Electrical and Computer Engineering Department
ECE 4510 Microcontroller Applications

Lab 9

Proximity Measurement

Isaac Bagley and Rohullah Sah

March 16, 2023

Introduction

In this lab, our goal was to learn and work and interface a digital ultrasonic proximity sensor to the STM32 microcontroller. Our first task was canceled. So starting from the second one, we had to develop a C program that would take calibrated measurements of objects in close proximity. Next we would display the output from the sensor and display the measured values on a LCD. Third, we would add a switch which would change the units for the ultrasonic sensor outputs from centimeters to inches.

Procedure

Task 2

Starting from this task, we had to develop a C program that would generate periodic trigger signals for the sensor. The detailed specs were given prior to the lab. We have to set up the ultrasonic sensor interfaced to the Timer channel. Two tri-state buffers were used to implement the trigger signal and receive echo signals to/from the sensor. Logic Analyzer was used to record the length of the echo pulses to the distance of the object from the sensor. We were to also create an excel document that can be filled with data obtained using a logic analyzer, as well as data gathered for calibration purposes using a tape measure.

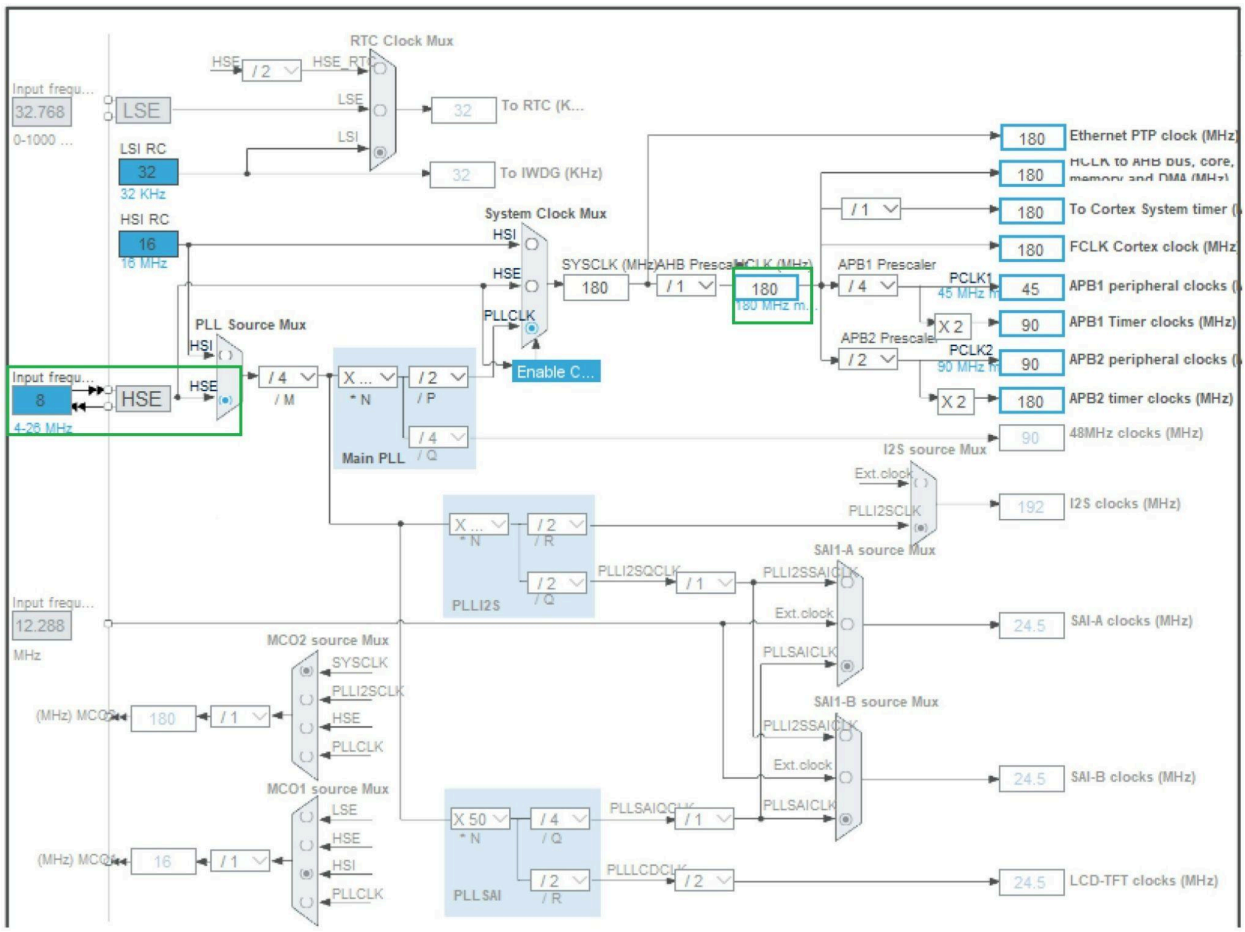


Figure 1: Clock Configuration

The HCLK and AHB clock rate were set up to 180MHz using the HSE clock at 8MHz just like mentioned in the prelab.

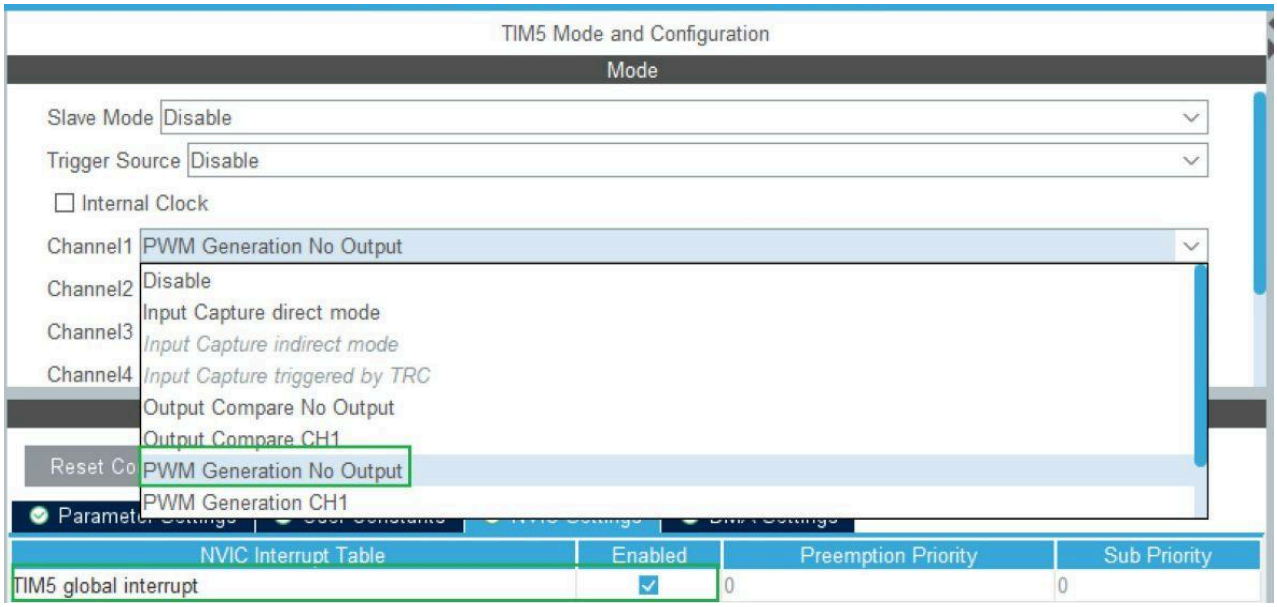


Figure 2: PWM Generation No Output enabled on Timer Channel with global interrupt.

We had to enable the PWM no output for TIM5 Channel 1, which is Port A Pin 0, because we used the PWM method to deliver the trigger signals.

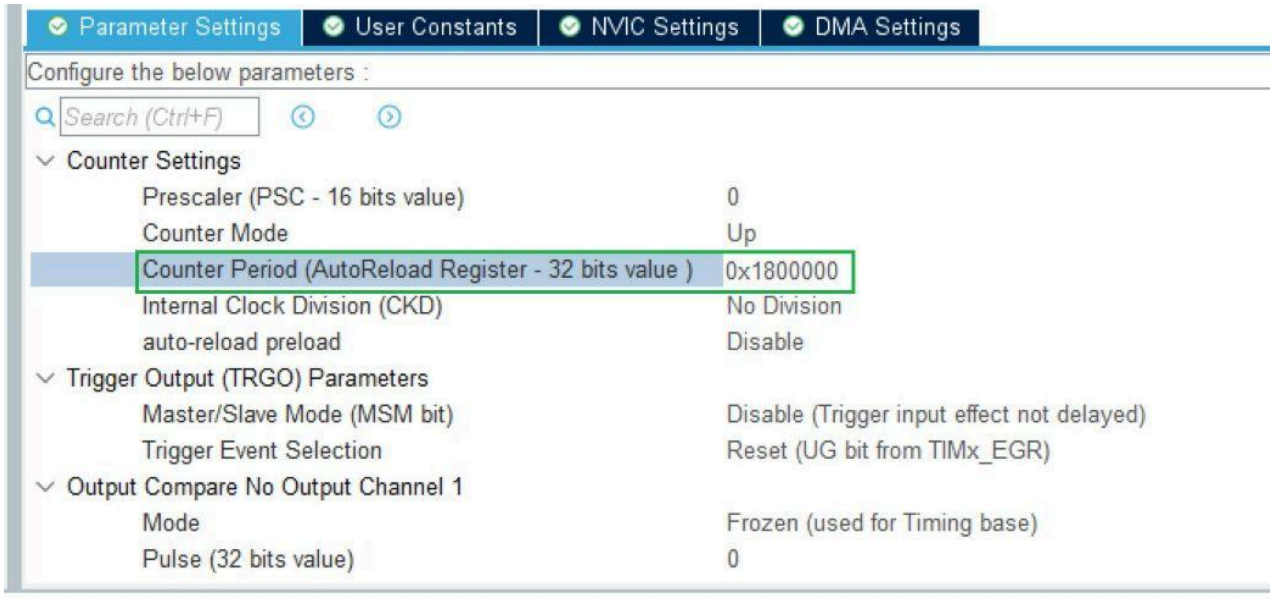


Figure 3: Counter Period - 20us

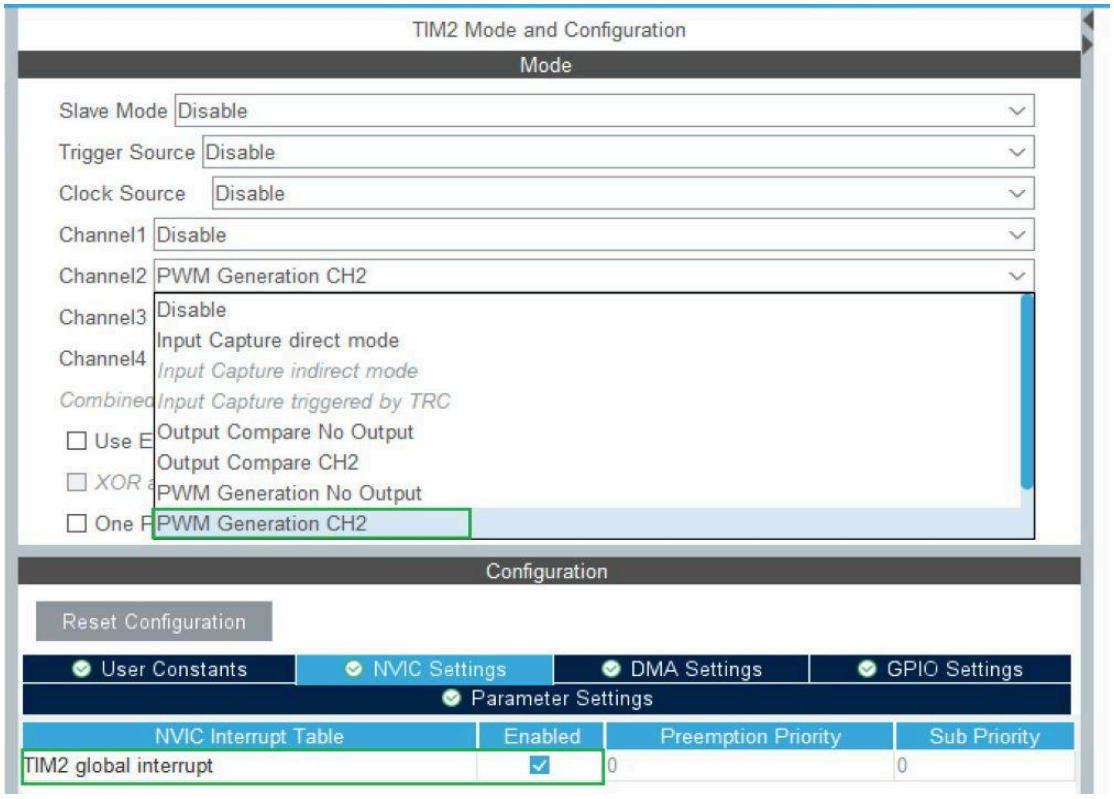


Figure 4: PWM Generation CH2 on Timer Channel with global interrupt.

We had to enable the PWM Generate for TIM2 Channel 2, which is Port A Pin 1, because we used the PWM method to trigger signals.

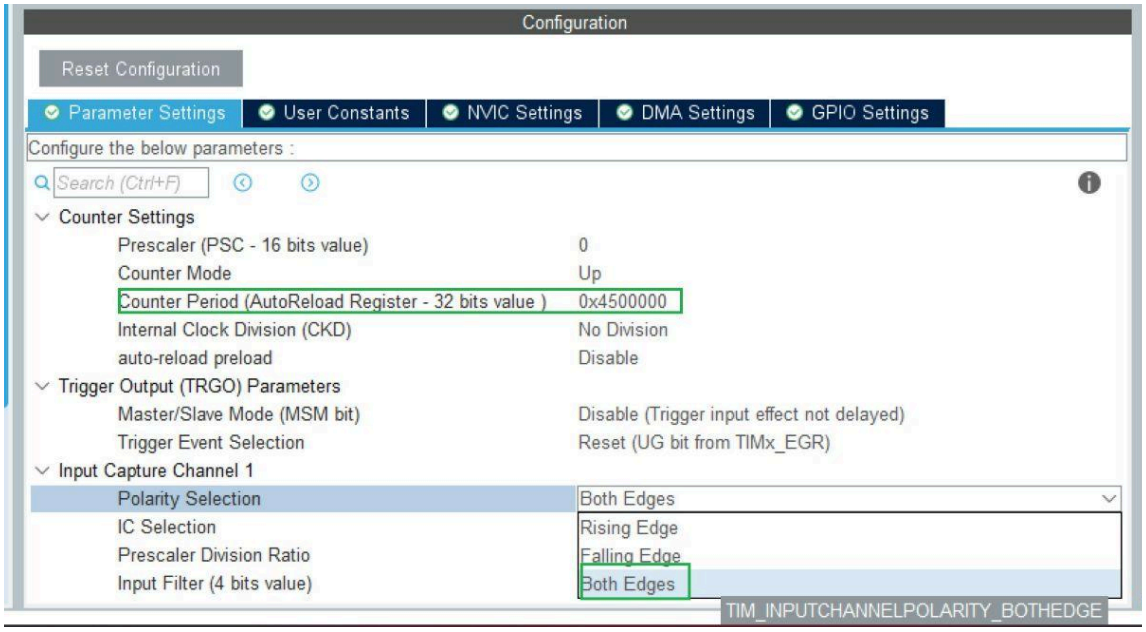


Figure 5: Change counter period - 50us and polarity selection to both edges.

Figure 6: Pinout Configuration

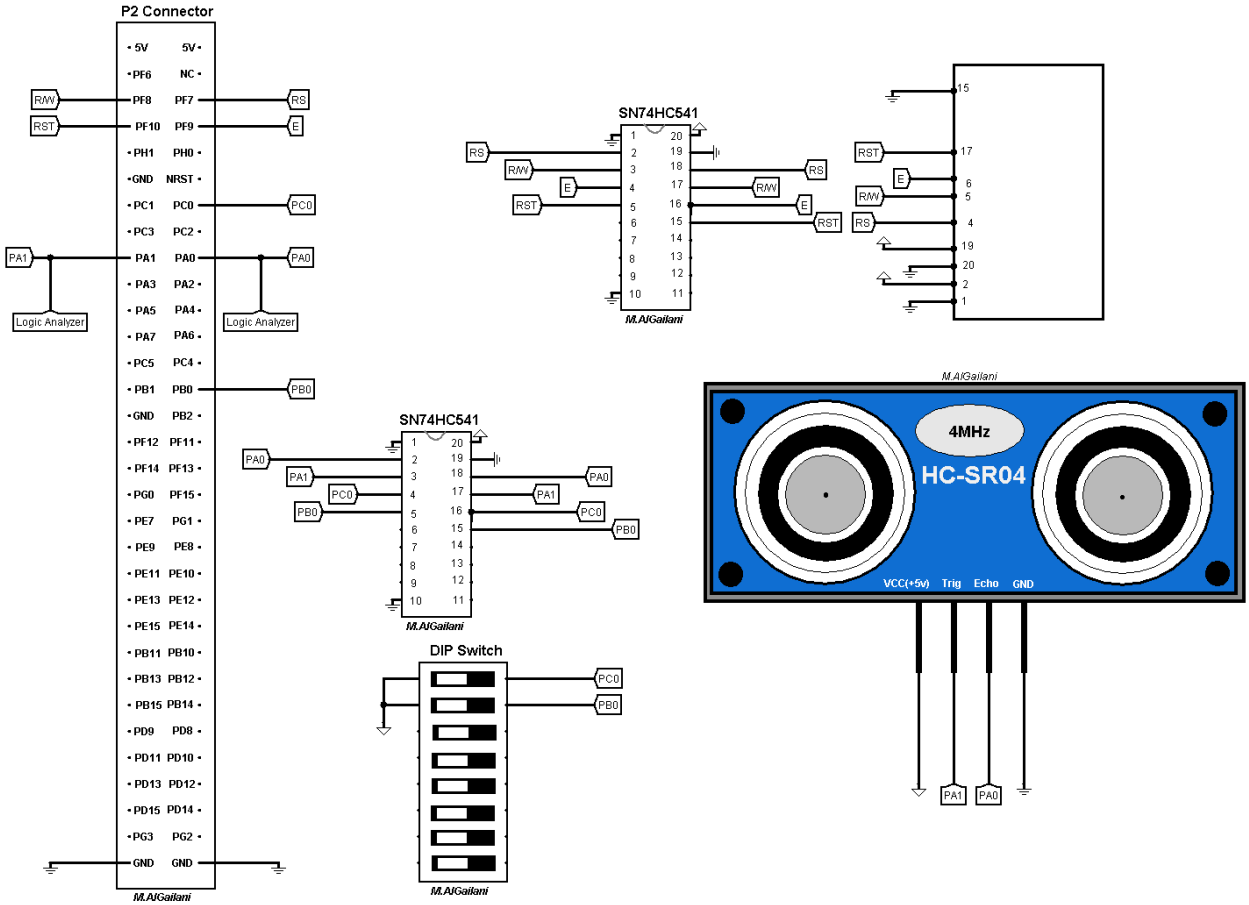


Figure 7: Schematic Diagram

Task 3

For this task, we were asked to revise the previous program to use the input capture feature of any timer channel to measure the width of the echo pulses. We were then asked to add a switch to go between centimeters and inches for the output of the microcontroller to the LCD display.

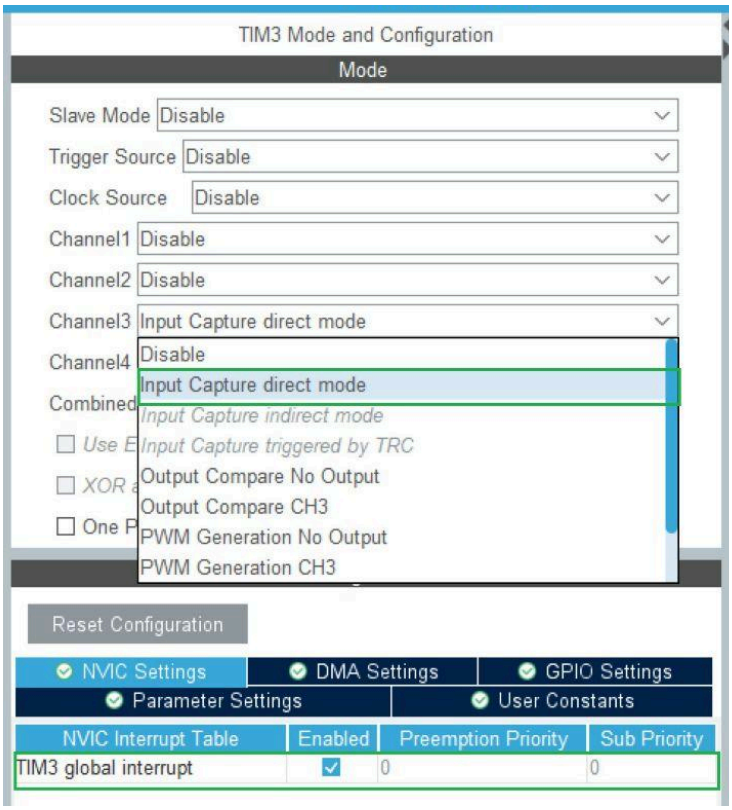
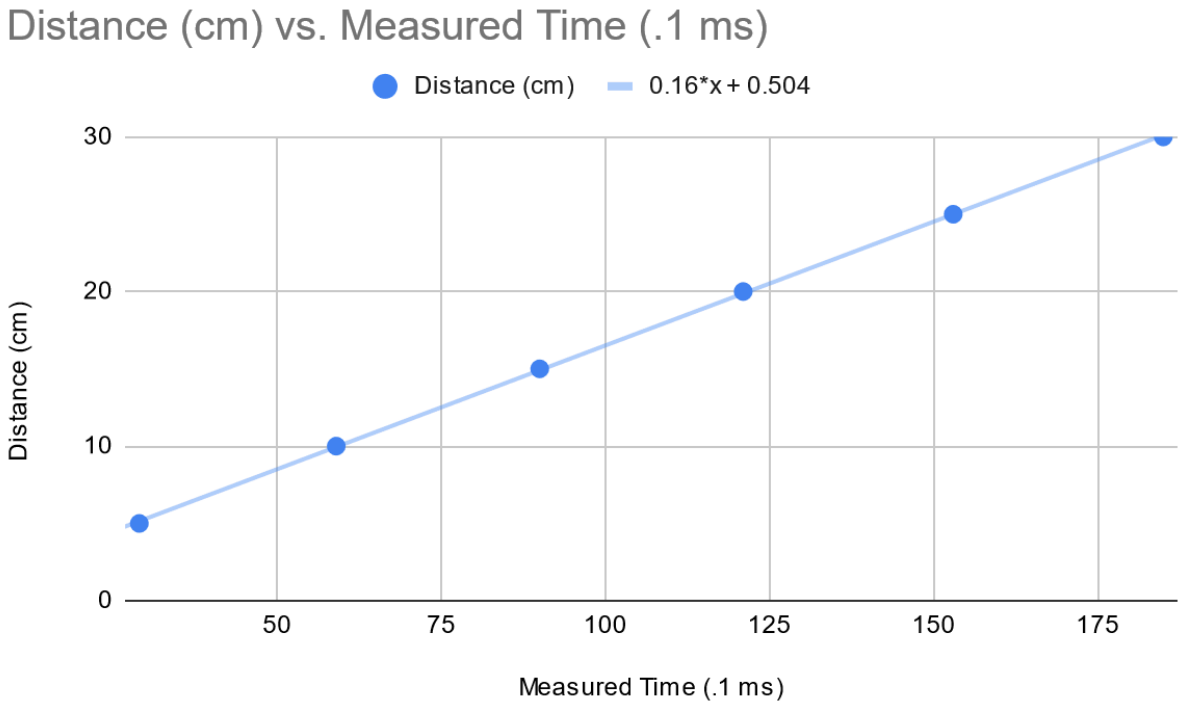


Figure 7: Enabling TIM3_CH3 and global interrupt.

Figure 8: Pinout Configuration

Results

Task 2



A graph showing the approximate conversion from one of our clock ticks to 1 cm as cm=tick/6 + 1. This calibration data was gathered in the lab using a meter stick and our sensor.

Expression	Value	Location	Type
echo_time	62	0x2000'0150	int
echo_dist	11	0x2000'00f0	int
⊞ num_out	<array>"...	0x2000'0004	uint8_t[3]
flag	0	0x2000'00f4	int
<click to add>			

This table shows the value of time read from the ultrasonic sensor and its conversion to distance in cm.

Task 3

Expression	Value	Location	Type
echo_time	32	0x2000'0150	int
echo_dist	6	0x2000'00f0	int
num_out	<array>"...	0x2000'0004	uint8_t[3]
[0]	'0' (0x30)	0x2000'0004	uint8_t
[1]	'0' (0x30)	0x2000'0005	uint8_t
[2]	'6' (0x36)	0x2000'0006	uint8_t
flag	0	0x2000'00f4	int
<click to add>			

A value of 6 cm shown to be read as the distance from the sensor and shown as the output value for the LCD screen (seen in num_out)

Expression	Value	Location	Type
echo_time	105	0x2000'0150	int
echo_dist	18	0x2000'00f0	int
num_out	<array>"...	0x2000'0004	uint8_t[3]
[0]	'0' (0x30)	0x2000'0004	uint8_t
[1]	'1' (0x31)	0x2000'0005	uint8_t
[2]	'8' (0x38)	0x2000'0006	uint8_t
flag	0	0x2000'00f4	int
<click to add>			

A measured value using the 10's place of num_out.

Expression	Value	Location	Type
echo_time	766	0x2000'0150	int
echo_dist	128	0x2000'00f0	int
num_out	<array>"...	0x2000'0004	uint8_t[3]
[0]	'1' (0x31)	0x2000'0004	uint8_t
[1]	'2' (0x32)	0x2000'0005	uint8_t
[2]	'8' (0x38)	0x2000'0006	uint8_t
flag	0	0x2000'00f4	int
<click to add>			

128 cm is the largest value we were able to reliably measure with the sensor. This also shows reliable usage of the 100's place of num_out.

Further places for num_out are not necessary since the sensor is only reliably accurate to less than 200 cm.

Expression	Value	Location	Type
echo_time	134	0x2000'0150	int
echo_dist	23	0x2000'00f0	int
num_out	<array>"...	0x2000'0004	uint8_t[3]
[0]	'0' (0x30)	0x2000'0004	uint8_t
[1]	'0' (0x30)	0x2000'0005	uint8_t
[2]	'7' (0x37)	0x2000'0006	uint8_t
flag	0	0x2000'00f4	int
<click to add>			

The difference between cm and inches is shown as the difference between echo_dist (in cm) and num_out (in inches).

Expression	Value	Location	Type
echo_time	192	0x2000'0150	int
echo_dist	33	0x2000'00f0	int
num_out	<array>"...	0x2000'0004	uint8_t[3]
[0]	'0' (0x30)	0x2000'0004	uint8_t
[1]	'1' (0x31)	0x2000'0005	uint8_t
[2]	'1' (0x31)	0x2000'0006	uint8_t
flag	1	0x2000'00f4	int
<click to add>			

Showing a number of inches using the tens place of the num_out value.

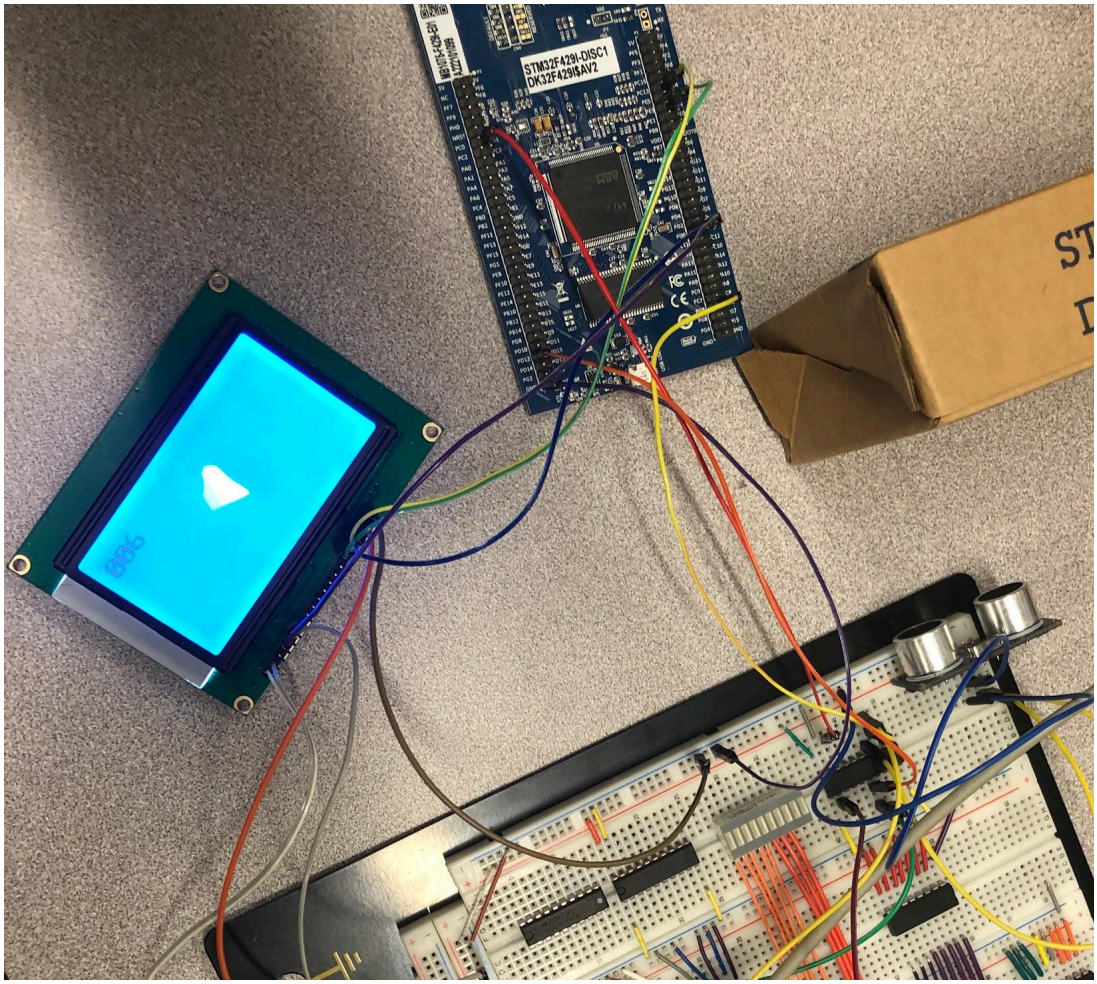


Figure 9: Our LCD showing the value of 6 cm between the sensor and box as read by the sensor.

