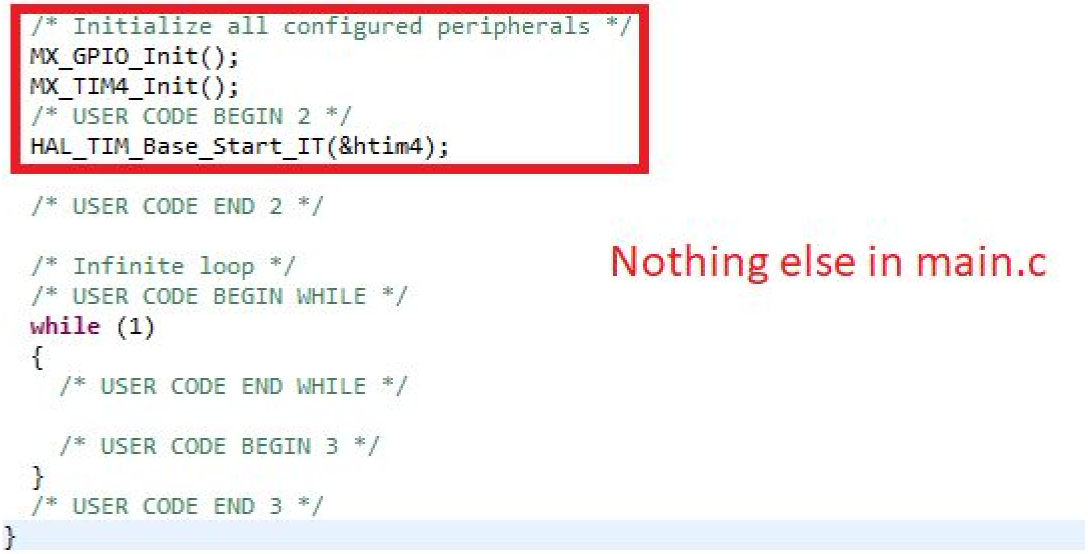
PWM Res. / 2

TIM4_CH1

Green LED

Muhammed Okumuş
151044017

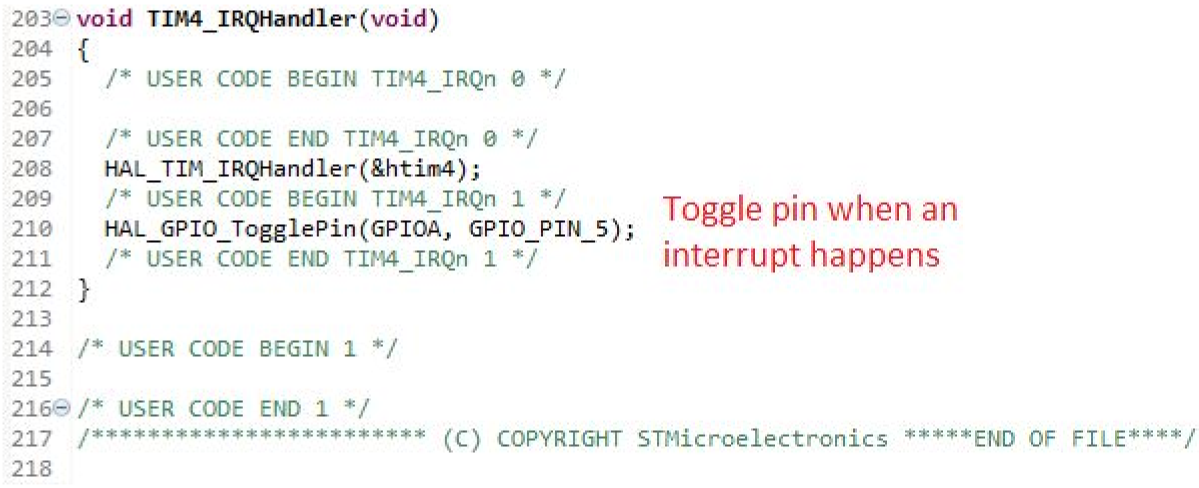
Step 4 : Initializing the timer in “main.c”



```
  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_TIM4_Init();
  /* USER CODE BEGIN 2 */
  HAL_TIM_Base_Start_IT(&htim4);

  /* USER CODE END 2 */

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

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```

Nothing else in main.c

Step 5 : Coding the timer handler in “stm32f4xx_it.c”



