蓝桥杯（电子）嵌入式

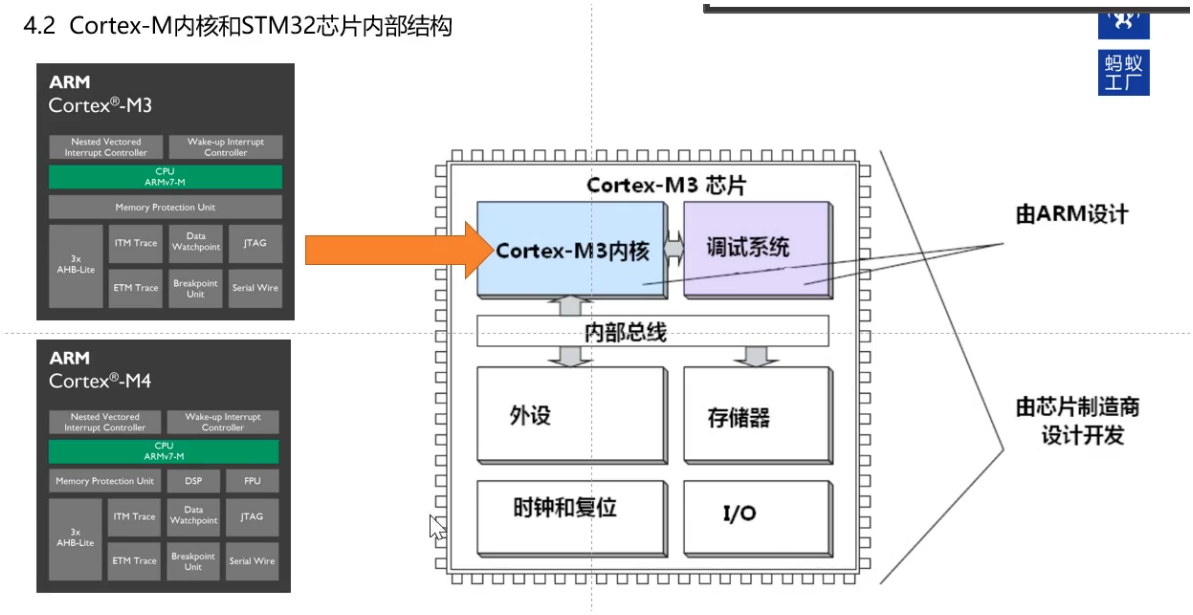
> 比赛介绍
>
> 
>

STM32基础知识

芯片型号==STM32G431RBT6==
STM 家族
F 产品类别（基础型）
051 特定功能（入门级）
R 引脚数（64&66）
B 闪存容量（64kbytes）
T 封装