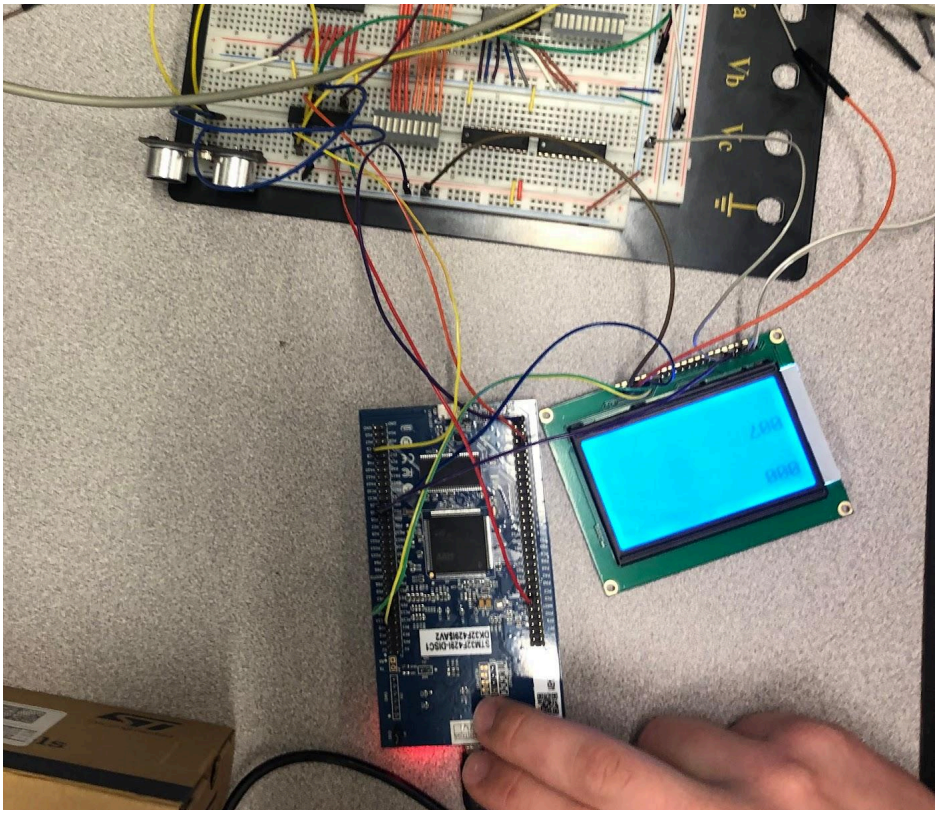


Figure 10: Our LCD showing a value of 7 inches to the box being read by the sensor. Some issues with the LCD screen interface caused us to not complete this portion of the lab.

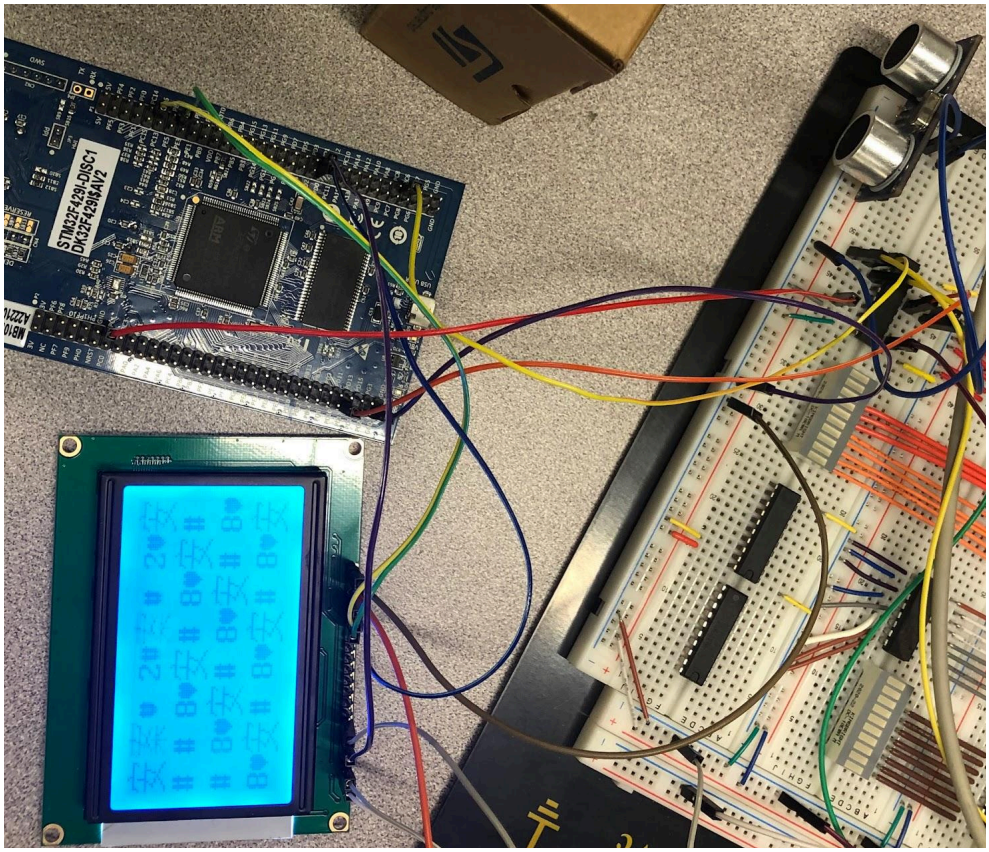


Figure 11: The continued measurement and output to the LCD caused our display to malfunction like in the above picture.

Conclusion

This lab taught us about how to use a digital ultrasonic sensor to provide data to the STM32 microcontroller along with a breadboard to interface with an LCD peripheral. Using data sheets, we can implement the trigger signals and receive echo signals to/from the sensor and interface with the LCD screen. Task 2 was done successfully and confirmed by the TA, Tony. As we continued along the lab, Task 3 was the most challenging and difficult we had experienced so far in all these labs. Unfortunately, we were unable to complete Task 3. We encountered problems with our setup and had to go to great lengths to troubleshoot, including changing out every part of the hardware and rereading every line of code but we still couldn't figure out what the issue was even though we tried countless times to fix it. In the end, the program was working, but there was some issue with the SPI interface which caused the LCD screen to become overwritten shortly after its initialization with correct values. Figures 9-10 show the progress with correct data and figure 11 shows the issues we encountered when trying to continuously update the LCD screen. The code used to drive the LCD and ultrasonic sensor is included in the appendix below.

Appendix

C Code

Task 2

Main.c

```
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @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"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */

/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */
#define Function_Set 0b00110000 //0x30
#define Display_On 0b00001100 //0x0C
```

```
#define Display_Clear 0b00000001 //0x01
#define Entry_Mode_Set 0b00000110 //0x06
/* USER CODE END PD */

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

/* USER CODE END PM */

/* Private variables ---------------------------------------------------------*/
SPI_HandleTypeDef hspi4;

TIM_HandleTypeDef htim3;
TIM_HandleTypeDef htim4;
TIM_HandleTypeDef htim5;

/* USER CODE BEGIN PV */
int echo_time = 0;
int echo_dist = 0;
uint8_t data [3] = "000";
uint8_t num_out [3] = "000";
int flag = 0;
/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_TIM4_Init(void);
static void MX_TIM5_Init(void);
static void MX_TIM3_Init(void);
static void MX_SPI4_Init(void);
/* USER CODE BEGIN PFP */
/* 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 */

  /* 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_TIM4_Init();
  MX_TIM5_Init();
  MX_TIM3_Init();
  MX_SPI4_Init();
  /* USER CODE BEGIN 2 */


  HAL_TIM_IC_Start_IT(&htim3, TIM_CHANNEL_1);
  HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_1);
  /* 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};

  /** Configure the main internal regulator output voltage
  */
  __HAL_RCC_PWR_CLK_ENABLE();

__HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1
);

  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
  RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
  RCC_OscInitStruct.PLL.PLLM = 4;
  RCC_OscInitStruct.PLL.PLLN = 180;
  RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
  RCC_OscInitStruct.PLL.PLLQ = 4;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
        Error_Handler();
```

```
  }

  /** Activate the Over-Drive mode
  */
  if (HAL_PWREx_EnableOverDrive() != 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_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV4;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV2;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_5) != HAL_OK)
  {
        Error_Handler();
  }
}

/**
  * @brief SPI4 Initialization Function
  * @param None
  * @retval None
  */
static void MX_SPI4_Init(void)
{

  /* USER CODE BEGIN SPI4_Init 0 */

  /* USER CODE END SPI4_Init 0 */

  /* USER CODE BEGIN SPI4_Init 1 */

  /* USER CODE END SPI4_Init 1 */
```

```
  /* SPI4 parameter configuration*/
  hspi4.Instance = SPI4;
  hspi4.Init.Mode = SPI_MODE_MASTER;
  hspi4.Init.Direction = SPI_DIRECTION_2LINES;
  hspi4.Init.DataSize = SPI_DATASIZE_8BIT;
  hspi4.Init.CLKPolarity = SPI_POLARITY_LOW;
  hspi4.Init.CLKPhase = SPI_PHASE_1EDGE;
  hspi4.Init.NSS = SPI_NSS_SOFT;
  hspi4.Init.BaudRatePrescaler = SPI_BAUDRATEPRESCALER_256;
  hspi4.Init.FirstBit = SPI_FIRSTBIT_MSB;
  hspi4.Init.TIMode = SPI_TIMODE_DISABLE;
  hspi4.Init.CRCCalculation = SPI_CRCCALCULATION_DISABLE;
  hspi4.Init.CRCPolynomial = 10;
  if (HAL_SPI_Init(&hspi4) != HAL_OK)
  {
          Error_Handler();
  }
  /* USER CODE BEGIN SPI4_Init 2 */

  /* USER CODE END SPI4_Init 2 */

}

/**
  * @brief TIM3 Initialization Function
  * @param None
  * @retval None
  */
static void MX_TIM3_Init(void)
{

  /* USER CODE BEGIN TIM3_Init 0 */

  /* USER CODE END TIM3_Init 0 */

  TIM_ClockConfigTypeDef sClockSourceConfig = {0};
  TIM_MasterConfigTypeDef sMasterConfig = {0};
  TIM_IC_InitTypeDef sConfigIC = {0};

  /* USER CODE BEGIN TIM3_Init 1 */
```

```
  /* USER CODE END TIM3_Init 1 */
  htim3.Instance = TIM3;
  htim3.Init.Prescaler = 900;
  htim3.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim3.Init.Period = 65535;
  htim3.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim3.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim3) != HAL_OK)
  {
      Error_Handler();
  }
  sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim3, &sClockSourceConfig) != HAL_OK)
  {
      Error_Handler();
  }
  if (HAL_TIM_IC_Init(&htim3) != HAL_OK)
  {
      Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim3, &sMasterConfig) != HAL_OK)
  {
      Error_Handler();
  }
  sConfigIC.ICPolarity = TIM_INPUTCHANNELPOLARITY_BOTHEDGE;
  sConfigIC.ICSelection = TIM_ICSELECTION_DIRECTTI;
  sConfigIC.ICPrescaler = TIM_ICPSC_DIV1;
  sConfigIC.ICFilter = 0;
  if (HAL_TIM_IC_ConfigChannel(&htim3, &sConfigIC, TIM_CHANNEL_1) != HAL_OK)
  {
      Error_Handler();
  }
  /* USER CODE BEGIN TIM3_Init 2 */

  /* USER CODE END TIM3_Init 2 */

}
```

```
/**
  * @brief TIM4 Initialization Function
  * @param None
  * @retval None
  */
static void MX_TIM4_Init(void)
{

  /* USER CODE BEGIN TIM4_Init 0 */

  /* USER CODE END TIM4_Init 0 */

  TIM_ClockConfigTypeDef sClockSourceConfig = {0};
  TIM_MasterConfigTypeDef sMasterConfig = {0};
  TIM_OC_InitTypeDef sConfigOC = {0};

  /* USER CODE BEGIN TIM4_Init 1 */

  /* USER CODE END TIM4_Init 1 */
  htim4.Instance = TIM4;
  htim4.Init.Prescaler = 1800;
  htim4.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim4.Init.Period = 2500;
  htim4.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim4.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim4) != HAL_OK)
  {
        Error_Handler();
  }
  sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim4, &sClockSourceConfig) != HAL_OK)
  {
        Error_Handler();
  }
  if (HAL_TIM_PWM_Init(&htim4) != HAL_OK)
  {
        Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
```

```
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim4, &sMasterConfig) != HAL_OK)
  {
        Error_Handler();
  }
  sConfigOC.OCMode = TIM_OCMODE_PWM1;
  sConfigOC.Pulse = 1;
  sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
  sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
  if (HAL_TIM_PWM_ConfigChannel(&htim4, &sConfigOC, TIM_CHANNEL_1) !=
HAL_OK)
  {
        Error_Handler();
  }
  /* USER CODE BEGIN TIM4_Init 2 */

  /* USER CODE END TIM4_Init 2 */
  HAL_TIM_MspPostInit(&htim4);

}

/**
  * @brief TIM5 Initialization Function
  * @param None
  * @retval None
  */
static void MX_TIM5_Init(void)
{

  /* USER CODE BEGIN TIM5_Init 0 */

  /* USER CODE END TIM5_Init 0 */

  TIM_ClockConfigTypeDef sClockSourceConfig = {0};
  TIM_MasterConfigTypeDef sMasterConfig = {0};
  TIM_IC_InitTypeDef sConfigIC = {0};

  /* USER CODE BEGIN TIM5_Init 1 */

  /* USER CODE END TIM5_Init 1 */
```

```
  htim5.Instance = TIM5;
  htim5.Init.Prescaler = 0;
  htim5.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim5.Init.Period = 4294967295;
  htim5.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim5.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim5) != HAL_OK)
  {
      Error_Handler();
  }
  sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim5, &sClockSourceConfig) != HAL_OK)
  {
      Error_Handler();
  }
  if (HAL_TIM_IC_Init(&htim5) != HAL_OK)
  {
      Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim5, &sMasterConfig) != HAL_OK)
  {
      Error_Handler();
  }
  sConfigIC.ICPolarity = TIM_INPUTCHANNELPOLARITY_BOTHEDGE;
  sConfigIC.ICSelection = TIM_ICSELECTION_DIRECTTI;
  sConfigIC.ICPrescaler = TIM_ICPSC_DIV1;
  sConfigIC.ICFilter = 0;
  if (HAL_TIM_IC_ConfigChannel(&htim5, &sConfigIC, TIM_CHANNEL_3) != HAL_OK)
  {
      Error_Handler();
  }
  /* USER CODE BEGIN TIM5_Init 2 */

  /* USER CODE END TIM5_Init 2 */

}

/**
```

```
  * @brief GPIO Initialization Function
  * @param None
  * @retval None
  */
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOE_CLK_ENABLE();
  __HAL_RCC_GPIOH_CLK_ENABLE();
  __HAL_RCC_GPIOC_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOD_CLK_ENABLE();

  /*Configure GPIO pin : PC1 */
  GPIO_InitStruct.Pin = GPIO_PIN_1;
  GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);

}

/* USER CODE BEGIN 4 */

/* 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 */
```

it.c

```
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file         stm32f4xx_it.c
  * @brief   Interrupt Service Routines.
  ******************************************************************************
  * @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 "stm32f4xx_it.h"
/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN TD */

/* USER CODE END TD */

/* 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 */
int rising = 0;
/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

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

/* USER CODE END 0 */

/* External variables --------------------------------------------------------*/
```

```
extern TIM_HandleTypeDef htim3;
extern TIM_HandleTypeDef htim4;
extern TIM_HandleTypeDef htim5;
/* USER CODE BEGIN EV */
extern int echo_time;
extern int echo_dist;
extern uint8_t num_out [3];
extern int flag;
/* USER CODE END EV */

/******************************************************************************/
/*           Cortex-M4 Processor Interruption and Exception Handlers          */
/******************************************************************************/
/**
  * @brief This function handles Non maskable interrupt.
  */
void NMI_Handler(void)
{
  /* USER CODE BEGIN NonMaskableInt_IRQn 0 */

  /* USER CODE END NonMaskableInt_IRQn 0 */
  /* USER CODE BEGIN NonMaskableInt_IRQn 1 */
  while (1)
  {
  }
  /* USER CODE END NonMaskableInt_IRQn 1 */
}

/**
  * @brief This function handles Hard fault interrupt.
  */
void HardFault_Handler(void)
{
  /* USER CODE BEGIN HardFault_IRQn 0 */

  /* USER CODE END HardFault_IRQn 0 */
  while (1)
  {
        /* USER CODE BEGIN W1_HardFault_IRQn 0 */
        /* USER CODE END W1_HardFault_IRQn 0 */
```

```
  }
}

/**
  * @brief This function handles Memory management fault.
  */
void MemManage_Handler(void)
{
  /* USER CODE BEGIN MemoryManagement_IRQn 0 */

  /* USER CODE END MemoryManagement_IRQn 0 */
  while (1)
  {
        /* USER CODE BEGIN W1_MemoryManagement_IRQn 0 */
        /* USER CODE END W1_MemoryManagement_IRQn 0 */
  }
}

/**
  * @brief This function handles Pre-fetch fault, memory access fault.
  */
void BusFault_Handler(void)
{
  /* USER CODE BEGIN BusFault_IRQn 0 */

  /* USER CODE END BusFault_IRQn 0 */
  while (1)
  {
        /* USER CODE BEGIN W1_BusFault_IRQn 0 */
        /* USER CODE END W1_BusFault_IRQn 0 */
  }
}

/**
  * @brief This function handles Undefined instruction or illegal state.
  */
void UsageFault_Handler(void)
{
  /* USER CODE BEGIN UsageFault_IRQn 0 */
```

```
  /* USER CODE END UsageFault_IRQn 0 */
  while (1)
  {
        /* USER CODE BEGIN W1_UsageFault_IRQn 0 */
        /* USER CODE END W1_UsageFault_IRQn 0 */
  }
}

/**
  * @brief This function handles System service call via SWI instruction.
  */
void SVC_Handler(void)
{
  /* USER CODE BEGIN SVCall_IRQn 0 */

  /* USER CODE END SVCall_IRQn 0 */
  /* USER CODE BEGIN SVCall_IRQn 1 */

  /* USER CODE END SVCall_IRQn 1 */
}

/**
  * @brief This function handles Debug monitor.
  */
void DebugMon_Handler(void)
{
  /* USER CODE BEGIN DebugMonitor_IRQn 0 */

  /* USER CODE END DebugMonitor_IRQn 0 */
  /* USER CODE BEGIN DebugMonitor_IRQn 1 */

  /* USER CODE END DebugMonitor_IRQn 1 */
}

/**
  * @brief This function handles Pendable request for system service.
  */
void PendSV_Handler(void)
{
  /* USER CODE BEGIN PendSV_IRQn 0 */
```

```
  /* USER CODE END PendSV_IRQn 0 */
  /* USER CODE BEGIN PendSV_IRQn 1 */

  /* USER CODE END PendSV_IRQn 1 */
}

/**
  * @brief This function handles System tick timer.
  */
void SysTick_Handler(void)
{
  /* USER CODE BEGIN SysTick_IRQn 0 */

  /* USER CODE END SysTick_IRQn 0 */
  HAL_IncTick();
  /* USER CODE BEGIN SysTick_IRQn 1 */

  /* USER CODE END SysTick_IRQn 1 */
}

/******************************************************************************/
/* STM32F4xx Peripheral Interrupt Handlers                                    */
/* Add here the Interrupt Handlers for the used peripherals.                  */
/* For the available peripheral interrupt handler names,                      */
/* please refer to the startup file (startup_stm32f4xx.s).                    */
/******************************************************************************/

/**
  * @brief This function handles TIM3 global interrupt.
  */
void TIM3_IRQHandler(void)
{
  /* USER CODE BEGIN TIM3_IRQn 0 */
  if(flag == 0){
        rising = TIM3 -> CNT;
        flag = 1;
  }
  else{
        echo_time = (TIM3 -> CNT) - rising;
```

```
            echo_dist = echo_time / 6 + 1; //convert to cm
            TIM3 -> CNT = 0;
            flag = 0;

  }
  /* USER CODE END TIM3_IRQn 0 */
  HAL_TIM_IRQHandler(&htim3);
  /* USER CODE BEGIN TIM3_IRQn 1 */

  /* USER CODE END TIM3_IRQn 1 */
}

/**
  * @brief This function handles TIM4 global interrupt.
  */
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 */

  /* USER CODE END TIM4_IRQn 1 */
}

/**
  * @brief This function handles TIM5 global interrupt.
  */
void TIM5_IRQHandler(void)
{
  /* USER CODE BEGIN TIM5_IRQn 0 */

  /* USER CODE END TIM5_IRQn 0 */
  HAL_TIM_IRQHandler(&htim5);
  /* USER CODE BEGIN TIM5_IRQn 1 */

  /* USER CODE END TIM5_IRQn 1 */
}
```

```
/* USER CODE BEGIN 1 */

/* USER CODE END 1 */
```

Task 3

Main.c

```
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @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"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */

/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */
```

```
/* USER CODE END PTD */

/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */
#define Function_Set 0b00110000 //0x30
#define Display_On 0b00001100 //0x0C
#define Display_Clear 0b00000001 //0x01
#define Entry_Mode_Set 0b00000110 //0x06
/* USER CODE END PD */

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

/* USER CODE END PM */

/* Private variables ---------------------------------------------------------*/
SPI_HandleTypeDef hspi4;

TIM_HandleTypeDef htim3;
TIM_HandleTypeDef htim4;
TIM_HandleTypeDef htim5;

/* USER CODE BEGIN PV */
int echo_time = 0;
int echo_dist = 0;
uint8_t data [3] = "000";
uint8_t num_out [3] = "000";
int flag = 0;
/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_TIM4_Init(void);
static void MX_TIM5_Init(void);
static void MX_TIM3_Init(void);
static void MX_SPI4_Init(void);
/* USER CODE BEGIN PFP */
void LCD_Init();
void LCD_SendCommand(uint8_t cmd);
```

```
void LCD_SendData(uint8_t data);
void LCD_Sel_Row_Col(uint8_t row, uint8_t col);
void LCD_sendString(char* in, int length);
void LCD_Clear();
void int_to_str(int number);
/* 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 */

  /* 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_TIM4_Init();
  MX_TIM5_Init();
  MX_TIM3_Init();
  MX_SPI4_Init();
  /* USER CODE BEGIN 2 */

  HAL_SPI_Init(&hspi4);
  LCD_Init();
  HAL_TIM_IC_Start_IT(&htim3, TIM_CHANNEL_1);
  HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_1);
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
        if(flag == 1){
 if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_1) == GPIO_PIN_SET) {
        echo_dist = echo_dist / 3;
        int_to_str(echo_dist);
        for (int i = 0; i<3; i++){
        LCD_SendData(num_out[i]);}
        LCD_Sel_Row_Col(2, 0);
        char *inches = "inches";
        LCD_sendString(inches, 6);
        }
        else {
        int_to_str(echo_dist);
        for (int i = 0; i<3; i++){
        LCD_SendData(num_out[i]);}
        LCD_Sel_Row_Col(2, 0);
        char *cm = "cm";
        LCD_sendString(cm, 2);
        }
  }
        /* 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};

  /** Configure the main internal regulator output voltage
  */
  __HAL_RCC_PWR_CLK_ENABLE();

__HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1
);

  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
  RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
  RCC_OscInitStruct.PLL.PLLM = 4;
  RCC_OscInitStruct.PLL.PLLN = 180;
  RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
  RCC_OscInitStruct.PLL.PLLQ = 4;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
        Error_Handler();
  }

  /** Activate the Over-Drive mode
  */
  if (HAL_PWREx_EnableOverDrive() != 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_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV4;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV2;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_5) != HAL_OK)
  {
        Error_Handler();
  }
}

/**
  * @brief SPI4 Initialization Function
  * @param None
  * @retval None
  */
static void MX_SPI4_Init(void)
{

  /* USER CODE BEGIN SPI4_Init 0 */

  /* USER CODE END SPI4_Init 0 */

  /* USER CODE BEGIN SPI4_Init 1 */

  /* USER CODE END SPI4_Init 1 */
  /* SPI4 parameter configuration*/
  hspi4.Instance = SPI4;
  hspi4.Init.Mode = SPI_MODE_MASTER;
  hspi4.Init.Direction = SPI_DIRECTION_2LINES;
  hspi4.Init.DataSize = SPI_DATASIZE_8BIT;
  hspi4.Init.CLKPolarity = SPI_POLARITY_LOW;
  hspi4.Init.CLKPhase = SPI_PHASE_1EDGE;
  hspi4.Init.NSS = SPI_NSS_SOFT;
```

```
  hspi4.Init.BaudRatePrescaler = SPI_BAUDRATEPRESCALER_256;
  hspi4.Init.FirstBit = SPI_FIRSTBIT_MSB;
  hspi4.Init.TIMode = SPI_TIMODE_DISABLE;
  hspi4.Init.CRCCalculation = SPI_CRCCALCULATION_DISABLE;
  hspi4.Init.CRCPolynomial = 10;
  if (HAL_SPI_Init(&hspi4) != HAL_OK)
  {
        Error_Handler();
  }
  /* USER CODE BEGIN SPI4_Init 2 */

  /* USER CODE END SPI4_Init 2 */

}

/**
  * @brief TIM3 Initialization Function
  * @param None
  * @retval None
  */
static void MX_TIM3_Init(void)
{

  /* USER CODE BEGIN TIM3_Init 0 */

  /* USER CODE END TIM3_Init 0 */

  TIM_ClockConfigTypeDef sClockSourceConfig = {0};
  TIM_MasterConfigTypeDef sMasterConfig = {0};
  TIM_IC_InitTypeDef sConfigIC = {0};

  /* USER CODE BEGIN TIM3_Init 1 */

  /* USER CODE END TIM3_Init 1 */
  htim3.Instance = TIM3;
  htim3.Init.Prescaler = 900;
  htim3.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim3.Init.Period = 65535;
  htim3.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim3.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
```

```
  if (HAL_TIM_Base_Init(&htim3) != HAL_OK)
  {
        Error_Handler();
  }
  sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim3, &sClockSourceConfig) != HAL_OK)
  {
        Error_Handler();
  }
  if (HAL_TIM_IC_Init(&htim3) != HAL_OK)
  {
        Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim3, &sMasterConfig) != HAL_OK)
  {
        Error_Handler();
  }
  sConfigIC.ICPolarity = TIM_INPUTCHANNELPOLARITY_BOTHEDGE;
  sConfigIC.ICSelection = TIM_ICSELECTION_DIRECTTI;
  sConfigIC.ICPrescaler = TIM_ICPSC_DIV1;
  sConfigIC.ICFilter = 0;
  if (HAL_TIM_IC_ConfigChannel(&htim3, &sConfigIC, TIM_CHANNEL_1) != HAL_OK)
  {
        Error_Handler();
  }
  /* USER CODE BEGIN TIM3_Init 2 */

  /* USER CODE END TIM3_Init 2 */

}

/**
  * @brief TIM4 Initialization Function
  * @param None
  * @retval None
  */
static void MX_TIM4_Init(void)
{
```

```
/* USER CODE BEGIN TIM4_Init 0 */

/* USER CODE END TIM4_Init 0 */

TIM_ClockConfigTypeDef sClockSourceConfig = {0};
TIM_MasterConfigTypeDef sMasterConfig = {0};
TIM_OC_InitTypeDef sConfigOC = {0};

/* USER CODE BEGIN TIM4_Init 1 */

/* USER CODE END TIM4_Init 1 */
htim4.Instance = TIM4;
htim4.Init.Prescaler = 1800;
htim4.Init.CounterMode = TIM_COUNTERMODE_UP;
htim4.Init.Period = 2500;
htim4.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
htim4.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
if (HAL_TIM_Base_Init(&htim4) != HAL_OK)
{
	Error_Handler();
}
sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
if (HAL_TIM_ConfigClockSource(&htim4, &sClockSourceConfig) != HAL_OK)
{
	Error_Handler();
}
if (HAL_TIM_PWM_Init(&htim4) != HAL_OK)
{
	Error_Handler();
}
sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
if (HAL_TIMEx_MasterConfigSynchronization(&htim4, &sMasterConfig) != HAL_OK)
{
	Error_Handler();
}
sConfigOC.OCMode = TIM_OCMODE_PWM1;
sConfigOC.Pulse = 1;
sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
```

```
  sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
  if (HAL_TIM_PWM_ConfigChannel(&htim4, &sConfigOC, TIM_CHANNEL_1) !=
HAL_OK)
  {
        Error_Handler();
  }
  /* USER CODE BEGIN TIM4_Init 2 */

  /* USER CODE END TIM4_Init 2 */
  HAL_TIM_MspPostInit(&htim4);

}

/**
  * @brief TIM5 Initialization Function
  * @param None
  * @retval None
  */
static void MX_TIM5_Init(void)
{

  /* USER CODE BEGIN TIM5_Init 0 */

  /* USER CODE END TIM5_Init 0 */

  TIM_ClockConfigTypeDef sClockSourceConfig = {0};
  TIM_MasterConfigTypeDef sMasterConfig = {0};
  TIM_IC_InitTypeDef sConfigIC = {0};

  /* USER CODE BEGIN TIM5_Init 1 */

  /* USER CODE END TIM5_Init 1 */
  htim5.Instance = TIM5;
  htim5.Init.Prescaler = 0;
  htim5.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim5.Init.Period = 4294967295;
  htim5.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim5.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim5) != HAL_OK)
  {
```

```
        Error_Handler();
  }
  sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim5, &sClockSourceConfig) != HAL_OK)
  {
        Error_Handler();
  }
  if (HAL_TIM_IC_Init(&htim5) != HAL_OK)
  {
        Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim5, &sMasterConfig) != HAL_OK)
  {
        Error_Handler();
  }
  sConfigIC.ICPolarity = TIM_INPUTCHANNELPOLARITY_BOTHEDGE;
  sConfigIC.ICSelection = TIM_ICSELECTION_DIRECTTI;
  sConfigIC.ICPrescaler = TIM_ICPSC_DIV1;
  sConfigIC.ICFilter = 0;
  if (HAL_TIM_IC_ConfigChannel(&htim5, &sConfigIC, TIM_CHANNEL_3) != HAL_OK)
  {
        Error_Handler();
  }
  /* USER CODE BEGIN TIM5_Init 2 */

  /* USER CODE END TIM5_Init 2 */

}

/**
  * @brief GPIO Initialization Function
  * @param None
  * @retval None
  */
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};
```

```
  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOE_CLK_ENABLE();
  __HAL_RCC_GPIOH_CLK_ENABLE();
  __HAL_RCC_GPIOC_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOD_CLK_ENABLE();

  /*Configure GPIO pin : PC1 */
  GPIO_InitStruct.Pin = GPIO_PIN_1;
  GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);

}

/* USER CODE BEGIN 4 */
void LCD_SendCommand (uint8_t cmd){
   data[0] = 0xF8;
   data[1] = (cmd & 0xF0);
   data[2] = ((cmd << 4) & 0xF0);
   HAL_GPIO_WritePin(GPIOD, GPIO_PIN_0, GPIO_PIN_SET); // CS to high
        HAL_SPI_Transmit(&hspi4, data, 3, 1);
   HAL_GPIO_WritePin(GPIOD, GPIO_PIN_0, GPIO_PIN_RESET); // CS to low
}

void LCD_Init() {
        HAL_GPIO_WritePin(GPIOD, GPIO_PIN_1, GPIO_PIN_RESET); // Make sure
reset is low
        HAL_Delay(100);
        HAL_GPIO_WritePin(GPIOD, GPIO_PIN_1, GPIO_PIN_SET); // Reset Low to
high
        HAL_Delay(50); // greater than 40ms
        LCD_SendCommand(Function_Set);
        HAL_Delay(10); // greater than 100us
        LCD_SendCommand(Function_Set);
        HAL_Delay(10); // greater than 37us
        LCD_SendCommand(Display_On);
        HAL_Delay(10); // greater than 100us
        LCD_SendCommand(Display_Clear);
        HAL_Delay(15); // greater than 10ms
```

```
	LCD_SendCommand(Entry_Mode_Set);
	HAL_Delay(10); // greater than 100us
}

void LCD_Clear(){
	LCD_SendCommand(Display_Clear);
}

void LCD_SendData (uint8_t ascii){
   data[0] = 0xFA;
   data[1] = (ascii & 0xF0);
   data[2] = ((ascii << 4) & 0xF0);
   HAL_GPIO_WritePin(GPIOD, GPIO_PIN_0, GPIO_PIN_SET); // CS to high
	HAL_SPI_Transmit(&hspi4, data, 3, 1);
   HAL_GPIO_WritePin(GPIOD, GPIO_PIN_0, GPIO_PIN_RESET); // CS to low
}

void LCD_sendString(char* in, int length)
{
  uint8_t letter;
  for (int i; i<length; i++) //iterate through the input string
  {
	letter = in[i]; // assign the letter to the int, getting its ascii value, this may be
unnecessary
	LCD_SendData(letter); // send the data to the board
  }
}

void LCD_Sel_Row_Col(uint8_t row, uint8_t col)
{
	switch(row)
   {
   case 0:
	 col /= 2; //cursor is 2 spaces wide
	 col |= 0x80; // 0b10000000 first row
		break;
	case 1:
		col /= 2;
		col |= 0x90; //second column starts at 0x90
		break;
```

```
        case 2:
                col /= 2;
                col |= 0x88; // third row is 0x88
                break;
        case 3:
                col /= 2;
                col |= 0x98; //fourth row is 0x98
                break;
        }
        LCD_SendCommand(col);
        HAL_Delay(5);
}

void int_to_str(int number) {
  int Hex_offset = 0x30;
  int first_dig = (number % 1000)/100;  //separate the hundreds digit
  first_dig = first_dig + Hex_offset;
  num_out[0] = first_dig;
  int second_dig = (number % 100)/10 ; //separate the tens digit
  second_dig = second_dig + Hex_offset;
  num_out[1] = second_dig;
  int third_dig = number % 10;  // separate the ones place
  third_dig = third_dig + Hex_offset;
  num_out[2] = third_dig;
}
/* 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 */
```

