```
/* USER CODE BEGIN Header */
/**
  * GitHub: https://github.com/Naikthelegend/
3096_pracs_FRSKIA001_CLRCAM007/tree/master/Prac4

******************************************************************************
  * @file           : main.c
  * @brief          : Main program body

******************************************************************************
  * @attention
  *
  * Copyright (c) 2023 STMicroelectronics.
  * All rights reserved.
  *
  * This software is licensed under terms that can be found in the
LICENSE file
  * in the root directory of this software component.
  * If no LICENSE file comes with this software, it is provided AS-
IS.
  *

******************************************************************************
  */
/* USER CODE END Header */
/* Includes
------------------------------------------------------------------*/
#include "main.h"
#include "dma.h"
#include "tim.h"
#include "gpio.h"

/* Private includes
----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include <stdio.h>
#include "stm32f0xx.h"
#include <lcd_stm32f0.c>
/* USER CODE END Includes */

/* Private typedef
-----------------------------------------------------------*/
/* USER CODE BEGIN PTD */
#define NS 256       // Number of samples in LUT
#define TIM2CLK 8000000  // STM Clock frequency
#define F_SIGNAL 50  // Frequency of output analog signal
/* USER CODE END PTD */

/* Private define
------------------------------------------------------------*/
/* USER CODE BEGIN PD */
```

```
/* USER CODE END PD */

/* Private macro
-------------------------------------------------------------*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables
---------------------------------------------------------*/

/* USER CODE BEGIN PV */
// TODO: Add code for global variables, including LUTs

uint32_t Sin_LUT[NS] =
{512,524,537,549,562,574,587,599,611,624,636,648,660,672,684,696,707
,719,730,742,753,764,775,785,796,806,816,826,836,846,855,865,874,882
,891,899,907,915,923,930,937,944,951,957,963,969,974,980,985,989,994
,998,1001,1005,1008,1011,1014,1016,1018,1020,1021,1022,1023,1023,102
4,1023,1023,1022,1021,1020,1018,1016,1014,1011,1008,1005,1001,998,99
4,989,985,980,974,969,963,957,951,944,937,930,923,915,907,899,891,88
2,874,865,855,846,836,826,816,806,796,785,775,764,753,742,730,719,70
7,696,684,672,660,648,636,624,611,599,587,574,562,549,537,524,512,49
9,486,474,461,449,436,424,412,399,387,375,363,351,339,327,316,304,29
3,281,270,259,248,238,227,217,207,197,187,177,168,158,149,141,132,12
4,116,108,100,93,86,79,72,66,60,54,49,43,38,34,29,25,22,18,15,12,9,7
,5,3,2,1,0,0,0,0,0,1,2,3,5,7,9,12,15,18,22,25,29,34,38,43,49,54,60,6
6,72,79,86,93,100,108,116,124,132,141,149,158,168,177,187,197,207,21
7,227,238,248,259,270,281,293,304,316,327,339,351,363,375,387,399,41
2,424,436,449,461,474,486,499};

uint32_t Saw_LUT[NS] =
{0,3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63,67,71,75,79,83,87,9
1,95,99,103,107,111,115,119,123,127,131,135,139,143,147,151,155,159,
163,167,171,175,179,183,187,191,195,199,203,207,211,215,219,223,227,
231,235,239,243,247,251,255,259,263,267,271,275,279,283,287,291,295,
299,303,307,311,315,319,323,327,331,335,339,343,347,351,355,359,363,
367,371,375,379,383,387,391,395,399,403,407,411,415,419,423,427,431,
435,439,443,447,451,455,459,463,467,471,475,479,483,487,491,495,499,
503,507,511,515,519,523,527,531,535,539,543,547,551,555,559,563,567,
571,575,579,583,587,591,595,599,603,607,611,615,619,623,627,631,635,
639,643,647,651,655,659,663,667,671,675,679,683,687,691,695,699,703,
707,711,715,719,723,727,731,735,739,743,747,751,755,759,763,767,771,
775,779,783,787,791,795,799,803,807,811,815,819,823,827,831,835,839,
843,847,851,855,859,863,867,871,875,879,883,887,891,895,899,903,907,
911,915,919,923,927,931,935,939,943,947,951,955,959,963,967,971,975,
979,983,987,991,995,999,1003,1007,1011,1015,1019};

uint32_t Triangle_LUT[NS] =
{0,7,15,23,31,39,47,55,63,71,79,87,95,103,111,119,127,135,143,151,15
9,167,175,183,191,199,207,215,223,231,239,247,255,263,271,279,287,29
5,303,311,319,327,335,343,351,359,367,375,383,391,399,407,415,423,43
1,439,447,455,463,471,479,487,495,503,511,519,527,535,543,551,559,56
```

```
7,575,583,591,599,607,615,623,631,639,647,655,663,671,679,687,695,70
3,711,719,727,735,743,751,759,767,775,783,791,799,807,815,823,831,83
9,847,855,863,871,879,887,895,903,911,919,927,935,943,951,959,967,97
5,983,991,999,1007,1015,1015,1007,999,991,983,975,967,959,951,943,93
5,927,919,911,903,895,887,879,871,863,855,847,839,831,823,815,807,79
9,791,783,775,767,759,751,743,735,727,719,711,703,695,687,679,671,66
3,655,647,639,631,623,615,607,599,591,583,575,567,559,551,543,535,52
7,519,511,503,495,487,479,471,463,455,447,439,431,423,415,407,399,39
1,383,375,367,359,351,343,335,327,319,311,303,295,287,279,271,263,25
5,247,239,231,223,215,207,199,191,183,175,167,159,151,143,135,127,11
9,111,103,95,87,79,71,63,55,47,39,31,23,15,7,0};

// TODO: Equation to calculate TIM2_Ticks
uint32_t TIM2_Ticks = TIM2CLK/(NS*F_SIGNAL); // How often to write
new LUT value
uint32_t DestAddress = (uint32_t) &(TIM3->CCR3); // Write LUT TO
TIM3->CCR3 to modify PWM duty cycle

uint8_t wave = 0;
uint32_t previous_tick = 0;
/* USER CODE END PV */

/* Private function prototypes
-----------------------------------------------*/
void SystemClock_Config(void);
static void MX_NVIC_Init(void);
/* USER CODE BEGIN PFP */
void EXTI0_1_IRQHandler(void);
/* USER CODE END PFP */

/* Private user code
---------------------------------------------------------*/
/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

/**
  * @brief  The application entry point.