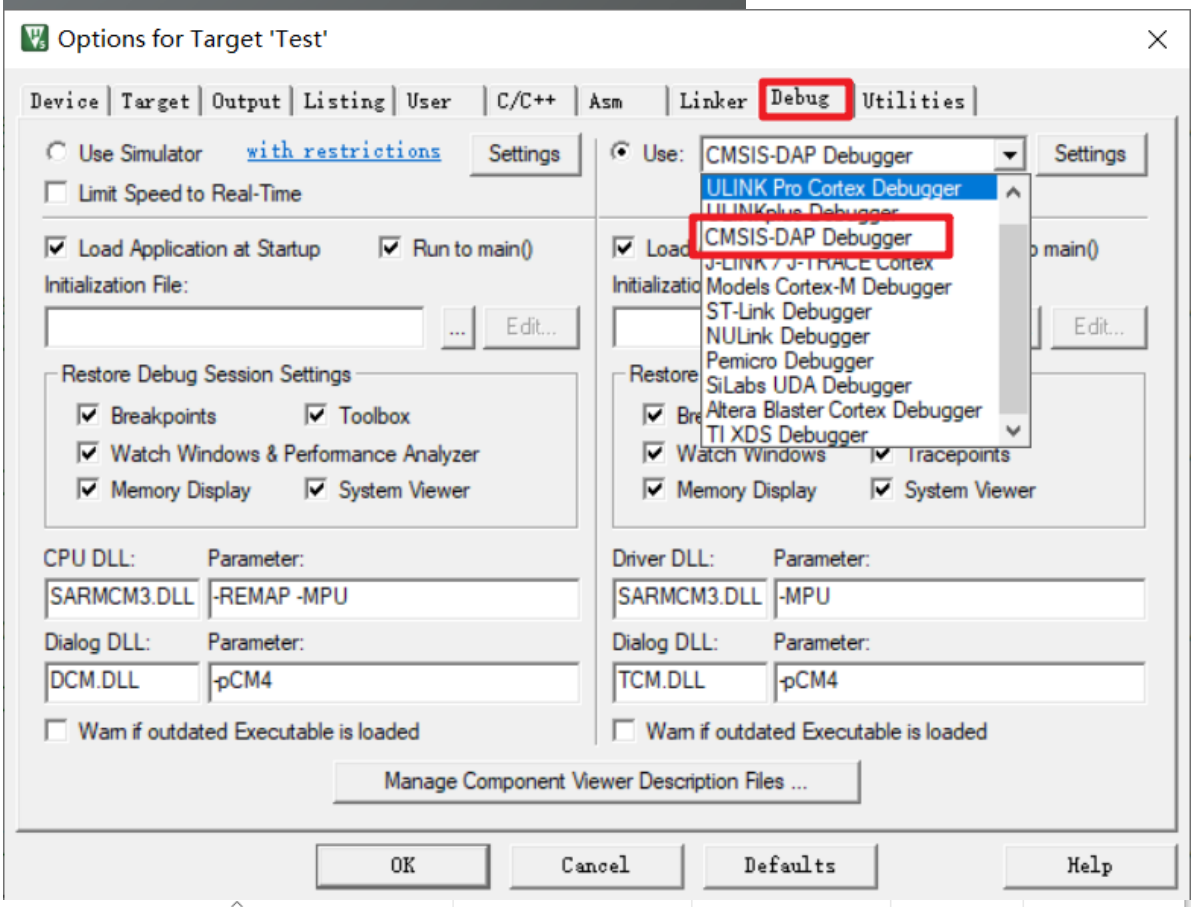
6 温度范围

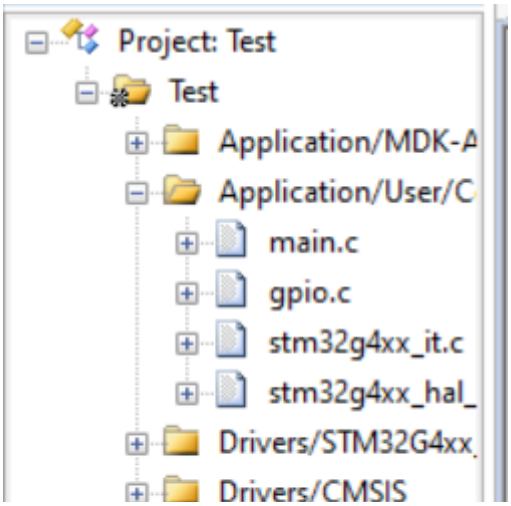
4.2 Cortex-M内核和STM32芯片内部结构

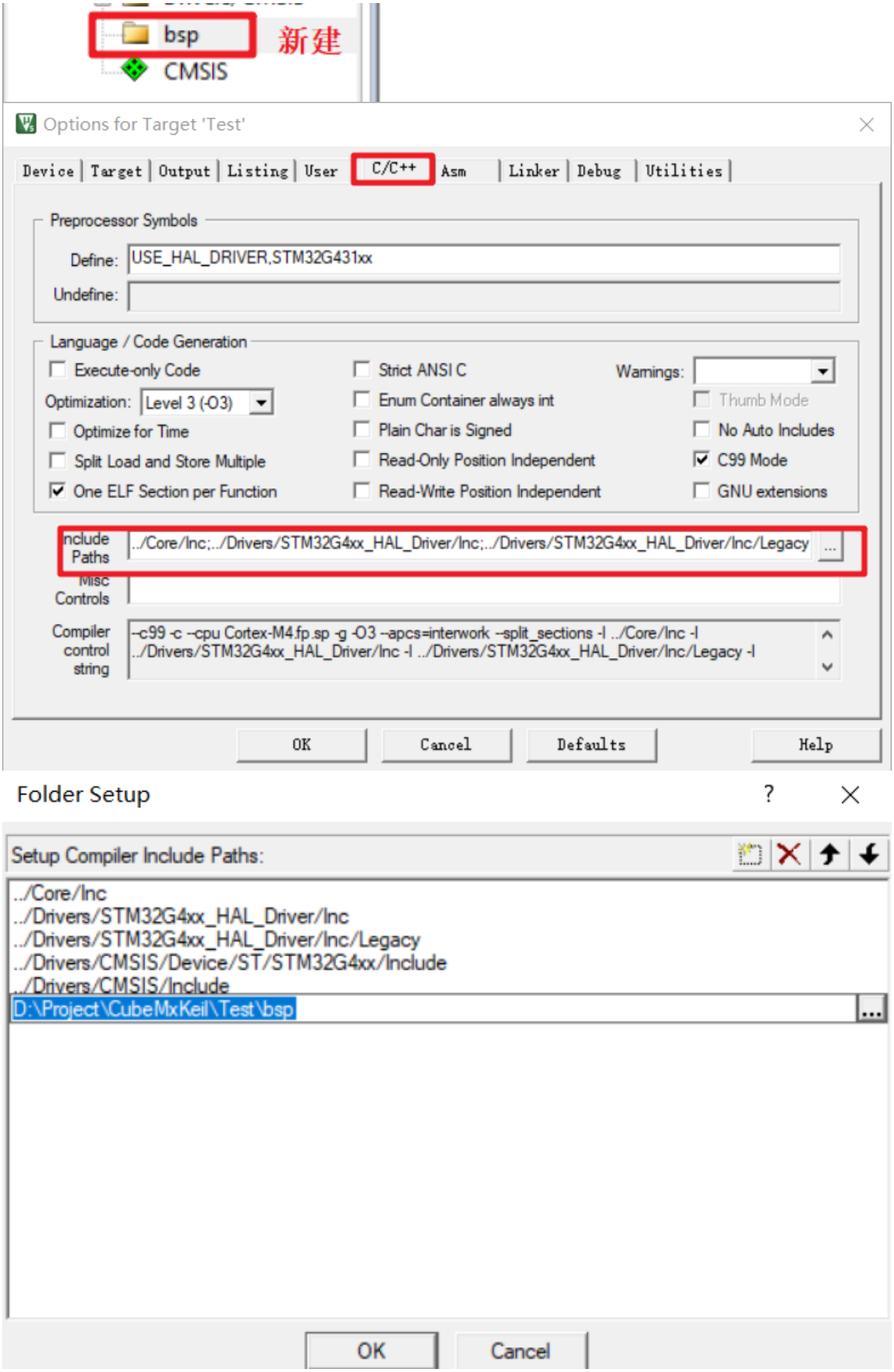


一、环境配置及新建工程

STM32G431RBTx
LQFP64

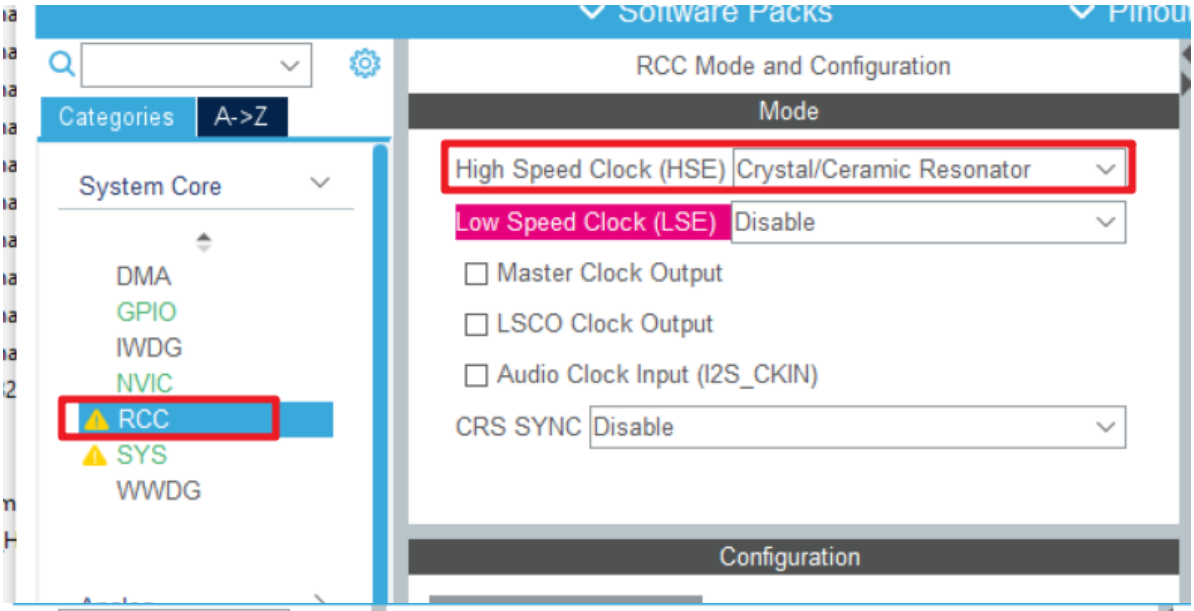


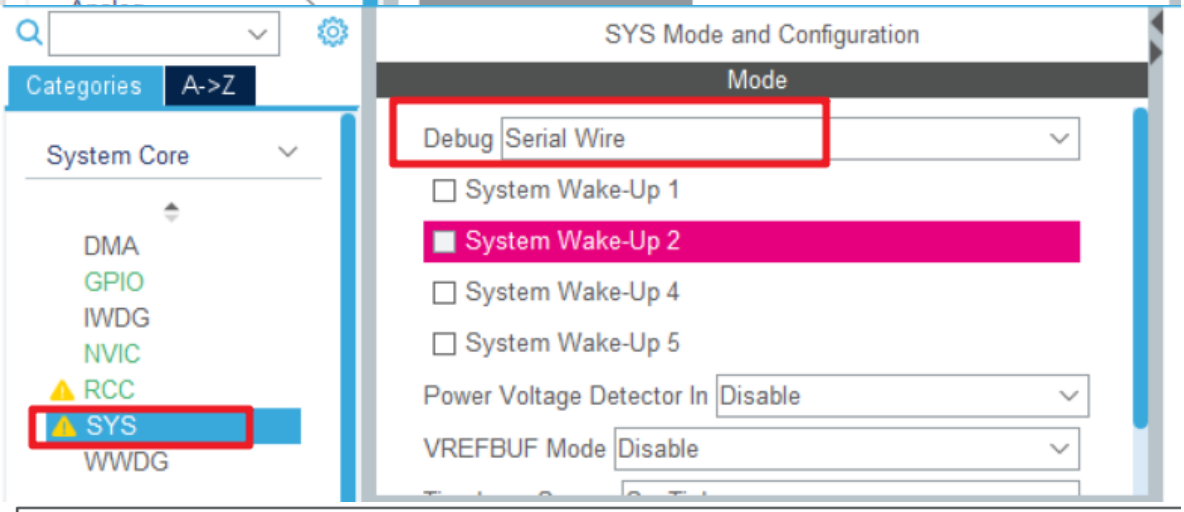


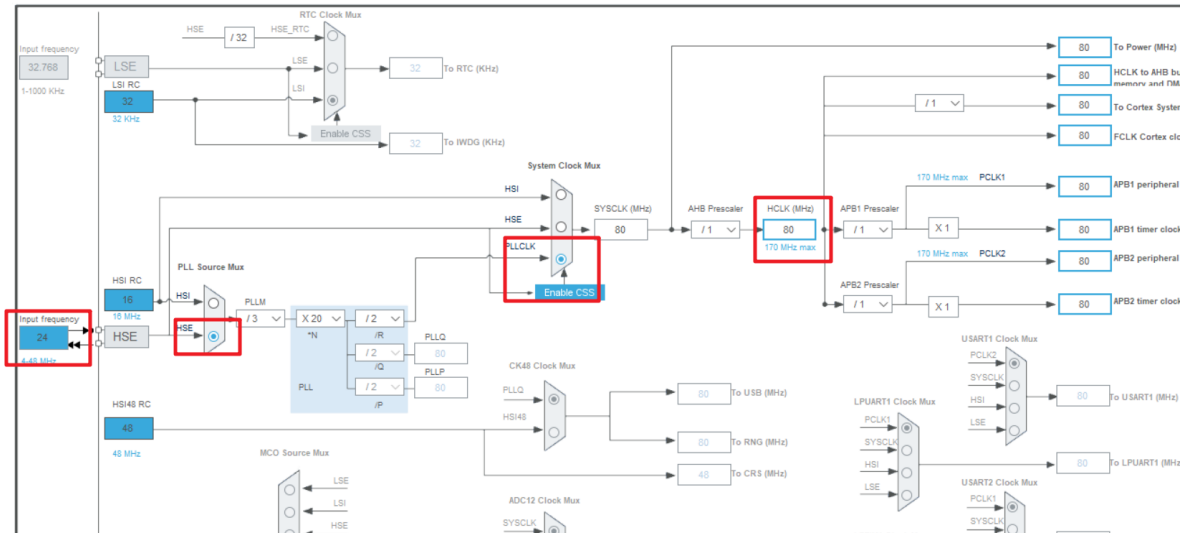


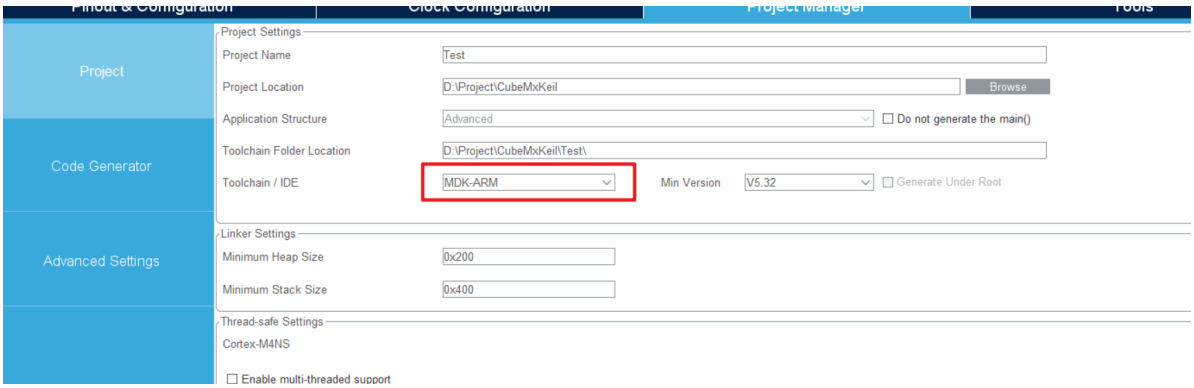
！注意CN2才是烧录口，别插到CN5上去了

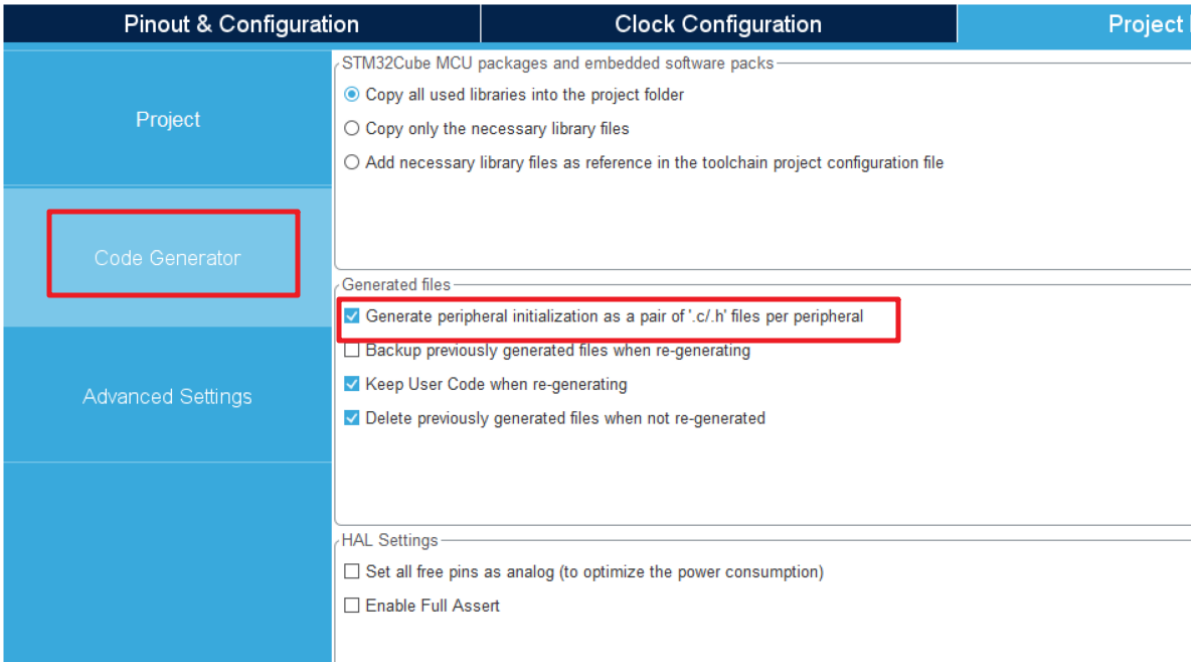
CubeMx的配置


Software Packs
Pinou
Categories
A->Z
System Core
DMA
GPIO
IWDG
NVIC
RCC
SYS
WWDG
RCC Mode and Configuration
Mode
High Speed Clock (HSE)
Crystal/Ceramic Resonator
Low Speed Clock (LSE)
Disable
Master Clock Output
LSCO Clock Output
Audio Clock Input (I2S_CKIN)
CRS SYNC
Disable
Configuration


Categories
A->Z
System Core
DMA
GPIO
IWDG
NVIC
RCC
SYS
WWDG
SYS Mode and Configuration
Mode
Debug
Serial Wire
System Wake-Up 1
System Wake-Up 2
System Wake-Up 4
System Wake-Up 5
Power Voltage Detector In
Disable
VREFBUF Mode
Disable


RTC Clock Mux
System Clock Mux
PLL Source Mux
MCO Source Mux
CK48 Clock Mux
ADC12 Clock Mux
USART1 Clock Mux
LPUART1 Clock Mux
USART2 Clock Mux
SYSCLK (MHz)
AHB Prescaler
HCLK (MHz)
APB1 Prescaler
APB2 Prescaler
PLLCLK
Enable CSS
To Power (MHz)
To RTC (KHz)
To IWDG (KHz)
To USB (MHz)
To RNG (MHz)
To CRS (MHz)
To USART1 (MHz)
To LPUART1 (MHz)
APB1 peripheral
APB1 timer clock
APB2 peripheral
APB2 timer clock