It.c

```
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file         stm32f4xx_it.c
  * @brief   Interrupt Service Routines.
  ******************************************************************************
  * @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 "stm32f4xx_it.h"
/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN TD */

/* USER CODE END TD */

/* 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 */
int rising = 0;
/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

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

```
/* USER CODE END 0 */

/* External variables ---------------------------------------------------------*/
extern TIM_HandleTypeDef htim3;
extern TIM_HandleTypeDef htim4;
extern TIM_HandleTypeDef htim5;
/* USER CODE BEGIN EV */
extern int echo_time;
extern int echo_dist;
extern void LCD_Init();
extern void LCD_SendCommand(uint8_t cmd);
extern void LCD_SendData(uint8_t data);
extern void LCD_Sel_Row_Col(uint8_t row, uint8_t col);
extern void LCD_sendString(char* in, int length);
extern void LCD_Clear();
extern void int_to_str(int number);
extern uint8_t num_out [3];
extern int flag;
/* USER CODE END EV */

/******************************************************************************/
/*        Cortex-M4 Processor Interruption and Exception Handlers             */
/******************************************************************************/
/**
  * @brief This function handles Non maskable interrupt.
  */
void NMI_Handler(void)
{
  /* USER CODE BEGIN NonMaskableInt_IRQn 0 */

  /* USER CODE END NonMaskableInt_IRQn 0 */
  /* USER CODE BEGIN NonMaskableInt_IRQn 1 */
  while (1)
  {
  }
  /* USER CODE END NonMaskableInt_IRQn 1 */
}

/**
  * @brief This function handles Hard fault interrupt.
```

```
  */
void HardFault_Handler(void)
{
  /* USER CODE BEGIN HardFault_IRQn 0 */

  /* USER CODE END HardFault_IRQn 0 */
  while (1)
  {
        /* USER CODE BEGIN W1_HardFault_IRQn 0 */
        /* USER CODE END W1_HardFault_IRQn 0 */
  }
}

/**
  * @brief This function handles Memory management fault.
  */
void MemManage_Handler(void)
{
  /* USER CODE BEGIN MemoryManagement_IRQn 0 */

  /* USER CODE END MemoryManagement_IRQn 0 */
  while (1)
  {
        /* USER CODE BEGIN W1_MemoryManagement_IRQn 0 */
        /* USER CODE END W1_MemoryManagement_IRQn 0 */
  }
}

/**
  * @brief This function handles Pre-fetch fault, memory access fault.
  */
void BusFault_Handler(void)
{
  /* USER CODE BEGIN BusFault_IRQn 0 */

  /* USER CODE END BusFault_IRQn 0 */
  while (1)
  {
        /* USER CODE BEGIN W1_BusFault_IRQn 0 */
        /* USER CODE END W1_BusFault_IRQn 0 */
```

```
  }
}

/**
  * @brief This function handles Undefined instruction or illegal state.
  */
void UsageFault_Handler(void)
{
  /* USER CODE BEGIN UsageFault_IRQn 0 */

  /* USER CODE END UsageFault_IRQn 0 */
  while (1)
  {
        /* USER CODE BEGIN W1_UsageFault_IRQn 0 */
        /* USER CODE END W1_UsageFault_IRQn 0 */
  }
}

/**
  * @brief This function handles System service call via SWI instruction.
  */
void SVC_Handler(void)
{
  /* USER CODE BEGIN SVCall_IRQn 0 */

  /* USER CODE END SVCall_IRQn 0 */
  /* USER CODE BEGIN SVCall_IRQn 1 */

  /* USER CODE END SVCall_IRQn 1 */
}

/**
  * @brief This function handles Debug monitor.
  */
void DebugMon_Handler(void)
{
  /* USER CODE BEGIN DebugMonitor_IRQn 0 */

  /* USER CODE END DebugMonitor_IRQn 0 */
  /* USER CODE BEGIN DebugMonitor_IRQn 1 */
```

```
  /* USER CODE END DebugMonitor_IRQn 1 */
}

/**
  * @brief This function handles Pendable request for system service.
  */
void PendSV_Handler(void)
{
  /* USER CODE BEGIN PendSV_IRQn 0 */

  /* USER CODE END PendSV_IRQn 0 */
  /* USER CODE BEGIN PendSV_IRQn 1 */

  /* USER CODE END PendSV_IRQn 1 */
}

/**
  * @brief This function handles System tick timer.
  */
void SysTick_Handler(void)
{
  /* USER CODE BEGIN SysTick_IRQn 0 */

  /* USER CODE END SysTick_IRQn 0 */
  HAL_IncTick();
  /* USER CODE BEGIN SysTick_IRQn 1 */

  /* USER CODE END SysTick_IRQn 1 */
}

/******************************************************************************/
/* STM32F4xx Peripheral Interrupt Handlers                                    */
/* Add here the Interrupt Handlers for the used peripherals.                  */
/* For the available peripheral interrupt handler names,                      */
/* please refer to the startup file (startup_stm32f4xx.s).                    */
/******************************************************************************/

/**
  * @brief This function handles TIM3 global interrupt.
```

```
  */
void TIM3_IRQHandler(void)
{
  /* USER CODE BEGIN TIM3_IRQn 0 */
  if(flag == 0){
          rising = TIM3 -> CNT;
          flag = 1;
  }
  else{
          echo_time = (TIM3 -> CNT) - rising;
          echo_dist = echo_time / 6 + 1; //convert to cm
          TIM3 -> CNT = 0;
          flag = 0;

  }
  /* USER CODE END TIM3_IRQn 0 */
  HAL_TIM_IRQHandler(&htim3);
  /* USER CODE BEGIN TIM3_IRQn 1 */

  /* USER CODE END TIM3_IRQn 1 */
}

/**
  * @brief This function handles TIM4 global interrupt.
  */
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 */

  /* USER CODE END TIM4_IRQn 1 */
}

/**
  * @brief This function handles TIM5 global interrupt.
  */
void TIM5_IRQHandler(void)
```

```
{
  /* USER CODE BEGIN TIM5_IRQn 0 */

  /* USER CODE END TIM5_IRQn 0 */
  HAL_TIM_IRQHandler(&htim5);
  /* USER CODE BEGIN TIM5_IRQn 1 */

  /* USER CODE END TIM5_IRQn 1 */
}

/* USER CODE BEGIN 1 */

/* USER CODE END 1 */
```

.lst Files

Task 2

Main.lst

```
###############################################################################
#########
#
# IAR ANSI C/C++ Compiler V9.20.4.327/W64 for ARM         16/Mar/2023  18:02:46
# Copyright 1999-2022 IAR Systems AB.
#
#       Cpu mode             =  thumb
#       Endian          =  little
#       Source file     =
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\Cor
e\Src\main.c
#       Command line         =
#       -f
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core\main.o.rsp
#
(S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\Cor
e\Src\main.c
```

```
#        -D USE_HAL_DRIVER -D STM32F429xx -lC
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\List\Application\User\Core
#        -o
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core
#        --debug --endian=little --cpu=Cortex-M4 -e --fpu=VFPv4_sp
#        --dlib_config S:\School_Work\arm\inc\c\DLib_Config_Full.h -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Core/Inc\
#        -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/STM32F4xx_HAL_Driver/Inc\
#        -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/STM32F4xx_HAL_Driver/Inc/Legacy\
#        -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/CMSIS/Device/ST/STM32F4xx/Include\
#        -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/CMSIS/Include\
#        -Ohz) --dependencies=n
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core\main.o.d
#        Locale         =  C
#        List file      =
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\List\Application\User\Core\main.lst
#        Object file    =
```

```
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core\main.o
#       Runtime model:
#       __CPP_Runtime   =  1
#       __SystemLibrary =  DLib
#       __dlib_version  =  6
#       __size_limit    =  32768|ARM.EW.LINKER
#
################################################################################
#########

S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\Cor
e\Src\main.c
        1       /* USER CODE BEGIN Header */
        2       /**
        3       ******************************************************************************
        4       * @file           : main.c
        5       * @brief          : Main program body
        6       ******************************************************************************
        7       * @attention
        8       *
        9       * Copyright (c) 2023 STMicroelectronics.
        10              * All rights reserved.
        11              *
        12              * This software is licensed under terms that can be found in the
LICENSE file
        13              * in the root directory of this software component.
        14              * If no LICENSE file comes with this software, it is provided AS-IS.
        15              *
        16
******************************************************************************
        17              */
        18      /* USER CODE END Header */
        19      /* Includes ------------------------------------------------------------*/
        20      #include "main.h"
        21
        22      /* Private includes ----------------------------------------------------------*/
        23      /* USER CODE BEGIN Includes */
        24
```

```
      25     /* USER CODE END Includes */
      26
      27     /* Private typedef -----------------------------------------------------------*/
      28     /* USER CODE BEGIN PTD */
      29
      30     /* USER CODE END PTD */
      31
      32     /* Private define ------------------------------------------------------------*/
      33     /* USER CODE BEGIN PD */
      34     #define Function_Set 0b00110000 //0x30
      35     #define Display_On 0b00001100 //0x0C
      36     #define Display_Clear 0b00000001 //0x01
      37     #define Entry_Mode_Set 0b00000110 //0x06
      38     /* USER CODE END PD */
      39
      40     /* Private macro -------------------------------------------------------------*/
      41     /* USER CODE BEGIN PM */
      42
      43     /* USER CODE END PM */
      44
      45     /* Private variables ---------------------------------------------------------*/

\                              In section .bss, align 4
      46     SPI_HandleTypeDef hspi4;
\            hspi4:
\     0x0                      DS8 88
      47
      48     TIM_HandleTypeDef htim3;
      49     TIM_HandleTypeDef htim4;
      50     TIM_HandleTypeDef htim5;
      51
      52     /* USER CODE BEGIN PV */
      53     int echo_time = 0;
      54     int echo_dist = 0;

\                              In section .data, align 4
      55     uint8_t data [3] = "000";
\            `data`:
\     0x0   0x30 0x30        DC8 0x30, 0x30, 0x30
```

```
\               0x30
\       0x3                     DS8 1

\                               In section .data, align 4
        56      uint8_t num_out [3] = "000";
\               num_out:
\       0x0   0x30 0x30         DC8 0x30, 0x30, 0x30

\               0x30
\       0x3                     DS8 1

\                               In section .bss, align 4
\               htim3:
\       0x0                     DS8 72
\               htim4:
\       0x48                    DS8 72
\               htim5:
\       0x90                    DS8 72
\               echo_dist:
\       0xD8                    DS8 4
        57      int flag = 0;
\               flag:
\       0xDC                    DS8 4

\                               In section .bss, align 4
\               echo_time:
\       0x0                     DS8 4
        58      /* USER CODE END PV */
        59
        60      /* Private function prototypes -----------------------------------------------*/
        61      void SystemClock_Config(void);
        62      static void MX_GPIO_Init(void);
        63      static void MX_TIM4_Init(void);
        64      static void MX_TIM5_Init(void);
        65      static void MX_TIM3_Init(void);
        66      static void MX_SPI4_Init(void);
        67      /* USER CODE BEGIN PFP */
        68      void LCD_Init();
        69      void LCD_SendCommand(uint8_t cmd);
        70      void LCD_SendData(uint8_t data);
```

```
        71      void LCD_Sel_Row_Col(uint8_t row, uint8_t col);
        72      void LCD_sendString(char* in, int length);
        73      void LCD_Clear();
        74      void int_to_str(int number);
        75      /* USER CODE END PFP */
        76
        77      /* Private user code ---------------------------------------------------------*/
        78      /* USER CODE BEGIN 0 */
        79
        80      /* USER CODE END 0 */
        81
        82      /**
        83              * @brief  The application entry point.
        84              * @retval int
        85              */

  \                             In section .text, align 4, keep-with-next
        86      int main(void)
        87      {
  \             main: (+1)
  \     0x0   0xE92D 0x47F8       PUSH          {R3-R10,LR}
  \     0x4   0xB08D              SUB   SP,SP,#+52
        88              /* USER CODE BEGIN 1 */
        89
        90              /* USER CODE END 1 */
        91
        92              /* MCU Configuration--------------------------------------------------------*/
        93
        94              /* Reset of all peripherals, Initializes the Flash interface and the
Systick. */
        95              HAL_Init();
  \     0x6   0x.... 0x....       BL      HAL_Init
        96
        97              /* USER CODE BEGIN Init */
        98
        99              /* USER CODE END Init */
       100
       101              /* Configure the system clock */
       102              SystemClock_Config();
  \     0xA   0x.... 0x....       BL      SystemClock_Config
```

```
      103
      104             /* USER CODE BEGIN SysInit */
      105
      106             /* USER CODE END SysInit */
      107
      108             /* Initialize all configured peripherals */
      109             MX_GPIO_Init();
\     0xE   0x.... 0x....     BL      ?Subroutine4
\             ??CrossCallReturnLabel_9: (+1)
\     0x12  0x2000        MOVS        R0,#+0
\     0x14  0x9000        STR   R0,[SP, #+0]
\     0x16  0x.... 0x....     LDR.W       R9,??DataTable7_1
\     0x1A  0x.... 0x....     LDR.W       R0,??DataTable7_2
\     0x1E  0x.... 0x....     LDR.W       R5,??DataTable7_3
\     0x22  0x6801        LDR   R1,[R0, #+0]
\     0x24  0xF041 0x0110     ORR   R1,R1,#0x10
\     0x28  0x6001        STR   R1,[R0, #+0]
\     0x2A  0x6802        LDR   R2,[R0, #+0]
\     0x2C  0xF002 0x0210     AND   R2,R2,#0x10
\     0x30  0x9200        STR   R2,[SP, #+0]
\     0x32  0x2200        MOVS        R2,#+0
\     0x34  0x9900        LDR   R1,[SP, #+0]
\     0x36  0x9200        STR   R2,[SP, #+0]
\     0x38  0x6803        LDR   R3,[R0, #+0]
\     0x3A  0xF043 0x0380     ORR   R3,R3,#0x80
\     0x3E  0x6003        STR   R3,[R0, #+0]
\     0x40  0x6801        LDR   R1,[R0, #+0]
\     0x42  0xF001 0x0180     AND   R1,R1,#0x80
\     0x46  0x9100        STR   R1,[SP, #+0]
\     0x48  0x9900        LDR   R1,[SP, #+0]
\     0x4A  0x9200        STR   R2,[SP, #+0]
\     0x4C  0x6803        LDR   R3,[R0, #+0]
\     0x4E  0xF043 0x0304     ORR   R3,R3,#0x4
\     0x52  0x6003        STR   R3,[R0, #+0]
\     0x54  0x6801        LDR   R1,[R0, #+0]
\     0x56  0xF001 0x0104     AND   R1,R1,#0x4
\     0x5A  0x9100        STR   R1,[SP, #+0]
\     0x5C  0x9900        LDR   R1,[SP, #+0]
\     0x5E  0x9200        STR   R2,[SP, #+0]
\     0x60  0x6803        LDR   R3,[R0, #+0]
```

```
\      0x62   0xF043 0x0301      ORR   R3,R3,#0x1
\      0x66   0x6003        STR   R3,[R0, #+0]
\      0x68   0x6801        LDR   R1,[R0, #+0]
\      0x6A   0xF001 0x0101      AND   R1,R1,#0x1
\      0x6E   0x9100        STR   R1,[SP, #+0]
\      0x70   0x9900        LDR   R1,[SP, #+0]
\      0x72   0x9200        STR   R2,[SP, #+0]
\      0x74   0x2102        MOVS        R1,#+2
\      0x76   0x6803        LDR   R3,[R0, #+0]
\      0x78   0xF043 0x0308      ORR   R3,R3,#0x8
\      0x7C   0x6003        STR   R3,[R0, #+0]
\      0x7E   0x6800        LDR   R0,[R0, #+0]
\      0x80   0xF000 0x0008      AND   R0,R0,#0x8
\      0x84   0x9000        STR   R0,[SP, #+0]
\      0x86   0x9800        LDR   R0,[SP, #+0]
\      0x88   0x9101        STR   R1,[SP, #+4]
\      0x8A   0x9202        STR   R2,[SP, #+8]
\      0x8C   0x9203        STR   R2,[SP, #+12]
\      0x8E   0xA901              ADD   R1,SP,#+4
\      0x90   0x4648        MOV   R0,R9
\      0x92   0x.... 0x....      BL    HAL_GPIO_Init
       110             MX_TIM4_Init();
\      0x96   0x2210        MOVS        R2,#+16
\      0x98   0x2100        MOVS        R1,#+0
\      0x9A   0xA809              ADD   R0,SP,#+36
\      0x9C   0x.... 0x....      BL    memset
\      0xA0   0x.... 0x....      BL    ?Subroutine1
\             ??CrossCallReturnLabel_0: (+1)
\      0xA4   0x221C              MOVS        R2,#+28
\      0xA6   0x.... 0x....      BL    ??Subroutine2_0
\             ??CrossCallReturnLabel_3: (+1)
\      0xAA   0x.... 0x....      LDR.W       R0,??DataTable7_4
\      0xAE   0x64A8              STR   R0,[R5, #+72]
\      0xB0   0xF44F 0x61E1      MOV   R1,#+1800
\      0xB4   0x64E9              STR   R1,[R5, #+76]
\      0xB6   0x2000        MOVS        R0,#+0
\      0xB8   0x6528        STR   R0,[R5, #+80]
\      0xBA   0xF640 0x11C4      MOVW        R1,#+2500
\      0xBE   0x65A8              STR   R0,[R5, #+88]
\      0xC0   0x6628        STR   R0,[R5, #+96]
```

```
\     0xC2   0x6569         STR    R1,[R5, #+84]
\     0xC4   0xF105 0x0048  ADD    R0,R5,#+72
\     0xC8   0x.... 0x....  BL     HAL_TIM_Base_Init
\     0xCC   0xB108         CBZ.N  R0,??main_0
\     0xCE   0x.... 0x....  BL     Error_Handler
\            ??main_0: (+1)
\     0xD2   0xF44F 0x5680  MOV    R6,#+4096
\     0xD6   0x9609         STR    R6,[SP, #+36]
\     0xD8   0xA909         ADD    R1,SP,#+36
\     0xDA   0xF105 0x0048  ADD    R0,R5,#+72
\     0xDE   0x.... 0x....  BL     HAL_TIM_ConfigClockSource
\     0xE2   0xB108         CBZ.N  R0,??main_1
\     0xE4   0x.... 0x....  BL     Error_Handler
\            ??main_1: (+1)
\     0xE8   0xF105 0x0048  ADD    R0,R5,#+72
\     0xEC   0x.... 0x....  BL     HAL_TIM_PWM_Init
\     0xF0   0xB108         CBZ.N  R0,??main_2
\     0xF2   0x.... 0x....  BL     Error_Handler
\            ??main_2: (+1)
\     0xF6   0x2100         MOVS   R1,#+0
\     0xF8   0x9100         STR    R1,[SP, #+0]
\     0xFA   0x9101         STR    R1,[SP, #+4]
\     0xFC   0xF105 0x0048  ADD    R0,R5,#+72
\     0x100  0x4669         MOV    R1,SP
\     0x102  0x.... 0x....  BL     HAL_TIMEx_MasterConfigSynchronization
\     0x106  0xB108         CBZ.N  R0,??main_3
\     0x108  0x.... 0x....  BL     Error_Handler
\            ??main_3: (+1)
\     0x10C  0x2160         MOVS   R1,#+96
\     0x10E  0x.... 0x....  BL     ?Subroutine5
\            ??CrossCallReturnLabel_11: (+1)
\     0x112  0x9206         STR    R2,[SP, #+24]
\     0x114  0xA902         ADD    R1,SP,#+8
\     0x116  0xF105 0x0048  ADD    R0,R5,#+72
\     0x11A  0x.... 0x....  BL     HAL_TIM_PWM_ConfigChannel
\     0x11E  0xB108         CBZ.N  R0,??main_4
\     0x120  0x.... 0x....  BL     Error_Handler
\            ??main_4: (+1)
\     0x124  0xF105 0x0048  ADD    R0,R5,#+72
\     0x128  0x.... 0x....  BL     HAL_TIM_MspPostInit
```

```
      111               MX_TIM5_Init();
\     0x12C   0x.... 0x....  BL      ?Subroutine3
\             ??CrossCallReturnLabel_6: (+1)
\     0x130   0x.... 0x....  BL      ?Subroutine1
\             ??CrossCallReturnLabel_1: (+1)
\     0x134   0x.... 0x....  BL      ?Subroutine2
\             ??CrossCallReturnLabel_4: (+1)
\     0x138   0xF105 0x0790  ADD   R7,R5,#+144
\     0x13C   0x.... 0x....  LDR.W        R0,??DataTable7_5
\     0x140   0x6038         STR   R0,[R7, #+0]
\     0x142   0x2100         MOVS        R1,#+0
\     0x144   0xF04F 0x30FF  MOV   R0,#+4294967295
\     0x148   0x60F8         STR   R0,[R7, #+12]
\     0x14A   0x6079         STR   R1,[R7, #+4]
\     0x14C   0x60B9         STR   R1,[R7, #+8]
\     0x14E   0x6139         STR   R1,[R7, #+16]
\     0x150   0x61B9         STR   R1,[R7, #+24]
\     0x152   0x4638         MOV   R0,R7
\     0x154   0x.... 0x....  BL      HAL_TIM_Base_Init
\     0x158   0xB108         CBZ.N       R0,??main_5
\     0x15A   0x.... 0x....  BL      Error_Handler
\             ??main_5: (+1)
\     0x15E   0x9606         STR   R6,[SP, #+24]
\     0x160   0xA906         ADD   R1,SP,#+24
\     0x162   0x4638         MOV   R0,R7
\     0x164   0x.... 0x....  BL      HAL_TIM_ConfigClockSource
\     0x168   0xB108         CBZ.N       R0,??main_6
\     0x16A   0x.... 0x....  BL      Error_Handler
\             ??main_6: (+1)
\     0x16E   0x4638         MOV   R0,R7
\     0x170   0x.... 0x....  BL      HAL_TIM_IC_Init
\     0x174   0xB108         CBZ.N       R0,??main_7
\     0x176   0x.... 0x....  BL      Error_Handler
\             ??main_7: (+1)
\     0x17A   0x2100         MOVS        R1,#+0
\     0x17C   0x9100         STR   R1,[SP, #+0]
\     0x17E   0x9101         STR   R1,[SP, #+4]
\     0x180   0x4638         MOV   R0,R7
\     0x182   0x4669         MOV   R1,SP
\     0x184   0x.... 0x....  BL      HAL_TIMEx_MasterConfigSynchronization
```

