****

A级达标线下测试报告



**学院** **专业**

**行政班级**  ， ，

**学号**  ， ，

**姓名** ， ，

**邮箱**

**手机** **完成日期** 20YY-MM-DD

**成绩**

# 题目名称: xxxx 红色提示句子可以删除

一、题目要求

A级达标测试题目要求内容复制到这里，字体宋体，小四，1.5倍行距。

二、设计思路

叙述一下你如何来实现或者解决这问题，编程的算法，流程图等等

三、项目管理

1. 项目组成员表

2. WBS表

3. 项目进度计划表

4. 项目会议纪要

四、参考文献

撰写报告所参考的图书，论文，网页资料等等。

五、程序代码

注：本代码由STM32CubeMX辅助完成，仅展示改动部分（USER CODE）

1. main.c

/\* USER CODE BEGIN Header \*/

/\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @file : main.c

\* @brief : Main program body

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @attention

\*

\* <h2><center>&copy; Copyright (c) 2021 STMicroelectronics.

\* All rights reserved.</center></h2>

\*

\* This software component is licensed by ST under BSD 3-Clause license,

\* the "License"; You may not use this file except in compliance with the

\* License. You may obtain a copy of the License at:

\* opensource.org/licenses/BSD-3-Clause

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

/\* 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 \*/

/\* USER CODE END PD \*/

/\* Private macro -------------------------------------------------------------\*/

/\* USER CODE BEGIN PM \*/

/\* USER CODE END PM \*/

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

TIM\_HandleTypeDef htim2;

TIM\_HandleTypeDef htim3;

TIM\_HandleTypeDef htim4;

UART\_HandleTypeDef huart1;

/\* USER CODE BEGIN PV \*/

/\* USER CODE END PV \*/

/\* Private function prototypes -----------------------------------------------\*/

void SystemClock\_Config(void);

static void MX\_GPIO\_Init(void);

static void MX\_TIM3\_Init(void);

static void MX\_TIM4\_Init(void);

static void MX\_TIM2\_Init(void);

static void MX\_USART1\_UART\_Init(void);

/\* USER CODE BEGIN PFP \*/

/\* USER CODE END PFP \*/

/\* Private user code ---------------------------------------------------------\*/

/\* USER CODE BEGIN 0 \*/

short count; //用来判断是第几次

short RGB[3];//用来存放读取回来的数据

short R,G,B; //存放比例系数

char Test1[]="#5P825S300#4P1475S300#3P1600S300#2P2000S300#1P570S300#0P1000S300\r\n";

char Test2[]="#5P1520S300#4P1240S300#3P1300S300#2P1670S300#1P570S300#0P1000S300\r\n";

char Test3[]="#5P1520S300#4P1240S300#3P1060S300#2P1670S300#1P570S300#0P1000S300\r\n";

char Test4[]="#5P1520S300#4P1240S300#3P1060S300#2P1670S300#1P570S300#0P2200S300\r\n";

char Test5[]="#5P1520S300#4P1240S300#3P1200S300#2P1670S300#1P570S300#0P2200S300\r\n";

char Test6[]="#5P1120S300#4P1370S300#3P990S300#2P1670S300#1P570S300#0P2200S300\r\n";

char Test7[]="#5P1120S300#4P1370S300#3P990S300#2P1670S300#1P570S300#0P1000S300\r\n";

void Set\_Speed(int speed)

{

TIM\_OC\_InitTypeDef sConfigOC = {0};

sConfigOC.OCMode = TIM\_OCMODE\_PWM1;

sConfigOC.Pulse = speed;

sConfigOC.OCPolarity = TIM\_OCPOLARITY\_HIGH;

sConfigOC.OCFastMode = TIM\_OCFAST\_DISABLE;

if (HAL\_TIM\_PWM\_ConfigChannel(&htim2, &sConfigOC, TIM\_CHANNEL\_1) != HAL\_OK)

{

Error\_Handler();

}

}

void Turn\_Forward(void)