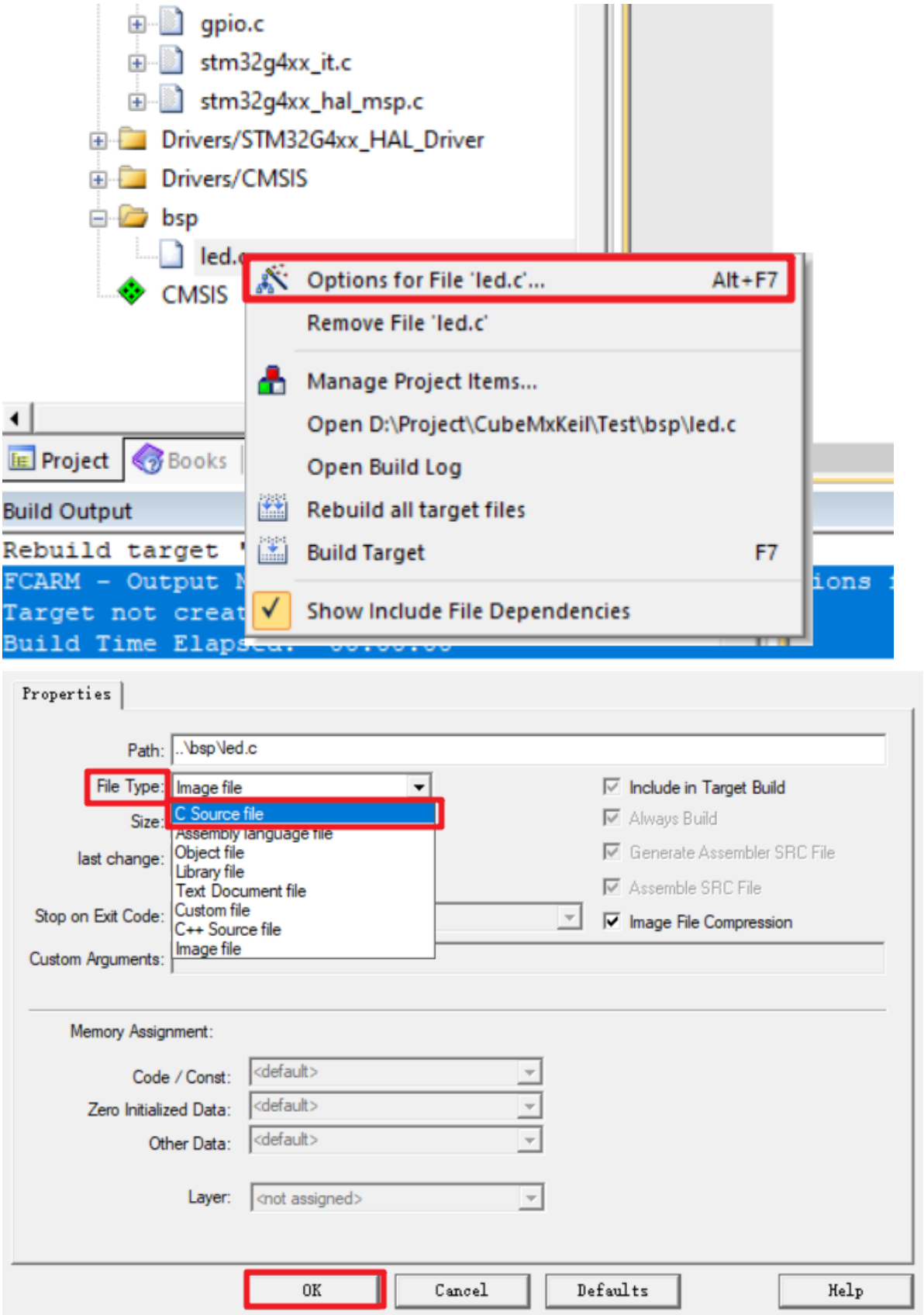


二、点亮一个LED

1. 问题解决

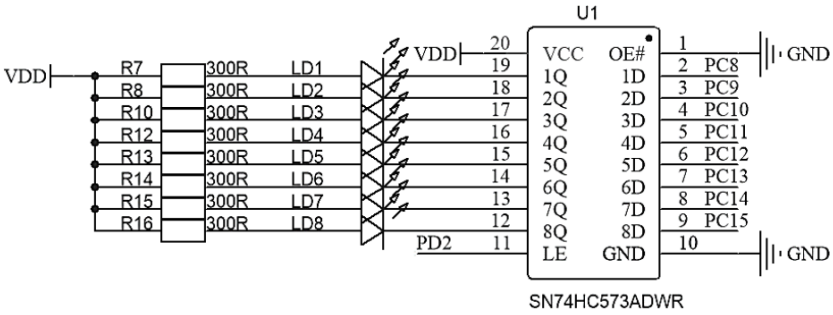
> FCARM - Output Name not specified, please check 'Options for Target - Utilities'
> Target not created.
> Build Time Elapsed: 00:00:00

