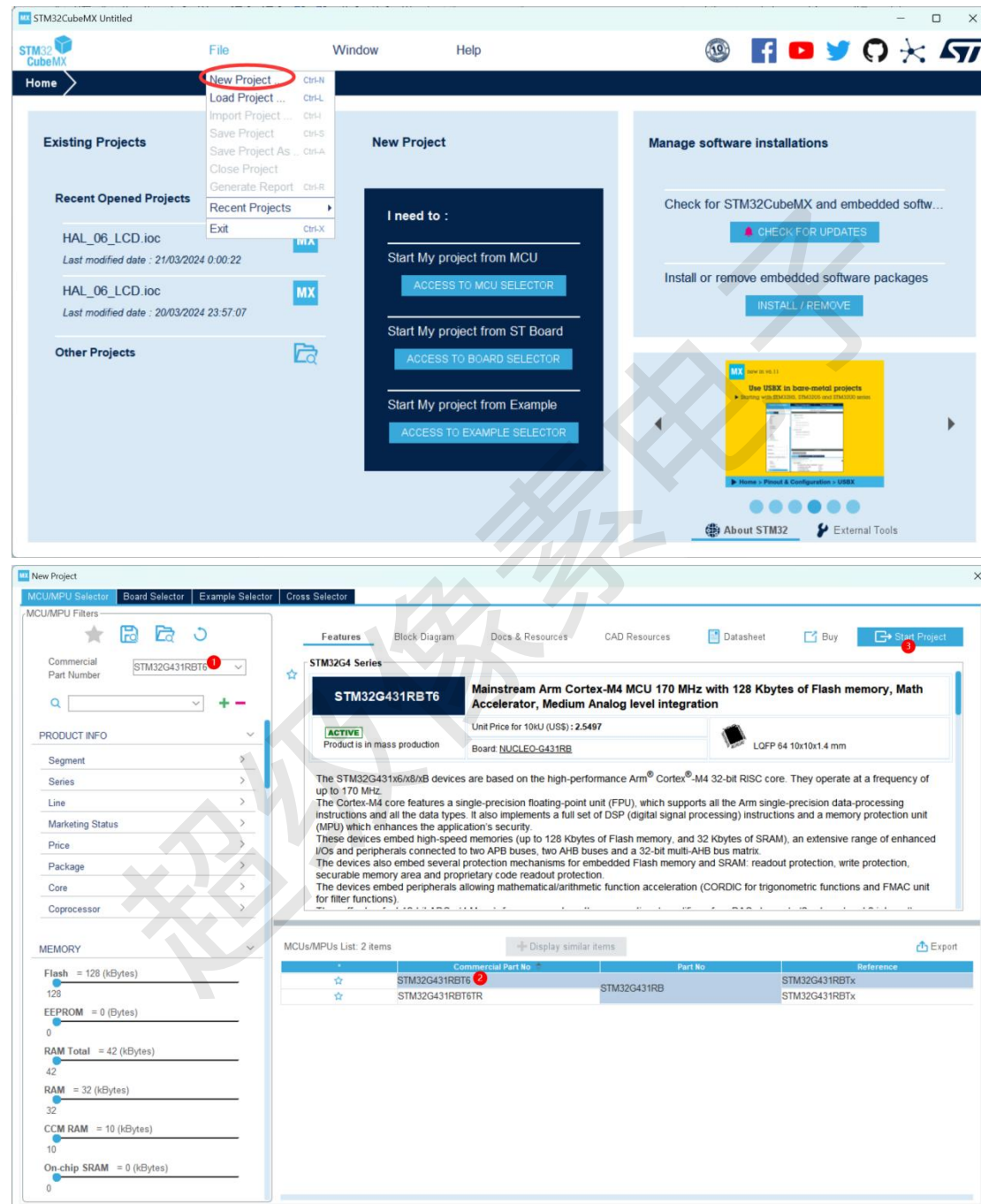
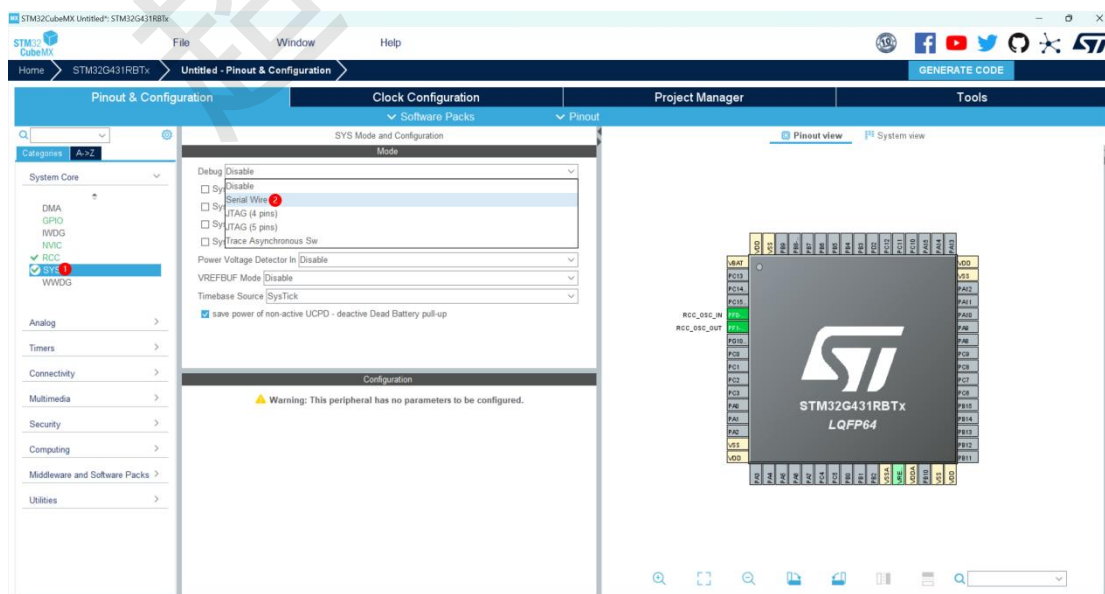


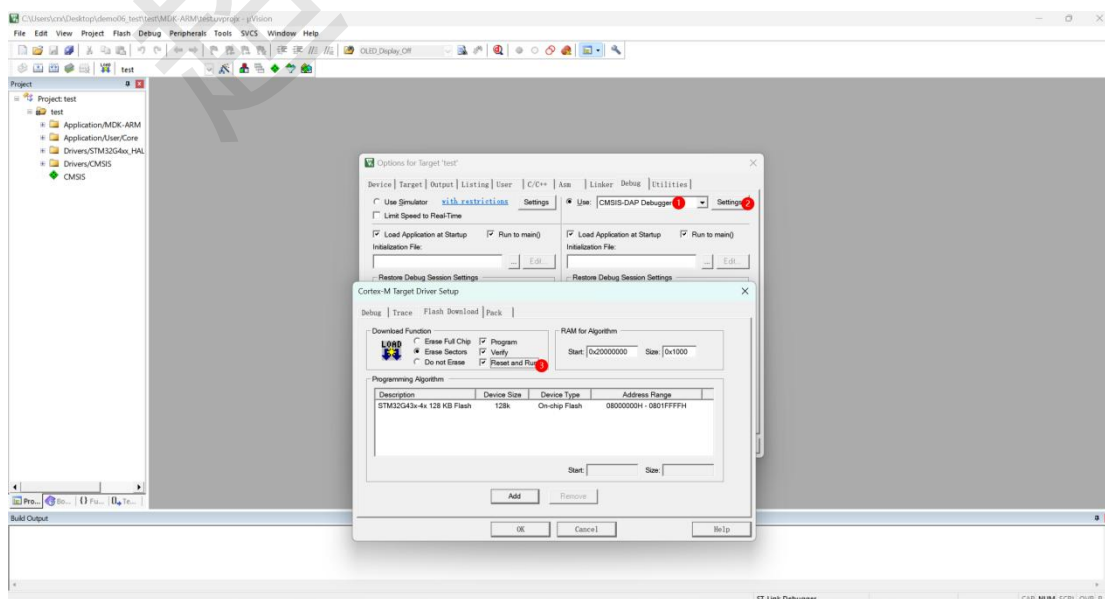
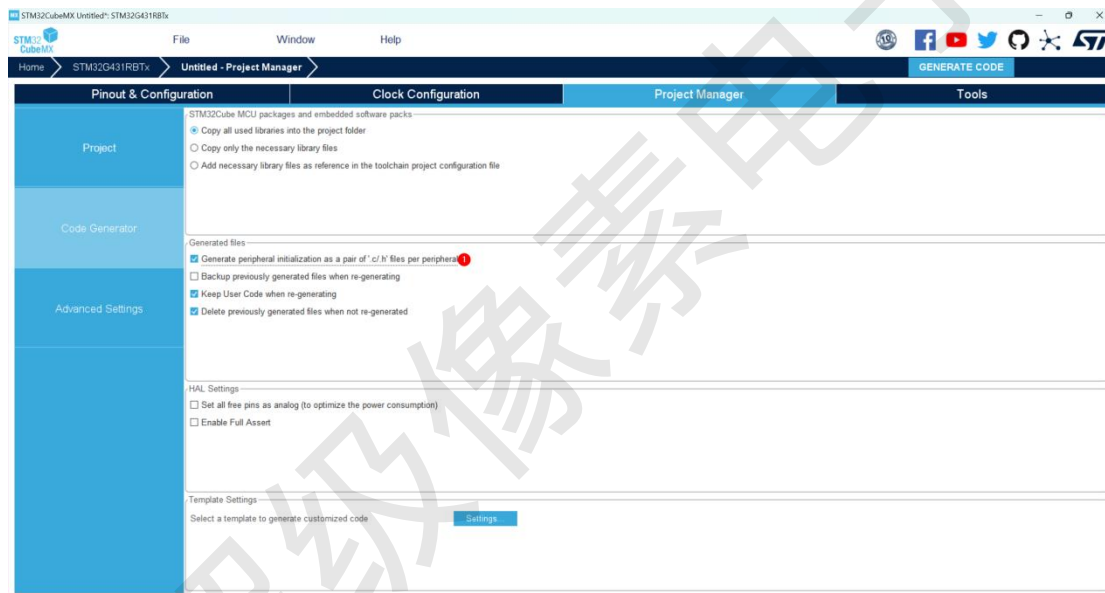
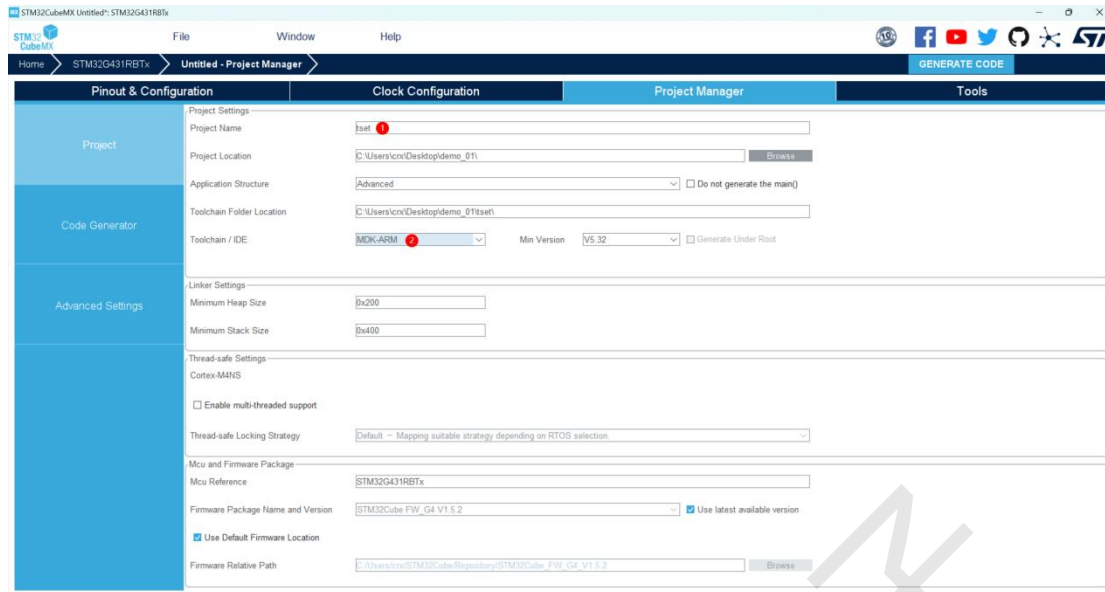
蓝桥杯-嵌入式赛道笔记