```
\     0x188   0xB108          CBZ.N       R0,??main_8
\     0x18A   0x.... 0x....   BL      Error_Handler
\             ??main_8: (+1)
\     0x18E   0x210A          MOVS        R1,#+10
\     0x190   0x9102          STR   R1,[SP, #+8]
\     0x192   0x2201          MOVS        R2,#+1
\     0x194   0x2100          MOVS        R1,#+0
\     0x196   0x9203          STR   R2,[SP, #+12]
\     0x198   0x9104          STR   R1,[SP, #+16]
\     0x19A   0x9105          STR   R1,[SP, #+20]
\     0x19C   0x2208          MOVS        R2,#+8
\     0x19E   0xA902          ADD   R1,SP,#+8
\     0x1A0   0x4638          MOV   R0,R7
\     0x1A2   0x.... 0x....   BL      HAL_TIM_IC_ConfigChannel
\     0x1A6   0xB108          CBZ.N       R0,??main_9
\     0x1A8   0x.... 0x....   BL      Error_Handler
      112             MX_TIM3_Init();
\             ??main_9: (+1)
\     0x1AC   0x.... 0x....           BL      ?Subroutine3
\             ??CrossCallReturnLabel_7: (+1)
\     0x1B0   0x.... 0x....   BL      ?Subroutine1
\             ??CrossCallReturnLabel_2: (+1)
\     0x1B4   0x.... 0x....   BL      ?Subroutine2
\             ??CrossCallReturnLabel_5: (+1)
\     0x1B8   0x....          LDR.N       R0,??DataTable7_6
\     0x1BA   0x6028          STR   R0,[R5, #+0]
\     0x1BC   0xF44F 0x7161   MOV   R1,#+900
\     0x1C0   0xF64F 0x70FF   MOVW        R0,#+65535
\     0x1C4   0x2200          MOVS        R2,#+0
\     0x1C6   0x60E8          STR   R0,[R5, #+12]
\     0x1C8   0x6069          STR   R1,[R5, #+4]
\     0x1CA   0x60AA          STR   R2,[R5, #+8]
\     0x1CC   0x612A          STR   R2,[R5, #+16]
\     0x1CE   0x61AA          STR   R2,[R5, #+24]
\     0x1D0   0x4628          MOV   R0,R5
\     0x1D2   0x.... 0x....   BL      HAL_TIM_Base_Init
\     0x1D6   0xB108          CBZ.N       R0,??main_10
\     0x1D8   0x.... 0x....   BL      Error_Handler
\             ??main_10: (+1)
\     0x1DC   0x9606          STR   R6,[SP, #+24]
```

```
\     0x1DE   0xA906            ADD   R1,SP,#+24
\     0x1E0   0x4628            MOV   R0,R5
\     0x1E2   0x.... 0x....  BL      HAL_TIM_ConfigClockSource
\     0x1E6   0xB108            CBZ.N       R0,??main_11
\     0x1E8   0x.... 0x....  BL      Error_Handler
\           ??main_11: (+1)
\     0x1EC   0x4628            MOV   R0,R5
\     0x1EE   0x.... 0x....  BL      HAL_TIM_IC_Init
\     0x1F2   0xB108            CBZ.N       R0,??main_12
\     0x1F4   0x.... 0x....  BL      Error_Handler
\           ??main_12: (+1)
\     0x1F8   0x2100            MOVS        R1,#+0
\     0x1FA   0x9100            STR   R1,[SP, #+0]
\     0x1FC   0x9101            STR   R1,[SP, #+4]
\     0x1FE   0x4628            MOV   R0,R5
\     0x200   0x4669            MOV   R1,SP
\     0x202   0x.... 0x....  BL      HAL_TIMEx_MasterConfigSynchronization
\     0x206   0xB108            CBZ.N       R0,??main_13
\     0x208   0x.... 0x....  BL      Error_Handler
\           ??main_13: (+1)
\     0x20C   0x210A            MOVS        R1,#+10
\     0x20E   0x.... 0x....  BL      ?Subroutine5
\           ??CrossCallReturnLabel_12: (+1)
\     0x212   0x9205            STR   R2,[SP, #+20]
\     0x214   0xA902            ADD   R1,SP,#+8
\     0x216   0x4628            MOV   R0,R5
\     0x218   0x.... 0x....  BL      HAL_TIM_IC_ConfigChannel
\     0x21C   0xB108            CBZ.N       R0,??main_14
\     0x21E   0x.... 0x....  BL      Error_Handler
      113             MX_SPI4_Init();
\           ??main_14: (+1)
\     0x222   0x....          LDR.N       R6,??DataTable7_7
\     0x224   0x....          LDR.N       R0,??DataTable7_8
\     0x226   0x6030            STR   R0,[R6, #+0]
\     0x228   0xF44F 0x7182     MOV   R1,#+260
\     0x22C   0x2038            MOVS        R0,#+56
\     0x22E   0x61F0            STR   R0,[R6, #+28]
\     0x230   0x6071            STR   R1,[R6, #+4]
\     0x232   0x200A            MOVS        R0,#+10
\     0x234   0x2200            MOVS        R2,#+0
```

```
\     0x236   0xF44F 0x7100     MOV   R1,#+512
\     0x23A   0x62F0            STR   R0,[R6, #+44]
\     0x23C   0x60B2            STR   R2,[R6, #+8]
\     0x23E   0x60F2            STR   R2,[R6, #+12]
\     0x240   0x6132            STR   R2,[R6, #+16]
\     0x242   0x6172            STR   R2,[R6, #+20]
\     0x244   0x61B1            STR   R1,[R6, #+24]
\     0x246   0x6232            STR   R2,[R6, #+32]
\     0x248   0x6272            STR   R2,[R6, #+36]
\     0x24A   0x62B2            STR   R2,[R6, #+40]
\     0x24C   0x4630            MOV   R0,R6
\     0x24E   0x.... 0x....   BL      HAL_SPI_Init
\     0x252   0xB108            CBZ.N         R0,??main_15
\     0x254   0x.... 0x....   BL      Error_Handler
      114             /* USER CODE BEGIN 2 */
      115
      116             HAL_SPI_Init(&hspi4);
\           ??main_15: (+1)
\     0x258   0x4630            MOV   R0,R6
\     0x25A   0x.... 0x....   BL      HAL_SPI_Init
      117             LCD_Init();
\     0x25E   0x.... 0x....   BL      LCD_Init
      118             HAL_TIM_IC_Start_IT(&htim3, TIM_CHANNEL_1);
\     0x262   0x2100            MOVS          R1,#+0
\     0x264   0x4628            MOV   R0,R5
\     0x266   0x.... 0x....   BL      HAL_TIM_IC_Start_IT
      119             HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_1);
\     0x26A   0x2100            MOVS          R1,#+0
\     0x26C   0xF105 0x0048     ADD   R0,R5,#+72
\     0x270   0x.... 0x....   BL      HAL_TIM_PWM_Start
\     0x274   0x.... 0x....   ADR.W        R5,??DataTable7
\     0x278   0x....            ADR.N         R6,?_0
\     0x27A   0x.... 0x....   LDR.W        R8,??DataTable7_9
\     0x27E   0xE017            B.N     ??main_16
      120             /* USER CODE END 2 */
      121
      122             /* Infinite loop */
      123             /* USER CODE BEGIN WHILE */
      124             while (1)
      125             {
```

```
        126             if(flag == 1){
        127             if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_1) ==
GPIO_PIN_SET) {
        128             echo_dist = echo_dist / 3;
  \             ??main_17: (+1)
  \     0x280   0x2103          MOVS    R1,#+3
  \     0x282   0xFB90 0xFAF1   SDIV  R10,R0,R1
        129             int_to_str(echo_dist);
  \     0x286   0x4650          MOV   R0,R10
  \     0x288   0x.... 0x....   BL      int_to_str
        130             for (int i = 0; i<3; i++){
  \     0x28C   0x2400          MOVS    R4,#+0
  \     0x28E   0xF8C7 0xA048   STR   R10,[R7, #+72]
        131             LCD_SendData(num_out[i]);}
  \             ??main_18: (+1)
  \     0x292   0xF818 0x0004   LDRB R0,[R8, R4]
  \     0x296   0x.... 0x....   BL      LCD_SendData
  \     0x29A   0x1C64          ADDS    R4,R4,#+1
  \     0x29C   0x2C02          CMP   R4,#+2
  \     0x29E   0xDDF8          BLE.N ??main_18
        132             LCD_Sel_Row_Col(2, 0);
  \     0x2A0   0x2100          MOVS    R1,#+0
  \     0x2A2   0x2002          MOVS    R0,#+2
  \     0x2A4   0x.... 0x....   BL      LCD_Sel_Row_Col
        133             char *inches = "inches";
        134             LCD_sendString(inches, 6);
  \     0x2A8   0x2106          MOVS    R1,#+6
  \     0x2AA   0x4630          MOV   R0,R6
  \             ??main_19: (+1)
  \     0x2AC   0x.... 0x....           BL      LCD_sendString
        135             }
  \             ??main_16: (+1)
  \     0x2B0   0x6CF8          LDR   R0,[R7, #+76]
  \     0x2B2   0x2801          CMP   R0,#+1
  \     0x2B4   0xD1FC          BNE.N         ??main_16
  \     0x2B6   0x2102          MOVS    R1,#+2
  \     0x2B8   0x4648          MOV   R0,R9
  \     0x2BA   0x.... 0x....   BL      HAL_GPIO_ReadPin
  \     0x2BE   0x2801          CMP   R0,#+1
  \     0x2C0   0x6CB8          LDR   R0,[R7, #+72]
```

```
\     0x2C2   0xD0DD            BEQ.N        ??main_17
      136             else {
      137             int_to_str(echo_dist);
\     0x2C4   0x.... 0x....  BL      int_to_str
      138             for (int i = 0; i<3; i++){
\     0x2C8   0x2400            MOVS         R4,#+0
      139             LCD_SendData(num_out[i]);}
\           ??main_20: (+1)
\     0x2CA   0xF818 0x0004    LDRB R0,[R8, R4]
\     0x2CE   0x.... 0x....          BL      LCD_SendData
\     0x2D2   0x1C64            ADDS         R4,R4,#+1
\     0x2D4   0x2C02            CMP   R4,#+2
\     0x2D6   0xDDF8            BLE.N??main_20
      140             LCD_Sel_Row_Col(2, 0);
\     0x2D8   0x2100            MOVS         R1,#+0
\     0x2DA   0x2002            MOVS         R0,#+2
\     0x2DC   0x.... 0x....          BL      LCD_Sel_Row_Col
      141             char *cm = "cm";
      142             LCD_sendString(cm, 2);
\     0x2E0   0x2102            MOVS         R1,#+2
\     0x2E2   0x4628            MOV   R0,R5
\     0x2E4   0xE7E2            B.N    ??main_19
      143             }
      144             }
      145             /* USER CODE END WHILE */
      146
      147             /* USER CODE BEGIN 3 */
      148             }
      149             /* USER CODE END 3 */
      150             }
      151
      152             /**
      153             * @brief System Clock Configuration
      154             * @retval None
      155             */

\                             In section .text, align 2, keep-with-next
\           ?Subroutine3: (+1)
\     0x0   0x2210          MOVS         R2,#+16
\           ??Subroutine3_0: (+1)
```

```
  \      0x2   0x2100          MOVS          R1,#+0
  \      0x4   0xA806          ADD   R0,SP,#+24
  \      0x6   0x.... 0x....   B.W   memset

  \                            In section .text, align 2, keep-with-next
         156                   void SystemClock_Config(void)
         157                   {
  \                SystemClock_Config: (+1)
  \      0x0   0xB580          PUSH          {R7,LR}
  \      0x2   0xB092          SUB   SP,SP,#+72
  \      0x4   0x2230          MOVS          R2,#+48
  \      0x6   0x.... 0x....   BL      ??Subroutine3_0
  \                ??CrossCallReturnLabel_8: (+1)
  \      0xA   0x.... 0x....   BL      ?Subroutine4
         158                   RCC_OscInitTypeDef RCC_OscInitStruct = {0};
         159                   RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
         160
         161                   /** Configure the main internal regulator output voltage
         162                   */
         163                   __HAL_RCC_PWR_CLK_ENABLE();
  \                ??CrossCallReturnLabel_10: (+1)
  \      0xE   0x2000          MOVS          R0,#+0
  \      0x10   0x9000         STR   R0,[SP, #+0]
         164
__HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1
);
         165
         166                   /** Initializes the RCC Oscillators according to the specified
parameters
         167                   * in the RCC_OscInitTypeDef structure.
         168                   */
         169                   RCC_OscInitStruct.OscillatorType =
RCC_OSCILLATORTYPE_HSE;
         170                   RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  \      0x12   0xF44F 0x3380       MOV   R3,#+65536
  \      0x16   0x....              LDR.N         R0,??DataTable7_10
  \      0x18   0x6801         LDR   R1,[R0, #+0]
  \      0x1A   0xF041 0x5180       ORR   R1,R1,#0x10000000
  \      0x1E   0x6001         STR   R1,[R0, #+0]
  \      0x20   0x2100         MOVS          R1,#+0
```

```
\     0x22   0x6800        LDR    R0,[R0, #+0]
\     0x24   0xF000 0x5080      AND    R0,R0,#0x10000000
\     0x28   0x9000        STR    R0,[SP, #+0]
\     0x2A   0x9800        LDR    R0,[SP, #+0]
\     0x2C   0x....        LDR.N          R0,??DataTable7_11
\     0x2E   0x9100        STR    R1,[SP, #+0]
\     0x30   0x6802        LDR    R2,[R0, #+0]
\     0x32   0xF442 0x4240      ORR    R2,R2,#0xC000
\     0x36   0x6002        STR    R2,[R0, #+0]
\     0x38   0x2201        MOVS           R2,#+1
\     0x3A   0x6800        LDR    R0,[R0, #+0]
\     0x3C   0xF400 0x4040      AND    R0,R0,#0xC000
\     0x40   0x9000        STR    R0,[SP, #+0]
      171              RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
      172              RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
      173              RCC_OscInitStruct.PLL.PLLM = 4;
      174              RCC_OscInitStruct.PLL.PLLN = 180;
      175              RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
      176              RCC_OscInitStruct.PLL.PLLQ = 4;
      177              if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
\     0x42   0xA806        ADD    R0,SP,#+24
\     0x44   0x9900        LDR    R1,[SP, #+0]
\     0x46   0x9206        STR    R2,[SP, #+24]
\     0x48   0x2102        MOVS           R1,#+2
\     0x4A   0xF44F 0x0280      MOV    R2,#+4194304
\     0x4E   0x910C                STR    R1,[SP, #+48]
\     0x50   0x920D        STR    R2,[SP, #+52]
\     0x52   0x2104        MOVS           R1,#+4
\     0x54   0x22B4        MOVS           R2,#+180
\     0x56   0x910E        STR    R1,[SP, #+56]
\     0x58   0x920F        STR    R2,[SP, #+60]
\     0x5A   0x2102        MOVS           R1,#+2
\     0x5C   0x2204        MOVS           R2,#+4
\     0x5E   0x9307        STR    R3,[SP, #+28]
\     0x60   0x9110        STR    R1,[SP, #+64]
\     0x62   0x9211        STR    R2,[SP, #+68]
\     0x64   0x.... 0x....      BL     HAL_RCC_OscConfig
\     0x68   0xB108        CBZ.N          R0,??SystemClock_Config_0
      178              {
      179              Error_Handler();
```

```
  \      0x6A  0x.... 0x....    BL      Error_Handler
         180             }
         181
         182             /** Activate the Over-Drive mode
         183             */
         184             if (HAL_PWREx_EnableOverDrive() != HAL_OK)
  \              ??SystemClock_Config_0: (+1)
  \      0x6E  0x.... 0x....    BL      HAL_PWREx_EnableOverDrive
  \      0x72  0xB108           CBZ.N           R0,??SystemClock_Config_1
         185             {
         186             Error_Handler();
  \      0x74  0xB672           CPSID           I
  \              ??SystemClock_Config_2: (+1)
  \      0x76  0xE7FE                   B.N     ??SystemClock_Config_2
         187             }
         188
         189             /** Initializes the CPU, AHB and APB buses clocks
         190             */
         191             RCC_ClkInitStruct.ClockType =
RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
         192
|RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
         193             RCC_ClkInitStruct.SYSCLKSource =
RCC_SYSCLKSOURCE_PLLCLK;
  \              ??SystemClock_Config_1: (+1)
  \      0x78  0x2102           MOVS            R1,#+2
  \      0x7A  0x9102           STR     R1,[SP, #+8]
         194             RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  \      0x7C  0x2200           MOVS            R2,#+0
  \      0x7E  0x9203           STR     R2,[SP, #+12]
  \      0x80  0x200F           MOVS            R0,#+15
         195             RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV4;
  \      0x82  0xF44F 0x51A0    MOV     R1,#+5120
  \      0x86  0x9001           STR     R0,[SP, #+4]
  \      0x88  0x9104           STR     R1,[SP, #+16]
         196             RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV2;
  \      0x8A  0xF44F 0x5280    MOV     R2,#+4096
  \      0x8E  0x9205           STR     R2,[SP, #+20]
         197
```

```
        198             if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct,
FLASH_LATENCY_5) != HAL_OK)
  \     0x90  0x2105        MOVS        R1,#+5
  \     0x92  0xA801        ADD   R0,SP,#+4
  \     0x94  0x.... 0x....     BL      HAL_RCC_ClockConfig
  \     0x98  0xB108        CBZ.N       R0,??SystemClock_Config_3
        199             {
        200             Error_Handler();
  \     0x9A  0xB672                CPSID       I
  \             ??SystemClock_Config_4: (+1)
  \     0x9C  0xE7FE                B.N     ??SystemClock_Config_4
        201             }
        202             }
  \             ??SystemClock_Config_3: (+1)
  \     0x9E  0xB013                ADD   SP,SP,#+76
  \     0xA0  0xBD00                POP   {PC}

  \                             In section .text, align 2, keep-with-next
  \             ?Subroutine4: (+1)
  \     0x0   0x2214        MOVS        R2,#+20
  \     0x2   0x2100        MOVS        R1,#+0
  \     0x4   0xA801        ADD   R0,SP,#+4
  \     0x6   0x.... 0x....     B.W     memset
        203
        204             /**
        205             * @brief SPI4 Initialization Function
        206             * @param None
        207             * @retval None
        208             */
        209             static void MX_SPI4_Init(void)
        210             {
        211
        212             /* USER CODE BEGIN SPI4_Init 0 */
        213
        214             /* USER CODE END SPI4_Init 0 */
        215
        216             /* USER CODE BEGIN SPI4_Init 1 */
        217
        218             /* USER CODE END SPI4_Init 1 */
        219             /* SPI4 parameter configuration*/
```

```
220         hspi4.Instance = SPI4;
221         hspi4.Init.Mode = SPI_MODE_MASTER;
222         hspi4.Init.Direction = SPI_DIRECTION_2LINES;
223         hspi4.Init.DataSize = SPI_DATASIZE_8BIT;
224         hspi4.Init.CLKPolarity = SPI_POLARITY_LOW;
225         hspi4.Init.CLKPhase = SPI_PHASE_1EDGE;
226         hspi4.Init.NSS = SPI_NSS_SOFT;
227         hspi4.Init.BaudRatePrescaler =
SPI_BAUDRATEPRESCALER_256;
228         hspi4.Init.FirstBit = SPI_FIRSTBIT_MSB;
229         hspi4.Init.TIMode = SPI_TIMODE_DISABLE;
230         hspi4.Init.CRCCalculation = SPI_CRCCALCULATION_DISABLE;
231         hspi4.Init.CRCPolynomial = 10;
232         if (HAL_SPI_Init(&hspi4) != HAL_OK)
233         {
234         Error_Handler();
235         }
236         /* USER CODE BEGIN SPI4_Init 2 */
237
238         /* USER CODE END SPI4_Init 2 */
239
240         }
241
242         /**
243         * @brief TIM3 Initialization Function
244         * @param None
245         * @retval None
246         */
247         static void MX_TIM3_Init(void)
248         {
249
250         /* USER CODE BEGIN TIM3_Init 0 */
251
252         /* USER CODE END TIM3_Init 0 */
253
254         TIM_ClockConfigTypeDef sClockSourceConfig = {0};
255         TIM_MasterConfigTypeDef sMasterConfig = {0};
256         TIM_IC_InitTypeDef sConfigIC = {0};
257
258         /* USER CODE BEGIN TIM3_Init 1 */
```

```
259
260         /* USER CODE END TIM3_Init 1 */
261         htim3.Instance = TIM3;
262         htim3.Init.Prescaler = 900;
263         htim3.Init.CounterMode = TIM_COUNTERMODE_UP;
264         htim3.Init.Period = 65535;
265         htim3.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
266         htim3.Init.AutoReloadPreload =
TIM_AUTORELOAD_PRELOAD_DISABLE;
267         if (HAL_TIM_Base_Init(&htim3) != HAL_OK)
268         {
269         Error_Handler();
270         }
271         sClockSourceConfig.ClockSource =
TIM_CLOCKSOURCE_INTERNAL;
272         if (HAL_TIM_ConfigClockSource(&htim3, &sClockSourceConfig) !=
HAL_OK)
273         {
274         Error_Handler();
275         }
276         if (HAL_TIM_IC_Init(&htim3) != HAL_OK)
277         {
278         Error_Handler();
279         }
280         sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
281         sMasterConfig.MasterSlaveMode =
TIM_MASTERSLAVEMODE_DISABLE;
282         if (HAL_TIMEx_MasterConfigSynchronization(&htim3,
&sMasterConfig) != HAL_OK)
283         {
284         Error_Handler();
285         }
286         sConfigIC.ICPolarity =
TIM_INPUTCHANNELPOLARITY_BOTHEDGE;
287         sConfigIC.ICSelection = TIM_ICSELECTION_DIRECTTI;
288         sConfigIC.ICPrescaler = TIM_ICPSC_DIV1;
289         sConfigIC.ICFilter = 0;
290         if (HAL_TIM_IC_ConfigChannel(&htim3, &sConfigIC,
TIM_CHANNEL_1) != HAL_OK)
291         {
```

```
292             Error_Handler();
293             }
294             /* USER CODE BEGIN TIM3_Init 2 */
295
296             /* USER CODE END TIM3_Init 2 */
297
298             }
299
300             /**
301             * @brief TIM4 Initialization Function
302             * @param None
303             * @retval None
304             */
305             static void MX_TIM4_Init(void)
306             {
307
308             /* USER CODE BEGIN TIM4_Init 0 */
309
310             /* USER CODE END TIM4_Init 0 */
311
312             TIM_ClockConfigTypeDef sClockSourceConfig = {0};
313             TIM_MasterConfigTypeDef sMasterConfig = {0};
314             TIM_OC_InitTypeDef sConfigOC = {0};
315
316             /* USER CODE BEGIN TIM4_Init 1 */
317
318             /* USER CODE END TIM4_Init 1 */
319             htim4.Instance = TIM4;
320             htim4.Init.Prescaler = 1800;
321             htim4.Init.CounterMode = TIM_COUNTERMODE_UP;
322             htim4.Init.Period = 2500;
323             htim4.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
324             htim4.Init.AutoReloadPreload =
TIM_AUTORELOAD_PRELOAD_DISABLE;
325             if (HAL_TIM_Base_Init(&htim4) != HAL_OK)
326             {
327             Error_Handler();
328             }
329             sClockSourceConfig.ClockSource =
TIM_CLOCKSOURCE_INTERNAL;
```

```
330         if (HAL_TIM_ConfigClockSource(&htim4, &sClockSourceConfig) !=
HAL_OK)
331         {
332         Error_Handler();
333         }
334         if (HAL_TIM_PWM_Init(&htim4) != HAL_OK)
335         {
336         Error_Handler();
337         }
338         sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
339         sMasterConfig.MasterSlaveMode =
TIM_MASTERSLAVEMODE_DISABLE;
340         if (HAL_TIMEx_MasterConfigSynchronization(&htim4,
&sMasterConfig) != HAL_OK)
341         {
342         Error_Handler();
343         }
344         sConfigOC.OCMode = TIM_OCMODE_PWM1;
345         sConfigOC.Pulse = 1;
346         sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
347         sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
348         if (HAL_TIM_PWM_ConfigChannel(&htim4, &sConfigOC,
TIM_CHANNEL_1) != HAL_OK)
349         {
350         Error_Handler();
351         }
352         /* USER CODE BEGIN TIM4_Init 2 */
353
354         /* USER CODE END TIM4_Init 2 */
355         HAL_TIM_MspPostInit(&htim4);
356
357         }
358
359         /**
360         * @brief TIM5 Initialization Function
361         * @param None
362         * @retval None
363         */
364         static void MX_TIM5_Init(void)
365         {
```

```
366
367         /* USER CODE BEGIN TIM5_Init 0 */
368
369         /* USER CODE END TIM5_Init 0 */
370
371         TIM_ClockConfigTypeDef sClockSourceConfig = {0};
372         TIM_MasterConfigTypeDef sMasterConfig = {0};
373         TIM_IC_InitTypeDef sConfigIC = {0};
374
375         /* USER CODE BEGIN TIM5_Init 1 */
376
377         /* USER CODE END TIM5_Init 1 */
378         htim5.Instance = TIM5;
379         htim5.Init.Prescaler = 0;
380         htim5.Init.CounterMode = TIM_COUNTERMODE_UP;
381         htim5.Init.Period = 4294967295;
382         htim5.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
383         htim5.Init.AutoReloadPreload =
TIM_AUTORELOAD_PRELOAD_DISABLE;
384         if (HAL_TIM_Base_Init(&htim5) != HAL_OK)
385         {
386         Error_Handler();
387         }
388         sClockSourceConfig.ClockSource =
TIM_CLOCKSOURCE_INTERNAL;
389         if (HAL_TIM_ConfigClockSource(&htim5, &sClockSourceConfig) !=
HAL_OK)
390         {
391         Error_Handler();
392         }
393         if (HAL_TIM_IC_Init(&htim5) != HAL_OK)
394         {
395         Error_Handler();
396         }
397         sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
398         sMasterConfig.MasterSlaveMode =
TIM_MASTERSLAVEMODE_DISABLE;
399         if (HAL_TIMEx_MasterConfigSynchronization(&htim5,
&sMasterConfig) != HAL_OK)
400         {
```

```
401             Error_Handler();
402             }
403             sConfigIC.ICPolarity =
TIM_INPUTCHANNELPOLARITY_BOTHEDGE;
404             sConfigIC.ICSelection = TIM_ICSELECTION_DIRECTTI;
405             sConfigIC.ICPrescaler = TIM_ICPSC_DIV1;
406             sConfigIC.ICFilter = 0;
407             if (HAL_TIM_IC_ConfigChannel(&htim5, &sConfigIC,
TIM_CHANNEL_3) != HAL_OK)
408             {
409             Error_Handler();
410             }
411             /* USER CODE BEGIN TIM5_Init 2 */
412
413             /* USER CODE END TIM5_Init 2 */
414
415             }
416
417             /**
418             * @brief GPIO Initialization Function
419             * @param None
420             * @retval None
421             */
422             static void MX_GPIO_Init(void)
423             {
424             GPIO_InitTypeDef GPIO_InitStruct = {0};
425
426             /* GPIO Ports Clock Enable */
427             __HAL_RCC_GPIOE_CLK_ENABLE();
428             __HAL_RCC_GPIOH_CLK_ENABLE();
429             __HAL_RCC_GPIOC_CLK_ENABLE();
430             __HAL_RCC_GPIOA_CLK_ENABLE();
431             __HAL_RCC_GPIOD_CLK_ENABLE();
432
433             /*Configure GPIO pin : PC1 */
434             GPIO_InitStruct.Pin = GPIO_PIN_1;
435             GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
436             GPIO_InitStruct.Pull = GPIO_NOPULL;
437             HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);
438
```

```
      439            }
      440
      441            /* USER CODE BEGIN 4 */

\                            In section .text, align 2, keep-with-next
      442            void LCD_SendCommand (uint8_t cmd){
\            LCD_SendCommand: (+1)
\     0x0  0xB538          PUSH         {R3-R5,LR}
      443                   data[0] = 0xF8;
\     0x2  0x....          LDR.N        R4,??DataTable7_12
\     0x4  0x21F8          MOVS         R1,#+248
      444                   data[1] = (cmd & 0xF0);
\     0x6  0x....          B.N    ?Subroutine0
      445                   data[2] = ((cmd << 4) & 0xF0);
      446                   HAL_GPIO_WritePin(GPIOD, GPIO_PIN_0,
GPIO_PIN_SET); // CS to high
      447            HAL_SPI_Transmit(&hspi4, data, 3, 1);
      448                   HAL_GPIO_WritePin(GPIOD, GPIO_PIN_0,
GPIO_PIN_RESET); // CS to low
      449            }

\                            In section .text, align 2, keep-with-next
\            ?Subroutine0: (+1)
\     0x0  0xF000 0x02F0        AND   R2,R0,#0xF0
\     0x4  0x0100          LSLS  R0,R0,#+4
\     0x6  0x7021          STRB R1,[R4, #+0]
\     0x8  0x7062          STRB R2,[R4, #+1]
\     0xA  0x70A0          STRB R0,[R4, #+2]
\     0xC  0x2201          MOVS         R2,#+1
\     0xE  0x....          LDR.N        R5,??DataTable7_13
\     0x10   0x2101        MOVS         R1,#+1
\     0x12   0x4628        MOV   R0,R5
\     0x14   0x.... 0x....     BL      HAL_GPIO_WritePin
\     0x18   0x2301        MOVS         R3,#+1
\     0x1A   0x2203        MOVS         R2,#+3
\     0x1C   0x4621        MOV   R1,R4
\     0x1E   0x....        LDR.N        R0,??DataTable7_7
\     0x20   0x.... 0x....     BL      HAL_SPI_Transmit
\     0x24   0x4628        MOV   R0,R5
\     0x26   0xE8BD 0x4038      POP   {R3-R5,LR}
```

```
\      0x2A   0x2200         MOVS          R2,#+0
\      0x2C   0x2101         MOVS          R1,#+1
\      0x2E   0x.... 0x....  B.W    HAL_GPIO_WritePin
       450