```
void TIM4_IRQHandler(void)
{
  /* USER CODE BEGIN TIM4_IRQn 0 */

  /* USER CODE END TIM4_IRQn 0 */
  HAL_TIM_IRQHandler(&htim4);
  /* USER CODE BEGIN TIM4_IRQn 1 */
  HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
  /* USER CODE END TIM4_IRQn 1 */
}

/* USER CODE BEGIN 1 */

/* USER CODE END 1 */
/************************ (C) COPYRIGHT STMicroelectronics *****END OF FILE****/

```

Toggle pin when an interrupt happens

Step 6 : Build & Run

10 second video, led gets toggled 20 times, 10 blinks.

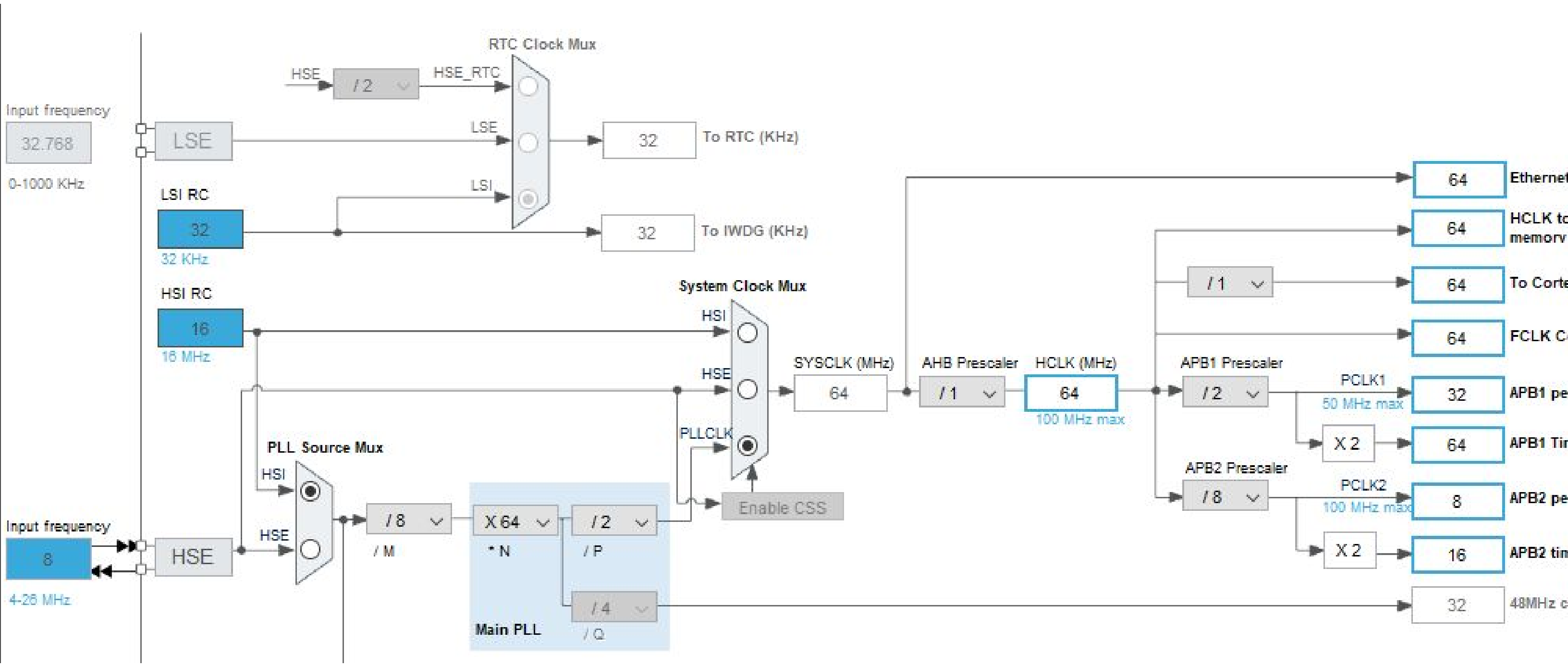