参考资料：01Studio 蓝桥杯教程 <https://space.bilibili.com/275419322>

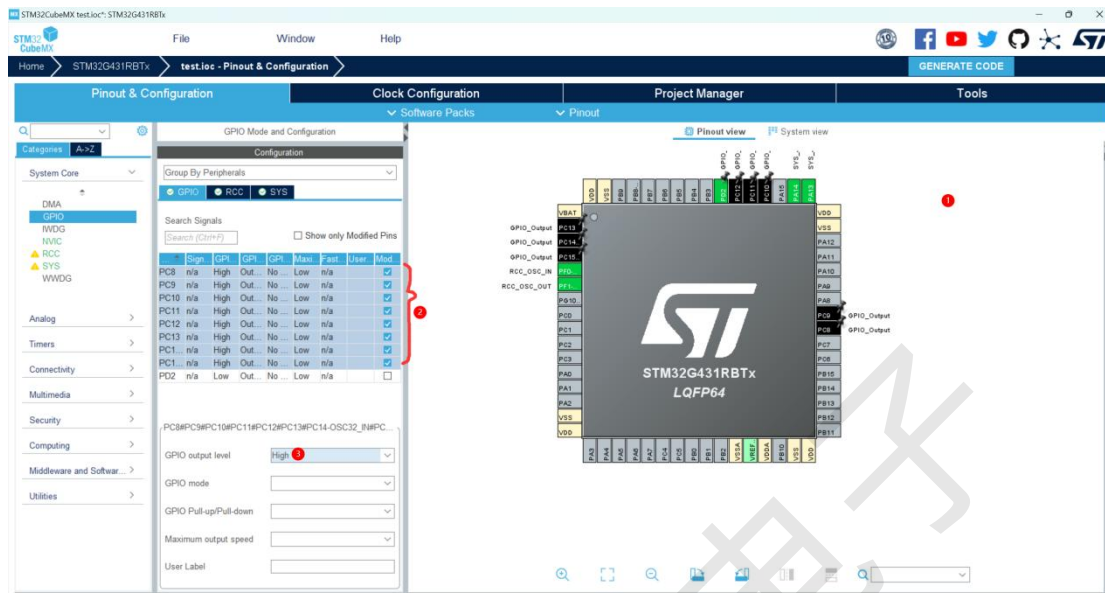
# 1.创建工程







## 2.LED

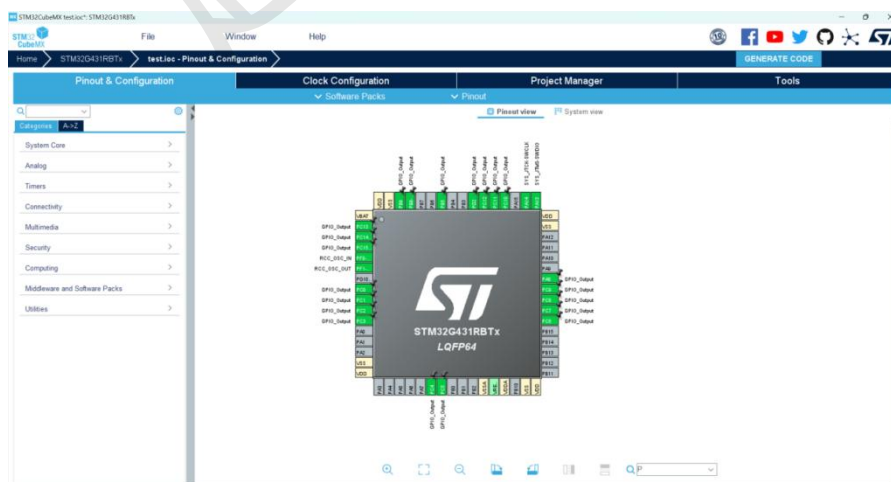


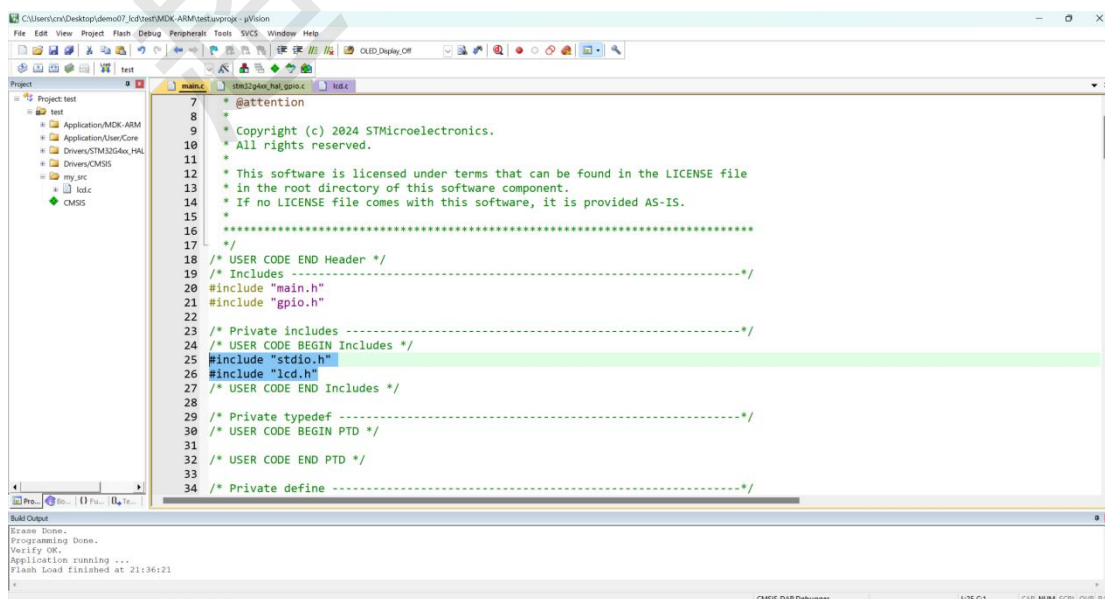
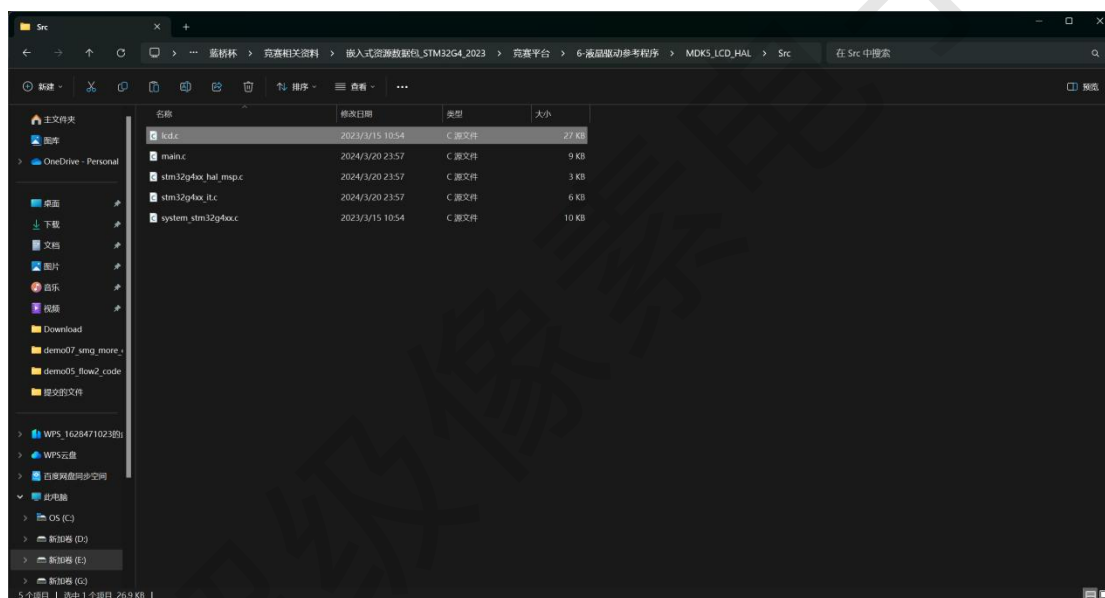
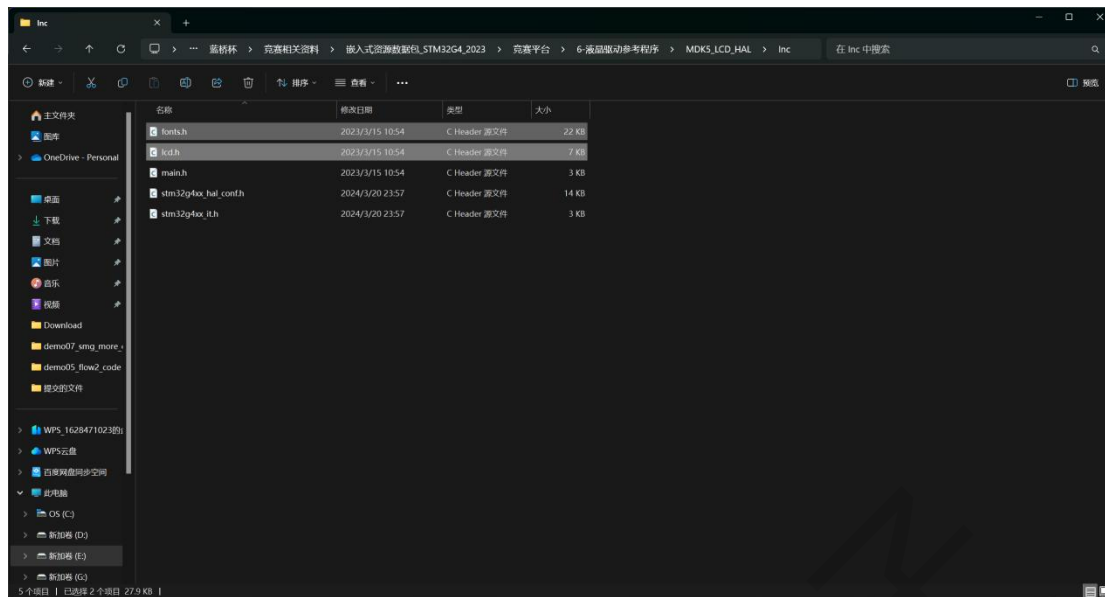
```
57 void LED(unsigned char my_data){  
58     HAL_GPIO_WritePin(GPIOC,GPIO_PIN_All,GPIO_PIN_SET);  
59     HAL_GPIO_WritePin(GPIOC,my_data<<8,GPIO_PIN_RESET);  
60     HAL_GPIO_WritePin(GPIOD,GPIO_PIN_2,GPIO_PIN_SET);  
61     HAL_GPIO_WritePin(GPIOD,GPIO_PIN_2,GPIO_PIN_RESET);  
62 }
```

```
HAL_Delay(500);
```