\                            In section .text, align 2, keep-with-next
       451              void LCD_Init() {
\               LCD_Init: (+1)
\      0x0    0xB510         PUSH          {R4,LR}
       452              HAL_GPIO_WritePin(GPIOD, GPIO_PIN_1, GPIO_PIN_RESET); //
Make sure reset is low
\      0x2    0x....         LDR.N         R4,??DataTable7_13
\      0x4    0x2200         MOVS          R2,#+0
\      0x6    0x2102         MOVS          R1,#+2
\      0x8    0x4620         MOV   R0,R4
\      0xA    0x.... 0x....  BL      HAL_GPIO_WritePin
       453              HAL_Delay(100);
\      0xE    0x2064         MOVS          R0,#+100
\      0x10   0x.... 0x....  BL      HAL_Delay
       454              HAL_GPIO_WritePin(GPIOD, GPIO_PIN_1, GPIO_PIN_SET); //
Reset Low to high
\      0x14   0x2201         MOVS          R2,#+1
\      0x16   0x2102         MOVS          R1,#+2
\      0x18   0x4620         MOV   R0,R4
\      0x1A   0x.... 0x....  BL      HAL_GPIO_WritePin
       455              HAL_Delay(50); // greater than 40ms
\      0x1E   0x2032         MOVS          R0,#+50
\      0x20   0x.... 0x....  BL      HAL_Delay
       456              LCD_SendCommand(Function_Set);
\      0x24   0x2030         MOVS          R0,#+48
\      0x26   0x.... 0x....  BL      LCD_SendCommand
       457              HAL_Delay(10); // greater than 100us
\      0x2A   0x200A                MOVS          R0,#+10
\      0x2C   0x.... 0x....  BL      HAL_Delay
       458              LCD_SendCommand(Function_Set);
\      0x30   0x2030         MOVS          R0,#+48
\      0x32   0x.... 0x....  BL      LCD_SendCommand
       459              HAL_Delay(10); // greater than 37us
\      0x36   0x200A         MOVS          R0,#+10
\      0x38   0x.... 0x....  BL      HAL_Delay
```

```
      460               LCD_SendCommand(Display_On);
\     0x3C   0x200C                MOVS          R0,#+12
\     0x3E   0x.... 0x....     BL       LCD_SendCommand
      461               HAL_Delay(10); // greater than 100us
\     0x42   0x200A          MOVS          R0,#+10
\     0x44   0x.... 0x....     BL       HAL_Delay
      462               LCD_SendCommand(Display_Clear);
\     0x48   0x2001          MOVS          R0,#+1
\     0x4A   0x.... 0x....     BL       LCD_SendCommand
      463               HAL_Delay(15); // greater than 10ms
\     0x4E   0x200F          MOVS          R0,#+15
\     0x50   0x.... 0x....     BL       HAL_Delay
      464               LCD_SendCommand(Entry_Mode_Set);
\     0x54   0x2006          MOVS          R0,#+6
\     0x56   0x.... 0x....     BL       LCD_SendCommand
      465               HAL_Delay(10); // greater than 100us
\     0x5A   0xE8BD 0x4010     POP    {R4,LR}
\     0x5E   0x200A                MOVS          R0,#+10
\     0x60   0x.... 0x....     B.W    HAL_Delay
      466               }
      467

\                               In section .text, align 2, keep-with-next
      468               void LCD_Clear(){
      469               LCD_SendCommand(Display_Clear);
\               LCD_Clear: (+1)
\     0x0    0x2001          MOVS          R0,#+1
\     0x2    0x....               B.N    LCD_SendCommand
      470               }
      471

\                               In section .text, align 2, keep-with-next
      472               void LCD_SendData (uint8_t ascii){
\               LCD_SendData: (+1)
\     0x0    0xB538          PUSH          {R3-R5,LR}
      473                        data[0] = 0xFA;
\     0x2    0x....               LDR.N         R4,??DataTable7_12
\     0x4    0x21FA          MOVS          R1,#+250
      474                        data[1] = (ascii & 0xF0);
\     0x6                        REQUIRE ?Subroutine0
```

```
\      0x6                     ;; // Fall through to label ?Subroutine0
       475                      data[2] = ((ascii << 4) & 0xF0);
       476                      HAL_GPIO_WritePin(GPIOD, GPIO_PIN_0,
GPIO_PIN_SET); // CS to high
       477              HAL_SPI_Transmit(&hspi4, data, 3, 1);
       478                      HAL_GPIO_WritePin(GPIOD, GPIO_PIN_0,
GPIO_PIN_RESET); // CS to low
       479              }
       480

\                               In section .text, align 2, keep-with-next
       481              void LCD_sendString(char* in, int length)
       482              {
\               LCD_sendString: (+1)
\      0x0   0xB570          PUSH          {R4-R6,LR}
\      0x2   0x4604          MOV   R4,R0
\      0x4   0x460D          MOV   R5,R1
\      0x6   0xE003          B.N     ??LCD_sendString_0
       483              uint8_t letter;
       484              for (int i; i<length; i++) //iterate through the input string
       485              {
       486              letter = in[i]; // assign the letter to the int, getting its ascii value, this
may be unnecessary
       487              LCD_SendData(letter); // send the data to the board
\               ??LCD_sendString_1: (+1)
\      0x8   0x5DA0          LDRB R0,[R4, R6]
\      0xA   0x.... 0x....       BL      LCD_SendData
       488              }
\      0xE   0x1C76          ADDS          R6,R6,#+1
\               ??LCD_sendString_0: (+1)
\      0x10   0x42AE                 CMP   R6,R5
\      0x12   0xDBF9                 BLT.N ??LCD_sendString_1
       489              }
\      0x14   0xBD70                 POP   {R4-R6,PC}
       490

\                               In section .text, align 4, keep-with-next
       491              void LCD_Sel_Row_Col(uint8_t row, uint8_t col)
       492              {
\               LCD_Sel_Row_Col: (+1)
```

```
\     0x0   0xB580          PUSH         {R7,LR}
      493             switch(row)
\     0x2   0x084A          LSRS R2,R1,#+1
\     0x4   0x2803          CMP  R0,#+3
\     0x6   0xD80E          BHI.N ??LCD_Sel_Row_Col_1
\     0x8   0xE8DF 0xF000       TBB    [PC, R0]
\             ??LCD_Sel_Row_Col_0:
\     0xC   0x02 0x05       DC8   0x2,0x5,0x8,0xB

\             0x08 0x0B
      494                     {
      495                     case 0:
      496                             col /= 2; //cursor is 2 spaces wide
      497                             col |= 0x80; // 0b10000000 first row
\             ??LCD_Sel_Row_Col_2: (+1)
\     0x10  0xF042 0x0180       ORR   R1,R2,#0x80
      498                     break;
\     0x14  0xE007          B.N    ??LCD_Sel_Row_Col_1
      499             case 1:
      500                     col /= 2;
      501                     col |= 0x90; //second column starts at 0x90
\             ??LCD_Sel_Row_Col_3: (+1)
\     0x16  0xF042 0x0190       ORR   R1,R2,#0x90
      502                     break;
\     0x1A  0xE004                  B.N    ??LCD_Sel_Row_Col_1
      503             case 2:
      504                     col /= 2;
      505                     col |= 0x88; // third row is 0x88
\             ??LCD_Sel_Row_Col_4: (+1)
\     0x1C  0xF042 0x0188       ORR   R1,R2,#0x88
      506                     break;
\     0x20  0xE001          B.N    ??LCD_Sel_Row_Col_1
      507             case 3:
      508                     col /= 2;
      509                     col |= 0x98; //fourth row is 0x98
\             ??LCD_Sel_Row_Col_5: (+1)
\     0x22  0xF042 0x0198       ORR   R1,R2,#0x98
      510                     break;
      511             }
      512             LCD_SendCommand(col);
```

```
\              ??LCD_Sel_Row_Col_1: (+1)
\      0x26   0x4608         MOV   R0,R1
\      0x28   0x.... 0x....  BL      LCD_SendCommand
       513              HAL_Delay(5);
\      0x2C   0xE8BD 0x4002  POP   {R1,LR}
\      0x30   0x2005         MOVS          R0,#+5
\      0x32   0x.... 0x....  B.W    HAL_Delay
       514              }
       515

\                            In section .text, align 2, keep-with-next
       516              void int_to_str(int number) {
\              int_to_str: (+1)
\      0x0    0xB510         PUSH          {R4,LR}
       517              int Hex_offset = 0x30;
       518              int first_dig = (number % 1000)/100;  //separate the hundreds digit
       519              first_dig = first_dig + Hex_offset;
       520              num_out[0] = first_dig;
\      0x2    0xF44F 0x737A         MOV   R3,#+1000
\      0x6    0xFB90 0xF3F3         SDIV  R3,R0,R3
\      0xA    0xF44F 0x717A         MOV   R1,#+1000
\      0xE    0xFB01 0x0313         MLS   R3,R1,R3,R0
\      0x12   0x2464         MOVS          R4,#+100
\      0x14   0x....         LDR.N         R2,??DataTable7_9
\      0x16   0xFB93 0xF4F4  SDIV  R4,R3,R4
\      0x1A   0x3430         ADDS          R4,R4,#+48
\      0x1C   0x7014         STRB R4,[R2, #+0]
       521              int second_dig = (number % 100)/10 ; //separate the tens digit
       522              second_dig = second_dig + Hex_offset;
       523              num_out[1] = second_dig;
\      0x1E   0x2364         MOVS          R3,#+100
\      0x20   0xFB90 0xF3F3  SDIV  R3,R0,R3
\      0x24   0x2164         MOVS          R1,#+100
\      0x26   0xFB01 0x0313  MLS   R3,R1,R3,R0
\      0x2A   0x240A                 MOVS          R4,#+10
\      0x2C   0xFB93 0xF4F4  SDIV  R4,R3,R4
       524              int third_dig = number % 10;  // separate the ones place
       525              third_dig = third_dig + Hex_offset;
       526              num_out[2] = third_dig;
\      0x30   0x230A         MOVS          R3,#+10
```

```
\      0x32   0xFB90 0xF3F3      SDIV  R3,R0,R3
\      0x36   0x210A        MOVS        R1,#+10
\      0x38   0x3430        ADDS        R4,R4,#+48
\      0x3A   0xFB01 0x0013      MLS   R0,R1,R3,R0
\      0x3E   0x7054        STRB R4,[R2, #+1]
\      0x40   0x3030        ADDS        R0,R0,#+48
\      0x42   0x7090        STRB R0,[R2, #+2]
       527            }
\      0x44   0xBD10              POP   {R4,PC}
       528            /* USER CODE END 4 */
       529
       530            /**
       531            * @brief  This function is executed in case of error occurrence.
       532            * @retval None
       533            */

\                             In section .text, align 2, keep-with-next
       534            void Error_Handler(void)
       535            {
       536            /* USER CODE BEGIN Error_Handler_Debug */
       537            /* User can add his own implementation to report the HAL error
return state */
       538            __disable_irq();
\             Error_Handler: (+1)
\      0x0   0xB672        CPSID       I
       539            while (1)
\             ??Error_Handler_0: (+1)
\      0x2   0xE7FE        B.N   ??Error_Handler_0
       540            {
       541            }
       542            /* USER CODE END Error_Handler_Debug */
       543            }

\                             In section .text, align 2, keep-with-next
\             ?Subroutine5: (+1)
\      0x0   0x9102        STR   R1,[SP, #+8]
\      0x2   0x2201        MOVS        R2,#+1
\      0x4   0x9203        STR   R2,[SP, #+12]
\      0x6   0x2100        MOVS        R1,#+0
\      0x8   0x9104        STR   R1,[SP, #+16]
```

```
\      0xA   0x2200          MOVS          R2,#+0
\      0xC   0x4770          BX      LR

\                            In section .text, align 2, keep-with-next
\              ?Subroutine2: (+1)
\      0x0   0x2210          MOVS          R2,#+16
\              ??Subroutine2_0: (+1)
\      0x2   0x2100          MOVS          R1,#+0
\      0x4   0xA802          ADD   R0,SP,#+8
\      0x6   0x.... 0x....   B.W     memset

\                            In section .text, align 2, keep-with-next
\              ?Subroutine1: (+1)
\      0x0   0x2208          MOVS          R2,#+8
\      0x2   0x2100          MOVS          R1,#+0
\      0x4   0x4668          MOV   R0,SP
\      0x6   0x.... 0x....   B.W     memset

\                            In section .text, align 4, keep-with-next
\              ??DataTable7:
\      0x0   0x63 0x6D       DC8     0x63, 0x6D, 0x00, 0x00

\              0x00 0x00

\                            In section .text, align 4, keep-with-next
\              ??DataTable7_1:
\      0x0   0x4002'0800             DC32  0x40020800

\                            In section .text, align 4, keep-with-next
\              ??DataTable7_2:
\      0x0   0x4002'3830             DC32  0x40023830

\                            In section .text, align 4, keep-with-next
\              ??DataTable7_3:
\      0x0   0x....'....     DC32  htim3

\                            In section .text, align 4, keep-with-next
\              ??DataTable7_4:
\      0x0   0x4000'0800             DC32  0x40000800
```

```
\                               In section .text, align 4, keep-with-next
\              ??DataTable7_5:
\      0x0   0x4000'0C00            DC32  0x40000c00

\                               In section .text, align 4, keep-with-next
\              ??DataTable7_6:
\      0x0   0x4000'0400            DC32  0x40000400

\                               In section .text, align 4, keep-with-next
\              ??DataTable7_7:
\      0x0   0x....'....            DC32  hspi4

\                               In section .text, align 4, keep-with-next
\              ??DataTable7_8:
\      0x0   0x4001'3400            DC32  0x40013400

\                               In section .text, align 4, keep-with-next
\              ??DataTable7_9:
\      0x0   0x....'....            DC32  num_out

\                               In section .text, align 4, keep-with-next
\              ??DataTable7_10:
\      0x0   0x4002'3840            DC32  0x40023840

\                               In section .text, align 4, keep-with-next
\              ??DataTable7_11:
\      0x0   0x4000'7000            DC32  0x40007000

\                               In section .text, align 4, keep-with-next
\              ??DataTable7_12:
\      0x0   0x....'....            DC32  `data`

\                               In section .text, align 4, keep-with-next
\              ??DataTable7_13:
\      0x0   0x4002'0C00            DC32  0x40020c00

\                               In section .rodata, align 4, keep-with-next
\              ?_1:
\      0x0   0x63 0x6D      DC8 "cm"
```

```
\              0x00
\      0x3                     DS8 1

\                              In section .text, align 4, keep-with-next
\              ?_0:
\      0x0   0x69 0x6E         DC8 "inches"

\              0x63 0x68

\              0x65 0x73

\              0x00
\      0x7                     DS8 1
       544
       545             #ifdef  USE_FULL_ASSERT
       546             /**
       547             * @brief  Reports the name of the source file and the source line
number
       548             *         where the assert_param error has occurred.
       549             * @param  file: pointer to the source file name
       550             * @param  line: assert_param error line source number
       551             * @retval None
       552             */
       553             void assert_failed(uint8_t *file, uint32_t line)
       554             {
       555             /* USER CODE BEGIN 6 */
       556             /* User can add his own implementation to report the file name and
line number,
       557             ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line)
*/
       558             /* USER CODE END 6 */
       559             }
       560             #endif /* USE_FULL_ASSERT */

  Maximum stack usage in bytes:

  .cstack Function
  ------- --------
       0   Error_Handler
       0   LCD_Clear
```

0 -> LCD_SendCommand
8 LCD_Init
0 -> HAL_Delay
8 -> HAL_Delay
8 -> HAL_GPIO_WritePin
8 -> LCD_SendCommand
8 LCD_Sel_Row_Col
0 -> HAL_Delay
8 -> LCD_SendCommand
16 LCD_SendCommand
0 -> HAL_GPIO_WritePin
16 -> HAL_GPIO_WritePin
16 -> HAL_SPI_Transmit
16 LCD_SendData
0 -> HAL_GPIO_WritePin
16 -> HAL_GPIO_WritePin
16 -> HAL_SPI_Transmit
16 LCD_sendString
16 -> LCD_SendData
80 SystemClock_Config
80 -> Error_Handler
80 -> HAL_PWREx_EnableOverDrive
80 -> HAL_RCC_ClockConfig
80 -> HAL_RCC_OscConfig
80 -> memset
8 int_to_str
88 main
88 -> Error_Handler
88 -> HAL_GPIO_Init
88 -> HAL_GPIO_ReadPin
88 -> HAL_Init
88 -> HAL_SPI_Init
88 -> HAL_TIMEx_MasterConfigSynchronization
88 -> HAL_TIM_Base_Init
88 -> HAL_TIM_ConfigClockSource
88 -> HAL_TIM_IC_ConfigChannel
88 -> HAL_TIM_IC_Init
88 -> HAL_TIM_IC_Start_IT
88 -> HAL_TIM_MspPostInit
88 -> HAL_TIM_PWM_ConfigChannel

```
      88   -> HAL_TIM_PWM_Init
      88   -> HAL_TIM_PWM_Start
      88   -> LCD_Init
      88   -> LCD_Sel_Row_Col
      88   -> LCD_SendData
      88   -> LCD_sendString
      88   -> SystemClock_Config
      88   -> int_to_str
      88   -> memset


Section sizes:

Bytes  Function/Label
-----  --------------
       4  ??DataTable7
       4  ??DataTable7_1
       4  ??DataTable7_10
       4  ??DataTable7_11
       4  ??DataTable7_12
       4  ??DataTable7_13
       4  ??DataTable7_2
       4  ??DataTable7_3
       4  ??DataTable7_4
       4  ??DataTable7_5
       4  ??DataTable7_6
       4  ??DataTable7_7
       4  ??DataTable7_8
       4  ??DataTable7_9
      50  ?Subroutine0
      10  ?Subroutine1
      10  ?Subroutine2
      10  ?Subroutine3
      10  ?Subroutine4
      14  ?Subroutine5
       8  ?_0
       4  ?_1
       4  Error_Handler
       4  LCD_Clear
     100  LCD_Init
```

```
        54  LCD_Sel_Row_Col
        8  LCD_SendCommand
        6  LCD_SendData
        22  LCD_sendString
        162  SystemClock_Config
        4  data
        4  echo_time
        88  hspi4
        224  htim3
        htim4
        htim5
        echo_dist
        flag
        70  int_to_str
        742  main
        4  num_out


  316 bytes in section .bss
        8 bytes in section .data
        4 bytes in section .rodata
1'340 bytes in section .text

1'340 bytes of CODE  memory
        4 bytes of CONST memory
  324 bytes of DATA  memory

Errors: none
Warnings: none


lt.lst
#############################################################################
##########
#
# IAR ANSI C/C++ Compiler V9.20.4.327/W64 for ARM         16/Mar/2023  18:02:45
# Copyright 1999-2022 IAR Systems AB.
#
#        Cpu mode              =  thumb
#        Endian          =  little
```

```
#       Source file    =
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\Cor
e\Src\stm32f4xx_it.c
#       Command line       =
#       -f
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core\stm32f4xx_it.o.rsp
#
(S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\Cor
e\Src\stm32f4xx_it.c
#       -D USE_HAL_DRIVER -D STM32F429xx -lC
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\List\Application\User\Core
#       -o
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core
#       --debug --endian=little --cpu=Cortex-M4 -e --fpu=VFPv4_sp
#       --dlib_config S:\School_Work\arm\inc\c\DLib_Config_Full.h -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Core/Inc\
#       -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/STM32F4xx_HAL_Driver/Inc\
#       -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/STM32F4xx_HAL_Driver/Inc/Legacy\
#       -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/CMSIS/Device/ST/STM32F4xx/Include\
#       -I
```

```
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/CMSIS/Include\
#       -Ohz) --dependencies=n
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core\stm32f4xx_it.o.d
#       Locale         =  C
#       List file      =
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\List\Application\User\Core\stm32f4xx_it.lst
#       Object file    =
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core\stm32f4xx_it.o
#       Runtime model:
#       __CPP_Runtime   =  1
#       __SystemLibrary =  DLib
#       __dlib_version  =  6
#       __size_limit    =  32768|ARM.EW.LINKER
#
################################################################################
##########

S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\Cor
e\Src\stm32f4xx_it.c
        1       /* USER CODE BEGIN Header */
        2       /**
        3       ******************************************************************************
        4       * @filestm32f4xx_it.c
        5       * @brief   Interrupt Service Routines.
        6       ******************************************************************************
        7       * @attention
        8       *
        9       * Copyright (c) 2023 STMicroelectronics.
        10              * All rights reserved.
        11              *
        12              * This software is licensed under terms that can be found in the
LICENSE file
```

```
        13              * in the root directory of this software component.
        14              * If no LICENSE file comes with this software, it is provided AS-IS.
        15              *
        16
******************************************************************************
        17              */
        18      /* USER CODE END Header */
        19
        20      /* Includes ------------------------------------------------------------------*/
        21      #include "main.h"
        22      #include "stm32f4xx_it.h"
        23      /* Private includes ----------------------------------------------------------*/
        24      /* USER CODE BEGIN Includes */
        25      /* USER CODE END Includes */
        26
        27      /* Private typedef -----------------------------------------------------------*/
        28      /* USER CODE BEGIN TD */
        29
        30      /* USER CODE END TD */
        31
        32      /* Private define ------------------------------------------------------------*/
        33      /* USER CODE BEGIN PD */
        34
        35      /* USER CODE END PD */
        36
        37      /* Private macro -------------------------------------------------------------*/
        38      /* USER CODE BEGIN PM */
        39
        40      /* USER CODE END PM */
        41
        42      /* Private variables ---------------------------------------------------------*/
        43      /* USER CODE BEGIN PV */

  \                             In section .bss, align 4
        44      int rising = 0;
  \             rising:
  \     0x0                     DS8 4
        45      /* USER CODE END PV */
        46
        47      /* Private function prototypes -----------------------------------------------*/
```

```
48      /* USER CODE BEGIN PFP */
49
50      /* USER CODE END PFP */
51
52      /* Private user code ---------------------------------------------------------*/
53      /* USER CODE BEGIN 0 */
54
55      /* USER CODE END 0 */
56
57      /* External variables --------------------------------------------------------*/
58      extern TIM_HandleTypeDef htim3;
59      extern TIM_HandleTypeDef htim4;
60      extern TIM_HandleTypeDef htim5;
61      /* USER CODE BEGIN EV */
62      extern int echo_time;
63      extern int echo_dist;
64      extern void LCD_Init();
65      extern void LCD_SendCommand(uint8_t cmd);
66      extern void LCD_SendData(uint8_t data);
67      extern void LCD_Sel_Row_Col(uint8_t row, uint8_t col);
68      extern void LCD_sendString(char* in, int length);
69      extern void LCD_Clear();
70      extern void int_to_str(int number);
71      extern uint8_t num_out [3];
72      extern int flag;
73      /* USER CODE END EV */
74
75      /******************************************************************************/
76      /*       Cortex-M4 Processor Interruption and Exception Handlers
*/
77      /******************************************************************************/
78      /**
79              * @brief This function handles Non maskable interrupt.
80              */

  \                     In section .text, align 2, keep-with-next
81      void NMI_Handler(void)
82      {
83              /* USER CODE BEGIN NonMaskableInt_IRQn 0 */
84
```

```
    85              /* USER CODE END NonMaskableInt_IRQn 0 */
    86              /* USER CODE BEGIN NonMaskableInt_IRQn 1 */
    87              while (1)
\               NMI_Handler: (+1)
\               ??NMI_Handler_0: (+1)
\       0x0  0xE7FE         B.N     ??NMI_Handler_0
    88              {
    89              }
    90              /* USER CODE END NonMaskableInt_IRQn 1 */
    91      }
    92
    93      /**
    94              * @brief This function handles Hard fault interrupt.
    95              */

\                               In section .text, align 2, keep-with-next
    96      void HardFault_Handler(void)
    97      {
    98              /* USER CODE BEGIN HardFault_IRQn 0 */
    99
   100              /* USER CODE END HardFault_IRQn 0 */
   101              while (1)
\               HardFault_Handler: (+1)
\               ??HardFault_Handler_0: (+1)
\       0x0  0xE7FE         B.N     ??HardFault_Handler_0
   102              {
   103              /* USER CODE BEGIN W1_HardFault_IRQn 0 */
   104              /* USER CODE END W1_HardFault_IRQn 0 */
   105              }
   106              }
   107
   108              /**
   109              * @brief This function handles Memory management fault.
   110              */

\                               In section .text, align 2, keep-with-next
   111              void MemManage_Handler(void)
   112              {
   113              /* USER CODE BEGIN MemoryManagement_IRQn 0 */
   114
```

```
    115           /* USER CODE END MemoryManagement_IRQn 0 */
    116           while (1)
\             MemManage_Handler: (+1)
\             ??MemManage_Handler_0: (+1)
\       0x0  0xE7FE         B.N    ??MemManage_Handler_0
    117           {
    118           /* USER CODE BEGIN W1_MemoryManagement_IRQn 0 */
    119           /* USER CODE END W1_MemoryManagement_IRQn 0 */
    120           }
    121           }
    122
    123           /**
    124           * @brief This function handles Pre-fetch fault, memory access fault.
    125           */

\                             In section .text, align 2, keep-with-next
    126           void BusFault_Handler(void)
    127           {
    128           /* USER CODE BEGIN BusFault_IRQn 0 */
    129
    130           /* USER CODE END BusFault_IRQn 0 */
    131           while (1)
\             BusFault_Handler: (+1)
\             ??BusFault_Handler_0: (+1)
\       0x0  0xE7FE         B.N    ??BusFault_Handler_0
    132           {
    133           /* USER CODE BEGIN W1_BusFault_IRQn 0 */
    134           /* USER CODE END W1_BusFault_IRQn 0 */
    135           }
    136           }
    137
    138           /**
    139           * @brief This function handles Undefined instruction or illegal state.
    140           */

\                             In section .text, align 2, keep-with-next
    141           void UsageFault_Handler(void)
    142           {
    143           /* USER CODE BEGIN UsageFault_IRQn 0 */
    144
```

```
      145           /* USER CODE END UsageFault_IRQn 0 */
      146           while (1)
\           UsageFault_Handler: (+1)
\           ??UsageFault_Handler_0: (+1)
\     0x0  0xE7FE         B.N    ??UsageFault_Handler_0
      147           {
      148           /* USER CODE BEGIN W1_UsageFault_IRQn 0 */
      149           /* USER CODE END W1_UsageFault_IRQn 0 */
      150           }
      151           }
      152
      153           /**
      154           * @brief This function handles System service call via SWI
instruction.
      155           */

\                           In section .text, align 2, keep-with-next
      156           void SVC_Handler(void)
      157           {
      158           /* USER CODE BEGIN SVCall_IRQn 0 */
      159
      160           /* USER CODE END SVCall_IRQn 0 */
      161           /* USER CODE BEGIN SVCall_IRQn 1 */
      162
      163           /* USER CODE END SVCall_IRQn 1 */
      164           }
\           SVC_Handler: (+1)
\     0x0  0x4770         BX     LR
      165
      166           /**
      167           * @brief This function handles Debug monitor.
      168           */

\                           In section .text, align 2, keep-with-next
      169           void DebugMon_Handler(void)
      170           {
      171           /* USER CODE BEGIN DebugMonitor_IRQn 0 */
      172
      173           /* USER CODE END DebugMonitor_IRQn 0 */
      174           /* USER CODE BEGIN DebugMonitor_IRQn 1 */
```

```
        175
        176             /* USER CODE END DebugMonitor_IRQn 1 */
        177             }
  \             DebugMon_Handler: (+1)
  \     0x0  0x4770         BX      LR
        178
        179             /**
        180             * @brief This function handles Pendable request for system
service.
        181             */

  \                             In section .text, align 2, keep-with-next
        182             void PendSV_Handler(void)
        183             {
        184             /* USER CODE BEGIN PendSV_IRQn 0 */
        185
        186             /* USER CODE END PendSV_IRQn 0 */
        187             /* USER CODE BEGIN PendSV_IRQn 1 */
        188
        189             /* USER CODE END PendSV_IRQn 1 */
        190             }
  \             PendSV_Handler: (+1)
  \     0x0  0x4770         BX      LR
        191
        192             /**
        193             * @brief This function handles System tick timer.
        194             */

  \                             In section .text, align 2, keep-with-next
        195             void SysTick_Handler(void)
        196             {
        197             /* USER CODE BEGIN SysTick_IRQn 0 */
        198
        199             /* USER CODE END SysTick_IRQn 0 */
        200             HAL_IncTick();
  \             SysTick_Handler: (+1)
  \     0x0  0x.... 0x....      B.W    HAL_IncTick
        201             /* USER CODE BEGIN SysTick_IRQn 1 */
        202
        203             /* USER CODE END SysTick_IRQn 1 */
```

```
        204             }
        205
        206
/******************************************************************************/
        207             /* STM32F4xx Peripheral Interrupt Handlers
*/
        208             /* Add here the Interrupt Handlers for the used peripherals.
*/
        209             /* For the available peripheral interrupt handler names,
*/
        210             /* please refer to the startup file (startup_stm32f4xx.s).
*/
        211
/******************************************************************************/
        212
        213             /**
        214             * @brief This function handles TIM3 global interrupt.
        215             */

  \                             In section .text, align 2, keep-with-next
        216             void TIM3_IRQHandler(void)
        217             {
  \              TIM3_IRQHandler: (+1)
  \      0x0   0xB538           PUSH         {R3-R5,LR}
        218             /* USER CODE BEGIN TIM3_IRQn 0 */
        219             if(flag == 0){
  \      0x2   0x....           LDR.N        R3,??DataTable3
  \      0x4   0x....           LDR.N        R1,??DataTable3_1
  \      0x6   0x6818           LDR    R0,[R3, #+0]
  \      0x8   0x....           LDR.N        R2,??DataTable3_2
  \      0xA   0xB918           CBNZ.N   R0,??TIM3_IRQHandler_0
        220             rising = TIM3 -> CNT;
  \      0xC   0x6812           LDR    R2,[R2, #+0]
  \      0xE   0x600A           STR    R2,[R1, #+0]
        221             flag = 1;
  \      0x10   0x2001          MOVS         R0,#+1
  \      0x12   0xE00C                 B.N    ??TIM3_IRQHandler_1
        222             }
        223             else{
        224             echo_time = (TIM3 -> CNT) - rising;
```