https://www.youtube.com/watch?v=Hrgz08-pKfU&feature=youtu.be

Part 2 & 3

Minimum frequency that can be measure at 16Mhz:
(TIMx CLK) / ARR
45Mhz / 65535 = **244 Hz**

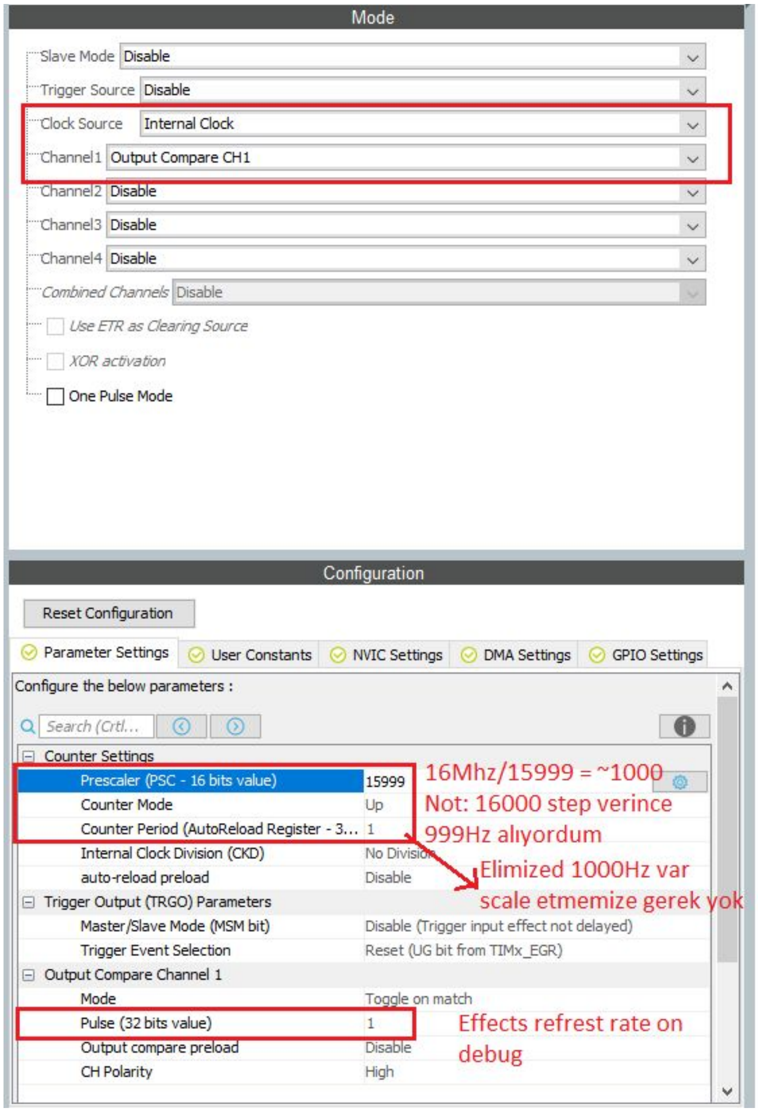
We need to measure 1ms which is **1000 Hz**

Step 1 : Clock configuration



Muhammed Okumuş
151044017

Step 2 : Pinout configuration (TIM2 : Output Compare)



Mode

Slave Mode	Disable
Trigger Source	Disable
Clock Source	Internal Clock
Channel1	Output Compare CH1
Channel2	Disable
Channel3	Disable
Channel4	Disable
Combined Channels	Disable

□ *Use ETR as Clearing Source*