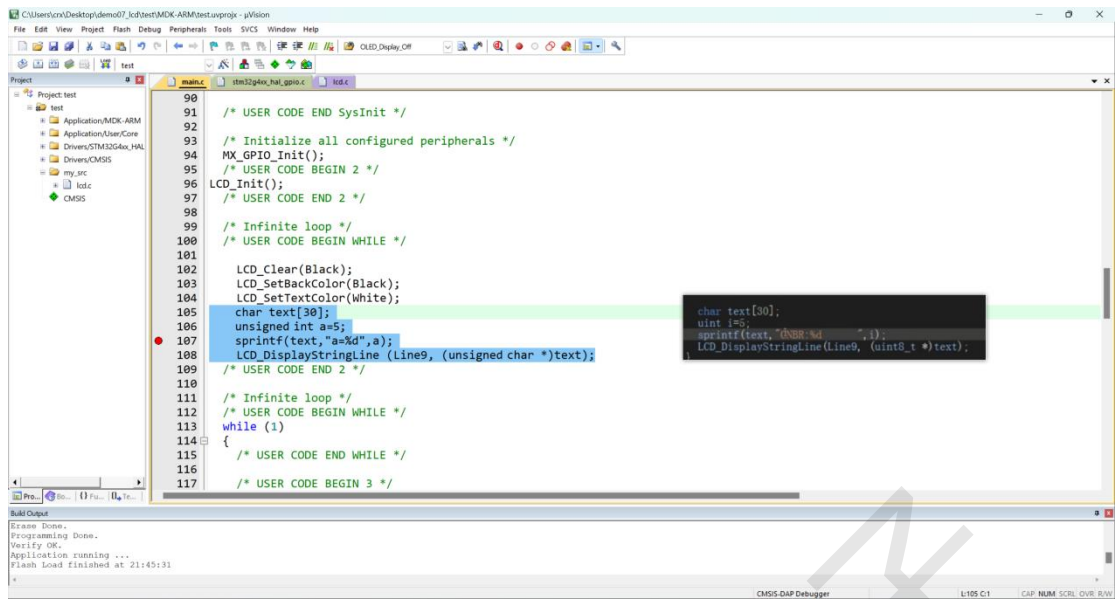
## 3.LCD

1.LCD 的初始化要放在 HAL 的初始化后才能执行

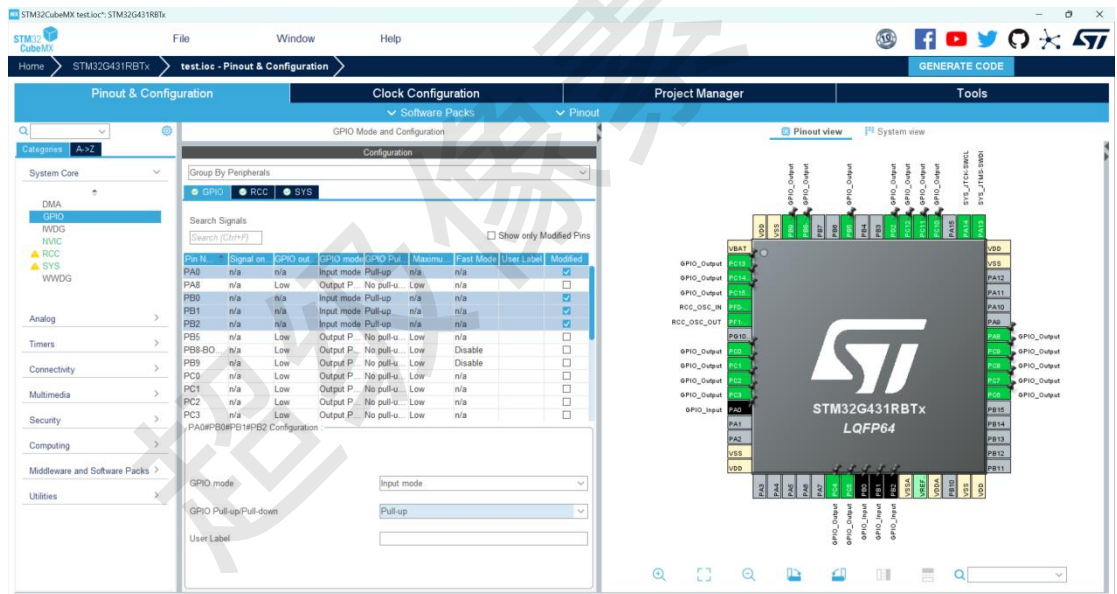


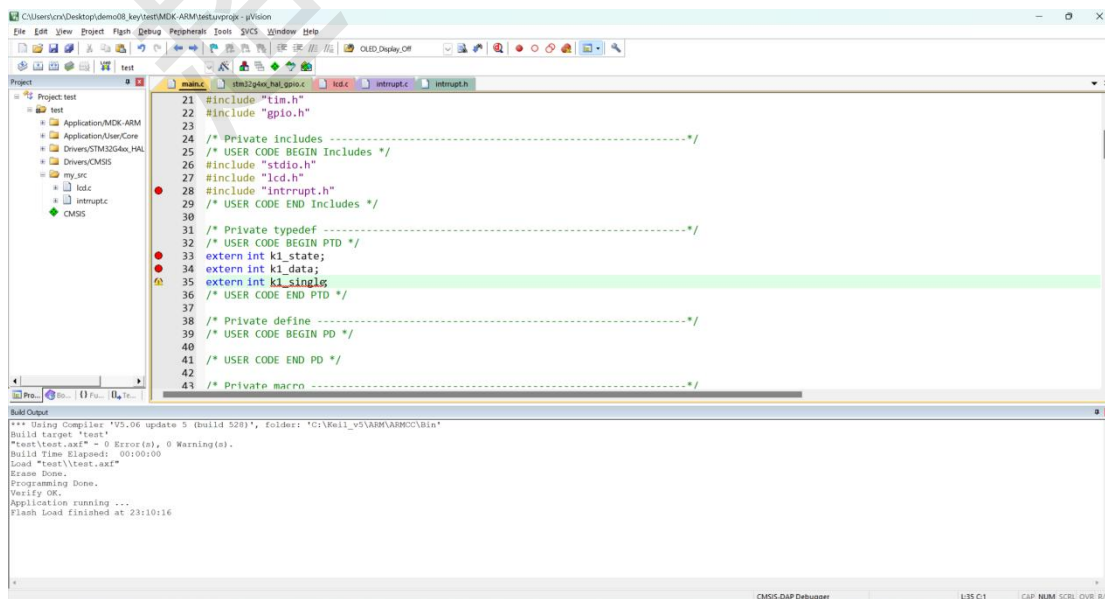
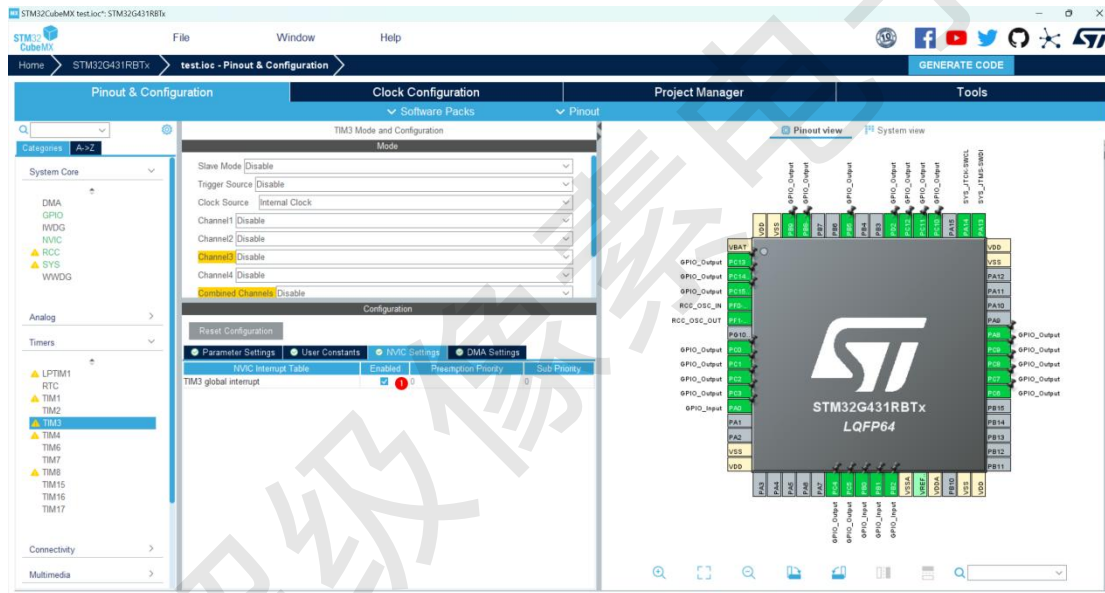
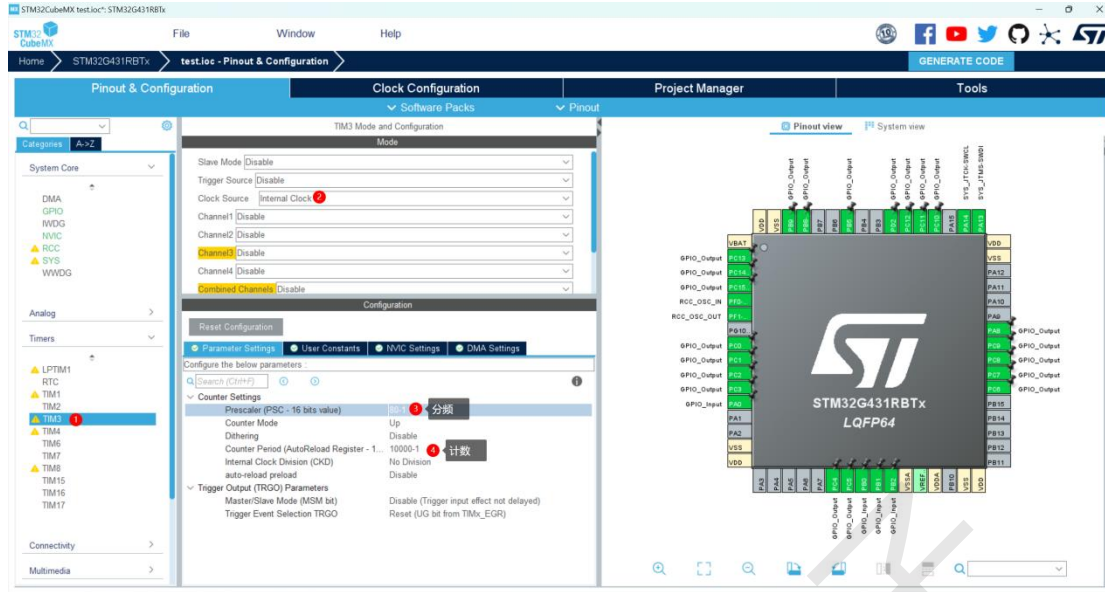


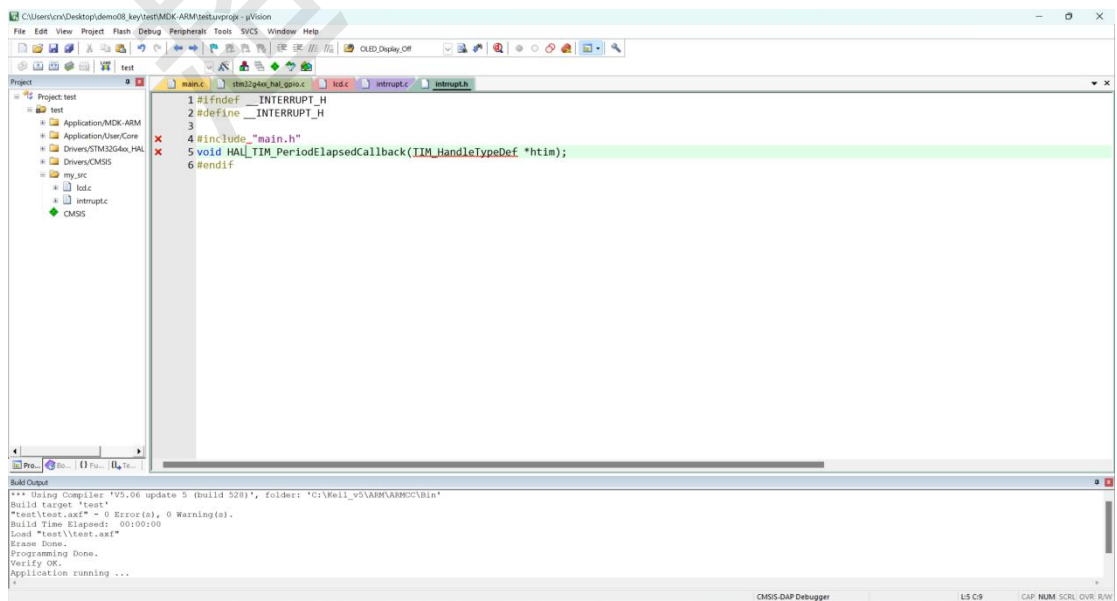
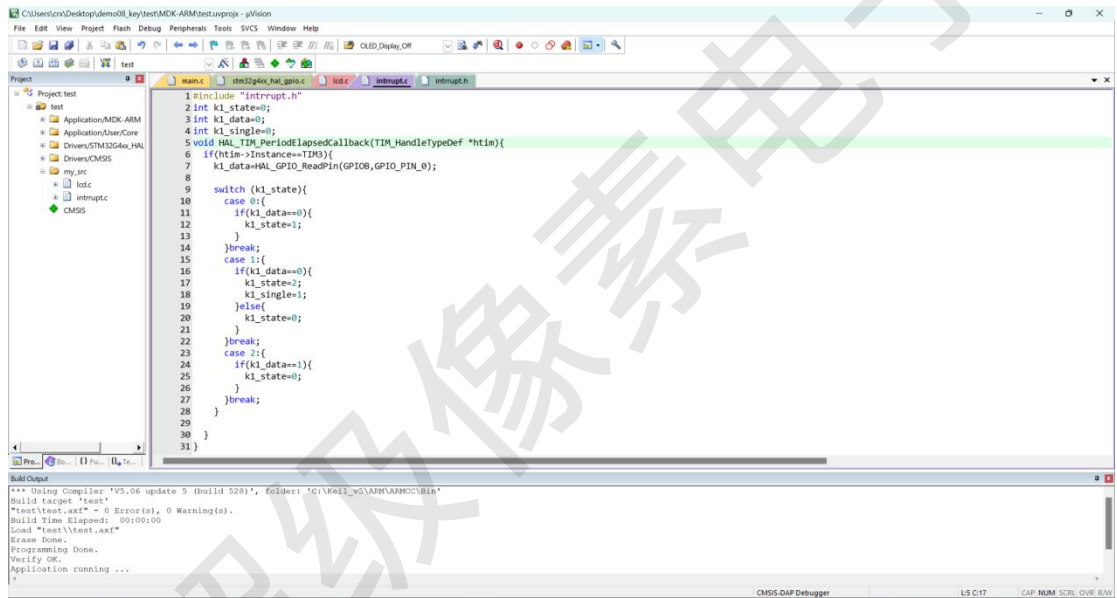
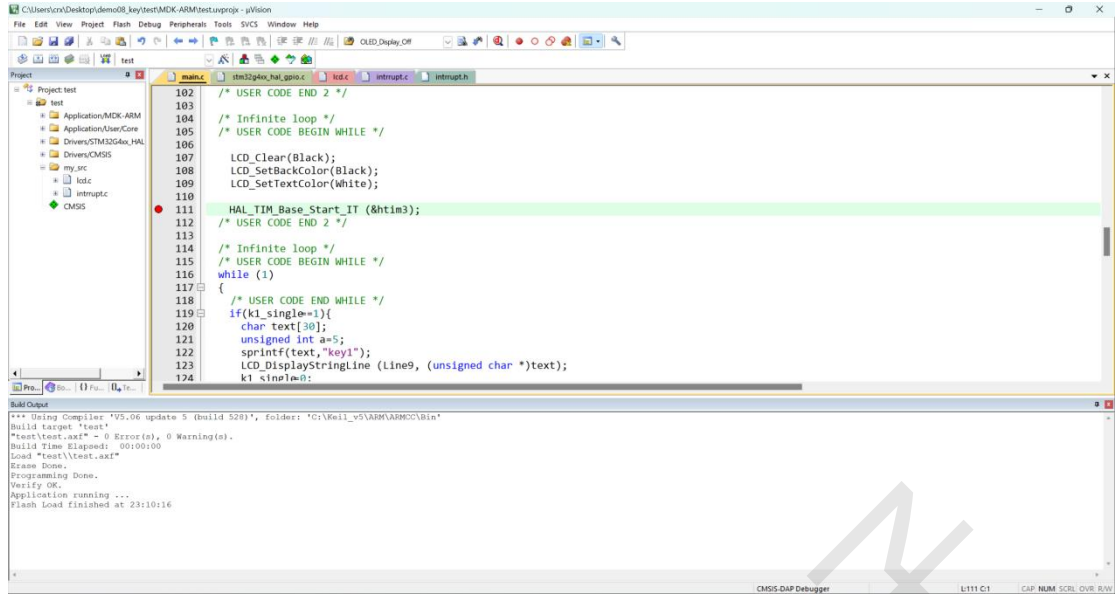