  * @retval int
  */
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 */
    init_LCD();
  /* 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_DMA_Init();
  MX_TIM2_Init();
  MX_TIM3_Init();

  /* Initialize interrupts */
  MX_NVIC_Init();
  /* USER CODE BEGIN 2 */
    // TODO: Start TIM3 in PWM mode on channel 3
    HAL_TIM_PWM_Start(&htim3, TIM_CHANNEL_3);

    // TODO: Start TIM2 in Output Compare (OC) mode on channel 1.
    HAL_TIM_OC_Start(&htim2, TIM_CHANNEL_1);

    // TODO: Start DMA in IT mode on TIM2->CH1; Source is LUT and
Dest is TIM3->CCR3; start with Sine LUT
    HAL_DMA_Start_IT(&hdma_tim2_ch1 , (uint32_t) &Sin_LUT,
DestAddress, NS);

    // TODO: Write current waveform to LCD ("Sine")
    delay(3000);
    lcd_command(CLEAR);
    lcd_putstring("Sine");

    // TODO: Enable DMA (start transfer from LUT to CCR)
    __HAL_TIM_ENABLE_DMA(&htim2, TIM_DMA_CC1);

  /* USER CODE END 2 */

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

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

/**
  * @brief System Clock Configuration
  * @retval None
```

```
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Initializes the RCC Oscillators according to the specified
parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSICalibrationValue =
RCC_HSICALIBRATION_DEFAULT;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_NONE;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|
RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_HSI;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_0) !=
HAL_OK)
  {
    Error_Handler();
  }
}

/**
  * @brief NVIC Configuration.
  * @retval None
  */
static void MX_NVIC_Init(void)
{
  /* DMA1_Channel4_5_IRQn interrupt configuration */
  HAL_NVIC_SetPriority(DMA1_Channel4_5_IRQn, 0, 0);
  HAL_NVIC_EnableIRQ(DMA1_Channel4_5_IRQn);
  /* EXTI0_1_IRQn interrupt configuration */
  HAL_NVIC_SetPriority(EXTI0_1_IRQn, 1, 0);
  HAL_NVIC_EnableIRQ(EXTI0_1_IRQn);
}

/* USER CODE BEGIN 4 */
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin) {
    if (GPIO_Pin == Button0_Pin) {
        if (HAL_GetTick() - previous_tick > 10) {
```

```
            wave = (wave + 1) % 3;
            __HAL_TIM_DISABLE_DMA(&htim2, TIM_DMA_CC1);
            HAL_DMA_Abort_IT(&hdma_tim2_ch1);
            lcd_command(CLEAR);
            switch (wave) {
                case 0:
                default:
                    HAL_DMA_Start_IT(&hdma_tim2_ch1 , (uint32_t)
&Sin_LUT, DestAddress, NS);
                    lcd_putstring("Sine");
                    break;
                case 1:
                    HAL_DMA_Start_IT(&hdma_tim2_ch1 , (uint32_t)
&Triangle_LUT, DestAddress, NS);
                    lcd_putstring("Triangle");
                    break;
                case 2:
                    HAL_DMA_Start_IT(&hdma_tim2_ch1 , (uint32_t)
&Saw_LUT, DestAddress, NS);
                    lcd_putstring("Saw");
                    break;
            }
            __HAL_TIM_ENABLE_DMA(&htim2, TIM_DMA_CC1);
        }
        previous_tick = HAL_GetTick();
    }
}
/* USER CODE END 4 */

/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
    /* User can add his own implementation to report the HAL error
return state */
    __disable_irq();
    while (1) {
    }
  /* USER CODE END Error_Handler_Debug */
}

#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line
number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
void assert_failed(uint8_t *file, uint32_t line)
```

```
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and
line number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n",
file, line) */
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```