```
\             ??TIM3_IRQHandler_0: (+1)
\     0x14  0x6814        LDR   R4,[R2, #+0]
\     0x16  0x6808        LDR   R0,[R1, #+0]
\     0x18  0x1A20        SUBS        R0,R4,R0
      225             echo_dist = echo_time / 6 + 1; //convert to cm
\     0x1A  0x2106        MOVS        R1,#+6
\     0x1C  0xFB90 0xF1F1     SDIV  R1,R0,R1
\     0x20  0x....        LDR.N       R4,??DataTable3_3
      226             TIM3 -> CNT = 0;
\     0x22  0x....        LDR.N       R5,??DataTable3_4
\     0x24  0x1C49        ADDS        R1,R1,#+1
\     0x26  0x6021        STR   R1,[R4, #+0]
\     0x28  0x6028        STR   R0,[R5, #+0]
\     0x2A  0x2000        MOVS        R0,#+0
\     0x2C  0x6010        STR   R0,[R2, #+0]
      227             flag = 0;
      228
      229             }
      230             /* USER CODE END TIM3_IRQn 0 */
      231             HAL_TIM_IRQHandler(&htim3);
\             ??TIM3_IRQHandler_1: (+1)
\     0x2E  0x6018        STR   R0,[R3, #+0]
\     0x30  0xE8BD 0x4032     POP   {R1,R4,R5,LR}
\     0x34  0x....        LDR.N       R0,??DataTable3_5
\     0x36  0x.... 0x....     B.W   HAL_TIM_IRQHandler
      232             /* USER CODE BEGIN TIM3_IRQn 1 */
      233
      234             /* USER CODE END TIM3_IRQn 1 */
      235             }
      236
      237             /**
      238             * @brief This function handles TIM4 global interrupt.
      239             */

\                             In section .text, align 2, keep-with-next
      240             void TIM4_IRQHandler(void)
      241             {
      242             /* USER CODE BEGIN TIM4_IRQn 0 */
      243
      244             /* USER CODE END TIM4_IRQn 0 */
```

```
    245            HAL_TIM_IRQHandler(&htim4);
\           TIM4_IRQHandler: (+1)
\   0x0  0x....            LDR.N          R0,??DataTable3_6
\   0x2  0x.... 0x....     B.W    HAL_TIM_IRQHandler
    246            /* USER CODE BEGIN TIM4_IRQn 1 */
    247
    248            /* USER CODE END TIM4_IRQn 1 */
    249            }
    250
    251            /**
    252            * @brief This function handles TIM5 global interrupt.
    253            */

\                          In section .text, align 2, keep-with-next
    254            void TIM5_IRQHandler(void)
    255            {
    256            /* USER CODE BEGIN TIM5_IRQn 0 */
    257
    258            /* USER CODE END TIM5_IRQn 0 */
    259            HAL_TIM_IRQHandler(&htim5);
\           TIM5_IRQHandler: (+1)
\   0x0  0x....            LDR.N          R0,??DataTable3_7
\   0x2  0x.... 0x....     B.W    HAL_TIM_IRQHandler
    260            /* USER CODE BEGIN TIM5_IRQn 1 */
    261
    262            /* USER CODE END TIM5_IRQn 1 */
    263            }

\                          In section .text, align 4, keep-with-next
\           ??DataTable3:
\   0x0  0x....'....       DC32  flag

\                          In section .text, align 4, keep-with-next
\           ??DataTable3_1:
\   0x0  0x....'....       DC32  rising

\                          In section .text, align 4, keep-with-next
\           ??DataTable3_2:
\   0x0  0x4000'0424            DC32  0x40000424
```

```
\                               In section .text, align 4, keep-with-next
\              ??DataTable3_3:
\      0x0   0x....'....        DC32  echo_dist

\                               In section .text, align 4, keep-with-next
\              ??DataTable3_4:
\      0x0   0x....'....        DC32  echo_time

\                               In section .text, align 4, keep-with-next
\              ??DataTable3_5:
\      0x0   0x....'....        DC32  htim3

\                               In section .text, align 4, keep-with-next
\              ??DataTable3_6:
\      0x0   0x....'....        DC32  htim4

\                               In section .text, align 4, keep-with-next
\              ??DataTable3_7:
\      0x0   0x....'....        DC32  htim5
       264
       265             /* USER CODE BEGIN 1 */
       266
       267             /* USER CODE END 1 */
```

Maximum stack usage in bytes:

```
.cstack Function
------- --------
       0   BusFault_Handler
       0   DebugMon_Handler
       0   HardFault_Handler
       0   MemManage_Handler
       0   NMI_Handler
       0   PendSV_Handler
       0   SVC_Handler
       0   SysTick_Handler
       0   -> HAL_IncTick
      16   TIM3_IRQHandler
       0   -> HAL_TIM_IRQHandler
       0   TIM4_IRQHandler
```

```
        0   -> HAL_TIM_IRQHandler
        0   TIM5_IRQHandler
        0   -> HAL_TIM_IRQHandler
        0   UsageFault_Handler


    Section sizes:

    Bytes  Function/Label
    -----  --------------
        4  ??DataTable3
        4  ??DataTable3_1
        4  ??DataTable3_2
        4  ??DataTable3_3
        4  ??DataTable3_4
        4  ??DataTable3_5
        4  ??DataTable3_6
        4  ??DataTable3_7
        2  BusFault_Handler
        2  DebugMon_Handler
        2  HardFault_Handler
        2  MemManage_Handler
        2  NMI_Handler
        2  PendSV_Handler
        2  SVC_Handler
        4  SysTick_Handler
       58  TIM3_IRQHandler
        6  TIM4_IRQHandler
        6  TIM5_IRQHandler
        2  UsageFault_Handler
        4  rising


    4 bytes in section .bss
  122 bytes in section .text

  122 bytes of CODE memory
    4 bytes of DATA memory

Errors: none
```

Warnings: none

Task 3

Main.lst

```
###############################################################################
#########
#
# IAR ANSI C/C++ Compiler V9.20.4.327/W64 for ARM         16/Mar/2023  18:02:46
# Copyright 1999-2022 IAR Systems AB.
#
#       Cpu mode              =  thumb
#       Endian          =  little
#       Source file     =
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\Cor
e\Src\main.c
#       Command line        =
#       -f
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core\main.o.rsp
#
(S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\Cor
e\Src\main.c
#       -D USE_HAL_DRIVER -D STM32F429xx -lC
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\List\Application\User\Core
#       -o
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core
#       --debug --endian=little --cpu=Cortex-M4 -e --fpu=VFPv4_sp
#       --dlib_config S:\School_Work\arm\inc\c\DLib_Config_Full.h -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Core/Inc\
#       -I
```

```
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/STM32F4xx_HAL_Driver/Inc\
#       -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/STM32F4xx_HAL_Driver/Inc/Legacy\
#       -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/CMSIS/Device/ST/STM32F4xx/Include\
#       -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/CMSIS/Include\
#       -Ohz) --dependencies=n
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core\main.o.d
#       Locale          =  C
#       List file       =
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\List\Application\User\Core\main.lst
#       Object file     =
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core\main.o
#       Runtime model:
#       __CPP_Runtime   =  1
#       __SystemLibrary =  DLib
#       __dlib_version  =  6
#       __size_limit   =  32768|ARM.EW.LINKER
#
################################################################################
#########

S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\Cor
e\Src\main.c
        1       /* USER CODE BEGIN Header */
```

```
2       /**
3       ******************************************************************************
4       * @file           : main.c
5       * @brief          : Main program body
6       ******************************************************************************
7       * @attention
8       *
9       * Copyright (c) 2023 STMicroelectronics.
10              * All rights reserved.
11              *
12              * This software is licensed under terms that can be found in the
LICENSE file
13              * in the root directory of this software component.
14              * If no LICENSE file comes with this software, it is provided AS-IS.
15              *
16
******************************************************************************
17              */
18      /* USER CODE END Header */
19      /* Includes ------------------------------------------------------------------*/
20      #include "main.h"
21
22      /* Private includes ----------------------------------------------------------*/
23      /* USER CODE BEGIN Includes */
24
25      /* USER CODE END Includes */
26
27      /* Private typedef -----------------------------------------------------------*/
28      /* USER CODE BEGIN PTD */
29
30      /* USER CODE END PTD */
31
32      /* Private define ------------------------------------------------------------*/
33      /* USER CODE BEGIN PD */
34      #define Function_Set 0b00110000 //0x30
35      #define Display_On 0b00001100 //0x0C
36      #define Display_Clear 0b00000001 //0x01
37      #define Entry_Mode_Set 0b00000110 //0x06
38      /* USER CODE END PD */
39
```

```
      40     /* Private macro -------------------------------------------------------*/
      41     /* USER CODE BEGIN PM */
      42
      43     /* USER CODE END PM */
      44
      45     /* Private variables ---------------------------------------------------*/

\                              In section .bss, align 4
      46     SPI_HandleTypeDef hspi4;
\            hspi4:
\     0x0                      DS8 88
      47
      48     TIM_HandleTypeDef htim3;
      49     TIM_HandleTypeDef htim4;
      50     TIM_HandleTypeDef htim5;
      51
      52     /* USER CODE BEGIN PV */
      53     int echo_time = 0;
      54     int echo_dist = 0;

\                              In section .data, align 4
      55     uint8_t data [3] = "000";
\            `data`:
\     0x0  0x30 0x30           DC8 0x30, 0x30, 0x30

\            0x30
\     0x3                      DS8 1

\                              In section .data, align 4
      56     uint8_t num_out [3] = "000";
\            num_out:
\     0x0  0x30 0x30           DC8 0x30, 0x30, 0x30

\            0x30
\     0x3                      DS8 1

\                              In section .bss, align 4
\            htim3:
\     0x0                      DS8 72
\            htim4:
```

```
\     0x48                    DS8 72
\             htim5:
\     0x90                    DS8 72
\             echo_dist:
\     0xD8                    DS8 4
      57      int flag = 0;
\             flag:
\     0xDC                    DS8 4

\                             In section .bss, align 4
\             echo_time:
\     0x0                     DS8 4
      58      /* USER CODE END PV */
      59
      60      /* Private function prototypes -----------------------------------------------*/
      61      void SystemClock_Config(void);
      62      static void MX_GPIO_Init(void);
      63      static void MX_TIM4_Init(void);
      64      static void MX_TIM5_Init(void);
      65      static void MX_TIM3_Init(void);
      66      static void MX_SPI4_Init(void);
      67      /* USER CODE BEGIN PFP */
      68      void LCD_Init();
      69      void LCD_SendCommand(uint8_t cmd);
      70      void LCD_SendData(uint8_t data);
      71      void LCD_Sel_Row_Col(uint8_t row, uint8_t col);
      72      void LCD_sendString(char* in, int length);
      73      void LCD_Clear();
      74      void int_to_str(int number);
      75      /* USER CODE END PFP */
      76
      77      /* Private user code ---------------------------------------------------------*/
      78      /* USER CODE BEGIN 0 */
      79
      80      /* USER CODE END 0 */
      81
      82      /**
      83              * @brief  The application entry point.
      84              * @retval int
      85              */
```

```
\                               In section .text, align 4, keep-with-next
      86      int main(void)
      87      {
\             main: (+1)
\     0x0   0xE92D 0x47F8       PUSH          {R3-R10,LR}
\     0x4   0xB08D              SUB   SP,SP,#+52
      88              /* USER CODE BEGIN 1 */
      89
      90              /* USER CODE END 1 */
      91
      92              /* MCU Configuration--------------------------------------------------------*/
      93
      94              /* Reset of all peripherals, Initializes the Flash interface and the
Systick. */
      95              HAL_Init();
\     0x6   0x.... 0x....       BL      HAL_Init
      96
      97              /* USER CODE BEGIN Init */
      98
      99              /* USER CODE END Init */
      100
      101             /* Configure the system clock */
      102             SystemClock_Config();
\     0xA   0x.... 0x....       BL      SystemClock_Config
      103
      104             /* USER CODE BEGIN SysInit */
      105
      106             /* USER CODE END SysInit */
      107
      108             /* Initialize all configured peripherals */
      109             MX_GPIO_Init();
\     0xE   0x.... 0x....       BL      ?Subroutine4
\             ??CrossCallReturnLabel_9: (+1)
\     0x12  0x2000              MOVS          R0,#+0
\     0x14  0x9000              STR   R0,[SP, #+0]
\     0x16  0x.... 0x....       LDR.W         R9,??DataTable7_1
\     0x1A  0x.... 0x....       LDR.W         R0,??DataTable7_2
\     0x1E  0x.... 0x....       LDR.W         R5,??DataTable7_3
\     0x22  0x6801              LDR   R1,[R0, #+0]
```

```
\      0x24   0xF041 0x0110      ORR   R1,R1,#0x10
\      0x28   0x6001       STR   R1,[R0, #+0]
\      0x2A   0x6802       LDR   R2,[R0, #+0]
\      0x2C   0xF002 0x0210      AND   R2,R2,#0x10
\      0x30   0x9200       STR   R2,[SP, #+0]
\      0x32   0x2200       MOVS        R2,#+0
\      0x34   0x9900       LDR   R1,[SP, #+0]
\      0x36   0x9200       STR   R2,[SP, #+0]
\      0x38   0x6803       LDR   R3,[R0, #+0]
\      0x3A   0xF043 0x0380      ORR   R3,R3,#0x80
\      0x3E   0x6003       STR   R3,[R0, #+0]
\      0x40   0x6801       LDR   R1,[R0, #+0]
\      0x42   0xF001 0x0180      AND   R1,R1,#0x80
\      0x46   0x9100       STR   R1,[SP, #+0]
\      0x48   0x9900       LDR   R1,[SP, #+0]
\      0x4A   0x9200       STR   R2,[SP, #+0]
\      0x4C   0x6803       LDR   R3,[R0, #+0]
\      0x4E   0xF043 0x0304      ORR   R3,R3,#0x4
\      0x52   0x6003       STR   R3,[R0, #+0]
\      0x54   0x6801       LDR   R1,[R0, #+0]
\      0x56   0xF001 0x0104      AND   R1,R1,#0x4
\      0x5A   0x9100       STR   R1,[SP, #+0]
\      0x5C   0x9900       LDR   R1,[SP, #+0]
\      0x5E   0x9200       STR   R2,[SP, #+0]
\      0x60   0x6803       LDR   R3,[R0, #+0]
\      0x62   0xF043 0x0301      ORR   R3,R3,#0x1
\      0x66   0x6003       STR   R3,[R0, #+0]
\      0x68   0x6801       LDR   R1,[R0, #+0]
\      0x6A   0xF001 0x0101      AND   R1,R1,#0x1
\      0x6E   0x9100       STR   R1,[SP, #+0]
\      0x70   0x9900       LDR   R1,[SP, #+0]
\      0x72   0x9200       STR   R2,[SP, #+0]
\      0x74   0x2102       MOVS        R1,#+2
\      0x76   0x6803       LDR   R3,[R0, #+0]
\      0x78   0xF043 0x0308      ORR   R3,R3,#0x8
\      0x7C   0x6003       STR   R3,[R0, #+0]
\      0x7E   0x6800       LDR   R0,[R0, #+0]
\      0x80   0xF000 0x0008      AND   R0,R0,#0x8
\      0x84   0x9000       STR   R0,[SP, #+0]
\      0x86   0x9800       LDR   R0,[SP, #+0]
```

```
\     0x88   0x9101         STR    R1,[SP, #+4]
\     0x8A   0x9202         STR    R2,[SP, #+8]
\     0x8C   0x9203         STR    R2,[SP, #+12]
\     0x8E   0xA901                ADD    R1,SP,#+4
\     0x90   0x4648         MOV    R0,R9
\     0x92   0x.... 0x....  BL     HAL_GPIO_Init
      110              MX_TIM4_Init();
\     0x96   0x2210         MOVS           R2,#+16
\     0x98   0x2100         MOVS           R1,#+0
\     0x9A   0xA809                ADD    R0,SP,#+36
\     0x9C   0x.... 0x....  BL     memset
\     0xA0   0x.... 0x....  BL     ?Subroutine1
\            ??CrossCallReturnLabel_0: (+1)
\     0xA4   0x221C                MOVS           R2,#+28
\     0xA6   0x.... 0x....  BL     ??Subroutine2_0
\            ??CrossCallReturnLabel_3: (+1)
\     0xAA   0x.... 0x....  LDR.W          R0,??DataTable7_4
\     0xAE   0x64A8                STR    R0,[R5, #+72]
\     0xB0   0xF44F 0x61E1         MOV    R1,#+1800
\     0xB4   0x64E9                STR    R1,[R5, #+76]
\     0xB6   0x2000         MOVS           R0,#+0
\     0xB8   0x6528         STR    R0,[R5, #+80]
\     0xBA   0xF640 0x11C4         MOVW           R1,#+2500
\     0xBE   0x65A8                STR    R0,[R5, #+88]
\     0xC0   0x6628         STR    R0,[R5, #+96]
\     0xC2   0x6569         STR    R1,[R5, #+84]
\     0xC4   0xF105 0x0048         ADD    R0,R5,#+72
\     0xC8   0x.... 0x....  BL     HAL_TIM_Base_Init
\     0xCC   0xB108                CBZ.N          R0,??main_0
\     0xCE   0x.... 0x....  BL     Error_Handler
\            ??main_0: (+1)
\     0xD2   0xF44F 0x5680         MOV    R6,#+4096
\     0xD6   0x9609         STR    R6,[SP, #+36]
\     0xD8   0xA909                ADD    R1,SP,#+36
\     0xDA   0xF105 0x0048         ADD    R0,R5,#+72
\     0xDE   0x.... 0x....  BL     HAL_TIM_ConfigClockSource
\     0xE2   0xB108                CBZ.N          R0,??main_1
\     0xE4   0x.... 0x....  BL     Error_Handler
\            ??main_1: (+1)
\     0xE8   0xF105 0x0048         ADD    R0,R5,#+72
```

```
\      0xEC   0x.... 0x....    BL      HAL_TIM_PWM_Init
\      0xF0   0xB108           CBZ.N           R0,??main_2
\      0xF2   0x.... 0x....    BL      Error_Handler
\              ??main_2: (+1)
\      0xF6   0x2100           MOVS            R1,#+0
\      0xF8   0x9100           STR     R1,[SP, #+0]
\      0xFA   0x9101           STR     R1,[SP, #+4]
\      0xFC   0xF105 0x0048    ADD     R0,R5,#+72
\      0x100   0x4669                  MOV     R1,SP
\      0x102   0x.... 0x....   BL      HAL_TIMEx_MasterConfigSynchronization
\      0x106   0xB108                  CBZ.N           R0,??main_3
\      0x108   0x.... 0x....   BL      Error_Handler
\              ??main_3: (+1)
\      0x10C   0x2160                  MOVS            R1,#+96
\      0x10E   0x.... 0x....   BL      ?Subroutine5
\              ??CrossCallReturnLabel_11: (+1)
\      0x112   0x9206                  STR     R2,[SP, #+24]
\      0x114   0xA902                  ADD     R1,SP,#+8
\      0x116   0xF105 0x0048    ADD     R0,R5,#+72
\      0x11A   0x.... 0x....   BL      HAL_TIM_PWM_ConfigChannel
\      0x11E   0xB108                  CBZ.N           R0,??main_4
\      0x120   0x.... 0x....   BL      Error_Handler
\              ??main_4: (+1)
\      0x124   0xF105 0x0048    ADD     R0,R5,#+72
\      0x128   0x.... 0x....   BL      HAL_TIM_MspPostInit
       111             MX_TIM5_Init();
\      0x12C   0x.... 0x....   BL      ?Subroutine3
\              ??CrossCallReturnLabel_6: (+1)
\      0x130   0x.... 0x....   BL      ?Subroutine1
\              ??CrossCallReturnLabel_1: (+1)
\      0x134   0x.... 0x....   BL      ?Subroutine2
\              ??CrossCallReturnLabel_4: (+1)
\      0x138   0xF105 0x0790    ADD     R7,R5,#+144
\      0x13C   0x.... 0x....   LDR.W            R0,??DataTable7_5
\      0x140   0x6038                  STR     R0,[R7, #+0]
\      0x142   0x2100                  MOVS            R1,#+0
\      0x144   0xF04F 0x30FF    MOV     R0,#+4294967295
\      0x148   0x60F8                  STR     R0,[R7, #+12]
\      0x14A   0x6079                  STR     R1,[R7, #+4]
\      0x14C   0x60B9                  STR     R1,[R7, #+8]
```

```
\     0x14E   0x6139          STR   R1,[R7, #+16]
\     0x150   0x61B9          STR   R1,[R7, #+24]
\     0x152   0x4638          MOV   R0,R7
\     0x154   0x.... 0x....   BL    HAL_TIM_Base_Init
\     0x158   0xB108          CBZ.N       R0,??main_5
\     0x15A   0x.... 0x....   BL    Error_Handler
\           ??main_5: (+1)
\     0x15E   0x9606          STR   R6,[SP, #+24]
\     0x160   0xA906          ADD   R1,SP,#+24
\     0x162   0x4638          MOV   R0,R7
\     0x164   0x.... 0x....   BL    HAL_TIM_ConfigClockSource
\     0x168   0xB108          CBZ.N       R0,??main_6
\     0x16A   0x.... 0x....   BL    Error_Handler
\           ??main_6: (+1)
\     0x16E   0x4638          MOV   R0,R7
\     0x170   0x.... 0x....   BL    HAL_TIM_IC_Init
\     0x174   0xB108          CBZ.N       R0,??main_7
\     0x176   0x.... 0x....   BL    Error_Handler
\           ??main_7: (+1)
\     0x17A   0x2100          MOVS        R1,#+0
\     0x17C   0x9100          STR   R1,[SP, #+0]
\     0x17E   0x9101          STR   R1,[SP, #+4]
\     0x180   0x4638          MOV   R0,R7
\     0x182   0x4669          MOV   R1,SP
\     0x184   0x.... 0x....   BL    HAL_TIMEx_MasterConfigSynchronization
\     0x188   0xB108          CBZ.N       R0,??main_8
\     0x18A   0x.... 0x....   BL    Error_Handler
\           ??main_8: (+1)
\     0x18E   0x210A          MOVS        R1,#+10
\     0x190   0x9102          STR   R1,[SP, #+8]
\     0x192   0x2201          MOVS        R2,#+1
\     0x194   0x2100          MOVS        R1,#+0
\     0x196   0x9203          STR   R2,[SP, #+12]
\     0x198   0x9104          STR   R1,[SP, #+16]
\     0x19A   0x9105          STR   R1,[SP, #+20]
\     0x19C   0x2208          MOVS        R2,#+8
\     0x19E   0xA902          ADD   R1,SP,#+8
\     0x1A0   0x4638          MOV   R0,R7
\     0x1A2   0x.... 0x....   BL    HAL_TIM_IC_ConfigChannel
\     0x1A6   0xB108          CBZ.N       R0,??main_9
```

```
\     0x1A8   0x.... 0x....  BL      Error_Handler
      112             MX_TIM3_Init();
\             ??main_9: (+1)
\     0x1AC   0x.... 0x....          BL      ?Subroutine3
\             ??CrossCallReturnLabel_7: (+1)
\     0x1B0   0x.... 0x....  BL      ?Subroutine1
\             ??CrossCallReturnLabel_2: (+1)
\     0x1B4   0x.... 0x....  BL      ?Subroutine2
\             ??CrossCallReturnLabel_5: (+1)
\     0x1B8   0x....         LDR.N         R0,??DataTable7_6
\     0x1BA   0x6028                STR   R0,[R5, #+0]
\     0x1BC   0xF44F 0x7161         MOV   R1,#+900
\     0x1C0   0xF64F 0x70FF         MOVW          R0,#+65535
\     0x1C4   0x2200                MOVS          R2,#+0
\     0x1C6   0x60E8                STR   R0,[R5, #+12]
\     0x1C8   0x6069                STR   R1,[R5, #+4]
\     0x1CA   0x60AA                STR   R2,[R5, #+8]
\     0x1CC   0x612A                STR   R2,[R5, #+16]
\     0x1CE   0x61AA                STR   R2,[R5, #+24]
\     0x1D0   0x4628                MOV   R0,R5
\     0x1D2   0x.... 0x....  BL      HAL_TIM_Base_Init
\     0x1D6   0xB108                CBZ.N         R0,??main_10
\     0x1D8   0x.... 0x....  BL      Error_Handler
\             ??main_10: (+1)
\     0x1DC   0x9606                STR   R6,[SP, #+24]
\     0x1DE   0xA906                ADD   R1,SP,#+24
\     0x1E0   0x4628                MOV   R0,R5
\     0x1E2   0x.... 0x....  BL      HAL_TIM_ConfigClockSource
\     0x1E6   0xB108                CBZ.N         R0,??main_11
\     0x1E8   0x.... 0x....  BL      Error_Handler
\             ??main_11: (+1)
\     0x1EC   0x4628                MOV   R0,R5
\     0x1EE   0x.... 0x....  BL      HAL_TIM_IC_Init
\     0x1F2   0xB108                CBZ.N         R0,??main_12
\     0x1F4   0x.... 0x....  BL      Error_Handler
\             ??main_12: (+1)
\     0x1F8   0x2100                MOVS          R1,#+0
\     0x1FA   0x9100                STR   R1,[SP, #+0]
\     0x1FC   0x9101                STR   R1,[SP, #+4]
\     0x1FE   0x4628                MOV   R0,R5
```

```
\     0x200   0x4669              MOV   R1,SP
\     0x202   0x.... 0x....   BL      HAL_TIMEx_MasterConfigSynchronization
\     0x206   0xB108              CBZ.N         R0,??main_13
\     0x208   0x.... 0x....   BL      Error_Handler
\             ??main_13: (+1)
\     0x20C   0x210A              MOVS          R1,#+10
\     0x20E   0x.... 0x....   BL      ?Subroutine5
\             ??CrossCallReturnLabel_12: (+1)
\     0x212   0x9205              STR   R2,[SP, #+20]
\     0x214   0xA902              ADD   R1,SP,#+8
\     0x216   0x4628              MOV   R0,R5
\     0x218   0x.... 0x....   BL      HAL_TIM_IC_ConfigChannel
\     0x21C   0xB108              CBZ.N         R0,??main_14
\     0x21E   0x.... 0x....   BL      Error_Handler
      113             MX_SPI4_Init();
\             ??main_14: (+1)
\     0x222   0x....          LDR.N         R6,??DataTable7_7
\     0x224   0x....          LDR.N         R0,??DataTable7_8
\     0x226   0x6030              STR   R0,[R6, #+0]
\     0x228   0xF44F 0x7182       MOV   R1,#+260
\     0x22C   0x2038              MOVS          R0,#+56
\     0x22E   0x61F0              STR   R0,[R6, #+28]
\     0x230   0x6071              STR   R1,[R6, #+4]
\     0x232   0x200A              MOVS          R0,#+10
\     0x234   0x2200              MOVS          R2,#+0
\     0x236   0xF44F 0x7100       MOV   R1,#+512
\     0x23A   0x62F0              STR   R0,[R6, #+44]
\     0x23C   0x60B2              STR   R2,[R6, #+8]
\     0x23E   0x60F2              STR   R2,[R6, #+12]
\     0x240   0x6132              STR   R2,[R6, #+16]
\     0x242   0x6172              STR   R2,[R6, #+20]
\     0x244   0x61B1              STR   R1,[R6, #+24]
\     0x246   0x6232              STR   R2,[R6, #+32]
\     0x248   0x6272              STR   R2,[R6, #+36]
\     0x24A   0x62B2              STR   R2,[R6, #+40]
\     0x24C   0x4630              MOV   R0,R6
\     0x24E   0x.... 0x....   BL      HAL_SPI_Init
\     0x252   0xB108              CBZ.N         R0,??main_15
\     0x254   0x.... 0x....   BL      Error_Handler
      114             /* USER CODE BEGIN 2 */
```

```
        115
        116             HAL_SPI_Init(&hspi4);
  \             ??main_15: (+1)
  \     0x258   0x4630            MOV   R0,R6
  \     0x25A   0x.... 0x....     BL    HAL_SPI_Init
        117             LCD_Init();
  \     0x25E   0x.... 0x....     BL    LCD_Init
        118             HAL_TIM_IC_Start_IT(&htim3, TIM_CHANNEL_1);
  \     0x262   0x2100            MOVS          R1,#+0
  \     0x264   0x4628            MOV   R0,R5
  \     0x266   0x.... 0x....     BL    HAL_TIM_IC_Start_IT
        119             HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_1);
  \     0x26A   0x2100            MOVS          R1,#+0
  \     0x26C   0xF105 0x0048     ADD   R0,R5,#+72
  \     0x270   0x.... 0x....     BL    HAL_TIM_PWM_Start
  \     0x274   0x.... 0x....     ADR.W         R5,??DataTable7
  \     0x278   0x....            ADR.N         R6,?_0
  \     0x27A   0x.... 0x....     LDR.W         R8,??DataTable7_9
  \     0x27E   0xE017            B.N     ??main_16
        120             /* USER CODE END 2 */
        121
        122             /* Infinite loop */
        123             /* USER CODE BEGIN WHILE */
        124             while (1)
        125             {
        126             if(flag == 1){
        127             if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_1) ==
GPIO_PIN_SET) {
        128             echo_dist = echo_dist / 3;
  \             ??main_17: (+1)
  \     0x280   0x2103            MOVS          R1,#+3
  \     0x282   0xFB90 0xFAF1     SDIV  R10,R0,R1
        129             int_to_str(echo_dist);
  \     0x286   0x4650            MOV   R0,R10
  \     0x288   0x.... 0x....     BL    int_to_str
        130             for (int i = 0; i<3; i++){
  \     0x28C   0x2400            MOVS          R4,#+0
  \     0x28E   0xF8C7 0xA048     STR   R10,[R7, #+72]
        131             LCD_SendData(num_out[i]);}
  \             ??main_18: (+1)
```