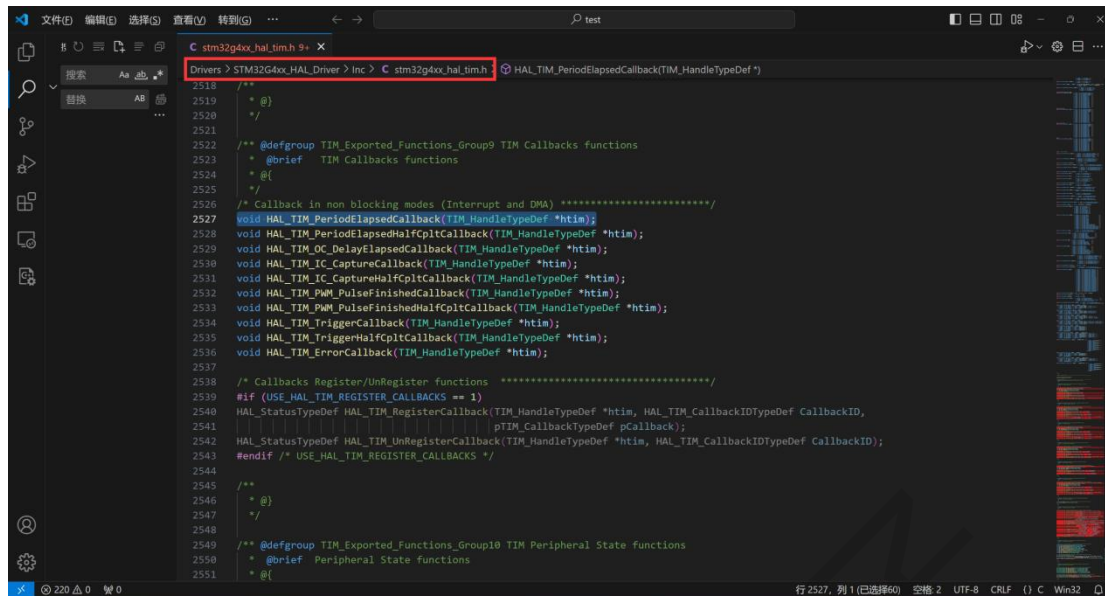
## 4.key,timer











```
2518 /**  
2519  * @}  
2520  */  
2521 /** @defgroup TIM_Exported_Functions_Group9 TIM Callbacks functions  
2522  * @brief TIM Callbacks functions  
2523  * @{  
2524  */  
2525 /* Callback in non blocking modes (Interrupt and DMA) *****  
2526 void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim);  
2527 void HAL_TIM_OC_DelayElapsedCallback(TIM_HandleTypeDef *htim);  
2528 void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim);  
2529 void HAL_TIM_IC_CaptureHalfCpltCallback(TIM_HandleTypeDef *htim);  
2530 void HAL_TIM_PWM_PulseFinishedCallback(TIM_HandleTypeDef *htim);  
2531 void HAL_TIM_PWM_PulseFinishedHalfCpltCallback(TIM_HandleTypeDef *htim);  
2532 void HAL_TIM_TriggerCallback(TIM_HandleTypeDef *htim);  
2533 void HAL_TIM_TriggerHalfCpltCallback(TIM_HandleTypeDef *htim);  
2534 void HAL_TIM_ErrorCallback(TIM_HandleTypeDef *htim);  
2535  
2536 /* Callbacks Register/UnRegister functions *****  
2537 #if (USE_HAL_TIM_REGISTER_CALLBACKS == 1)  
2538 HAL_StatusTypeDef HAL_TIM_RegisterCallback(TIM_HandleTypeDef *htim, HAL_TIM_CallbackIDTypeDef CallbackID,  
2539 TIM_CallbackTypeDef pCallback);  
2540 HAL_StatusTypeDef HAL_TIM_UnRegisterCallback(TIM_HandleTypeDef *htim, HAL_TIM_CallbackIDTypeDef CallbackID);  
2541 #endif /* USE_HAL_TIM_REGISTER_CALLBACKS */  
2542  
2543 /**  
2544  * @}  
2545  */  
2546 /** @defgroup TIM_Exported_Functions_Group10 TIM Peripheral State functions  
2547  * @brief Peripheral State functions  
2548  * @{  
2549  
2550  
2551
```

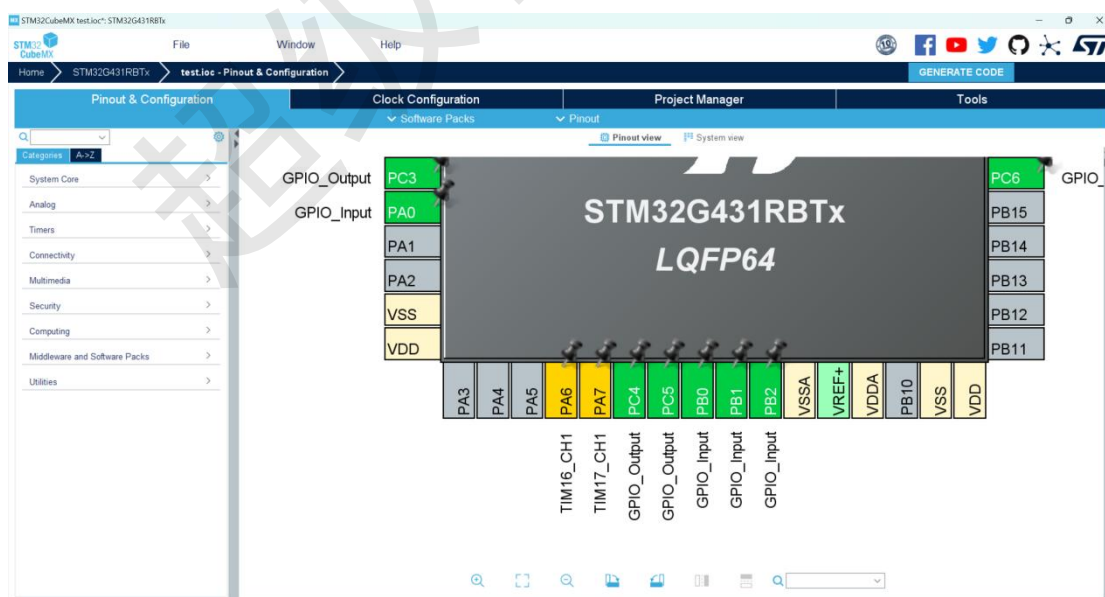
## 5.定时器

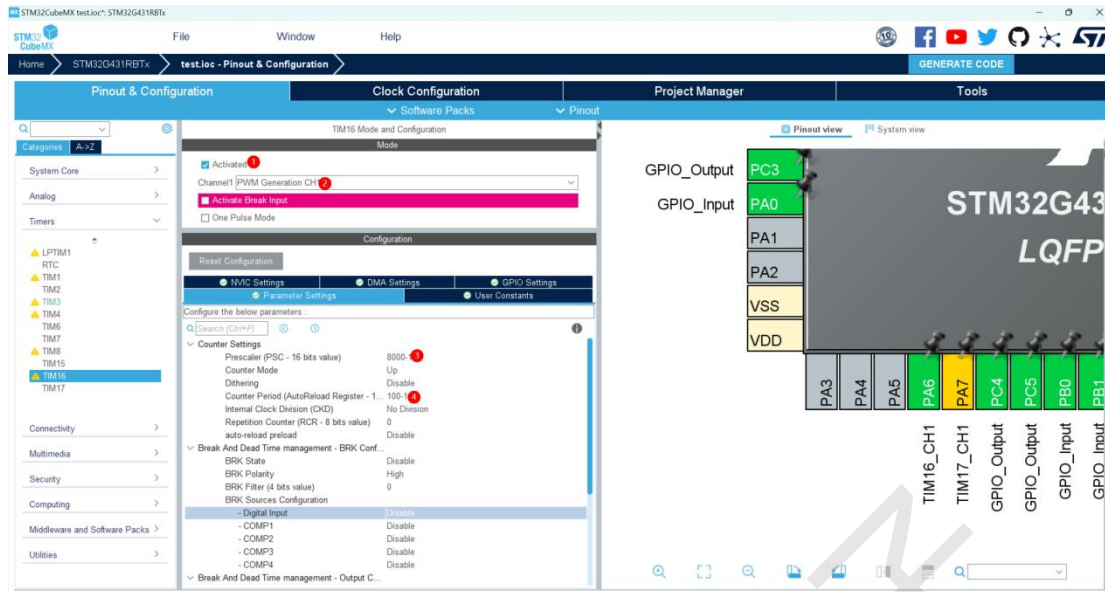
1. 时钟频率的装载值需要-1
2. 初始占空比记得设置,否则需要在软件里进行设置
3. 初始占空比的值是 重装载寄存器值\*占空比%