□ *XOR activation*

□ One Pulse Mode

Configuration

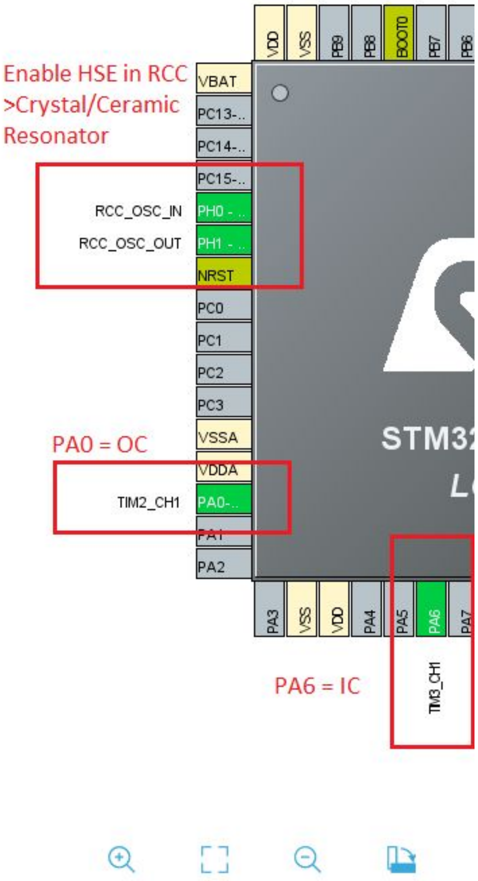
Configure the below parameters :

Counter Settings	
Prescaler (PSC - 16 bits value)	15999
Counter Mode	Up
Counter Period (AutoReload Register - 3...	1
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)
Output Compare Channel 1	
Mode	Toggle on match
Pulse (32 bits value)	1
Output compare preload	Disable
CH Polarity	High

16Mhz/15999 = ~1000
Not: 16000 step verince
999Hz alıyordum

Elimized 1000Hz var
scale etmemize gerek yok

Effects refrest rate on
debug