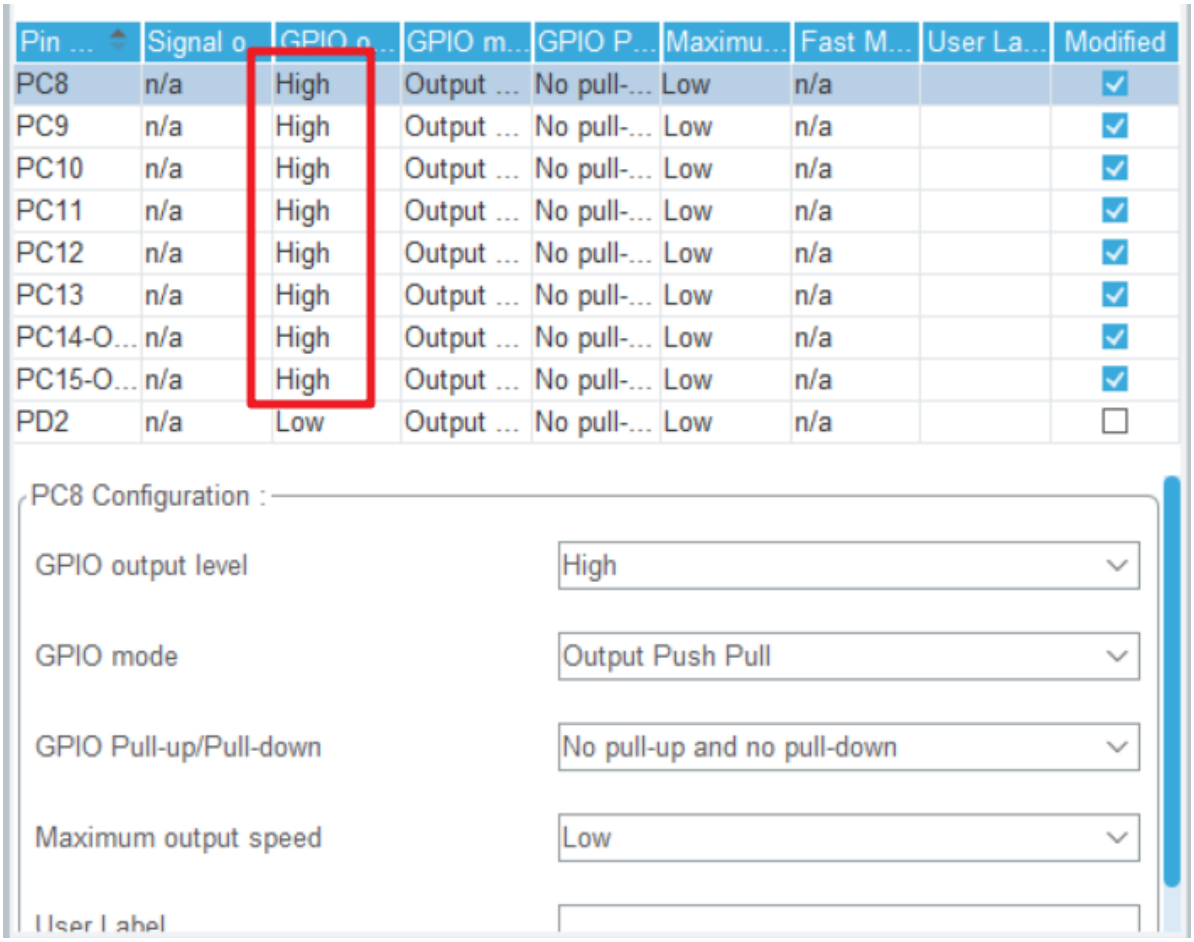
解决方法如下图所示：



2. 原理介绍

7.4 LED 指示灯





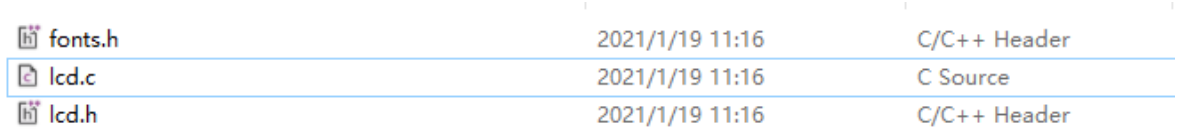
> 低电平点亮，高电平熄灭，这里默认高电平，防止上电点亮

3. 代码

```
#include "led.h"

void LED_Disp(uchar dsLED)
{
    HAL_GPIO_WritePin(GPIOC,GPIO_PIN_All,GPIO_PIN_SET);
    HAL_GPIO_WritePin(GPIOC,dsLED<<8,GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOD,GPIO_PIN_2,GPIO_PIN_SET);//打开锁存器
    HAL_GPIO_WritePin(GPIOD,GPIO_PIN_2,GPIO_PIN_RESET);
}
```

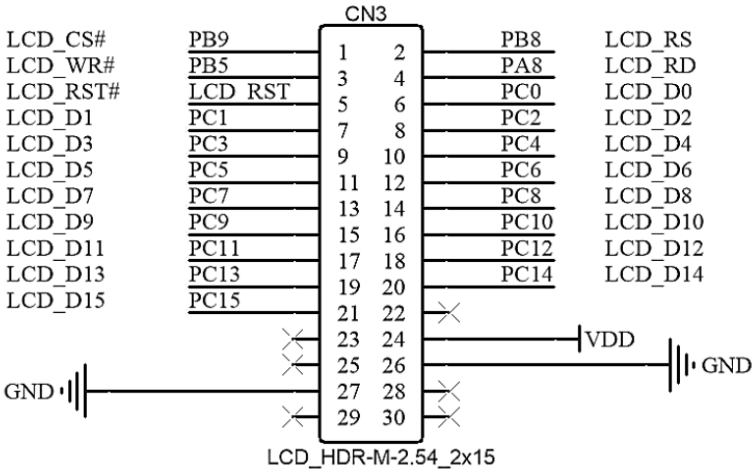
三、点亮一个LCD

> 路径：资源数据包_嵌入式（G431）\ 5-液晶驱动参考例程 \ HAL_06_LCD



2. 原理介绍

7.6 LCD

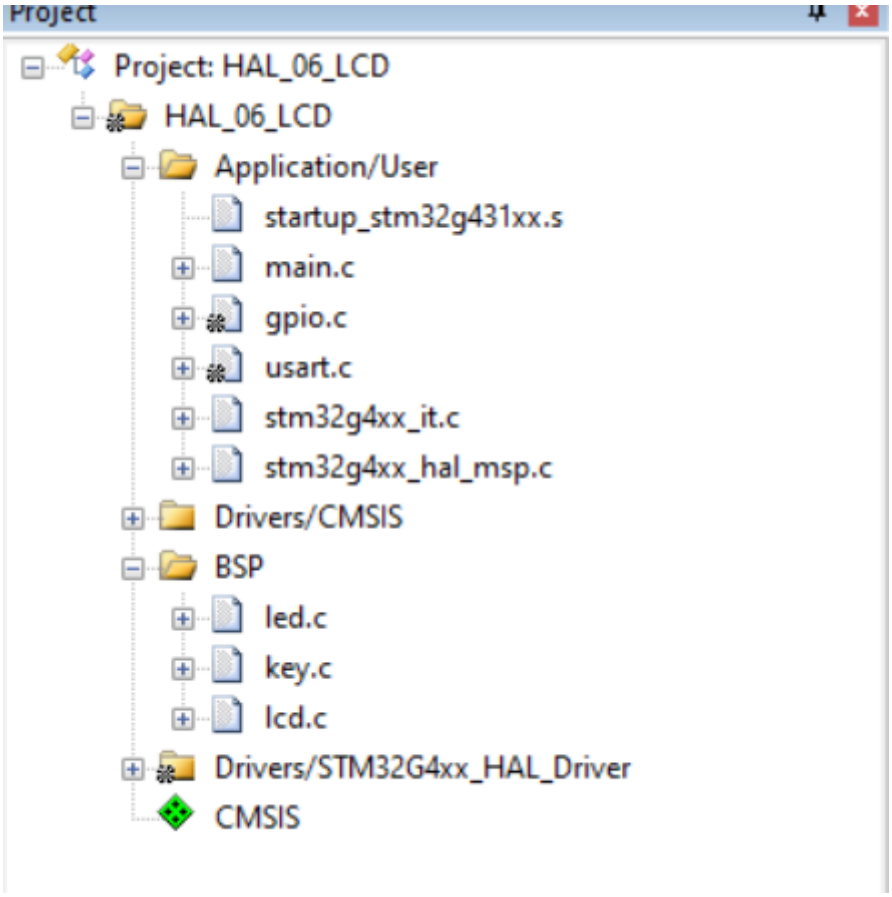


> 发现LCD屏持续白屏，尚未解决，应该是硬件损坏

- 可编程电阻是MCP4017 100K

四、以提供的LCD模板进行增添修改

程序结构如下所示



1将LCD部分放入到LCD_Proc(void) 中

```
//lcd变量
__IO uint32_t uwtick_lcd_set_point = 0;
unsigned char lcd_disp_string[22];
unsigned char i=0;

void LCD_Proc(void)
{
    if((uwTick - uwtick_lcd_set_point)<100)//减速函数
        return;
    uwtick_lcd_set_point = uwTick;
    i++;

    sprintf((char*)lcd_disp_string,"   i num: %03d      ",(unsigned int)i);
    LCD_DisplayStringLine(Line2,lcd_disp_string);

//
//  LCD_Clear(Blue);
//  LCD_SetBackColor(Blue);
//  LCD_SetTextColor(White);
//
//  LCD_DisplayStringLine(Line0, (uint8_t *)"                    ");
//  LCD_DisplayStringLine(Line1, (uint8_t *)"                    ");
//  LCD_DisplayStringLine(Line2, (uint8_t *)"      LCD Test      ");
//  LCD_DisplayStringLine(Line3, (uint8_t *)"                    ");
//  LCD_DisplayStringLine(Line4, (uint8_t *)"                    ");
//
//  LCD_SetBackColor(White);
//  LCD_SetTextColor(Blue);

//  LCD_DisplayStringLine(Line5, (uint8_t *)"                    ");
//  LCD_DisplayStringLine(Line6, (uint8_t *)"       HAL LSB      ");
//  LCD_DisplayStringLine(Line7, (uint8_t *)"                    ");
//  LCD_DisplayStringLine(Line8, (uint8_t *)"         @80        ");
//  LCD_DisplayStringLine(Line9, (uint8_t *)"                    ");
//
}
```

2.添加LED部分

设置PD2 GPIO_Output

```
void LED_Disp(unsigned char ucled)
{
    //将所有灯熄灭
    HAL_GPIO_WritePin(GPIOC,GPIO_PIN_All,GPIO_PIN_SET);
    HAL_GPIO_WritePin(GPIOD,GPIO_PIN_2,GPIO_PIN_SET);
    HAL_GPIO_WritePin(GPIOD,GPIO_PIN_2,GPIO_PIN_RESET);

    HAL_GPIO_WritePin(GPIOC,ucled<<8,GPIO_PIN_RESET);
    HAL_GPIO_WritePin(GPIOD,GPIO_PIN_2,GPIO_PIN_SET);
    HAL_GPIO_WritePin(GPIOD,GPIO_PIN_2,GPIO_PIN_RESET);
}
```