{

HAL\_GPIO\_WritePin(GPIOA,GPIO\_PIN\_4, GPIO\_PIN\_SET); //PA4=INB1 PA5=INA1

Set\_Speed(18);

HAL\_TIM\_PWM\_Start(&htim2,TIM\_CHANNEL\_1);

// HAL\_GPIO\_WritePin(GPIOA,GPIO\_PIN\_5, GPIO\_PIN\_SET);

}

void Turn\_Back(void)

{

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_4, GPIO\_PIN\_RESET);

Set\_Speed(30);

HAL\_TIM\_PWM\_Start(&htim2,TIM\_CHANNEL\_1);

}

void Turn\_Off(void)

{

HAL\_GPIO\_WritePin(GPIOA,GPIO\_PIN\_4, GPIO\_PIN\_RESET); //PA4=INB1 PA5=INA1

Set\_Speed(20);

HAL\_TIM\_PWM\_Start(&htim2,TIM\_CHANNEL\_1);

}

void Taking(void)

{

Turn\_Off();

HAL\_UART\_Transmit(&huart1,(uint8\_t \*)Test1,70,10);

HAL\_Delay(3000);

HAL\_UART\_Transmit(&huart1,(uint8\_t \*)Test2,70,10);

HAL\_Delay(3000);

HAL\_UART\_Transmit(&huart1,(uint8\_t \*)Test3,70,10);

HAL\_Delay(3000);

HAL\_UART\_Transmit(&huart1,(uint8\_t \*)Test4,70,10);

HAL\_Delay(3000);

HAL\_UART\_Transmit(&huart1,(uint8\_t \*)Test5,70,10);

HAL\_Delay(3000);

HAL\_UART\_Transmit(&huart1,(uint8\_t \*)Test6,70,10);

HAL\_Delay(3000);

HAL\_UART\_Transmit(&huart1,(uint8\_t \*)Test7,70,10);

HAL\_Delay(3000);

HAL\_UART\_Transmit(&huart1,(uint8\_t \*)Test1,70,10);

HAL\_Delay(3000);

}

/\* 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\_TIM3\_Init();

MX\_TIM4\_Init();

MX\_TIM2\_Init();

MX\_USART1\_UART\_Init();

/\* USER CODE BEGIN 2 \*/

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_4, GPIO\_PIN\_SET); //INA1

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_0, GPIO\_PIN\_RESET); //S2

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_1, GPIO\_PIN\_RESET); //S3

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_7, GPIO\_PIN\_SET); //S0

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_8, GPIO\_PIN\_SET); //S1

\_\_HAL\_TIM\_CLEAR\_IT(&htim4,TIM\_IT\_UPDATE);

HAL\_TIM\_Base\_Start\_IT(&htim4);

\_\_HAL\_TIM\_CLEAR\_IT(&htim3,TIM\_IT\_UPDATE);

HAL\_TIM\_Base\_Start(&htim3);

\_\_HAL\_TIM\_CLEAR\_IT(&htim2,TIM\_IT\_UPDATE);

// HAL\_TIM\_PWM\_Start(&htim2,TIM\_CHANNEL\_1);

count = 0;

RGB[0]=0;

RGB[1]=0;

RGB[2]=0;

R=3;

G=1;

B=2;

HAL\_UART\_Transmit(&huart1,(uint8\_t \*)Test1,70,10);

HAL\_Delay(3000);

/\* USER CODE END 2 \*/

/\* Infinite loop \*/

/\* USER CODE BEGIN WHILE \*/

while (1)

{

/\* USER CODE END WHILE \*/

Turn\_Forward();

if(RGB[0]>0x50&&RGB[2]<0x60)

{

Taking();

}

/\* 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};

RCC\_PeriphCLKInitTypeDef PeriphClkInit = {0};

/\*\* Initializes the CPU, AHB and APB busses clocks

\*/

RCC\_OscInitStruct.OscillatorType = RCC\_OSCILLATORTYPE\_MSI;

RCC\_OscInitStruct.MSIState = RCC\_MSI\_ON;