```
\     0x292   0xF818 0x0004     LDRB R0,[R8, R4]
\     0x296   0x.... 0x....   BL      LCD_SendData
\     0x29A   0x1C64            ADDS          R4,R4,#+1
\     0x29C   0x2C02            CMP   R4,#+2
\     0x29E   0xDDF8            BLE.N ??main_18
      132             LCD_Sel_Row_Col(2, 0);
\     0x2A0   0x2100            MOVS          R1,#+0
\     0x2A2   0x2002            MOVS          R0,#+2
\     0x2A4   0x.... 0x....   BL      LCD_Sel_Row_Col
      133             char *inches = "inches";
      134             LCD_sendString(inches, 6);
\     0x2A8   0x2106            MOVS          R1,#+6
\     0x2AA   0x4630            MOV   R0,R6
\           ??main_19: (+1)
\     0x2AC   0x.... 0x....           BL      LCD_sendString
      135             }
\           ??main_16: (+1)
\     0x2B0   0x6CF8            LDR   R0,[R7, #+76]
\     0x2B2   0x2801            CMP   R0,#+1
\     0x2B4   0xD1FC            BNE.N         ??main_16
\     0x2B6   0x2102            MOVS          R1,#+2
\     0x2B8   0x4648            MOV   R0,R9
\     0x2BA   0x.... 0x....   BL      HAL_GPIO_ReadPin
\     0x2BE   0x2801            CMP   R0,#+1
\     0x2C0   0x6CB8            LDR   R0,[R7, #+72]
\     0x2C2   0xD0DD            BEQ.N         ??main_17
      136             else {
      137             int_to_str(echo_dist);
\     0x2C4   0x.... 0x....   BL      int_to_str
      138             for (int i = 0; i<3; i++){
\     0x2C8   0x2400            MOVS          R4,#+0
      139             LCD_SendData(num_out[i]);}
\           ??main_20: (+1)
\     0x2CA   0xF818 0x0004     LDRB R0,[R8, R4]
\     0x2CE   0x.... 0x....           BL      LCD_SendData
\     0x2D2   0x1C64            ADDS          R4,R4,#+1
\     0x2D4   0x2C02            CMP   R4,#+2
\     0x2D6   0xDDF8            BLE.N ??main_20
      140             LCD_Sel_Row_Col(2, 0);
\     0x2D8   0x2100            MOVS          R1,#+0
```

```
\     0x2DA   0x2002          MOVS     R0,#+2
\     0x2DC   0x.... 0x....   BL       LCD_Sel_Row_Col
      141             char *cm = "cm";
      142             LCD_sendString(cm, 2);
\     0x2E0   0x2102          MOVS     R1,#+2
\     0x2E2   0x4628          MOV      R0,R5
\     0x2E4   0xE7E2          B.N      ??main_19
      143             }
      144             }
      145             /* USER CODE END WHILE */
      146
      147             /* USER CODE BEGIN 3 */
      148             }
      149             /* USER CODE END 3 */
      150             }
      151
      152             /**
      153             * @brief System Clock Configuration
      154             * @retval None
      155             */

\                             In section .text, align 2, keep-with-next
\           ?Subroutine3: (+1)
\     0x0   0x2210          MOVS     R2,#+16
\           ??Subroutine3_0: (+1)
\     0x2   0x2100          MOVS     R1,#+0
\     0x4   0xA806          ADD      R0,SP,#+24
\     0x6   0x.... 0x....   B.W      memset

\                             In section .text, align 2, keep-with-next
      156             void SystemClock_Config(void)
      157             {
\           SystemClock_Config: (+1)
\     0x0   0xB580          PUSH     {R7,LR}
\     0x2   0xB092          SUB      SP,SP,#+72
\     0x4   0x2230          MOVS     R2,#+48
\     0x6   0x.... 0x....   BL       ??Subroutine3_0
\           ??CrossCallReturnLabel_8: (+1)
\     0xA   0x.... 0x....   BL       ?Subroutine4
      158             RCC_OscInitTypeDef RCC_OscInitStruct = {0};
```

```
        159             RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
        160
        161             /** Configure the main internal regulator output voltage
        162             */
        163             __HAL_RCC_PWR_CLK_ENABLE();
  \             ??CrossCallReturnLabel_10: (+1)
  \     0xE   0x2000          MOVS          R0,#+0
  \     0x10  0x9000          STR    R0,[SP, #+0]
        164
__HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1
);
        165
        166             /** Initializes the RCC Oscillators according to the specified
parameters
        167             * in the RCC_OscInitTypeDef structure.
        168             */
        169             RCC_OscInitStruct.OscillatorType =
RCC_OSCILLATORTYPE_HSE;
        170             RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  \     0x12  0xF44F 0x3380       MOV   R3,#+65536
  \     0x16  0x....              LDR.N         R0,??DataTable7_10
  \     0x18  0x6801          LDR    R1,[R0, #+0]
  \     0x1A  0xF041 0x5180       ORR   R1,R1,#0x10000000
  \     0x1E  0x6001          STR    R1,[R0, #+0]
  \     0x20  0x2100          MOVS          R1,#+0
  \     0x22  0x6800          LDR    R0,[R0, #+0]
  \     0x24  0xF000 0x5080       AND   R0,R0,#0x10000000
  \     0x28  0x9000          STR    R0,[SP, #+0]
  \     0x2A  0x9800          LDR    R0,[SP, #+0]
  \     0x2C  0x....              LDR.N         R0,??DataTable7_11
  \     0x2E  0x9100          STR    R1,[SP, #+0]
  \     0x30  0x6802          LDR    R2,[R0, #+0]
  \     0x32  0xF442 0x4240       ORR   R2,R2,#0xC000
  \     0x36  0x6002          STR    R2,[R0, #+0]
  \     0x38  0x2201          MOVS          R2,#+1
  \     0x3A  0x6800          LDR    R0,[R0, #+0]
  \     0x3C  0xF400 0x4040       AND   R0,R0,#0xC000
  \     0x40  0x9000          STR    R0,[SP, #+0]
        171             RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
        172             RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
```

```
    173             RCC_OscInitStruct.PLL.PLLM = 4;
    174             RCC_OscInitStruct.PLL.PLLN = 180;
    175             RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
    176             RCC_OscInitStruct.PLL.PLLQ = 4;
    177             if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
\   0x42  0xA806        ADD   R0,SP,#+24
\   0x44  0x9900        LDR   R1,[SP, #+0]
\   0x46  0x9206        STR   R2,[SP, #+24]
\   0x48  0x2102        MOVS        R1,#+2
\   0x4A  0xF44F 0x0280     MOV   R2,#+4194304
\   0x4E  0x910C                STR   R1,[SP, #+48]
\   0x50  0x920D        STR   R2,[SP, #+52]
\   0x52  0x2104        MOVS        R1,#+4
\   0x54  0x22B4        MOVS        R2,#+180
\   0x56  0x910E        STR   R1,[SP, #+56]
\   0x58  0x920F        STR   R2,[SP, #+60]
\   0x5A  0x2102        MOVS        R1,#+2
\   0x5C  0x2204        MOVS        R2,#+4
\   0x5E  0x9307        STR   R3,[SP, #+28]
\   0x60  0x9110        STR   R1,[SP, #+64]
\   0x62  0x9211        STR   R2,[SP, #+68]
\   0x64  0x.... 0x....     BL    HAL_RCC_OscConfig
\   0x68  0xB108        CBZ.N       R0,??SystemClock_Config_0
    178             {
    179             Error_Handler();
\   0x6A  0x.... 0x....     BL    Error_Handler
    180             }
    181
    182             /** Activate the Over-Drive mode
    183             */
    184             if (HAL_PWREx_EnableOverDrive() != HAL_OK)
\           ??SystemClock_Config_0: (+1)
\   0x6E  0x.... 0x....     BL    HAL_PWREx_EnableOverDrive
\   0x72  0xB108        CBZ.N       R0,??SystemClock_Config_1
    185             {
    186             Error_Handler();
\   0x74  0xB672        CPSID       I
\           ??SystemClock_Config_2: (+1)
\   0x76  0xE7FE                B.N   ??SystemClock_Config_2
    187             }
```

```
        188
        189             /** Initializes the CPU, AHB and APB buses clocks
        190             */
        191             RCC_ClkInitStruct.ClockType =
RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
        192
|RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
        193             RCC_ClkInitStruct.SYSCLKSource =
RCC_SYSCLKSOURCE_PLLCLK;
  \             ??SystemClock_Config_1: (+1)
  \     0x78  0x2102        MOVS        R1,#+2
  \     0x7A  0x9102        STR   R1,[SP, #+8]
        194             RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  \     0x7C  0x2200        MOVS        R2,#+0
  \     0x7E  0x9203        STR   R2,[SP, #+12]
  \     0x80  0x200F        MOVS        R0,#+15
        195             RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV4;
  \     0x82  0xF44F 0x51A0       MOV   R1,#+5120
  \     0x86  0x9001        STR   R0,[SP, #+4]
  \     0x88  0x9104        STR   R1,[SP, #+16]
        196             RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV2;
  \     0x8A  0xF44F 0x5280       MOV   R2,#+4096
  \     0x8E  0x9205        STR   R2,[SP, #+20]
        197
        198             if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct,
FLASH_LATENCY_5) != HAL_OK)
  \     0x90  0x2105        MOVS        R1,#+5
  \     0x92  0xA801        ADD   R0,SP,#+4
  \     0x94  0x.... 0x....     BL      HAL_RCC_ClockConfig
  \     0x98  0xB108        CBZ.N       R0,??SystemClock_Config_3
        199             {
        200             Error_Handler();
  \     0x9A  0xB672                CPSID         I
  \             ??SystemClock_Config_4: (+1)
  \     0x9C  0xE7FE                B.N   ??SystemClock_Config_4
        201             }
        202             }
  \             ??SystemClock_Config_3: (+1)
  \     0x9E  0xB013                ADD   SP,SP,#+76
  \     0xA0  0xBD00                POP   {PC}
```

```
\                               In section .text, align 2, keep-with-next
\                 ?Subroutine4: (+1)
\        0x0  0x2214            MOVS     R2,#+20
\        0x2  0x2100            MOVS     R1,#+0
\        0x4  0xA801            ADD   R0,SP,#+4
\        0x6  0x.... 0x....     B.W    memset
         203
         204             /**
         205             * @brief SPI4 Initialization Function
         206             * @param None
         207             * @retval None
         208             */
         209             static void MX_SPI4_Init(void)
         210             {
         211
         212             /* USER CODE BEGIN SPI4_Init 0 */
         213
         214             /* USER CODE END SPI4_Init 0 */
         215
         216             /* USER CODE BEGIN SPI4_Init 1 */
         217
         218             /* USER CODE END SPI4_Init 1 */
         219             /* SPI4 parameter configuration*/
         220             hspi4.Instance = SPI4;
         221             hspi4.Init.Mode = SPI_MODE_MASTER;
         222             hspi4.Init.Direction = SPI_DIRECTION_2LINES;
         223             hspi4.Init.DataSize = SPI_DATASIZE_8BIT;
         224             hspi4.Init.CLKPolarity = SPI_POLARITY_LOW;
         225             hspi4.Init.CLKPhase = SPI_PHASE_1EDGE;
         226             hspi4.Init.NSS = SPI_NSS_SOFT;
         227             hspi4.Init.BaudRatePrescaler =
SPI_BAUDRATEPRESCALER_256;
         228             hspi4.Init.FirstBit = SPI_FIRSTBIT_MSB;
         229             hspi4.Init.TIMode = SPI_TIMODE_DISABLE;
         230             hspi4.Init.CRCCalculation = SPI_CRCCALCULATION_DISABLE;
         231             hspi4.Init.CRCPolynomial = 10;
         232             if (HAL_SPI_Init(&hspi4) != HAL_OK)
         233             {
         234             Error_Handler();
```

```
235         }
236         /* USER CODE BEGIN SPI4_Init 2 */
237
238         /* USER CODE END SPI4_Init 2 */
239
240         }
241
242         /**
243         * @brief TIM3 Initialization Function
244         * @param None
245         * @retval None
246         */
247         static void MX_TIM3_Init(void)
248         {
249
250         /* USER CODE BEGIN TIM3_Init 0 */
251
252         /* USER CODE END TIM3_Init 0 */
253
254         TIM_ClockConfigTypeDef sClockSourceConfig = {0};
255         TIM_MasterConfigTypeDef sMasterConfig = {0};
256         TIM_IC_InitTypeDef sConfigIC = {0};
257
258         /* USER CODE BEGIN TIM3_Init 1 */
259
260         /* USER CODE END TIM3_Init 1 */
261         htim3.Instance = TIM3;
262         htim3.Init.Prescaler = 900;
263         htim3.Init.CounterMode = TIM_COUNTERMODE_UP;
264         htim3.Init.Period = 65535;
265         htim3.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
266         htim3.Init.AutoReloadPreload =
TIM_AUTORELOAD_PRELOAD_DISABLE;
267         if (HAL_TIM_Base_Init(&htim3) != HAL_OK)
268         {
269         Error_Handler();
270         }
271         sClockSourceConfig.ClockSource =
TIM_CLOCKSOURCE_INTERNAL;
```

```
272             if (HAL_TIM_ConfigClockSource(&htim3, &sClockSourceConfig) !=
HAL_OK)
273             {
274             Error_Handler();
275             }
276             if (HAL_TIM_IC_Init(&htim3) != HAL_OK)
277             {
278             Error_Handler();
279             }
280             sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
281             sMasterConfig.MasterSlaveMode =
TIM_MASTERSLAVEMODE_DISABLE;
282             if (HAL_TIMEx_MasterConfigSynchronization(&htim3,
&sMasterConfig) != HAL_OK)
283             {
284             Error_Handler();
285             }
286             sConfigIC.ICPolarity =
TIM_INPUTCHANNELPOLARITY_BOTHEDGE;
287             sConfigIC.ICSelection = TIM_ICSELECTION_DIRECTTI;
288             sConfigIC.ICPrescaler = TIM_ICPSC_DIV1;
289             sConfigIC.ICFilter = 0;
290             if (HAL_TIM_IC_ConfigChannel(&htim3, &sConfigIC,
TIM_CHANNEL_1) != HAL_OK)
291             {
292             Error_Handler();
293             }
294             /* USER CODE BEGIN TIM3_Init 2 */
295
296             /* USER CODE END TIM3_Init 2 */
297
298             }
299
300             /**
301             * @brief TIM4 Initialization Function
302             * @param None
303             * @retval None
304             */
305             static void MX_TIM4_Init(void)
306             {
```

```
307
308         /* USER CODE BEGIN TIM4_Init 0 */
309
310         /* USER CODE END TIM4_Init 0 */
311
312         TIM_ClockConfigTypeDef sClockSourceConfig = {0};
313         TIM_MasterConfigTypeDef sMasterConfig = {0};
314         TIM_OC_InitTypeDef sConfigOC = {0};
315
316         /* USER CODE BEGIN TIM4_Init 1 */
317
318         /* USER CODE END TIM4_Init 1 */
319         htim4.Instance = TIM4;
320         htim4.Init.Prescaler = 1800;
321         htim4.Init.CounterMode = TIM_COUNTERMODE_UP;
322         htim4.Init.Period = 2500;
323         htim4.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
324         htim4.Init.AutoReloadPreload =
TIM_AUTORELOAD_PRELOAD_DISABLE;
325         if (HAL_TIM_Base_Init(&htim4) != HAL_OK)
326         {
327         Error_Handler();
328         }
329         sClockSourceConfig.ClockSource =
TIM_CLOCKSOURCE_INTERNAL;
330         if (HAL_TIM_ConfigClockSource(&htim4, &sClockSourceConfig) !=
HAL_OK)
331         {
332         Error_Handler();
333         }
334         if (HAL_TIM_PWM_Init(&htim4) != HAL_OK)
335         {
336         Error_Handler();
337         }
338         sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
339         sMasterConfig.MasterSlaveMode =
TIM_MASTERSLAVEMODE_DISABLE;
340         if (HAL_TIMEx_MasterConfigSynchronization(&htim4,
&sMasterConfig) != HAL_OK)
341         {
```

```
342             Error_Handler();
343             }
344             sConfigOC.OCMode = TIM_OCMODE_PWM1;
345             sConfigOC.Pulse = 1;
346             sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
347             sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
348             if (HAL_TIM_PWM_ConfigChannel(&htim4, &sConfigOC,
TIM_CHANNEL_1) != HAL_OK)
349             {
350             Error_Handler();
351             }
352             /* USER CODE BEGIN TIM4_Init 2 */
353
354             /* USER CODE END TIM4_Init 2 */
355             HAL_TIM_MspPostInit(&htim4);
356
357             }
358
359             /**
360             * @brief TIM5 Initialization Function
361             * @param None
362             * @retval None
363             */
364             static void MX_TIM5_Init(void)
365             {
366
367             /* USER CODE BEGIN TIM5_Init 0 */
368
369             /* USER CODE END TIM5_Init 0 */
370
371             TIM_ClockConfigTypeDef sClockSourceConfig = {0};
372             TIM_MasterConfigTypeDef sMasterConfig = {0};
373             TIM_IC_InitTypeDef sConfigIC = {0};
374
375             /* USER CODE BEGIN TIM5_Init 1 */
376
377             /* USER CODE END TIM5_Init 1 */
378             htim5.Instance = TIM5;
379             htim5.Init.Prescaler = 0;
380             htim5.Init.CounterMode = TIM_COUNTERMODE_UP;
```

```
  htim5.Init.Period = 4294967295;
  htim5.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim5.Init.AutoReloadPreload =
TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim5) != HAL_OK)
  {
  Error_Handler();
  }
  sClockSourceConfig.ClockSource =
TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim5, &sClockSourceConfig) !=
HAL_OK)
  {
  Error_Handler();
  }
  if (HAL_TIM_IC_Init(&htim5) != HAL_OK)
  {
  Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode =
TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim5,
&sMasterConfig) != HAL_OK)
  {
  Error_Handler();
  }
  sConfigIC.ICPolarity =
TIM_INPUTCHANNELPOLARITY_BOTHEDGE;
  sConfigIC.ICSelection = TIM_ICSELECTION_DIRECTTI;
  sConfigIC.ICPrescaler = TIM_ICPSC_DIV1;
  sConfigIC.ICFilter = 0;
  if (HAL_TIM_IC_ConfigChannel(&htim5, &sConfigIC,
TIM_CHANNEL_3) != HAL_OK)
  {
  Error_Handler();
  }
  /* USER CODE BEGIN TIM5_Init 2 */

  /* USER CODE END TIM5_Init 2 */
```

```
    414
    415             }
    416
    417             /**
    418             * @brief GPIO Initialization Function
    419             * @param None
    420             * @retval None
    421             */
    422             static void MX_GPIO_Init(void)
    423             {
    424             GPIO_InitTypeDef GPIO_InitStruct = {0};
    425
    426             /* GPIO Ports Clock Enable */
    427             __HAL_RCC_GPIOE_CLK_ENABLE();
    428             __HAL_RCC_GPIOH_CLK_ENABLE();
    429             __HAL_RCC_GPIOC_CLK_ENABLE();
    430             __HAL_RCC_GPIOA_CLK_ENABLE();
    431             __HAL_RCC_GPIOD_CLK_ENABLE();
    432
    433             /*Configure GPIO pin : PC1 */
    434             GPIO_InitStruct.Pin = GPIO_PIN_1;
    435             GPIO_InitStruct.Mode = GPIO_MODE_INPUT;
    436             GPIO_InitStruct.Pull = GPIO_NOPULL;
    437             HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);
    438
    439             }
    440
    441             /* USER CODE BEGIN 4 */

\                           In section .text, align 2, keep-with-next
    442             void LCD_SendCommand (uint8_t cmd){
\             LCD_SendCommand: (+1)
\       0x0  0xB538         PUSH         {R3-R5,LR}
    443                      data[0] = 0xF8;
\       0x2  0x....         LDR.N        R4,??DataTable7_12
\       0x4  0x21F8         MOVS         R1,#+248
    444                      data[1] = (cmd & 0xF0);
\       0x6  0x....         B.N    ?Subroutine0
    445                      data[2] = ((cmd << 4) & 0xF0);
```

```
      446                     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_0,
GPIO_PIN_SET); // CS to high
      447             HAL_SPI_Transmit(&hspi4, data, 3, 1);
      448                     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_0,
GPIO_PIN_RESET); // CS to low
      449             }

  \                             In section .text, align 2, keep-with-next
  \              ?Subroutine0: (+1)
  \      0x0   0xF000 0x02F0        AND   R2,R0,#0xF0
  \      0x4   0x0100               LSLS  R0,R0,#+4
  \      0x6   0x7021               STRB R1,[R4, #+0]
  \      0x8   0x7062               STRB R2,[R4, #+1]
  \      0xA   0x70A0               STRB R0,[R4, #+2]
  \      0xC   0x2201               MOVS          R2,#+1
  \      0xE   0x....               LDR.N         R5,??DataTable7_13
  \      0x10   0x2101              MOVS          R1,#+1
  \      0x12   0x4628              MOV   R0,R5
  \      0x14   0x.... 0x....       BL      HAL_GPIO_WritePin
  \      0x18   0x2301              MOVS          R3,#+1
  \      0x1A   0x2203              MOVS          R2,#+3
  \      0x1C   0x4621              MOV   R1,R4
  \      0x1E   0x....              LDR.N         R0,??DataTable7_7
  \      0x20   0x.... 0x....       BL      HAL_SPI_Transmit
  \      0x24   0x4628              MOV   R0,R5
  \      0x26   0xE8BD 0x4038       POP   {R3-R5,LR}
  \      0x2A   0x2200              MOVS          R2,#+0
  \      0x2C   0x2101              MOVS          R1,#+1
  \      0x2E   0x.... 0x....       B.W    HAL_GPIO_WritePin
      450

  \                             In section .text, align 2, keep-with-next
      451             void LCD_Init() {
  \              LCD_Init: (+1)
  \      0x0   0xB510               PUSH          {R4,LR}
      452             HAL_GPIO_WritePin(GPIOD, GPIO_PIN_1, GPIO_PIN_RESET); //
Make sure reset is low
  \      0x2   0x....               LDR.N         R4,??DataTable7_13
  \      0x4   0x2200               MOVS          R2,#+0
  \      0x6   0x2102               MOVS          R1,#+2
```

```
\      0x8   0x4620          MOV   R0,R4
\      0xA   0x.... 0x....   BL    HAL_GPIO_WritePin
       453             HAL_Delay(100);
\      0xE   0x2064          MOVS        R0,#+100
\      0x10  0x.... 0x....   BL    HAL_Delay
       454             HAL_GPIO_WritePin(GPIOD, GPIO_PIN_1, GPIO_PIN_SET); //
Reset Low to high
\      0x14  0x2201          MOVS        R2,#+1
\      0x16  0x2102          MOVS        R1,#+2
\      0x18  0x4620          MOV   R0,R4
\      0x1A  0x.... 0x....   BL    HAL_GPIO_WritePin
       455             HAL_Delay(50); // greater than 40ms
\      0x1E  0x2032          MOVS        R0,#+50
\      0x20  0x.... 0x....   BL    HAL_Delay
       456             LCD_SendCommand(Function_Set);
\      0x24  0x2030          MOVS        R0,#+48
\      0x26  0x.... 0x....   BL    LCD_SendCommand
       457             HAL_Delay(10); // greater than 100us
\      0x2A  0x200A                MOVS        R0,#+10
\      0x2C  0x.... 0x....   BL    HAL_Delay
       458             LCD_SendCommand(Function_Set);
\      0x30  0x2030          MOVS        R0,#+48
\      0x32  0x.... 0x....   BL    LCD_SendCommand
       459             HAL_Delay(10); // greater than 37us
\      0x36  0x200A          MOVS        R0,#+10
\      0x38  0x.... 0x....   BL    HAL_Delay
       460             LCD_SendCommand(Display_On);
\      0x3C  0x200C                MOVS        R0,#+12
\      0x3E  0x.... 0x....   BL    LCD_SendCommand
       461             HAL_Delay(10); // greater than 100us
\      0x42  0x200A          MOVS        R0,#+10
\      0x44  0x.... 0x....   BL    HAL_Delay
       462             LCD_SendCommand(Display_Clear);
\      0x48  0x2001          MOVS        R0,#+1
\      0x4A  0x.... 0x....   BL    LCD_SendCommand
       463             HAL_Delay(15); // greater than 10ms
\      0x4E  0x200F          MOVS        R0,#+15
\      0x50  0x.... 0x....   BL    HAL_Delay
       464             LCD_SendCommand(Entry_Mode_Set);
\      0x54  0x2006          MOVS        R0,#+6
```

```
\     0x56   0x.... 0x....     BL      LCD_SendCommand
      465            HAL_Delay(10); // greater than 100us
\     0x5A   0xE8BD 0x4010     POP   {R4,LR}
\     0x5E   0x200A            MOVS          R0,#+10
\     0x60   0x.... 0x....     B.W    HAL_Delay
      466            }
      467

\                              In section .text, align 2, keep-with-next
      468            void LCD_Clear(){
      469            LCD_SendCommand(Display_Clear);
\             LCD_Clear: (+1)
\     0x0   0x2001           MOVS          R0,#+1
\     0x2   0x....           B.N    LCD_SendCommand
      470            }
      471

\                              In section .text, align 2, keep-with-next
      472            void LCD_SendData (uint8_t ascii){
\             LCD_SendData: (+1)
\     0x0   0xB538           PUSH          {R3-R5,LR}
      473                     data[0] = 0xFA;
\     0x2   0x....           LDR.N         R4,??DataTable7_12
\     0x4   0x21FA           MOVS          R1,#+250
      474                     data[1] = (ascii & 0xF0);
\     0x6                    REQUIRE ?Subroutine0
\     0x6                    ;; // Fall through to label ?Subroutine0
      475                     data[2] = ((ascii << 4) & 0xF0);
      476                     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_0,
GPIO_PIN_SET); // CS to high
      477            HAL_SPI_Transmit(&hspi4, data, 3, 1);
      478                     HAL_GPIO_WritePin(GPIOD, GPIO_PIN_0,
GPIO_PIN_RESET); // CS to low
      479            }
      480

\                              In section .text, align 2, keep-with-next
      481            void LCD_sendString(char* in, int length)
      482            {
\             LCD_sendString: (+1)
```

```
\     0x0  0xB570        PUSH        {R4-R6,LR}
\     0x2  0x4604        MOV  R4,R0
\     0x4  0x460D        MOV  R5,R1
\     0x6  0xE003        B.N  ??LCD_sendString_0
      483           uint8_t letter;
      484           for (int i; i<length; i++) //iterate through the input string
      485           {
      486           letter = in[i]; // assign the letter to the int, getting its ascii value, this
may be unnecessary
      487           LCD_SendData(letter); // send the data to the board
\           ??LCD_sendString_1: (+1)
\     0x8  0x5DA0        LDRB R0,[R4, R6]
\     0xA  0x.... 0x....      BL      LCD_SendData
      488           }
\     0xE  0x1C76        ADDS        R6,R6,#+1
\           ??LCD_sendString_0: (+1)
\     0x10 0x42AE              CMP  R6,R5
\     0x12 0xDBF9              BLT.N ??LCD_sendString_1
      489           }
\     0x14 0xBD70              POP  {R4-R6,PC}
      490

\                        In section .text, align 4, keep-with-next
      491           void LCD_Sel_Row_Col(uint8_t row, uint8_t col)
      492           {
\           LCD_Sel_Row_Col: (+1)
\     0x0  0xB580        PUSH        {R7,LR}
      493           switch(row)
\     0x2  0x084A        LSRS R2,R1,#+1
\     0x4  0x2803        CMP  R0,#+3
\     0x6  0xD80E        BHI.N ??LCD_Sel_Row_Col_1
\     0x8  0xE8DF 0xF000       TBB  [PC, R0]
\           ??LCD_Sel_Row_Col_0:
\     0xC  0x02 0x05     DC8  0x2,0x5,0x8,0xB

\           0x08 0x0B
      494                  {
      495                  case 0:
      496                        col /= 2; //cursor is 2 spaces wide
      497                        col |= 0x80; // 0b10000000 first row
```