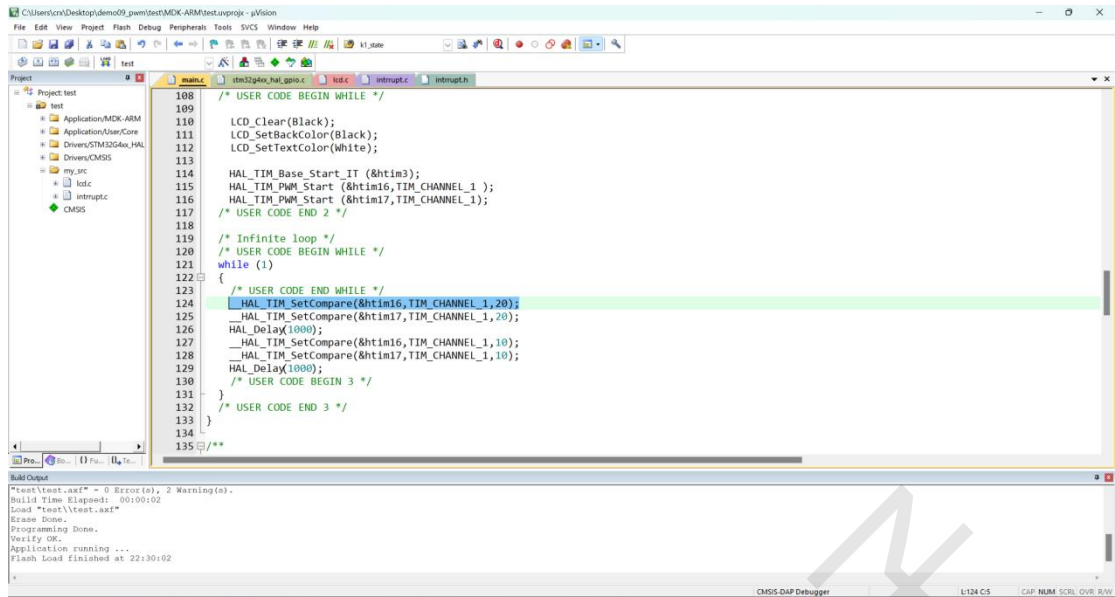
HAL\_TIM\_PWM\_Start(&htim2,TIM\_CHANNEL\_2);

\_\_HAL\_TIM\_SET\_PRESCALER(&htim2,400-1);

\_\_HAL\_TIM\_SetCompare(&htim2,TIM\_CHANNEL\_2,20);



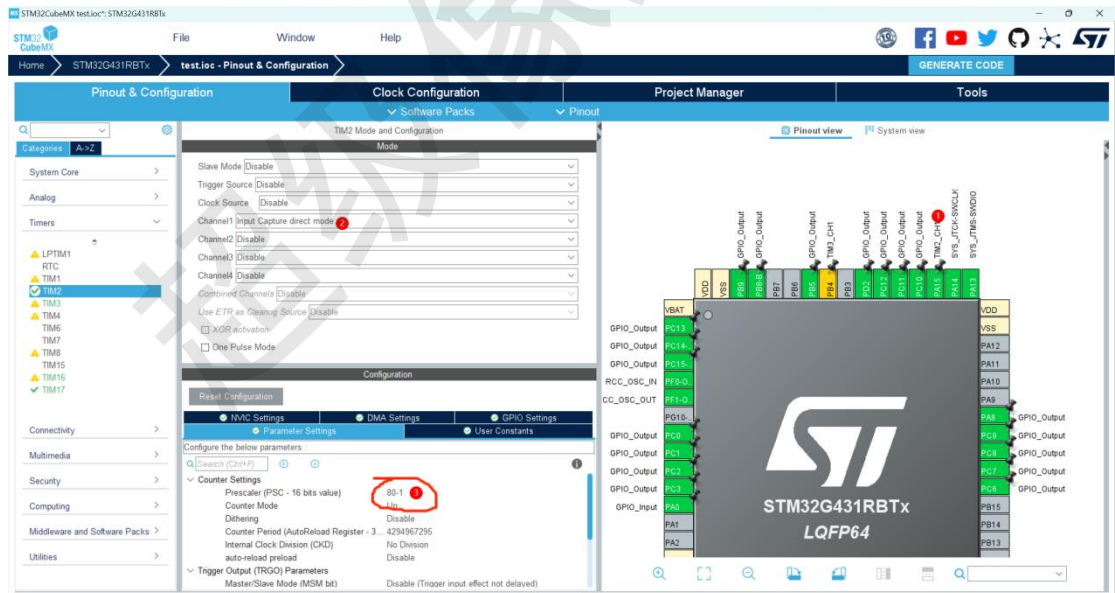


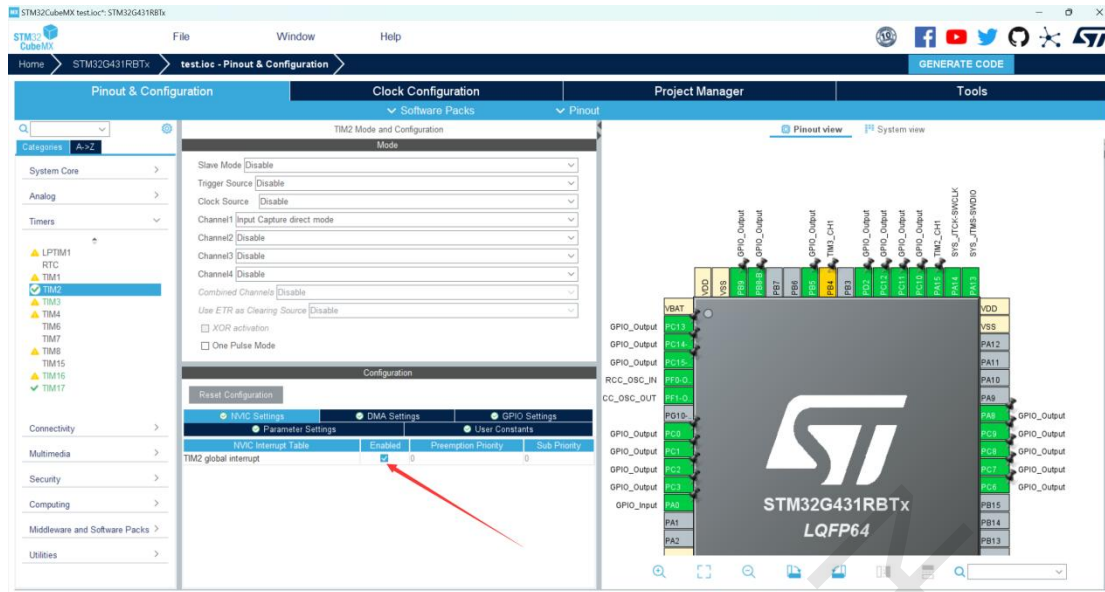


## 6. In

### 6.1 普通

#### 1.记得 HAL\_TIM\_IC\_Start\_IT

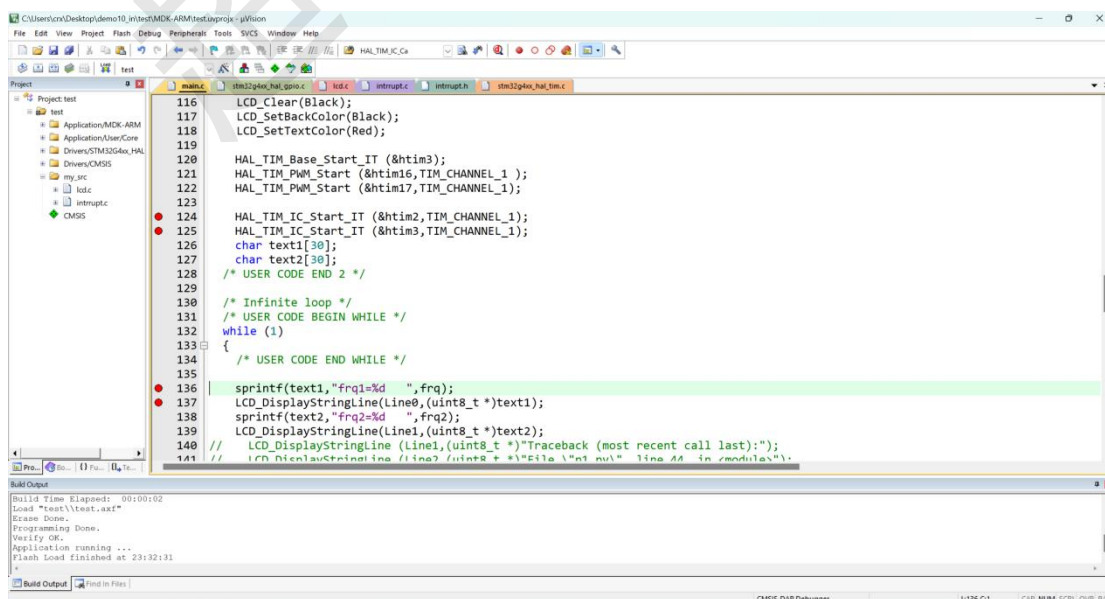




```

43 unsigned int val=0;
44 unsigned int frq=0;
45 unsigned int val2=0;
46 unsigned int frq2=0;
47 void HAL_TIM_IC_CaptureCallback (TIM_HandleTypeDef *htim){
48     if(htim->Instance==TIM2){
49         val=HAL_TIM_ReadCapturedValue(htim,TIM_CHANNEL_1);
50         __HAL_TIM_SetCounter(htim,0);
51         frq=(80000000/80)/val;
52         HAL_TIM_IC_Start (htim,TIM_CHANNEL_1);
53     }
54     if(htim->Instance==TIM3){
55         val=HAL_TIM_ReadCapturedValue(htim,TIM_CHANNEL_1);
56         __HAL_TIM_SetCounter(htim,0);
57         frq2=(80000000/80)/val;
58         HAL_TIM_IC_Start (htim,TIM_CHANNEL_1);
59     }
60 }

```

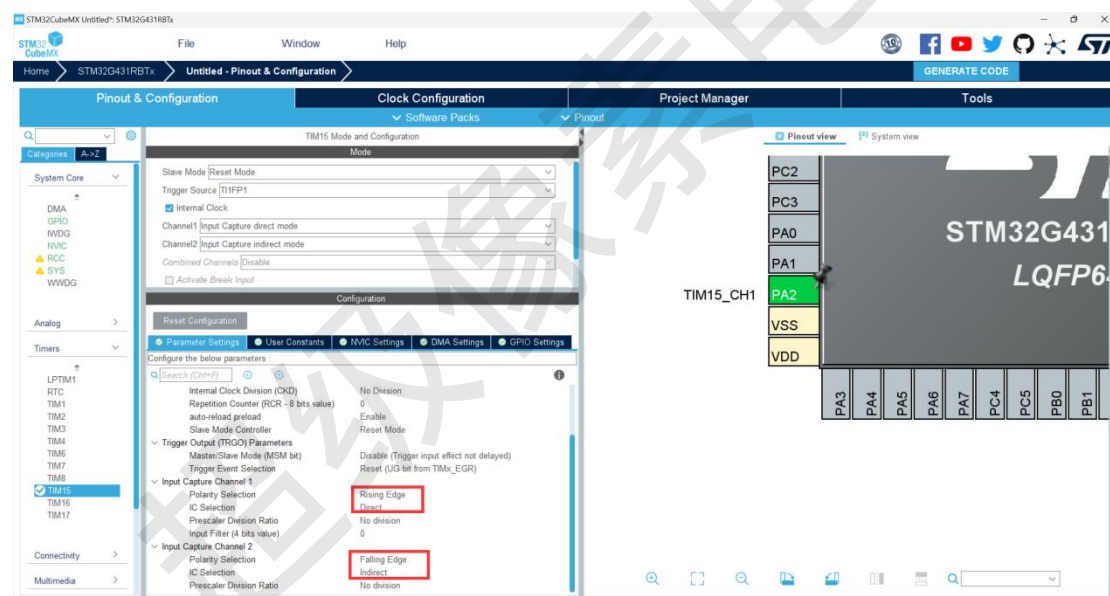
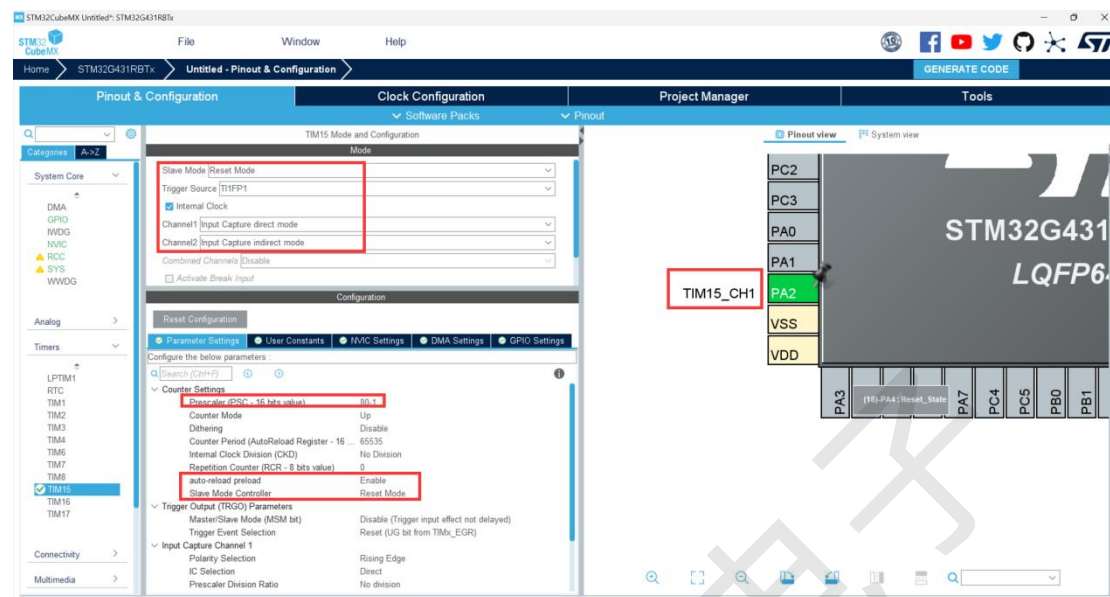