RCC\_OscInitStruct.MSICalibrationValue = 0;

RCC\_OscInitStruct.MSIClockRange = RCC\_MSIRANGE\_6;

RCC\_OscInitStruct.PLL.PLLState = RCC\_PLL\_NONE;

if (HAL\_RCC\_OscConfig(&RCC\_OscInitStruct) != HAL\_OK)

{

Error\_Handler();

}

/\*\* Initializes the CPU, AHB and APB busses clocks

\*/

RCC\_ClkInitStruct.ClockType = RCC\_CLOCKTYPE\_HCLK|RCC\_CLOCKTYPE\_SYSCLK

|RCC\_CLOCKTYPE\_PCLK1|RCC\_CLOCKTYPE\_PCLK2;

RCC\_ClkInitStruct.SYSCLKSource = RCC\_SYSCLKSOURCE\_MSI;

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\_0) != HAL\_OK)

{

Error\_Handler();

}

PeriphClkInit.PeriphClockSelection = RCC\_PERIPHCLK\_USART1;

PeriphClkInit.Usart1ClockSelection = RCC\_USART1CLKSOURCE\_PCLK2;

if (HAL\_RCCEx\_PeriphCLKConfig(&PeriphClkInit) != HAL\_OK)

{

Error\_Handler();

}

/\*\* Configure the main internal regulator output voltage

\*/

if (HAL\_PWREx\_ControlVoltageScaling(PWR\_REGULATOR\_VOLTAGE\_SCALE1) != HAL\_OK)

{

Error\_Handler();

}

}

/\*\*

\* @brief TIM2 Initialization Function

\* @param None

\* @retval None

\*/

static void MX\_TIM2\_Init(void)

{

/\* USER CODE BEGIN TIM2\_Init 0 \*/

/\* USER CODE END TIM2\_Init 0 \*/

TIM\_ClockConfigTypeDef sClockSourceConfig = {0};

TIM\_MasterConfigTypeDef sMasterConfig = {0};

TIM\_OC\_InitTypeDef sConfigOC = {0};

/\* USER CODE BEGIN TIM2\_Init 1 \*/

/\* USER CODE END TIM2\_Init 1 \*/

htim2.Instance = TIM2;

htim2.Init.Prescaler = 9;

htim2.Init.CounterMode = TIM\_COUNTERMODE\_UP;

htim2.Init.Period = 39;

htim2.Init.ClockDivision = TIM\_CLOCKDIVISION\_DIV1;

htim2.Init.AutoReloadPreload = TIM\_AUTORELOAD\_PRELOAD\_DISABLE;

if (HAL\_TIM\_Base\_Init(&htim2) != HAL\_OK)

{

Error\_Handler();

}

sClockSourceConfig.ClockSource = TIM\_CLOCKSOURCE\_INTERNAL;

if (HAL\_TIM\_ConfigClockSource(&htim2, &sClockSourceConfig) != HAL\_OK)

{

Error\_Handler();

}

if (HAL\_TIM\_PWM\_Init(&htim2) != HAL\_OK)

{

Error\_Handler();

}

sMasterConfig.MasterOutputTrigger = TIM\_TRGO\_RESET;

sMasterConfig.MasterSlaveMode = TIM\_MASTERSLAVEMODE\_DISABLE;

if (HAL\_TIMEx\_MasterConfigSynchronization(&htim2, &sMasterConfig) != HAL\_OK)

{

Error\_Handler();

}

sConfigOC.OCMode = TIM\_OCMODE\_PWM1;

sConfigOC.Pulse = 0;

sConfigOC.OCPolarity = TIM\_OCPOLARITY\_HIGH;

sConfigOC.OCFastMode = TIM\_OCFAST\_DISABLE;

if (HAL\_TIM\_PWM\_ConfigChannel(&htim2, &sConfigOC, TIM\_CHANNEL\_1) != HAL\_OK)

{

Error\_Handler();

}

/\* USER CODE BEGIN TIM2\_Init 2 \*/

/\* USER CODE END TIM2\_Init 2 \*/

HAL\_TIM\_MspPostInit(&htim2);