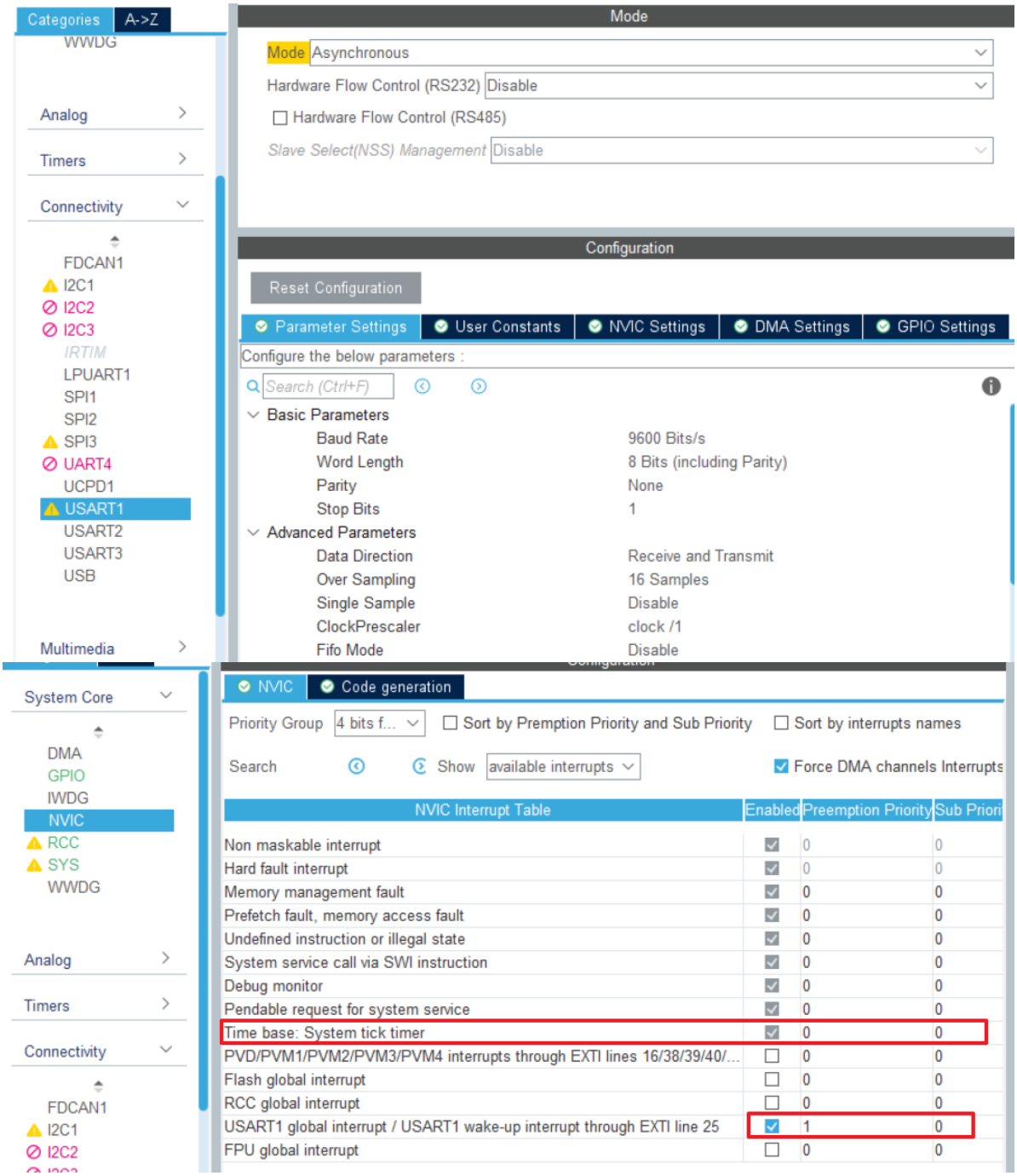
3.添加KEY部分

设置按键GPIO_Intput

```
unsigned char Key_scan(void)
{
    unsigned char unKey_Val=0;

    if(HAL_GPIO_ReadPin(GPIOA,GPIO_PIN_0)==GPIO_PIN_RESET)
        unKey_Val=4;
    if(HAL_GPIO_ReadPin(GPIOB,GPIO_PIN_0)==GPIO_PIN_RESET)
        unKey_Val=1;
    if(HAL_GPIO_ReadPin(GPIOB,GPIO_PIN_1)==GPIO_PIN_RESET)
        unKey_Val=2;
    if(HAL_GPIO_ReadPin(GPIOB,GPIO_PIN_2)==GPIO_PIN_RESET)
        unKey_Val=3;
    return unKey_Val;
}

//按键扫描变量
__IO uint32_t uwtick_key_set_point = 0;//控制Key_Proc的执行速度
unsigned char key_val,key_down,key_up,key_old;

void Key_Proc(void)
{

    if((uwTick - uwtick_key_set_point)<300)//减速函数
        return;
    uwtick_key_set_point = uwTick;
    key_val = Key_scan();
    key_down = key_val & (key_old^key_val);
    key_up = ~key_val & (key_old^key_val);
    key_old = key_val;
    if(key_down == 4)
    {
        LED_Disp(0x01);
    }
    if(key_down == 3)
    {
        LED_Disp(0x00);
    }
}
```

4.添加USART部分

> 注意滴答计时器的优先级为00，UART1为10




```c
//发送数据
__IO uint32_t uwtick_usart_set_point = 0;
int counter = 0;
char str[40];

void Usart_Proc(void)
{
    if((uwTick - uwtick_usart_set_point)<500)//减速函数
        return;
    uwtick_usart_set_point = uwTick;
    sprintf(str,"%04d:Hello,world.\r\n",counter);
    HAL_UART_Transmit(&huart1,(unsigned char*)str,strlen(str),50);
    if(++counter == 10000)
        counter = 0;
}

//接收数据
unsigned char rx_buffer;
```

```
HAL_UART_Receive_IT(&huart1,&rx_buffer,1);//加到while之前

void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{
    LED_Disp(0x88);
    HAL_Delay(300);
    LED_Disp(0x00);
    HAL_UART_Receive_IT(&huart1,&rx_buffer,1);

}
```

5.添加I2C部分

注意修改I2CWaitAck的顺序

```
SCL_Output(0);
delay1(DELAY_TIME);
SDA_Output_Mode();
```

6.添加EEPROM部分

```
//24C02的相关代码
void iic_24c02_write(unsigned char* pucBuf,unsigned char ucAddr,unsigned char
ucNum)
{
    I2CStart();
    I2CSendByte(0xa0);
    I2CWaitAck();

    I2CSendByte(ucAddr);
    I2CWaitAck();

    while(ucNum--)
    {
        I2CSendByte(*pucBuf++);
        I2CWaitAck();
    }
    I2CStop();
    delay1(500);
}

void iic_24c02_read(unsigned char* pucBuf,unsigned char ucAddr,unsigned char
ucNum)
{
    I2CStart();
    I2CSendByte(0xa0);
    I2CWaitAck();

    I2CSendByte(ucAddr);
    I2CWaitAck();

    I2CStart();
    I2CSendByte(0xa1);
    I2CWaitAck();
```