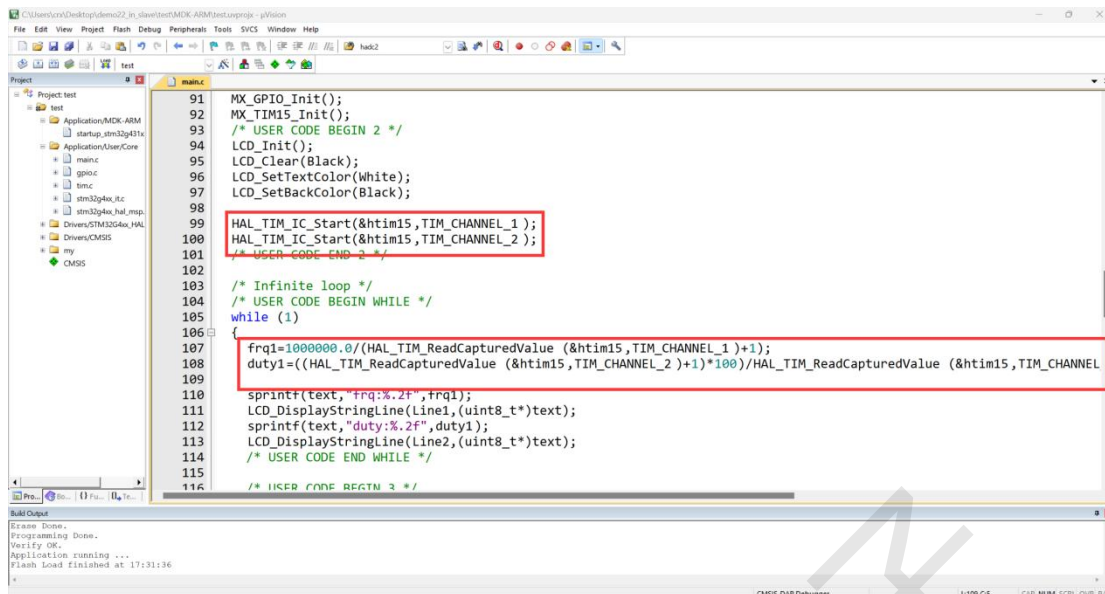
输入捕获从模式测量频率和占空比

从模式里，不需要中断函数去读取定时器数值，而且不需要清零和从新启动定时器

注意如果测占空比，两个通道都要 start







## 6.2 测占空比

```

8 void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim){
9     if(htim->Instance==TIM2){
10         if(htim->Channel==HAL_TIM_ACTIVE_CHANNEL_1){
11             ccr1_val1a=HAL_TIM_ReadCapturedValue (htim,TIM_CHANNEL_1);
12             ccr1_val1b=HAL_TIM_ReadCapturedValue (htim,TIM_CHANNEL_2);
13             __HAL_TIM_SetCounter(htim,0);
14             frq1=(80000000/80)/ccr1_val1a;
15             zhan1=ccr1_val1b/ccr1_val1a;
16             HAL_TIM_IC_Start(htim,TIM_CHANNEL_1);
17             HAL_TIM_IC_Start(htim,TIM_CHANNEL_2);
18         }
19     }

```

如果通道 2 是直接输入，通道 1 是间接输入：

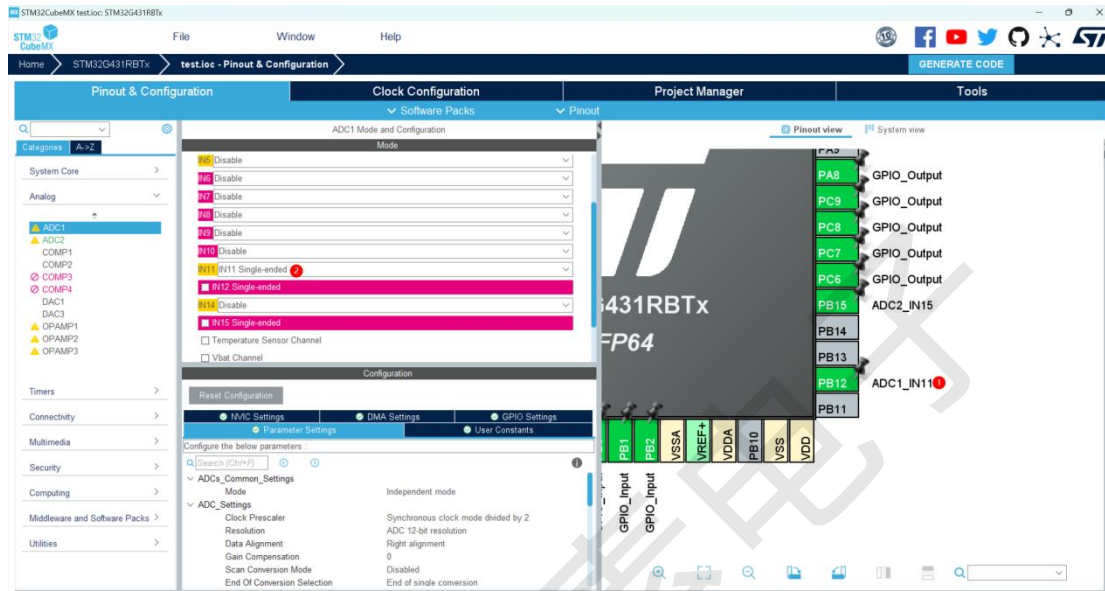
```

88 void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim){
89     if(htim->Instance==TIM2){
90         if(htim->Channel==HAL_TIM_ACTIVE_CHANNEL_2){
91             int temp1,temp2;
92             temp2=HAL_TIM_ReadCapturedValue(htim,TIM_CHANNEL_1);
93             temp1=HAL_TIM_ReadCapturedValue(htim,TIM_CHANNEL_2);
94
95             pa1_freq=(80000000/80)/temp1;
96             pa1_duty=(temp2*100)/temp1;
97             __HAL_TIM_SetCounter(htim,0);
98             HAL_TIM_IC_Start(htim,TIM_CHANNEL_1);
99             HAL_TIM_IC_Start(htim,TIM_CHANNEL_2);
100         }
101     }
102

```

## 7. Adc

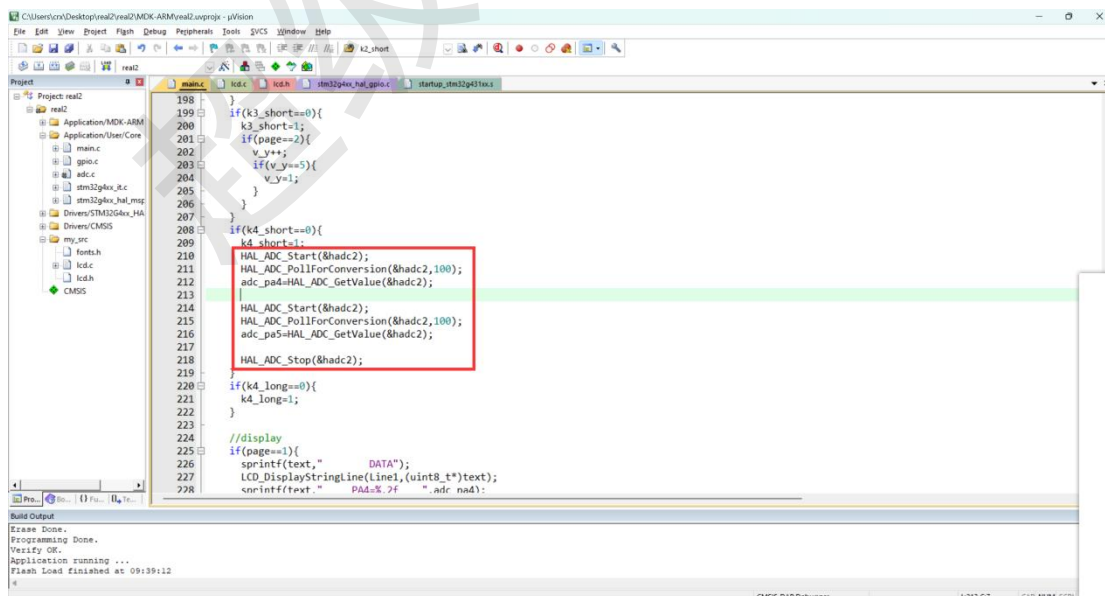
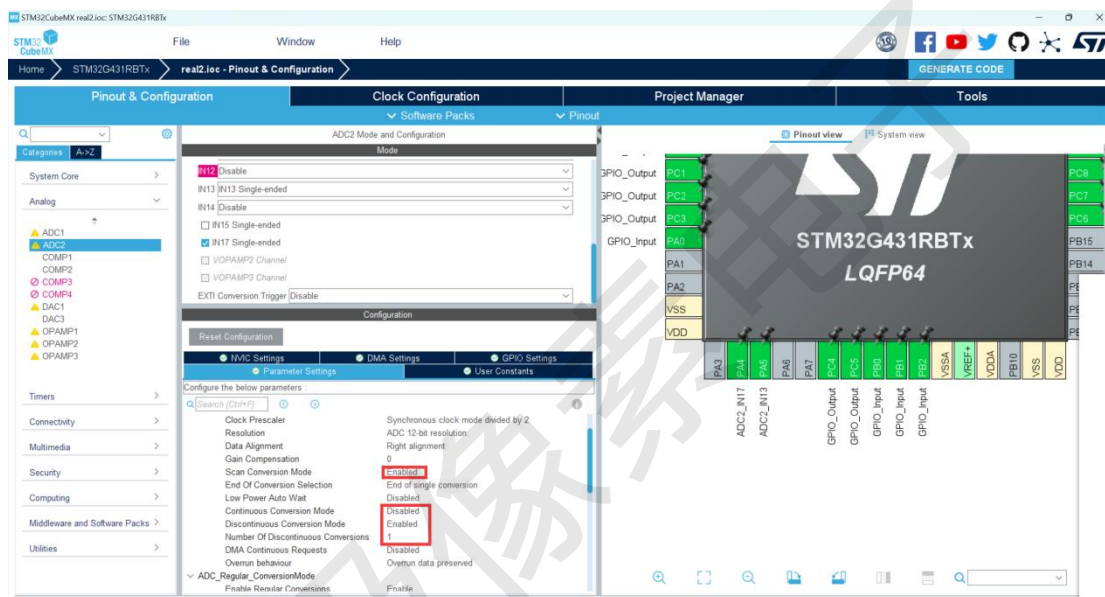
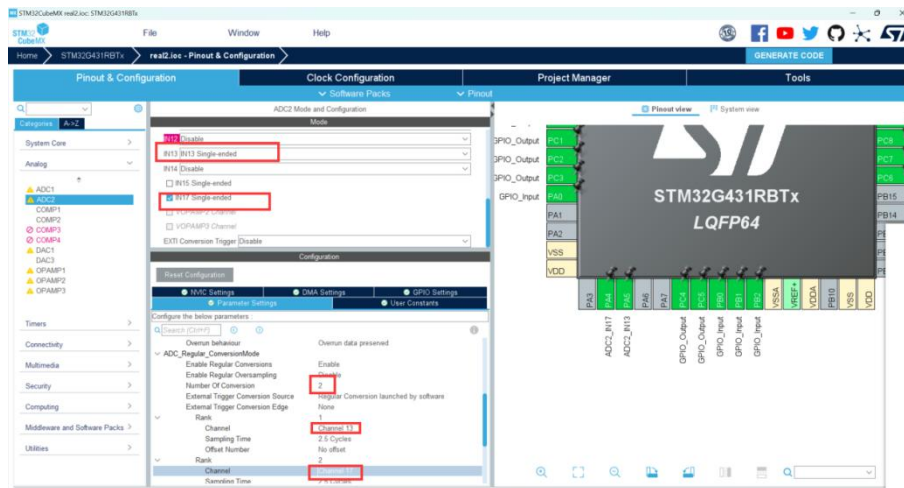
### 7.1ADC 单通道



```
double getADC(ADC_HandleTypeDef *pin){  
    unsigned int adc_val;  
    HAL_ADC_Start(pin);  
    adc_val=HAL_ADC_GetValue(pin);  
    return adc_val*3.3/4096;  
}  
  
char text1[30];  
char text2[30];  
sprintf(text1,"adc1=%.2f",getADC(&hadc1));  
sprintf(text2,"adc2=%.2f",getADC(&hadc2));  
LCD_DisplayStringLine(Line1,(uint8_t *)text1);  
LCD_DisplayStringLine(Line2,(uint8_t *)text2);
```