}

/\*\*

\* @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\_SlaveConfigTypeDef sSlaveConfig = {0};

TIM\_MasterConfigTypeDef sMasterConfig = {0};

/\* USER CODE BEGIN TIM3\_Init 1 \*/

/\* USER CODE END TIM3\_Init 1 \*/

htim3.Instance = TIM3;

htim3.Init.Prescaler = 0;

htim3.Init.CounterMode = TIM\_COUNTERMODE\_UP;

htim3.Init.Period = 0xffff;

htim3.Init.ClockDivision = TIM\_CLOCKDIVISION\_DIV1;

htim3.Init.AutoReloadPreload = TIM\_AUTORELOAD\_PRELOAD\_DISABLE;

if (HAL\_TIM\_Base\_Init(&htim3) != HAL\_OK)

{

Error\_Handler();

}

sSlaveConfig.SlaveMode = TIM\_SLAVEMODE\_EXTERNAL1;

sSlaveConfig.InputTrigger = TIM\_TS\_TI1FP1;

sSlaveConfig.TriggerPolarity = TIM\_TRIGGERPOLARITY\_FALLING;

sSlaveConfig.TriggerFilter = 0;

if (HAL\_TIM\_SlaveConfigSynchro(&htim3, &sSlaveConfig) != HAL\_OK)

{

Error\_Handler();

}

sMasterConfig.MasterOutputTrigger = TIM\_TRGO\_UPDATE;

sMasterConfig.MasterSlaveMode = TIM\_MASTERSLAVEMODE\_DISABLE;

if (HAL\_TIMEx\_MasterConfigSynchronization(&htim3, &sMasterConfig) != 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};

/\* USER CODE BEGIN TIM4\_Init 1 \*/

/\* USER CODE END TIM4\_Init 1 \*/

htim4.Instance = TIM4;

htim4.Init.Prescaler = 9;

htim4.Init.CounterMode = TIM\_COUNTERMODE\_UP;

htim4.Init.Period = 3999;

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();

}

sMasterConfig.MasterOutputTrigger = TIM\_TRGO\_RESET;

sMasterConfig.MasterSlaveMode = TIM\_MASTERSLAVEMODE\_DISABLE;

if (HAL\_TIMEx\_MasterConfigSynchronization(&htim4, &sMasterConfig) != HAL\_OK)

{

Error\_Handler();

}

/\* USER CODE BEGIN TIM4\_Init 2 \*/

/\* USER CODE END TIM4\_Init 2 \*/

}

/\*\*

\* @brief USART1 Initialization Function

\* @param None

\* @retval None

\*/

static void MX\_USART1\_UART\_Init(void)

{

/\* USER CODE BEGIN USART1\_Init 0 \*/

/\* USER CODE END USART1\_Init 0 \*/

/\* USER CODE BEGIN USART1\_Init 1 \*/

/\* USER CODE END USART1\_Init 1 \*/

huart1.Instance = USART1;

huart1.Init.BaudRate = 115200;

huart1.Init.WordLength = UART\_WORDLENGTH\_8B;

huart1.Init.StopBits = UART\_STOPBITS\_1;

huart1.Init.Parity = UART\_PARITY\_NONE;

huart1.Init.Mode = UART\_MODE\_TX\_RX;

huart1.Init.HwFlowCtl = UART\_HWCONTROL\_NONE;

huart1.Init.OverSampling = UART\_OVERSAMPLING\_16;

huart1.Init.OneBitSampling = UART\_ONE\_BIT\_SAMPLE\_DISABLE;

huart1.AdvancedInit.AdvFeatureInit = UART\_ADVFEATURE\_NO\_INIT;

if (HAL\_UART\_Init(&huart1) != HAL\_OK)

{

Error\_Handler();

}

/\* USER CODE BEGIN USART1\_Init 2 \*/

/\* USER CODE END USART1\_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\_GPIOA\_CLK\_ENABLE();