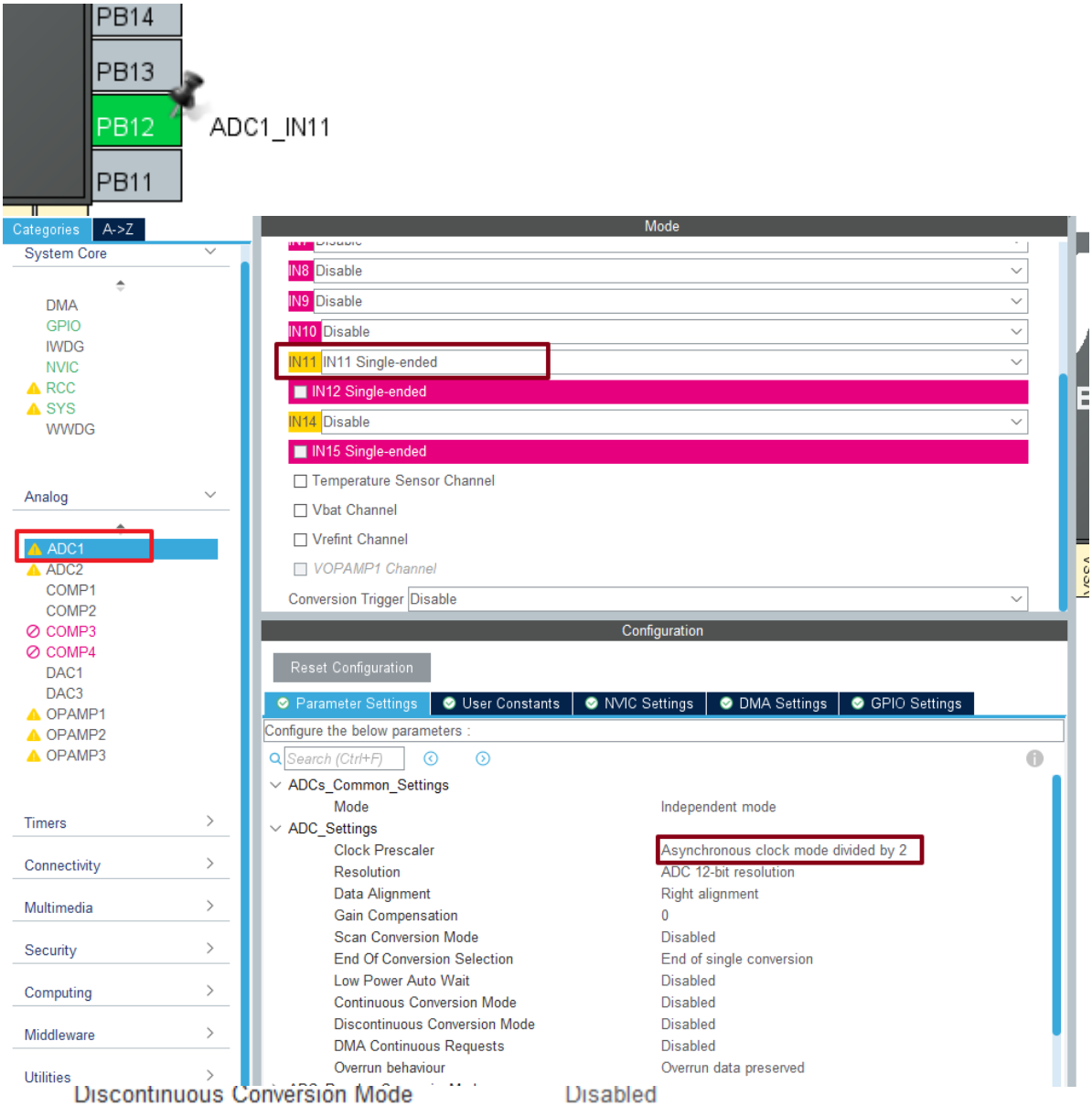
```
    while(ucNum--)
    {
        *pucBuf++=I2CReceiveByte();
        if(ucNum)
            I2CSendAck();
        else
            I2CSendNotAck();
    }
    I2CStop();
}

//*EEPROM的相关变量
unsigned char EEPROM_String_1[5] = {0x11,0x22,0x33,0x44,0x55};
unsigned char EEPROM_String_2[5] = {0};

I2CInit();
iic_24c02_write(EEPROM_String_1,0,5);
HAL_Delay(1);//注意延时1ms
iic_24c02_read(EEPROM_String_2,0,5);

void LCD_Proc(void)
{
    if((uwTick - uwtick_lcd_set_point)<100)//减速函数
        return;
    uwtick_lcd_set_point = uwTick;
    i++;
    LCD_Clear(White);

sprintf((char*)lcd_disp_string,"E:%x,%x,%x,%x,%x",EEPROM_String_2[0],EEPROM_Stri
ng_2[1],EEPROM_String_2[2],EEPROM_String_2[3],EEPROM_String_2[4]);
    LCD_DisplayStringLine(Line2,lcd_disp_string);
}
```

6.添加MCP4017可编程电阻部分

```
//MCP4017相关代码
void write_resistor(unsigned char vlaue)
{
    I2CStart();
    I2CSendByte(0x5E);
    I2CWaitAck();

    I2CSendByte(vlaue);
    I2CWaitAck();
    I2CStop();
}

unsigned char read_resistor(void)
{
    unsigned char value;
    I2CStart();
    I2CSendByte(0x5F);
    I2CWaitAck();
```

```
    value=I2CReceiveByte();
    I2CWaitAck();
    I2CStop();
    return value;
}

//4017的相关变量
unsigned char RES_4017;

//MCP4017测试
write_resistor(0x0D);
RES_4017=read_resistor();

void LCD_Proc(void)
{
    if((uwTick - uwtick_lcd_set_point)<1000)//减速函数
        return;
    uwtick_lcd_set_point = uwTick;
    i++;
    LCD_Clear(White);
    sprintf((char*)lcd_disp_string,"RES:%5.2fK",0.7874*RES_4017);
    LCD_DisplayStringLine(Line2,lcd_disp_string);
    sprintf((char*)lcd_disp_string,"VOL:%6.3fV",3.3*
(0.7874*RES_4017/(0.7874*RES_4017+10)));
    LCD_DisplayStringLine(Line3,lcd_disp_string);
}
```

7.添加ADC部分



Discontinuous Conversion Mode	Disabled
DMA Continuous Requests	Disabled
Overrun behaviour	Overrun data preserved
ADC_Regular_ConversionMode	
Enable Regular Conversions	Enable
Enable Regular Oversampling	Disable
Number Of Conversion	1
External Trigger Conversion Source	Regular Conversion launched by software
External Trigger Conversion Edge	None
Rank	1
Channel	Channel 11
Sampling Time	640.5 Cycles
Offset Number	No offset
ADC_Injected_ConversionMode	

> RANK: Samping Time (640.5Cycles)

```
uint16_t getADC(void)
{
    uint16_t adc = 0;
    HAL_ADC_Start(&hadc1);
    adc = HAL_ADC_GetValue(&hadc1);
    return adc;
}

sprintf((char*)lcd_disp_string,"R38_Vol:%6.3fV",3.3*getADC()/4096);//注意先*3.3的到浮点数
LCD_DisplayStringLine(Line4,lcd_disp_string);
```

> 第二路ADC PB15 R37 IN15 同理

8.添加TIM部分

基本定时器——定时
TIM6
Activated
7999
9999

```
//定时器6测试
HAL_TIM_Base_Start_IT(&htim6);

void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
    if(htim->Instance==TIM6)
    {
        i++;
        HAL_TIM_Base_Start_IT(&htim6);
    }
}
```