```
\             ??LCD_Sel_Row_Col_2: (+1)
\     0x10  0xF042 0x0180       ORR   R1,R2,#0x80
      498                 break;
\     0x14  0xE007        B.N     ??LCD_Sel_Row_Col_1
      499           case 1:
      500                 col /= 2;
      501                 col |= 0x90; //second column starts at 0x90
\             ??LCD_Sel_Row_Col_3: (+1)
\     0x16  0xF042 0x0190       ORR   R1,R2,#0x90
      502                 break;
\     0x1A  0xE004              B.N     ??LCD_Sel_Row_Col_1
      503           case 2:
      504                 col /= 2;
      505                 col |= 0x88; // third row is 0x88
\             ??LCD_Sel_Row_Col_4: (+1)
\     0x1C  0xF042 0x0188       ORR   R1,R2,#0x88
      506                 break;
\     0x20  0xE001        B.N     ??LCD_Sel_Row_Col_1
      507           case 3:
      508                 col /= 2;
      509                 col |= 0x98; //fourth row is 0x98
\             ??LCD_Sel_Row_Col_5: (+1)
\     0x22  0xF042 0x0198       ORR   R1,R2,#0x98
      510                 break;
      511           }
      512           LCD_SendCommand(col);
\             ??LCD_Sel_Row_Col_1: (+1)
\     0x26  0x4608        MOV   R0,R1
\     0x28  0x.... 0x....     BL      LCD_SendCommand
      513           HAL_Delay(5);
\     0x2C  0xE8BD 0x4002     POP   {R1,LR}
\     0x30  0x2005        MOVS          R0,#+5
\     0x32  0x.... 0x....     B.W     HAL_Delay
      514           }
      515


\                         In section .text, align 2, keep-with-next
      516           void int_to_str(int number) {
\             int_to_str: (+1)
\     0x0   0xB510          PUSH          {R4,LR}
```

```
    517             int Hex_offset = 0x30;
    518             int first_dig = (number % 1000)/100;  //separate the hundreds digit
    519             first_dig = first_dig + Hex_offset;
    520             num_out[0] = first_dig;
\   0x2   0xF44F 0x737A       MOV  R3,#+1000
\   0x6   0xFB90 0xF3F3       SDIV  R3,R0,R3
\   0xA   0xF44F 0x717A       MOV  R1,#+1000
\   0xE   0xFB01 0x0313       MLS  R3,R1,R3,R0
\   0x12  0x2464       MOVS       R4,#+100
\   0x14  0x....       LDR.N      R2,??DataTable7_9
\   0x16  0xFB93 0xF4F4       SDIV  R4,R3,R4
\   0x1A  0x3430       ADDS       R4,R4,#+48
\   0x1C  0x7014       STRB R4,[R2, #+0]
    521             int second_dig = (number % 100)/10 ; //separate the tens digit
    522             second_dig = second_dig + Hex_offset;
    523             num_out[1] = second_dig;
\   0x1E  0x2364       MOVS       R3,#+100
\   0x20  0xFB90 0xF3F3       SDIV  R3,R0,R3
\   0x24  0x2164       MOVS       R1,#+100
\   0x26  0xFB01 0x0313       MLS  R3,R1,R3,R0
\   0x2A  0x240A               MOVS       R4,#+10
\   0x2C  0xFB93 0xF4F4       SDIV  R4,R3,R4
    524             int third_dig = number % 10;  // separate the ones place
    525             third_dig = third_dig + Hex_offset;
    526             num_out[2] = third_dig;
\   0x30  0x230A       MOVS       R3,#+10
\   0x32  0xFB90 0xF3F3       SDIV  R3,R0,R3
\   0x36  0x210A       MOVS       R1,#+10
\   0x38  0x3430       ADDS       R4,R4,#+48
\   0x3A  0xFB01 0x0013       MLS  R0,R1,R3,R0
\   0x3E  0x7054       STRB R4,[R2, #+1]
\   0x40  0x3030       ADDS       R0,R0,#+48
\   0x42  0x7090       STRB R0,[R2, #+2]
    527             }
\   0x44  0xBD10               POP  {R4,PC}
    528             /* USER CODE END 4 */
    529
    530             /**
    531             * @brief  This function is executed in case of error occurrence.
    532             * @retval None
```

```
        533             */

  \                             In section .text, align 2, keep-with-next
        534             void Error_Handler(void)
        535             {
        536             /* USER CODE BEGIN Error_Handler_Debug */
        537             /* User can add his own implementation to report the HAL error
return state */
        538             __disable_irq();
  \               Error_Handler: (+1)
  \       0x0   0xB672          CPSID         I
        539             while (1)
  \               ??Error_Handler_0: (+1)
  \       0x2   0xE7FE          B.N     ??Error_Handler_0
        540             {
        541             }
        542             /* USER CODE END Error_Handler_Debug */
        543             }

  \                             In section .text, align 2, keep-with-next
  \               ?Subroutine5: (+1)
  \       0x0   0x9102          STR    R1,[SP, #+8]
  \       0x2   0x2201          MOVS          R2,#+1
  \       0x4   0x9203          STR    R2,[SP, #+12]
  \       0x6   0x2100          MOVS          R1,#+0
  \       0x8   0x9104          STR    R1,[SP, #+16]
  \       0xA   0x2200          MOVS          R2,#+0
  \       0xC   0x4770          BX      LR

  \                             In section .text, align 2, keep-with-next
  \               ?Subroutine2: (+1)
  \       0x0   0x2210          MOVS          R2,#+16
  \               ??Subroutine2_0: (+1)
  \       0x2   0x2100          MOVS          R1,#+0
  \       0x4   0xA802          ADD    R0,SP,#+8
  \       0x6   0x.... 0x....       B.W    memset

  \                             In section .text, align 2, keep-with-next
  \               ?Subroutine1: (+1)
  \       0x0   0x2208          MOVS          R2,#+8
```

```
\      0x2   0x2100          MOVS          R1,#+0
\      0x4   0x4668          MOV   R0,SP
\      0x6   0x.... 0x....   B.W   memset

\                            In section .text, align 4, keep-with-next
\             ??DataTable7:
\      0x0   0x63 0x6D       DC8   0x63, 0x6D, 0x00, 0x00

\             0x00 0x00

\                            In section .text, align 4, keep-with-next
\             ??DataTable7_1:
\      0x0   0x4002'0800             DC32  0x40020800

\                            In section .text, align 4, keep-with-next
\             ??DataTable7_2:
\      0x0   0x4002'3830             DC32  0x40023830

\                            In section .text, align 4, keep-with-next
\             ??DataTable7_3:
\      0x0   0x....'....     DC32  htim3

\                            In section .text, align 4, keep-with-next
\             ??DataTable7_4:
\      0x0   0x4000'0800             DC32  0x40000800

\                            In section .text, align 4, keep-with-next
\             ??DataTable7_5:
\      0x0   0x4000'0C00             DC32  0x40000c00

\                            In section .text, align 4, keep-with-next
\             ??DataTable7_6:
\      0x0   0x4000'0400             DC32  0x40000400

\                            In section .text, align 4, keep-with-next
\             ??DataTable7_7:
\      0x0   0x....'....     DC32  hspi4

\                            In section .text, align 4, keep-with-next
\             ??DataTable7_8:
```

```
\      0x0   0x4001'3400            DC32  0x40013400

\                                   In section .text, align 4, keep-with-next
\             ??DataTable7_9:
\      0x0   0x....'....           DC32  num_out

\                                   In section .text, align 4, keep-with-next
\             ??DataTable7_10:
\      0x0   0x4002'3840            DC32  0x40023840

\                                   In section .text, align 4, keep-with-next
\             ??DataTable7_11:
\      0x0   0x4000'7000            DC32  0x40007000

\                                   In section .text, align 4, keep-with-next
\             ??DataTable7_12:
\      0x0   0x....'....           DC32  `data`

\                                   In section .text, align 4, keep-with-next
\             ??DataTable7_13:
\      0x0   0x4002'0C00            DC32  0x40020c00

\                                   In section .rodata, align 4, keep-with-next
\             ?_1:
\      0x0   0x63 0x6D      DC8 "cm"

\             0x00
\      0x3                  DS8 1

\                                   In section .text, align 4, keep-with-next
\             ?_0:
\      0x0   0x69 0x6E      DC8 "inches"

\             0x63 0x68

\             0x65 0x73

\             0x00
\      0x7                  DS8 1
       544
```

```
        545             #ifdef  USE_FULL_ASSERT
        546             /**
        547             * @brief  Reports the name of the source file and the source line
number
        548             *         where the assert_param error has occurred.
        549             * @param  file: pointer to the source file name
        550             * @param  line: assert_param error line source number
        551             * @retval None
        552             */
        553             void assert_failed(uint8_t *file, uint32_t line)
        554             {
        555             /* USER CODE BEGIN 6 */
        556             /* User can add his own implementation to report the file name and
line number,
        557             ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line)
*/
        558             /* USER CODE END 6 */
        559             }
        560             #endif /* USE_FULL_ASSERT */
```

Maximum stack usage in bytes:

```
.cstack Function
------- --------
      0   Error_Handler
      0   LCD_Clear
      0   -> LCD_SendCommand
      8   LCD_Init
      0   -> HAL_Delay
      8   -> HAL_Delay
      8   -> HAL_GPIO_WritePin
      8   -> LCD_SendCommand
      8   LCD_Sel_Row_Col
      0   -> HAL_Delay
      8   -> LCD_SendCommand
     16   LCD_SendCommand
      0   -> HAL_GPIO_WritePin
     16   -> HAL_GPIO_WritePin
     16   -> HAL_SPI_Transmit
     16   LCD_SendData
```

```
0   -> HAL_GPIO_WritePin
16   -> HAL_GPIO_WritePin
16   -> HAL_SPI_Transmit
16   LCD_sendString
16   -> LCD_SendData
80   SystemClock_Config
80   -> Error_Handler
80   -> HAL_PWREx_EnableOverDrive
80   -> HAL_RCC_ClockConfig
80   -> HAL_RCC_OscConfig
80   -> memset
8   int_to_str
88   main
88   -> Error_Handler
88   -> HAL_GPIO_Init
88   -> HAL_GPIO_ReadPin
88   -> HAL_Init
88   -> HAL_SPI_Init
88   -> HAL_TIMEx_MasterConfigSynchronization
88   -> HAL_TIM_Base_Init
88   -> HAL_TIM_ConfigClockSource
88   -> HAL_TIM_IC_ConfigChannel
88   -> HAL_TIM_IC_Init
88   -> HAL_TIM_IC_Start_IT
88   -> HAL_TIM_MspPostInit
88   -> HAL_TIM_PWM_ConfigChannel
88   -> HAL_TIM_PWM_Init
88   -> HAL_TIM_PWM_Start
88   -> LCD_Init
88   -> LCD_Sel_Row_Col
88   -> LCD_SendData
88   -> LCD_sendString
88   -> SystemClock_Config
88   -> int_to_str
88   -> memset
```

Section sizes:

Bytes Function/Label

```
-----  --------------
       4  ??DataTable7
       4  ??DataTable7_1
       4  ??DataTable7_10
       4  ??DataTable7_11
       4  ??DataTable7_12
       4  ??DataTable7_13
       4  ??DataTable7_2
       4  ??DataTable7_3
       4  ??DataTable7_4
       4  ??DataTable7_5
       4  ??DataTable7_6
       4  ??DataTable7_7
       4  ??DataTable7_8
       4  ??DataTable7_9
       50  ?Subroutine0
       10  ?Subroutine1
       10  ?Subroutine2
       10  ?Subroutine3
       10  ?Subroutine4
       14  ?Subroutine5
       8  ?_0
       4  ?_1
       4  Error_Handler
       4  LCD_Clear
       100  LCD_Init
       54  LCD_Sel_Row_Col
       8  LCD_SendCommand
       6  LCD_SendData
       22  LCD_sendString
       162  SystemClock_Config
       4  data
       4  echo_time
       88  hspi4
       224  htim3
       htim4
       htim5
       echo_dist
       flag
       70  int_to_str
```

```
        742  main
          4  num_out


  316 bytes in section .bss
          8 bytes in section .data
          4 bytes in section .rodata
1'340 bytes in section .text

1'340 bytes of CODE  memory
          4 bytes of CONST memory
  324 bytes of DATA  memory

Errors: none
Warnings: none


lt.lst
#############################################################################
##########
#
# IAR ANSI C/C++ Compiler V9.20.4.327/W64 for ARM         16/Mar/2023  18:02:45
# Copyright 1999-2022 IAR Systems AB.
#
#       Cpu mode             =  thumb
#       Endian          =  little
#       Source file     =
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\Cor
e\Src\stm32f4xx_it.c
#       Command line         =
#       -f
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core\stm32f4xx_it.o.rsp
#
(S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\Cor
e\Src\stm32f4xx_it.c
#       -D USE_HAL_DRIVER -D STM32F429xx -lC
```

```
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\List\Application\User\Core
#        -o
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core
#        --debug --endian=little --cpu=Cortex-M4 -e --fpu=VFPv4_sp
#        --dlib_config S:\School_Work\arm\inc\c\DLib_Config_Full.h -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Core/Inc\
#        -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/STM32F4xx_HAL_Driver/Inc\
#        -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/STM32F4xx_HAL_Driver/Inc/Legacy\
#        -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/CMSIS/Device/ST/STM32F4xx/Include\
#        -I
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM/../Drivers/CMSIS/Include\
#        -Ohz) --dependencies=n
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core\stm32f4xx_it.o.d
#        Locale          =  C
#        List file       =
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\List\Application\User\Core\stm32f4xx_it.lst
#        Object file     =
```

```
#
S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\EW
ARM\Prelab9_task2\Obj\Application\User\Core\stm32f4xx_it.o
#       Runtime model:
#       __CPP_Runtime   =  1
#       __SystemLibrary =  DLib
#       __dlib_version  =  6
#       __size_limit    =  32768|ARM.EW.LINKER
#
#############################################################################
#########

S:\School_Work\Spring_2023\MicroconrollerApps\Prelabs\Prelab_9\Prelab9_task2\Cor
e\Src\stm32f4xx_it.c
        1       /* USER CODE BEGIN Header */
        2       /**
        3       ******************************************************************************
        4       * @filestm32f4xx_it.c
        5       * @brief   Interrupt Service Routines.
        6       ******************************************************************************
        7       * @attention
        8       *
        9       * Copyright (c) 2023 STMicroelectronics.
        10              * All rights reserved.
        11              *
        12              * This software is licensed under terms that can be found in the
LICENSE file
        13              * in the root directory of this software component.
        14              * If no LICENSE file comes with this software, it is provided AS-IS.
        15              *
        16
******************************************************************************
        17              */
        18      /* USER CODE END Header */
        19
        20      /* Includes ------------------------------------------------------------*/
        21      #include "main.h"
        22      #include "stm32f4xx_it.h"
        23      /* Private includes ----------------------------------------------------------*/
        24      /* USER CODE BEGIN Includes */
```

```
   25      /* USER CODE END Includes */
   26
   27      /* Private typedef -----------------------------------------------------------*/
   28      /* USER CODE BEGIN TD */
   29
   30      /* USER CODE END TD */
   31
   32      /* Private define ------------------------------------------------------------*/
   33      /* USER CODE BEGIN PD */
   34
   35      /* USER CODE END PD */
   36
   37      /* Private macro -------------------------------------------------------------*/
   38      /* USER CODE BEGIN PM */
   39
   40      /* USER CODE END PM */
   41
   42      /* Private variables ---------------------------------------------------------*/
   43      /* USER CODE BEGIN PV */

\                               In section .bss, align 4
   44      int rising = 0;
\          rising:
\  0x0                          DS8 4
   45      /* USER CODE END PV */
   46
   47      /* Private function prototypes -----------------------------------------------*/
   48      /* USER CODE BEGIN PFP */
   49
   50      /* USER CODE END PFP */
   51
   52      /* Private user code ---------------------------------------------------------*/
   53      /* USER CODE BEGIN 0 */
   54
   55      /* USER CODE END 0 */
   56
   57      /* External variables --------------------------------------------------------*/
   58      extern TIM_HandleTypeDef htim3;
   59      extern TIM_HandleTypeDef htim4;
   60      extern TIM_HandleTypeDef htim5;
```

```
        61      /* USER CODE BEGIN EV */
        62      extern int echo_time;
        63      extern int echo_dist;
        64      extern void LCD_Init();
        65      extern void LCD_SendCommand(uint8_t cmd);
        66      extern void LCD_SendData(uint8_t data);
        67      extern void LCD_Sel_Row_Col(uint8_t row, uint8_t col);
        68      extern void LCD_sendString(char* in, int length);
        69      extern void LCD_Clear();
        70      extern void int_to_str(int number);
        71      extern uint8_t num_out [3];
        72      extern int flag;
        73      /* USER CODE END EV */
        74
        75      /******************************************************************************/
        76      /*       Cortex-M4 Processor Interruption and Exception Handlers
*/
        77      /******************************************************************************/
        78      /**
        79              * @brief This function handles Non maskable interrupt.
        80              */

  \                             In section .text, align 2, keep-with-next
        81      void NMI_Handler(void)
        82      {
        83              /* USER CODE BEGIN NonMaskableInt_IRQn 0 */
        84
        85              /* USER CODE END NonMaskableInt_IRQn 0 */
        86              /* USER CODE BEGIN NonMaskableInt_IRQn 1 */
        87              while (1)
  \             NMI_Handler: (+1)
  \             ??NMI_Handler_0: (+1)
  \     0x0   0xE7FE          B.N     ??NMI_Handler_0
        88              {
        89              }
        90              /* USER CODE END NonMaskableInt_IRQn 1 */
        91      }
        92
        93      /**
        94              * @brief This function handles Hard fault interrupt.
```

```
     95             */

\                           In section .text, align 2, keep-with-next
     96     void HardFault_Handler(void)
     97     {
     98             /* USER CODE BEGIN HardFault_IRQn 0 */
     99
     100            /* USER CODE END HardFault_IRQn 0 */
     101            while (1)
\            HardFault_Handler: (+1)
\            ??HardFault_Handler_0: (+1)
\    0x0  0xE7FE         B.N    ??HardFault_Handler_0
     102            {
     103            /* USER CODE BEGIN W1_HardFault_IRQn 0 */
     104            /* USER CODE END W1_HardFault_IRQn 0 */
     105            }
     106            }
     107
     108            /**
     109            * @brief This function handles Memory management fault.
     110            */

\                           In section .text, align 2, keep-with-next
     111            void MemManage_Handler(void)
     112            {
     113            /* USER CODE BEGIN MemoryManagement_IRQn 0 */
     114
     115            /* USER CODE END MemoryManagement_IRQn 0 */
     116            while (1)
\            MemManage_Handler: (+1)
\            ??MemManage_Handler_0: (+1)
\    0x0  0xE7FE         B.N    ??MemManage_Handler_0
     117            {
     118            /* USER CODE BEGIN W1_MemoryManagement_IRQn 0 */
     119            /* USER CODE END W1_MemoryManagement_IRQn 0 */
     120            }
     121            }
     122
     123            /**
     124            * @brief This function handles Pre-fetch fault, memory access fault.
```

```
      125           */

\                           In section .text, align 2, keep-with-next
      126           void BusFault_Handler(void)
      127           {
      128           /* USER CODE BEGIN BusFault_IRQn 0 */
      129
      130           /* USER CODE END BusFault_IRQn 0 */
      131           while (1)
\             BusFault_Handler: (+1)
\             ??BusFault_Handler_0: (+1)
\         0x0 0xE7FE        B.N     ??BusFault_Handler_0
      132           {
      133           /* USER CODE BEGIN W1_BusFault_IRQn 0 */
      134           /* USER CODE END W1_BusFault_IRQn 0 */
      135           }
      136           }
      137
      138           /**
      139           * @brief This function handles Undefined instruction or illegal state.
      140           */

\                           In section .text, align 2, keep-with-next
      141           void UsageFault_Handler(void)
      142           {
      143           /* USER CODE BEGIN UsageFault_IRQn 0 */
      144
      145           /* USER CODE END UsageFault_IRQn 0 */
      146           while (1)
\             UsageFault_Handler: (+1)
\             ??UsageFault_Handler_0: (+1)
\         0x0 0xE7FE        B.N     ??UsageFault_Handler_0
      147           {
      148           /* USER CODE BEGIN W1_UsageFault_IRQn 0 */
      149           /* USER CODE END W1_UsageFault_IRQn 0 */
      150           }
      151           }
      152
      153           /**
```

```
        154             * @brief This function handles System service call via SWI
instruction.
        155             */

  \                             In section .text, align 2, keep-with-next
        156             void SVC_Handler(void)
        157             {
        158             /* USER CODE BEGIN SVCall_IRQn 0 */
        159
        160             /* USER CODE END SVCall_IRQn 0 */
        161             /* USER CODE BEGIN SVCall_IRQn 1 */
        162
        163             /* USER CODE END SVCall_IRQn 1 */
        164             }
  \               SVC_Handler: (+1)
  \       0x0   0x4770          BX      LR
        165
        166             /**
        167             * @brief This function handles Debug monitor.
        168             */

  \                             In section .text, align 2, keep-with-next
        169             void DebugMon_Handler(void)
        170             {
        171             /* USER CODE BEGIN DebugMonitor_IRQn 0 */
        172
        173             /* USER CODE END DebugMonitor_IRQn 0 */
        174             /* USER CODE BEGIN DebugMonitor_IRQn 1 */
        175
        176             /* USER CODE END DebugMonitor_IRQn 1 */
        177             }
  \               DebugMon_Handler: (+1)
  \       0x0   0x4770          BX      LR
        178
        179             /**
        180             * @brief This function handles Pendable request for system
service.
        181             */

  \                             In section .text, align 2, keep-with-next
```

```
       182            void PendSV_Handler(void)
       183            {
       184            /* USER CODE BEGIN PendSV_IRQn 0 */
       185
       186            /* USER CODE END PendSV_IRQn 0 */
       187            /* USER CODE BEGIN PendSV_IRQn 1 */
       188
       189            /* USER CODE END PendSV_IRQn 1 */
       190            }
 \              PendSV_Handler: (+1)
 \      0x0  0x4770          BX      LR
       191
       192            /**
       193            * @brief This function handles System tick timer.
       194            */

 \                              In section .text, align 2, keep-with-next
       195            void SysTick_Handler(void)
       196            {
       197            /* USER CODE BEGIN SysTick_IRQn 0 */
       198
       199            /* USER CODE END SysTick_IRQn 0 */
       200            HAL_IncTick();
 \              SysTick_Handler: (+1)
 \      0x0  0x.... 0x....       B.W    HAL_IncTick
       201            /* USER CODE BEGIN SysTick_IRQn 1 */
       202
       203            /* USER CODE END SysTick_IRQn 1 */
       204            }
       205
       206
/******************************************************************************/
       207            /* STM32F4xx Peripheral Interrupt Handlers
*/
       208            /* Add here the Interrupt Handlers for the used peripherals.
*/
       209            /* For the available peripheral interrupt handler names,
*/
       210            /* please refer to the startup file (startup_stm32f4xx.s).
*/
```

```
      211
/*******************************************************************************/
      212
      213             /**
      214             * @brief This function handles TIM3 global interrupt.
      215             */

  \                                 In section .text, align 2, keep-with-next
      216             void TIM3_IRQHandler(void)
      217             {
  \             TIM3_IRQHandler: (+1)
  \     0x0  0xB538         PUSH         {R3-R5,LR}
      218             /* USER CODE BEGIN TIM3_IRQn 0 */
      219             if(flag == 0){
  \     0x2  0x....         LDR.N        R3,??DataTable3
  \     0x4  0x....         LDR.N        R1,??DataTable3_1
  \     0x6  0x6818         LDR   R0,[R3, #+0]
  \     0x8  0x....         LDR.N        R2,??DataTable3_2
  \     0xA  0xB918         CBNZ.N  R0,??TIM3_IRQHandler_0
      220             rising = TIM3 -> CNT;
  \     0xC  0x6812         LDR   R2,[R2, #+0]
  \     0xE  0x600A         STR   R2,[R1, #+0]
      221             flag = 1;
  \     0x10  0x2001        MOVS         R0,#+1
  \     0x12  0xE00C              B.N    ??TIM3_IRQHandler_1
      222             }
      223             else{
      224             echo_time = (TIM3 -> CNT) - rising;
  \             ??TIM3_IRQHandler_0: (+1)
  \     0x14  0x6814        LDR   R4,[R2, #+0]
  \     0x16  0x6808        LDR   R0,[R1, #+0]
  \     0x18  0x1A20        SUBS         R0,R4,R0
      225             echo_dist = echo_time / 6 + 1; //convert to cm
  \     0x1A  0x2106        MOVS         R1,#+6
  \     0x1C  0xFB90 0xF1F1      SDIV  R1,R0,R1
  \     0x20  0x....        LDR.N        R4,??DataTable3_3
      226             TIM3 -> CNT = 0;
  \     0x22  0x....        LDR.N        R5,??DataTable3_4
  \     0x24  0x1C49        ADDS         R1,R1,#+1
  \     0x26  0x6021        STR   R1,[R4, #+0]
```

```
\     0x28   0x6028         STR    R0,[R5, #+0]
\     0x2A   0x2000         MOVS           R0,#+0
\     0x2C   0x6010         STR    R0,[R2, #+0]
      227              flag = 0;
      228
      229              }
      230              /* USER CODE END TIM3_IRQn 0 */
      231              HAL_TIM_IRQHandler(&htim3);
\             ??TIM3_IRQHandler_1: (+1)
\     0x2E   0x6018         STR    R0,[R3, #+0]
\     0x30   0xE8BD 0x4032      POP    {R1,R4,R5,LR}
\     0x34   0x....             LDR.N          R0,??DataTable3_5
\     0x36   0x.... 0x....      B.W    HAL_TIM_IRQHandler
      232              /* USER CODE BEGIN TIM3_IRQn 1 */
      233
      234              /* USER CODE END TIM3_IRQn 1 */
      235              }
      236
      237              /**
      238              * @brief This function handles TIM4 global interrupt.
      239              */

\                              In section .text, align 2, keep-with-next
      240              void TIM4_IRQHandler(void)
      241              {
      242              /* USER CODE BEGIN TIM4_IRQn 0 */
      243
      244              /* USER CODE END TIM4_IRQn 0 */
      245              HAL_TIM_IRQHandler(&htim4);
\             TIM4_IRQHandler: (+1)
\     0x0    0x....             LDR.N          R0,??DataTable3_6
\     0x2    0x.... 0x....      B.W    HAL_TIM_IRQHandler
      246              /* USER CODE BEGIN TIM4_IRQn 1 */
      247
      248              /* USER CODE END TIM4_IRQn 1 */
      249              }
      250
      251              /**
      252              * @brief This function handles TIM5 global interrupt.
      253              */
```

```
\                               In section .text, align 2, keep-with-next
     254            void TIM5_IRQHandler(void)
     255            {
     256            /* USER CODE BEGIN TIM5_IRQn 0 */
     257
     258            /* USER CODE END TIM5_IRQn 0 */
     259            HAL_TIM_IRQHandler(&htim5);
\             TIM5_IRQHandler: (+1)
\     0x0  0x....             LDR.N          R0,??DataTable3_7
\     0x2  0x.... 0x....      B.W    HAL_TIM_IRQHandler
     260            /* USER CODE BEGIN TIM5_IRQn 1 */
     261
     262            /* USER CODE END TIM5_IRQn 1 */
     263            }

\                               In section .text, align 4, keep-with-next
\             ??DataTable3:
\     0x0  0x....'....        DC32  flag

\                               In section .text, align 4, keep-with-next
\             ??DataTable3_1:
\     0x0  0x....'....        DC32  rising

\                               In section .text, align 4, keep-with-next
\             ??DataTable3_2:
\     0x0  0x4000'0424            DC32  0x40000424

\                               In section .text, align 4, keep-with-next
\             ??DataTable3_3:
\     0x0  0x....'....        DC32  echo_dist

\                               In section .text, align 4, keep-with-next
\             ??DataTable3_4:
\     0x0  0x....'....        DC32  echo_time

\                               In section .text, align 4, keep-with-next
\             ??DataTable3_5:
\     0x0  0x....'....        DC32  htim3
```

```
\                               In section .text, align 4, keep-with-next
\              ??DataTable3_6:
\      0x0   0x....'....        DC32  htim4

\                               In section .text, align 4, keep-with-next
\              ??DataTable3_7:
\      0x0   0x....'....        DC32  htim5
       264
       265             /* USER CODE BEGIN 1 */
       266
       267             /* USER CODE END 1 */
```

Maximum stack usage in bytes:

```
.cstack Function
------- --------
       0   BusFault_Handler
       0   DebugMon_Handler
       0   HardFault_Handler
       0   MemManage_Handler
       0   NMI_Handler
       0   PendSV_Handler
       0   SVC_Handler
       0   SysTick_Handler
       0   -> HAL_IncTick
      16   TIM3_IRQHandler
       0   -> HAL_TIM_IRQHandler
       0   TIM4_IRQHandler
       0   -> HAL_TIM_IRQHandler
       0   TIM5_IRQHandler
       0   -> HAL_TIM_IRQHandler
       0   UsageFault_Handler
```

Section sizes:

```
Bytes  Function/Label
-----  --------------
       4  ??DataTable3
       4  ??DataTable3_1
```

```
   4  ??DataTable3_2
   4  ??DataTable3_3
   4  ??DataTable3_4
   4  ??DataTable3_5
   4  ??DataTable3_6
   4  ??DataTable3_7
   2  BusFault_Handler
   2  DebugMon_Handler
   2  HardFault_Handler
   2  MemManage_Handler
   2  NMI_Handler
   2  PendSV_Handler
   2  SVC_Handler
   4  SysTick_Handler
  58  TIM3_IRQHandler
   6  TIM4_IRQHandler
   6  TIM5_IRQHandler
   2  UsageFault_Handler
   4  rising


   4 bytes in section .bss
 122 bytes in section .text

 122 bytes of CODE memory
   4 bytes of DATA memory

Errors: none
Warnings: none
```