/\*Configure GPIO pin Output Level \*/

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_0|GPIO\_PIN\_1|GPIO\_PIN\_4|GPIO\_PIN\_7

|GPIO\_PIN\_8, GPIO\_PIN\_RESET);

/\*Configure GPIO pins : PA0 PA1 PA4 PA7

PA8 \*/

GPIO\_InitStruct.Pin = GPIO\_PIN\_0|GPIO\_PIN\_1|GPIO\_PIN\_4|GPIO\_PIN\_7

|GPIO\_PIN\_8;

GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Pull = GPIO\_NOPULL;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;

HAL\_GPIO\_Init(GPIOA, &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 \*/

/\* 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,

tex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) \*/

/\* USER CODE END 6 \*/

}

#endif /\* USE\_FULL\_ASSERT \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* (C) COPYRIGHT STMicroelectronics \*\*\*\*\*END OF FILE\*\*\*\*/

2. stm32l4xx\_it.c

/\* USER CODE BEGIN Header \*/

/\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @file stm32l4xx\_it.c

\* @brief Interrupt Service Routines.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* @attention

\*

\* <h2><center>&copy; Copyright (c) 2021 STMicroelectronics.

\* All rights reserved.</center></h2>

\*

\* This software component is licensed by ST under BSD 3-Clause license,

\* the "License"; You may not use this file except in compliance with the

\* License. You may obtain a copy of the License at:

\* opensource.org/licenses/BSD-3-Clause

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*/

/\* USER CODE END Header \*/

/\* Includes ------------------------------------------------------------------\*/

#include "main.h"

#include "stm32l4xx\_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 \*/

/\* 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 htim4;

/\* USER CODE BEGIN EV \*/

extern TIM\_HandleTypeDef htim3;

extern short count;//用来判断是第几次

extern short RGB[3];//用来存放读取回来的数据

extern short R,G,B;//存放比例系数

/\* 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 \*/

/\* 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 Prefetch 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 \*/

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* STM32L4xx 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\_stm32l4xx.s). \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*

\* @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 \*/

}

/\* USER CODE BEGIN 1 \*/

int test=0;

void HAL\_TIM\_PeriodElapsedCallback(TIM\_HandleTypeDef \*htim)

{

if (htim == (&htim4))

{

// HAL\_GPIO\_TogglePin(GPIOA, GPIO\_PIN\_5); //引脚极性反转 ?

HAL\_TIM\_Base\_Stop\_IT(&htim4);

HAL\_TIM\_Base\_Stop(&htim3);

switch(count)

{

case 0:

{ //读取红色数据，设定为绿色过滤

test = \_\_HAL\_TIM\_GetCounter(&htim3);

test /= R;

RGB[count] = test;

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_0, GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_1, GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_7, GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_8, GPIO\_PIN\_SET);

count = 1;

break;

}

case 1:

{

//读取绿色数据，设定为蓝色过滤

test = \_\_HAL\_TIM\_GetCounter(&htim3); //数据

test /= G;

RGB[count] = test;

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_0, GPIO\_PIN\_RESET); //S2

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_1, GPIO\_PIN\_SET); //S3

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_7, GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_8, GPIO\_PIN\_SET);

count = 2;

break;

}

case 2:

{ //读取蓝色数据，设定为红色过滤

test = \_\_HAL\_TIM\_GetCounter(&htim3);

test /= B;

RGB[count] = test;

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_0, GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_1, GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_7, GPIO\_PIN\_SET); //S0 or S1

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_8, GPIO\_PIN\_SET); //S0 or S1

count = 0;

break;

}

}

\_\_HAL\_TIM\_SetCounter(&htim3,0);

HAL\_TIM\_Base\_Start\_IT(&htim4);

HAL\_TIM\_Base\_Start(&htim3);

}

}

/\* USER CODE END 1 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* (C) COPYRIGHT STMicroelectronics \*\*\*\*\*END OF FILE\*\*\*\*/

文件命名 组号\_学院\_线下A测报告.pdf

每一组交一份报告 名字都写在封面上就行 电子版

测试报告上传到指定网站。