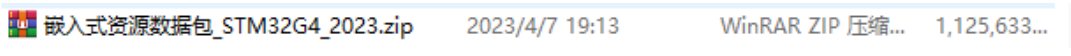
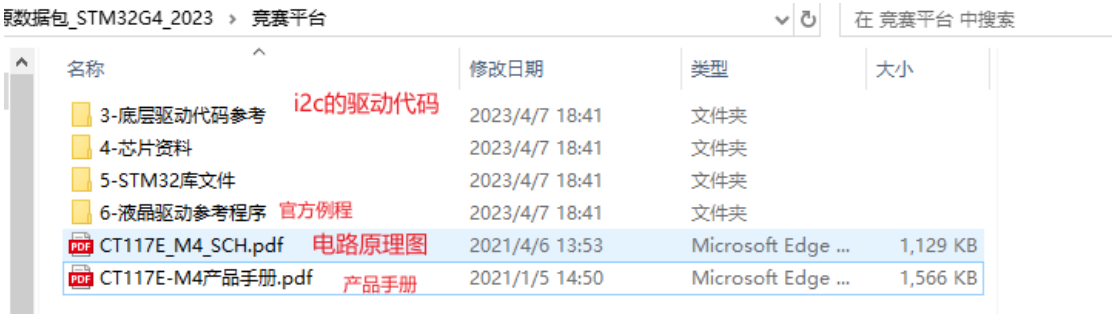
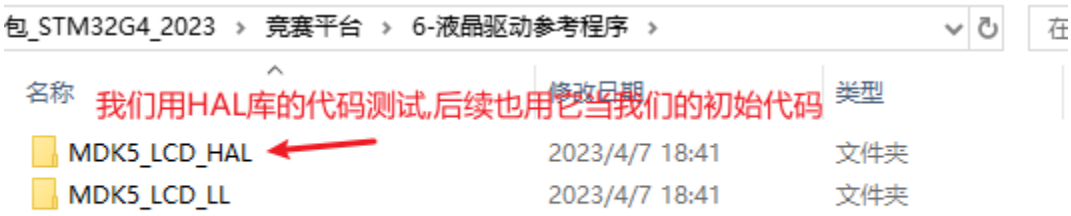
目录

