```c
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file           : main.c
  * @brief          : Main program body
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2024 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 "adc.h"
#include "dac.h"
#include "dma.h"
#include "tim.h"
#include "usart.h"
#include "gpio.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include "lcd.h"
#include "led.h"
#include "interrupt.h"
#include "string.h"
#include "stdio.h"
#include "i2c_hal.h"

/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */
extern struct keys key[4];
extern char rx_arry[50];
extern char rx_data;
extern char rx_pointer;
extern uint PA15_rise,PA15_fall;//TIM2
extern uint PA15_freq,PA15_duty;
extern uint PB4_rise,PB4_fall;//TIM3
extern uint PB4_freq,PB4_duty;
/* USER CODE END PTD */

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

/* USER CODE END PD */

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

/* USER CODE END PM */

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

/* USER CODE BEGIN PV */
__IO uint32_t key_uwTick;
__IO uint32_t adc_uwTick;
__IO uint32_t led_uwTick;
u16 adc1_arry[2];
u16 adc2_arry[1];
char lcd_arry[50];
char lcd_view;

```

```c
float R37_vol;
float distance;
uint direction;//0直行，1左，2右
__IO uint32_t dir_uwTick;
uint left_turn_flag;
uint right_turn_flag;
char led_num;



/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
void key_proc();
void rx_proc();
void adc_proc();
void lcd_proc();
void led_proc();


/* 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_DMA_Init();
  MX_TIM6_Init();
  MX_USART1_UART_Init();
  MX_ADC2_Init();
  MX_ADC1_Init();
  MX_DAC1_Init();
  MX_TIM2_Init();
  MX_TIM3_Init();
  MX_TIM8_Init();
  /* USER CODE BEGIN 2 */

  HAL_TIM_Base_Start_IT(&htim6);

  LCD_Init();
  LCD_Clear(Black);
  LCD_SetBackColor(Black);
  LCD_SetTextColor(White);

  HAL_UART_Receive_IT(&huart1, (uint8_t *)&rx_data, 1);
  HAL_ADC_Start_DMA(&hadc1,( uint32_t *)adc1_arry,2);
  HAL_ADC_Start_DMA(&hadc2,( uint32_t *)adc2_arry,1);
  HAL_DAC_Start(&hdac1,DAC_CHANNEL_1);

  HAL_TIM_IC_Start_IT(&htim8, TIM_CHANNEL_1);
  HAL_TIM_IC_Start_IT(&htim8, TIM_CHANNEL_2);
  HAL_TIM_IC_Start_IT(&htim3, TIM_CHANNEL_1);
  HAL_TIM_IC_Start_IT(&htim3, TIM_CHANNEL_2);

  HAL_TIM_PWM_Start(&htim2,TIM_CHANNEL_2);

  led_disp(0x00);

  /* USER CODE END 2 */

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

    /* USER CODE BEGIN 3 */
    key_proc();
    lcd_proc();
    rx_proc();
    adc_proc();
    led_proc();
  }
  /* 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_PWREx_ControlVoltageScaling(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 = RCC_PLLM_DIV3;
  RCC_OscInitStruct.PLL.PLLN = 20;
  RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
  RCC_OscInitStruct.PLL.PLLQ = RCC_PLLQ_DIV2;
  RCC_OscInitStruct.PLL.PLLR = RCC_PLLR_DIV2;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
```

```c
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_2) != HAL_OK)
  {
    Error_Handler();
  }
}

/* USER CODE BEGIN 4 */
void led_proc()
{
    if(uwTick - led_uwTick <100)
        return;
    led_uwTick = uwTick;

    if(lcd_view == 0)
    {
        if(direction == 1)
            led_num = led_num^0x01;
        else
            led_num = led_num&0xfe;
        if(direction == 2)
            led_num = led_num^0x02;
        else
            led_num = led_num&0xfd;
    }
    else
    {
        led_num = led_num&0xfc;
    }



    if(lcd_view == 1)
        led_num = led_num|0x80;
    else led_num = led_num&0x7f;

    led_disp(led_num);

}
void lcd_proc()
{
    if(lcd_view == 0)
    {
        sprintf(lcd_arry,"        DATA");