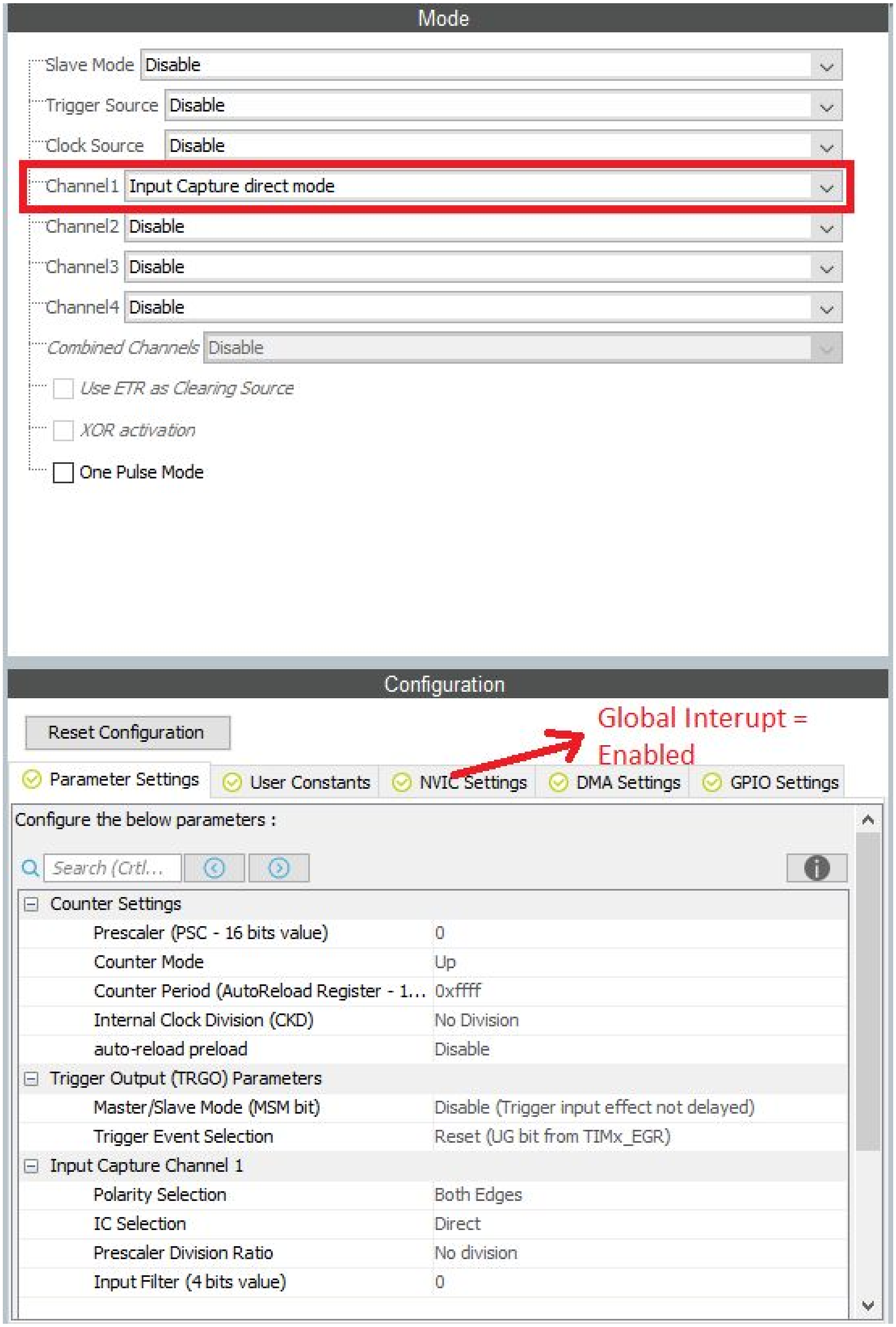
Enable HSE in RCC
>Crystal/Ceramic
Resonator

PA0 = OC

PA6 = IC

Muhammed Okumuş
151044017

Step 3 : Pinout configuration (TIM3 : Input Capture)



Mode

Slave Mode: Disable

Trigger Source: Disable

Clock Source: Disable

Channel1: Input Capture direct mode

Channel2: Disable

Channel3: Disable

Channel4: Disable

Combined Channels: Disable

□ *Use ETR as Clearing Source*

□ *XOR activation*

□ One Pulse Mode

Configuration

Reset Configuration

Global Interupt = Enabled

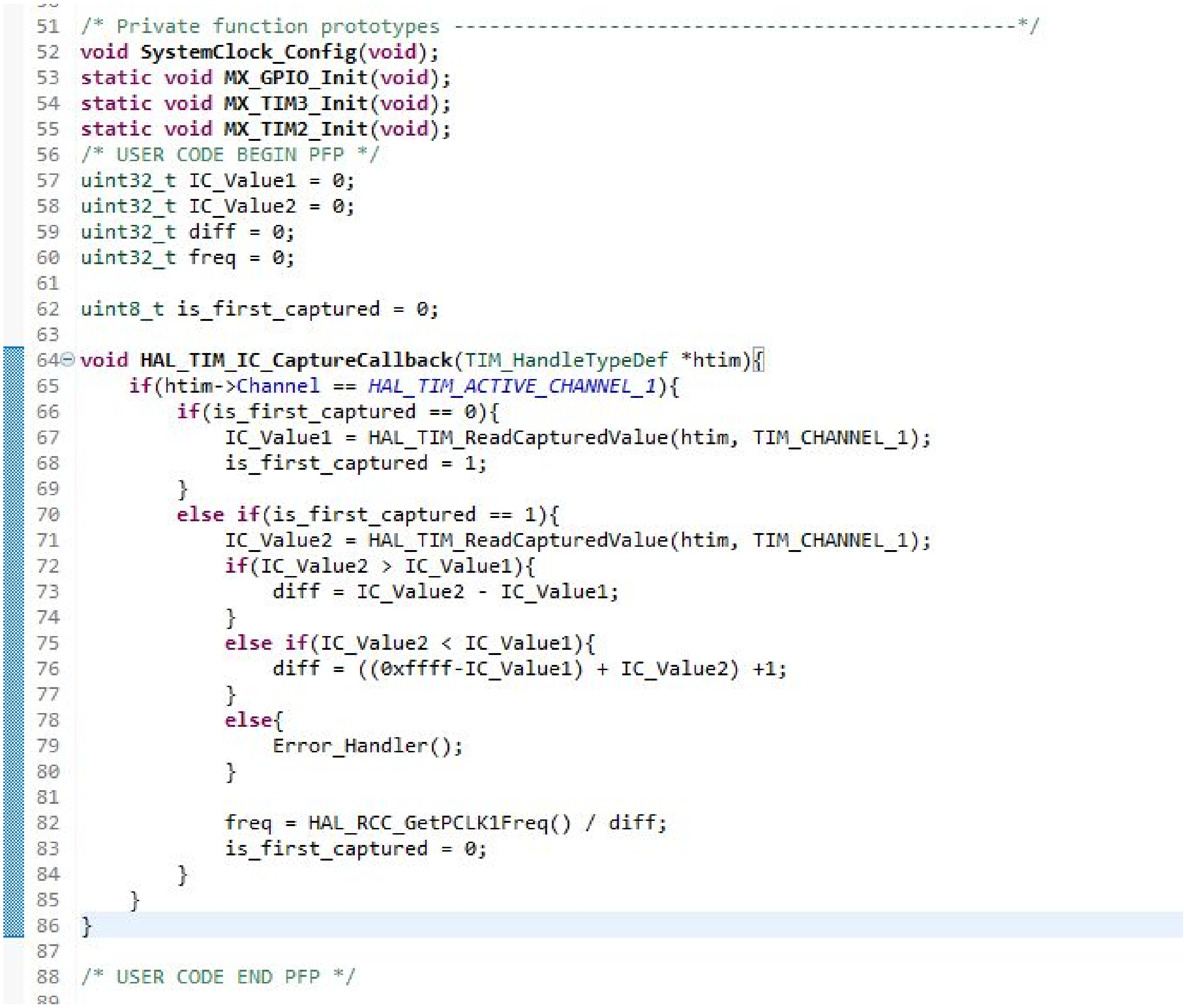