### 7.2ADC 多通道

记得使能 Continuous Conversion Mode

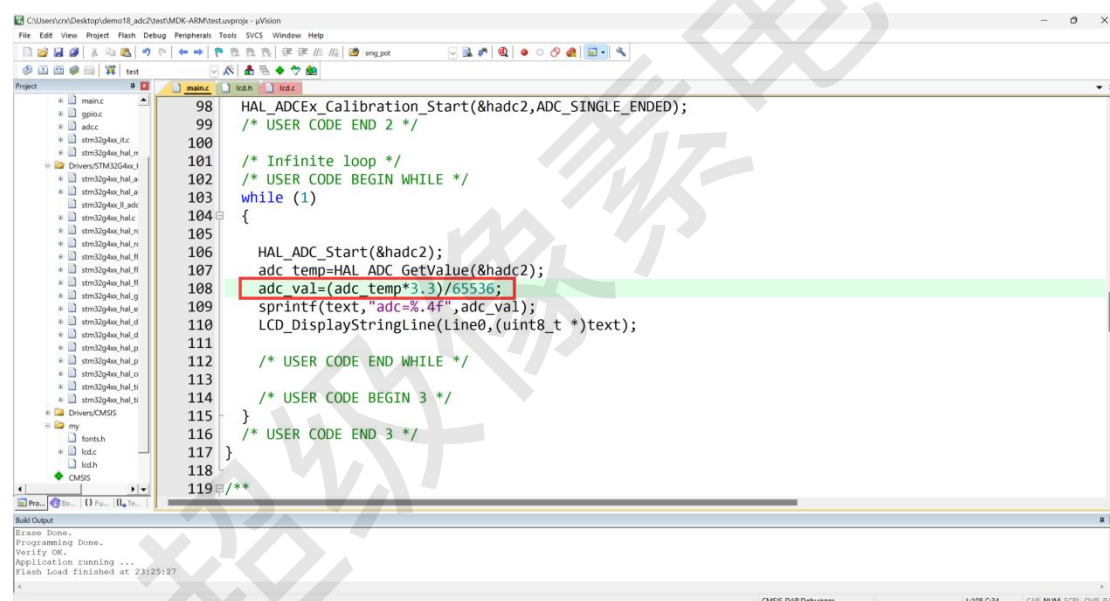
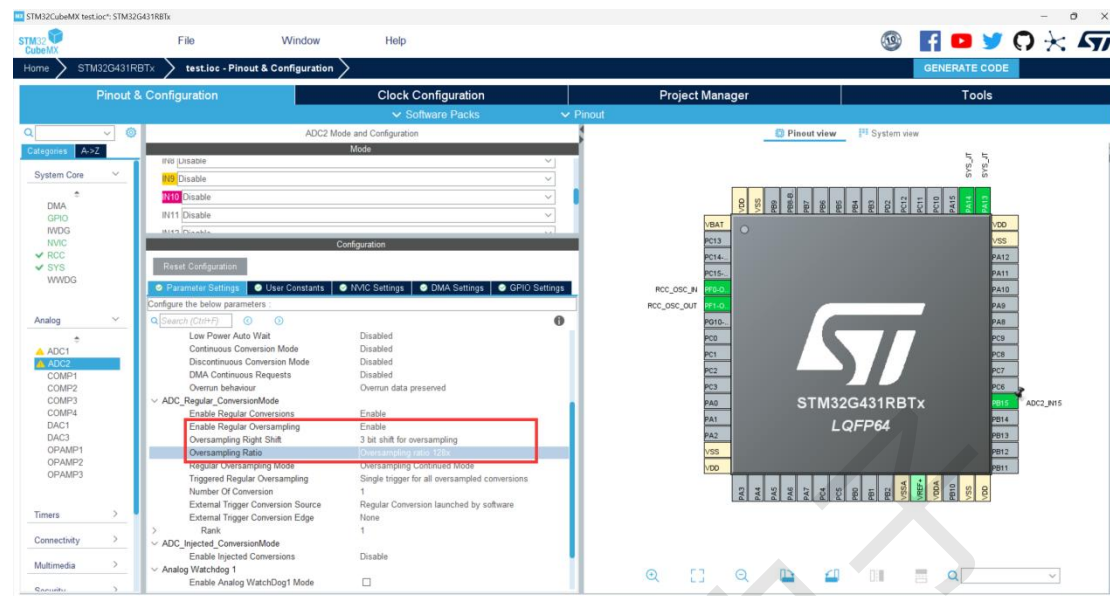


## Adc 校准

HAL\_ADCEx\_Calibration\_Start(&adc2, ADC\_SINGLE\_ENDED)

adc 滤波

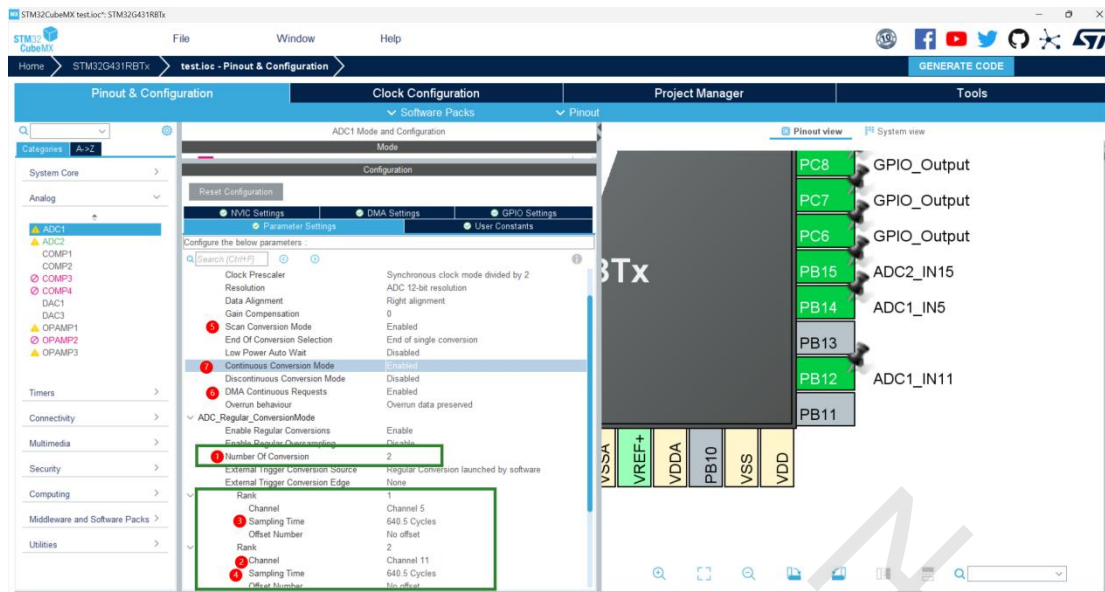
不能使用 `HAL_ADC_Stop(&hadcx);`



## 7.3 ADC\_DMA

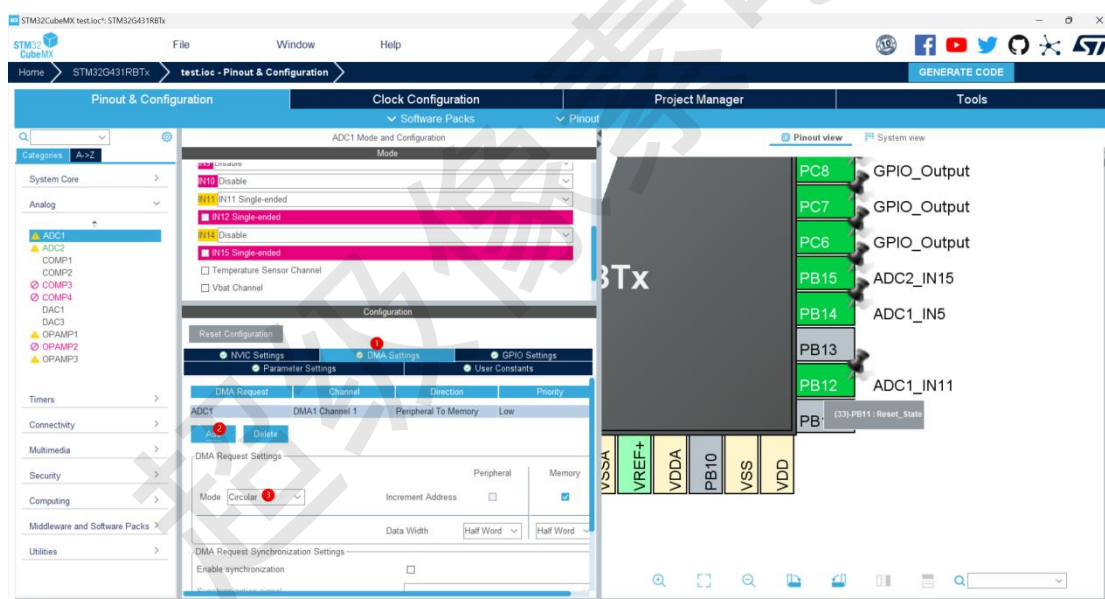
1. 缓冲区使用 `uint16_t`, 在调用 `HAL_ADC_Start_DMA` 时使用 `(uint32_t *)` 强制转换
2. 调大采样周期





```
HAL_ADC_Start_DMA(&hadc1,(uint32_t*)adc_buf,2);
```

```
58 uint16_t adc_buf[2];
```



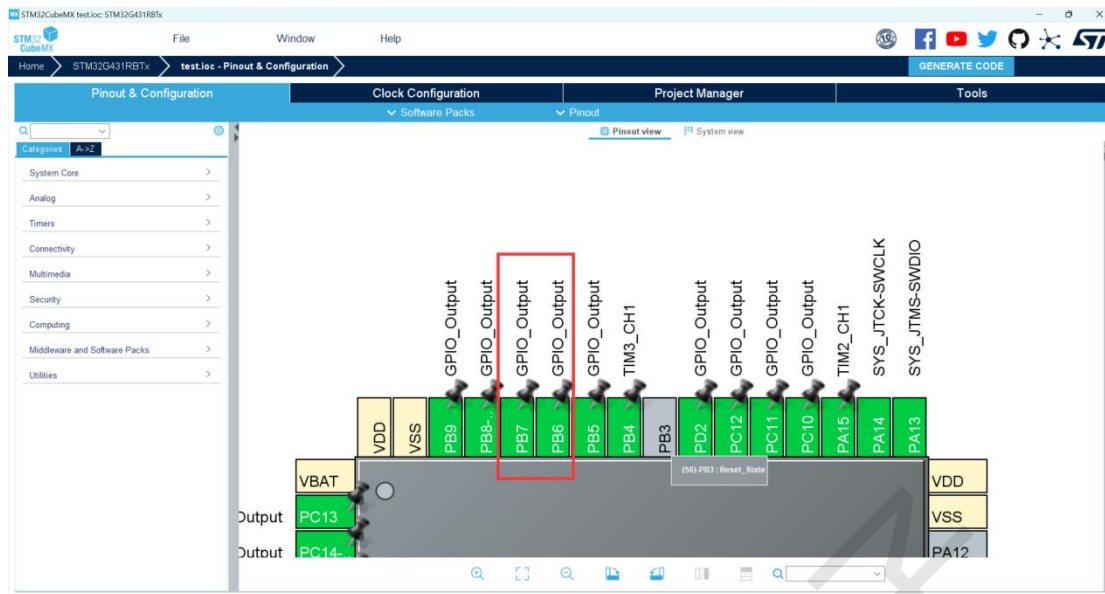
## 8.lic

1. 数据类型都为 unsigned char

2. 忘记写 return

3. 变量的命名要加上自己的前缀，避免和系统原有变量冲突（编译器不会报错，但是运行会出现 bug）





```

- unsigned char eeprom_read(unsigned char addr){
    unsigned char dat;
    I2CStart();
    I2CSendByte(0xa0);
    I2CWaitAck();
    I2CSendByte(addr);
    I2CWaitAck();
    I2CStop();

    I2CStart();
    I2CSendByte(0xa1);
    I2CWaitAck();
    dat=I2CReceiveByte();
    I2CWaitAck();
    I2CStop();
    return dat;
}

- void eeprom_write(unsigned char addr,unsigned char dat){
    I2CStart();
    I2CSendByte(0xa0);
    I2CWaitAck();
    I2CSendByte(addr);
    I2CWaitAck();
    I2CSendByte(dat);
    I2CWaitAck();
    I2CStop();
}