        LCD_DisplayStringLine(Line1, (u8 *)lcd_arry);

        if(direction == 0)
            sprintf(lcd_arry,"       N:S");
        else if(direction == 1)
            sprintf(lcd_arry,"       N:L");
        else if(direction == 2)
            sprintf(lcd_arry,"       N:R");
        LCD_DisplayStringLine(Line3, (u8 *)lcd_arry);

        sprintf(lcd_arry,"       D:%-6.1f",distance);
        LCD_DisplayStringLine(Line4, (u8 *)lcd_arry);
    }
    if(lcd_view == 1)
    {
        sprintf(lcd_arry,"        WARN");
        LCD_DisplayStringLine(Line4, (u8 *)lcd_arry);
    }
}

void adc_proc()
{
    R37_vol = adc2_arry[0]*3.3/4096;
    if(R37_vol>3)
```

```c
        distance = 300;
    else distance = 100*R37_vol;
}
void rx_proc()
{
    if(rx_pointer!=0)
    {
        int temp = rx_pointer;
        HAL_Delay(1);
        if(temp == rx_pointer)//接收完毕
        {
            if(lcd_view == 0)
            {
                if((strcmp(rx_arry,"L") == 0) && ((uwTick - dir_uwTick) > 5000 || dir_uwTick == 0))
                    //5s内不能再次改变方向
                {
                    direction = 1;//导航方向切换为左
                    left_turn_flag = 1;//
                    dir_uwTick = uwTick;//开始计时

                }
                else if((strcmp(rx_arry,"R") == 0) && ((uwTick - dir_uwTick) > 5000 || dir_uwTick == 0))
                {
                    direction = 2;//导航方向切换为右
                    right_turn_flag = 1;
                    dir_uwTick = uwTick;//开始计时
                }
                else printf("fei_fa_zi_fu:ERROR\r\n");
            }
            else printf("WAIT\r\n");//偏离导航界面下


            rx_pointer = 0;memset(rx_arry,0,50);

        }
    }
}
void key_proc()
{

    for(int i=0;i<4;i++)
        if(key[i].short_flag == 1)
            LCD_Clear(Black);

    if(key[0].short_flag == 1)
    {
        key[0].short_flag = 0;
        if(lcd_view == 1)//偏离导航界面下
        {
            lcd_view = 0;
            direction = 0;
            printf("diao_tou:Success\r\n");//车辆掉头成功
        }
    }

    if(key[2].short_flag == 1)
    {
        key[2].short_flag = 0;
        if(lcd_view == 0)
        {
            if(direction == 0)
            {
                lcd_view = 1;
                printf("no_rx_but_turn:Warn\r\n");
            }
            else if(direction == 1)
            {
                if(uwTick - dir_uwTick<5000)
                {
                    left_turn_flag = 0;//转弯标志结束
                    printf("rx_and_turn:Success\r\n");//车辆转向成功
                    direction = 0;//导航方向切换为直行
                }
            }
        }
    }

    if(key[3].short_flag == 1)
    {
        key[3].short_flag = 0;
        if(lcd_view == 0)
        {
            if(direction == 0)
            {
                lcd_view = 1;
                printf("no_rx_but_turn:Warn\r\n");
            }
            else if(direction == 2)
            {
                if(uwTick - dir_uwTick<5000)
                {
                    right_turn_flag = 0;//转弯标志结束
                    printf("rx_and_turn:Success\r\n");//车辆转向成功
                    direction = 0;//导航方向切换为直行
                }
            }
        }
    }

    if((uwTick - dir_uwTick>5000) && ((right_turn_flag == 1) || (left_turn_flag == 1)))
    {
        right_turn_flag = 0;
        left_turn_flag = 0;
        lcd_view = 1;
        printf("rx_but_not_turn:Warn\r\n");
        LCD_Clear(Black);
    }
}

int fputc(int ch, FILE *f)
{
  HAL_UART_Transmit(&huart1, (const uint8_t *)&ch, 1, 20);
  return ch;
}


/* 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
  */
```

```c
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 */

```