Parameter Settings | User Constants | NVIC Settings | DMA Settings | GPIO Settings

Configure the below parameters :

Counter Settings	
Prescaler (PSC - 16 bits value)	0
Counter Mode	Up
Counter Period (AutoReload Register - 1...	0xffff
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)
Input Capture Channel 1	
Polarity Selection	Both Edges
IC Selection	Direct
Prescaler Division Ratio	No division
Input Filter (4 bits value)	0

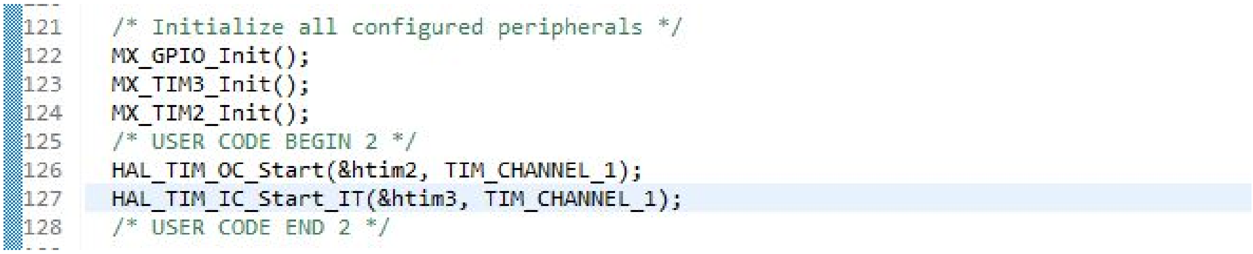
Muhammed Okumuş
151044017

Step 4 : Code in “main.c” , frequency is calculated here.[1]