```

小数的存储和读取

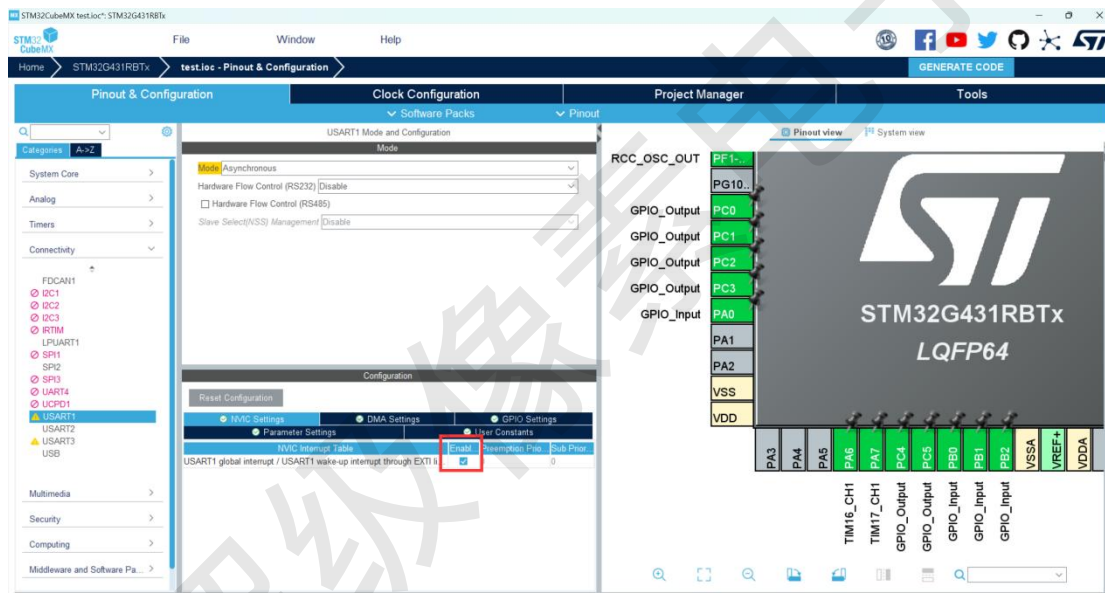
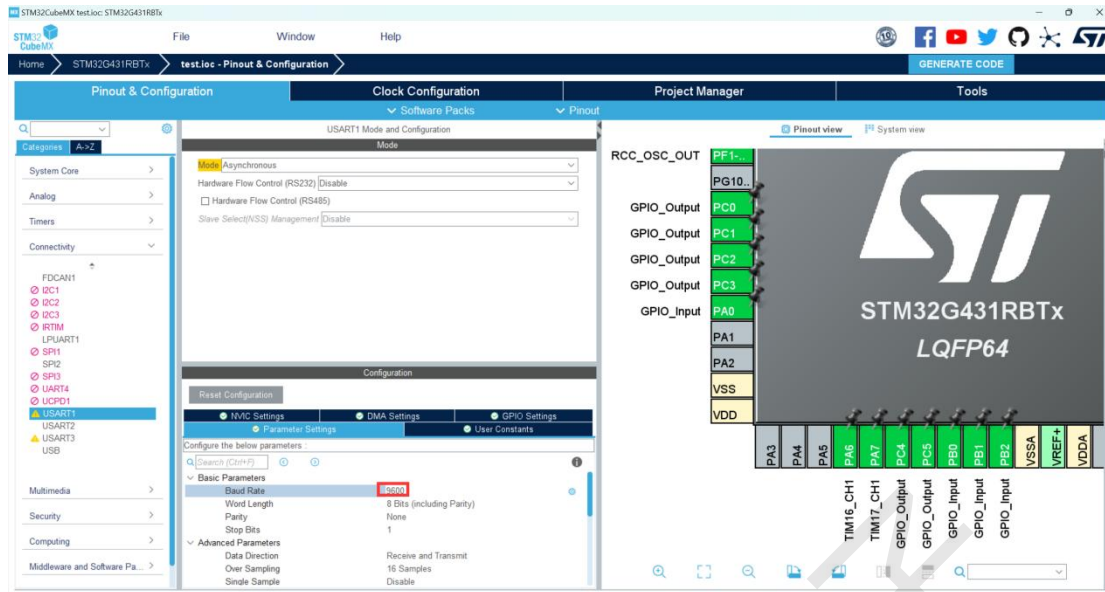
1. 取地址

2. 强制类型转换

3. 数组下标取指针

## 9.Uart

-



```
int a=5;
char temp[20];
sprintf(temp,"a=%d\r\n",a);
HAL_UART_Transmit(&huart1,(uint8_t *)temp,strlen(temp),50);
```

```
#include "usart.h"
char rxdata[30];
uint8_t rxdat;
unsigned char rx_p;
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart){
    rxdata[rx_p]=rxdat;
    HAL_UART_Receive_IT(&huart1,&rxdat,1);
    rx_p++;
}
```

```
HAL_UART_Receive_IT(&huart1,&rxdat,1);

void uart_receive(){
    if(rx_p>0){
        if(rx_p==9){
            sscanf(rxdata,"%4s:%4s",data1,data2);
        }else{
            char temp[20];
            sprintf(temp,"error");
            HAL_UART_Transmit(&huart1,(uint8_t *)temp,strlen(temp),50);
        }
        rx_p=0;memset(rxdata,0,30);
    }
}
```

## 10.总结

GPIO 设置: HAL\_GPIO\_WritePin(GPIOx,GPIO\_PIN\_x,GPIO\_PIN\_SET/RESET)

a=HAL\_GPIO\_ReadPin(GPIOx,GPIO\_PIN\_x)

延时: HAL\_Delay(ms);

定时器中断函数: HAL\_TIM\_Base\_Start\_IT(&htimx)

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

定时器 pwm: HAL\_TIM\_PWM\_Start(&htimx,TIM\_CHANNEL\_x)

\_\_HAL\_TIM\_SET\_PRESCALER(&htimx,xxx);

\_\_HAL\_TIM\_SET\_COMPARE(&htimx,TIM\_CHANNEL\_x,xxx);

\_\_HAL\_TIM\_SET\_AUTORELOAD(&htimx,x);

定时器 in: HAL\_TIM\_IC\_Start\_IT(&htimx,TIM\_CHANNEL\_x)

从模式使用 HAL\_TIM\_IC\_Start(&htimx,TIM\_CHANNEL\_x)

**void HAL\_TIM\_IC\_CaptureCallback(TIM\_HandleTypeDef \*htim)**

if(htim->Instance==TIMx){

**if(htim->Channel==HAL\_TIM\_ACTIVE\_CHANNEL\_x){**

a=HAL\_TIM\_ReadCapturedValue(htim,TIM\_CHANNEL\_x)

\_\_HAL\_TIM\_SetCounter(htim,0)

HAL\_TIM\_IC\_Start(htim,TIM\_CHANNEL\_x)

ADC: HAL\_ADC\_Start(&hadcx)

HAL\_ADCEX\_Calibration\_Start(&hadcx,ADC\_SINGLE\_ENDED);

HAL\_ADC\_PollForConversion(&hadcx,100)

data=HAL\_ADC\_GetValue(&hadcx)

HAL\_ADC\_Stop(&hadcx)

IIC: 读: 开始->发 0xa0 等应答->发地址等应答->停止

开始->发 0xa1 等应答->接收后等应答->停止

写: 开始->发 0xa0 等应答->发地址等应答->发数据等应答->停止

串口: HAL\_UART\_Receive\_IT(&huartx,&my\_one\_char,1)

发: HAL\_UART\_Transmit(&huartx,(uint8\_t\*)my\_data,strlen(my\_data),50)

收: #include "usart.h"

**HAL\_UART\_RxCpltCallback(UART\_HandleTypeDef \*huart)**

HAL\_UART\_Receive\_IT(&huartx,&my\_one\_char,1)

Memset(...)

数字电位器: 开始->发 0x5e 等应答->发 0~0x7f 等应答->停止

USB: #include "usb\_device.h" "usb\_cdc\_if.h"

发送: CDC\_Transmit\_FS(text,sizeof(text))

接收: CDC\_Receive\_FS 内添加转运 Len 和 Buf 代码

C 语言字符串: 1.连接字符串 strcat (string1, string2)

2.拷贝字符串 strcpy (string1, string2)

3.比较字符串 strcmp(string1, string2)

4.求字符串长度 strlen(string)

5.字符串转数据: #include "stdlib.h"

转化为 int 型: atoi()



转换为 double 类型:atof()

转化为 long 类型:atol()

6. 识别数据:

```
sscanf(receive_data,"%3s,%3s",&string1,&string2);
```

7. 组装数据:

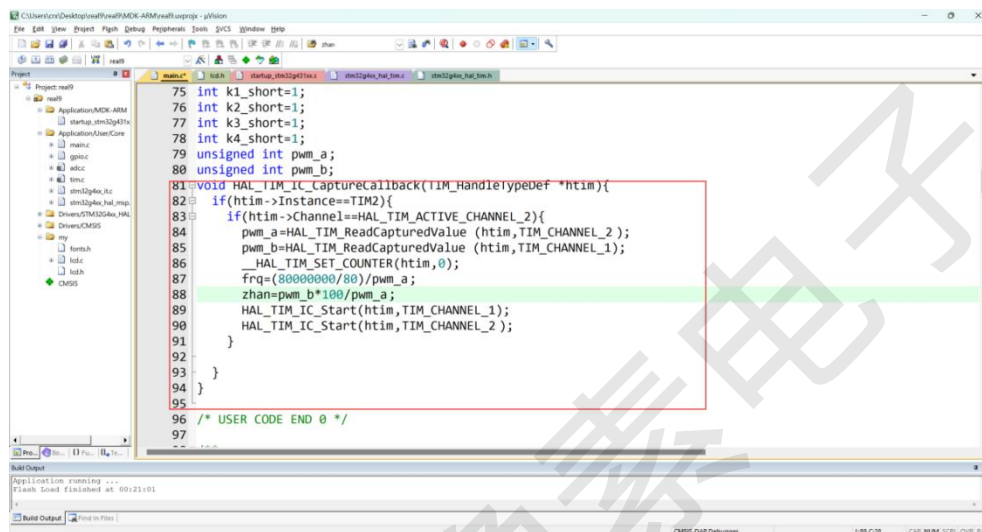
```
sprintf(temp,"1:%3s;2:%3s",&string1,&string2);
```

1. 题意理解错

2. 用定时器和 systick 来控制有关时间的操作, 频率更精确

3. 题意要求看漏

终极输入捕获



终极串口

```
int a=5;
char temp[20];
sprintf(temp,"a=%d\r\n",a);
HAL_UART_Transmit(&huart1,(uint8_t *)temp,strlen(temp),50);
```

```
#include "usart.h"
char rxdata[30];
uint8_t rxdat;
unsigned char rx_p;
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart){
    rxdata[rx_p]=rxdat;
    HAL_UART_Receive_IT(&huart1,&rxdat,1);
    rx_p++;
}

HAL_UART_Receive_IT(&huart1,&rxdat,1);

void uart_receive(){
    if(rx_p>0){
        if(rx_p==9){
            sscanf(rxdata,"%4s:%4s",data1,data2);
        }else{
            char temp[20];
            sprintf(temp,"error");
            HAL_UART_Transmit(&huart1,(uint8_t *)temp,strlen(temp),50);
        }
        rx_p=0;memset(rxdata,0,30);
    }
}
```