通用定时器——测量PWM频率

> TIM2_CH1(PB4),TIM3_CH1(PA15)
> Slave Mode: Reset Mode
> Trigger Source: TI1FP1
> Channel 1: Input Capture direct mode
> PSC:79,ARR:65535

```
//定时器3
HAL_TIM_Base_Start_IT(&htim3);
HAL_TIM_IC_Start_IT(&htim3,TIM_CHANNEL_1);//启动定时器通道输入捕获并开启中断
//定时器2
HAL_TIM_Base_Start_IT(&htim2);
HAL_TIM_IC_Start_IT(&htim2,TIM_CHANNEL_1);//启动定时器通道输入捕获并开启中断

//pwm的相关变量
uint16_t pwm1_T_count;
uint16_t pwm2_T_count;
```

```
void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim)
{
    if(htim->Instance==TIM3)
    {

        if(htim->Channel==HAL_TIM_ACTIVE_CHANNEL_1)
        {
            pwm1_T_count=HAL_TIM_ReadCapturedValue(htim,TIM_CHANNEL_1)+1;
        }
    }
    if(htim->Instance==TIM2)
    {

        if(htim->Channel==HAL_TIM_ACTIVE_CHANNEL_1)
        {
            pwm2_T_count=HAL_TIM_ReadCapturedValue(htim,TIM_CHANNEL_1)+1;
        }
    }
}
```

> 注意点：if(htim->Instance==TIM?) 多个定时器时一定要进行判断

占空比

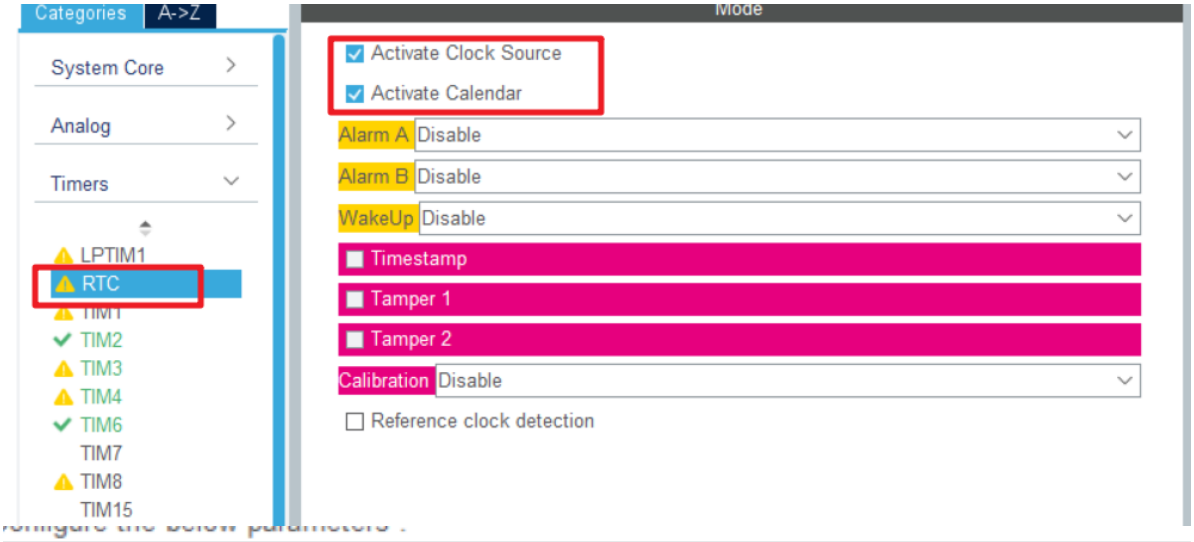
> TIM2_CH2 Falling Edge（可以不改）

```
void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim)
{
    if(htim->Instance==TIM3)
    {

        if(htim->Channel==HAL_TIM_ACTIVE_CHANNEL_1)
        {
            pwm1_T_count=HAL_TIM_ReadCapturedValue(htim,TIM_CHANNEL_1)+1;
            PWM1_Duty=(float)pwm1_D_count/pwm1_T_count;
        }

        else if(htim->Channel==HAL_TIM_ACTIVE_CHANNEL_2)
        {
            pwm1_D_count=HAL_TIM_ReadCapturedValue(htim,TIM_CHANNEL_2)+1;
        }
    }
    if(htim->Instance==TIM2)
    {

        if(htim->Channel==HAL_TIM_ACTIVE_CHANNEL_1)
        {
            pwm2_T_count=HAL_TIM_ReadCapturedValue(htim,TIM_CHANNEL_1)+1;
            PWM2_Duty=(float)pwm2_D_count/pwm2_T_count;
        }
        else if(htim->Channel==HAL_TIM_ACTIVE_CHANNEL_2)
        {
            pwm2_D_count=HAL_TIM_ReadCapturedValue(htim,TIM_CHANNEL_2)+1;
        }
    }
```

```
}

sprintf((char*)lcd_disp_string,"PWM1:%05d FD:%4.2f%%",(unsigned int)
(1000000/pwm1_T_count),PWM1_Duty*100);
LCD_DisplayStringLine(Line6,lcd_disp_string);

sprintf((char*)lcd_disp_string,"PWM2:%05d FD:%4.2f%%",(unsigned int)
(1000000/pwm2_T_count),PWM2_Duty*100);
LCD_DisplayStringLine(Line7,lcd_disp_string);
```

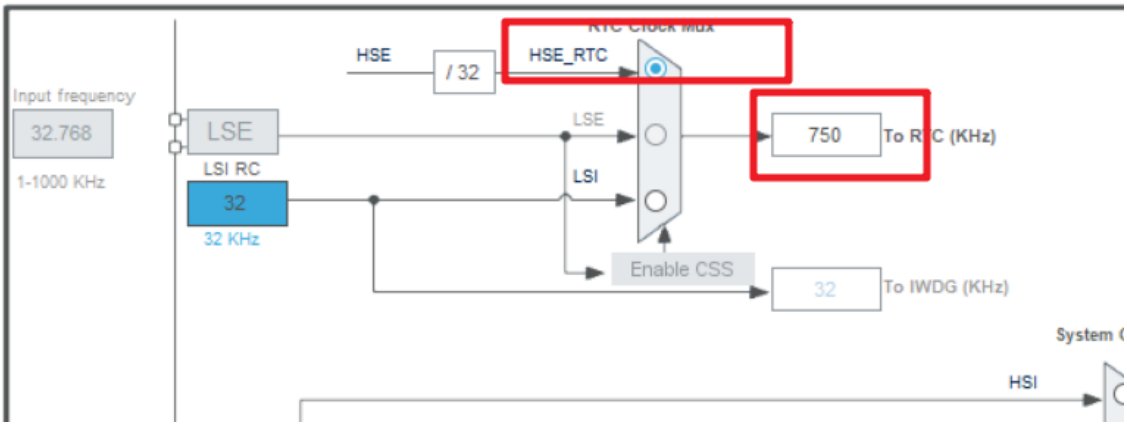
> ```
> __HAL_TIM_SET_AUTORELOAD(&htim2,499);//频率
> ```
>
> __HAL_TIM_SET_COMPARE(&htim2,TIM_CHANNEL_2,50);//占空比

9.添加RTC部分



Search (Ctrl+F)

General	
Hour Format	Hourformat 24
Asynchronous Predivider value	125
Synchronous Predivider value	6000
Calendar Time	
Data Format	BCD data format
Hours	15
Minutes	10
Seconds	5
Day Light Saving: value of hour adjus...	Daylightsaving None
Store Operation	Storeoperation Reset
Calendar Date	
Week Day	Monday
Month	January
Date	3
Year	0



```
//rtc相关变量
RTC_TimeTypeDef H_M_S_Time;
RTC_DateTypeDef Y_M_D_Time;

HAL_RTC_GetTime(&hrtc,&H_M_S_Time,RTC_FORMAT_BIN);
HAL_RTC_GetDate(&hrtc,&Y_M_D_Time,RTC_FORMAT_BIN);//必须同时使用
sprintf((char*)lcd_disp_string,"Time:%02d-%02d-%02d",(unsigned
int)H_M_S_Time.Hours,(unsigned int)H_M_S_Time.Minutes,(unsigned
int)H_M_S_Time.Seconds);
LCD_DisplayStringLine(Line0,lcd_disp_string);
```