```c
/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_TIM3_Init(void);
static void MX_TIM2_Init(void);
/* USER CODE BEGIN PFP */
uint32_t IC_Value1 = 0;
uint32_t IC_Value2 = 0;
uint32_t diff = 0;
uint32_t freq = 0;

uint8_t is_first_captured = 0;

void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim){
    if(htim->Channel == HAL_TIM_ACTIVE_CHANNEL_1){
        if(is_first_captured == 0){
            IC_Value1 = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1);
            is_first_captured = 1;
        }
        else if(is_first_captured == 1){
            IC_Value2 = HAL_TIM_ReadCapturedValue(htim, TIM_CHANNEL_1);
            if(IC_Value2 > IC_Value1){
                diff = IC_Value2 - IC_Value1;
            }
            else if(IC_Value2 < IC_Value1){
                diff = ((0xffff-IC_Value1) + IC_Value2) +1;
            }
            else{
                Error_Handler();
            }

            freq = HAL_RCC_GetPCLK1Freq() / diff;
            is_first_captured = 0;
        }
    }
}

/* USER CODE END PFP */
```

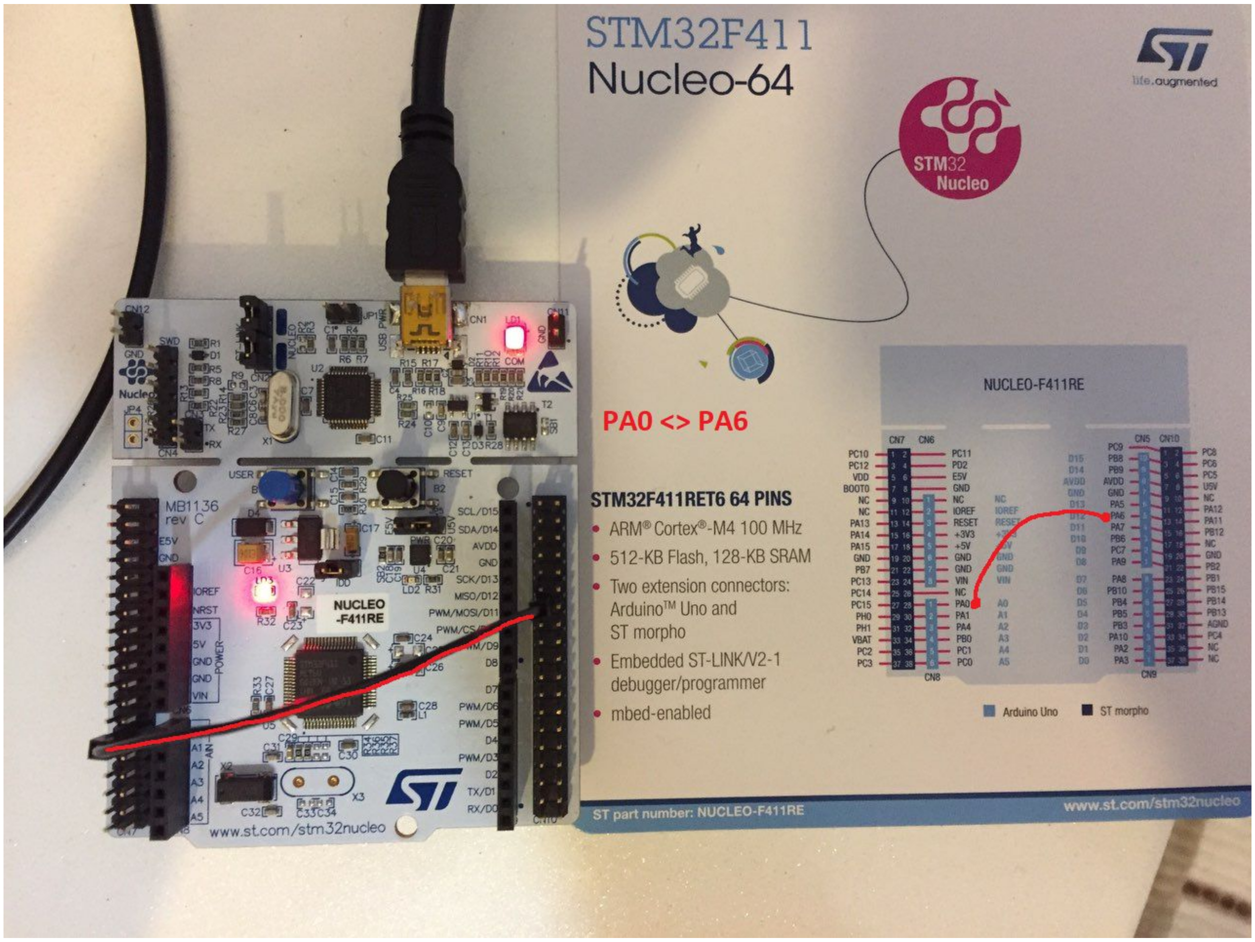
Initialize the timer modules



```c
  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_TIM3_Init();
  MX_TIM2_Init();
  /* USER CODE BEGIN 2 */
  HAL_TIM_OC_Start(&htim2, TIM_CHANNEL_1);
  HAL_TIM_IC_Start_IT(&htim3, TIM_CHANNEL_1);
  /* USER CODE END 2 */
```

Step 5 : Build & Run

1.40 minute video showing the frequency using STM Studio.
(Mikrofondan bilgisayarın gürültüsünü almış, sessiz alıp izleyin lütfen)
https://www.youtube.com/watch?v=1-klcD1dpyo&feature=youtu.be



Muhammed Okumuş
151044017

Resources

STM32F4 Discovery board - Keil 5 IDE with CubeMX: Tutorial 17 Timers - Output Compare
https://www.youtube.com/watch?v=NXGjhOa5oR8

Using input capture in STM32 || HAL || CubeMx || Frequency measure **(Code used from this video)**
https://www.youtube.com/watch?v=de4tfSzXrGM

Stm32f103 Pwm(Pulse width modulation) signal generation using internal timers, keil MDK-ARMv6 and Stmcubemx Ide
https://www.engineersgarage.com/stm32/stm32-pwm-generation-using-timers/

Thanks for reading!

Muhammed Okumuş
151044017