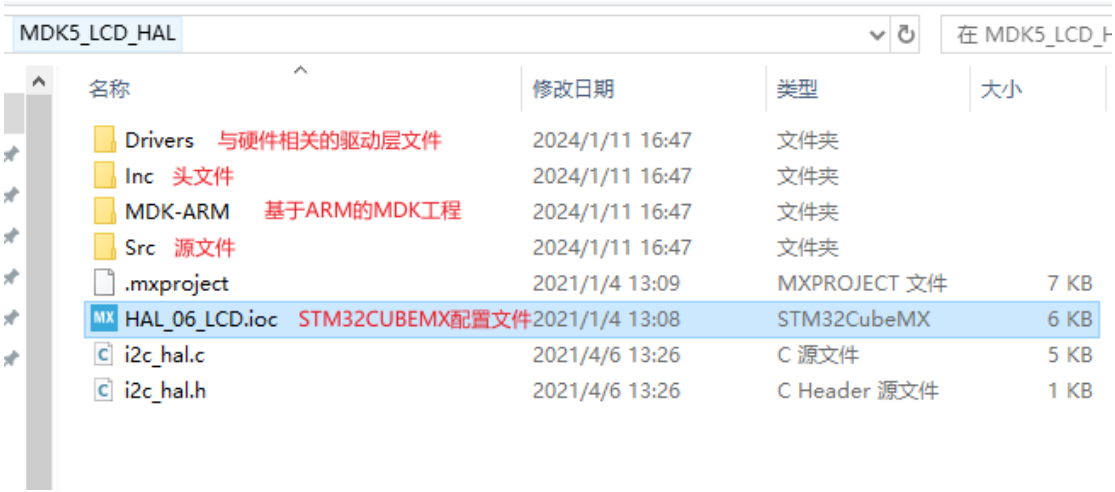
初始化以及下载程序

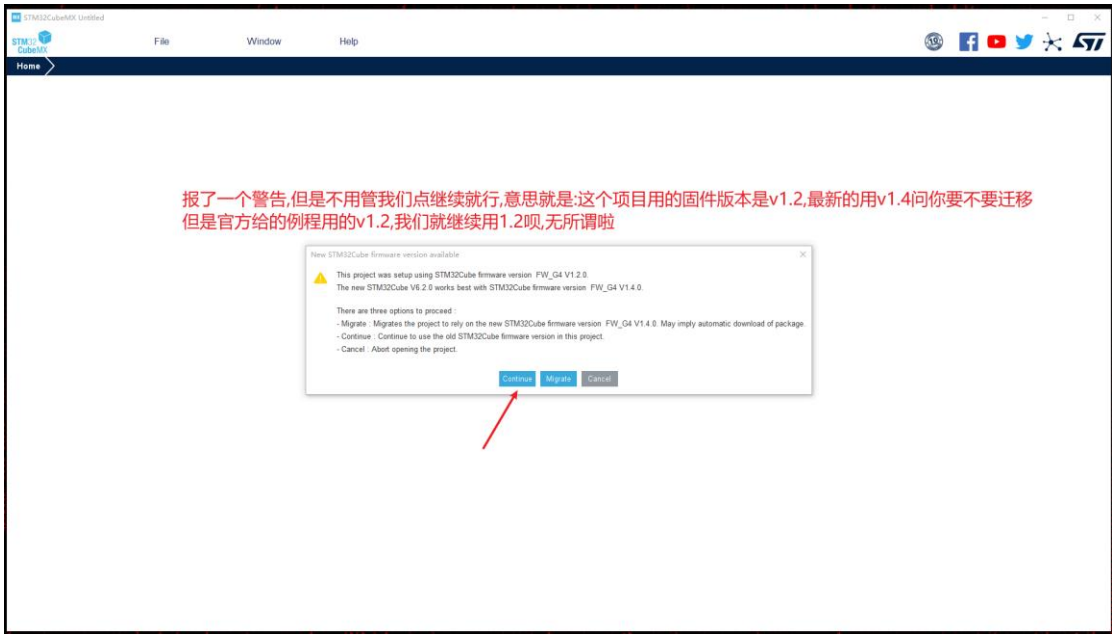


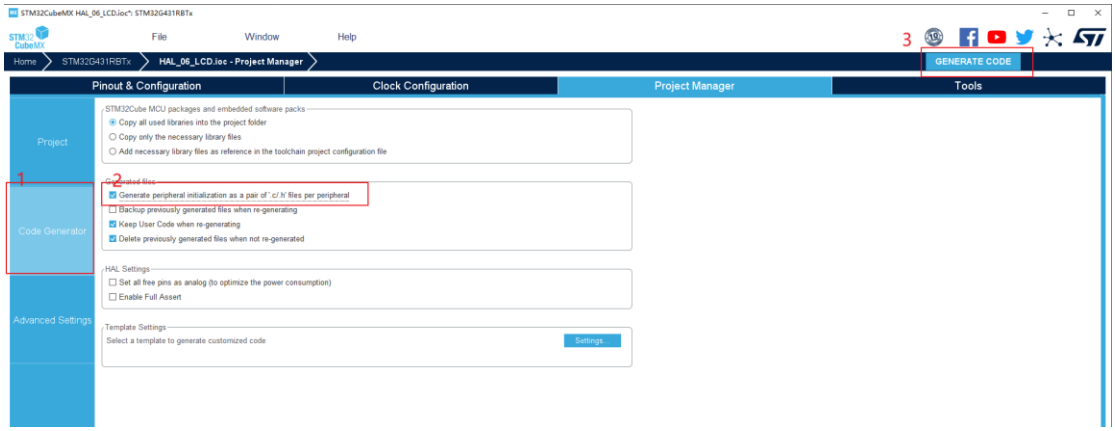
竞赛平台:

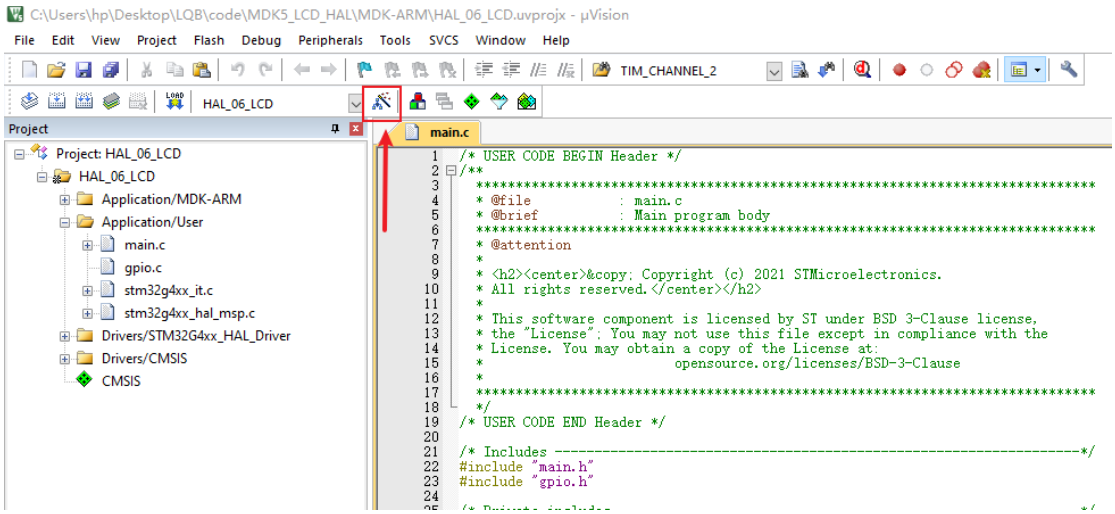


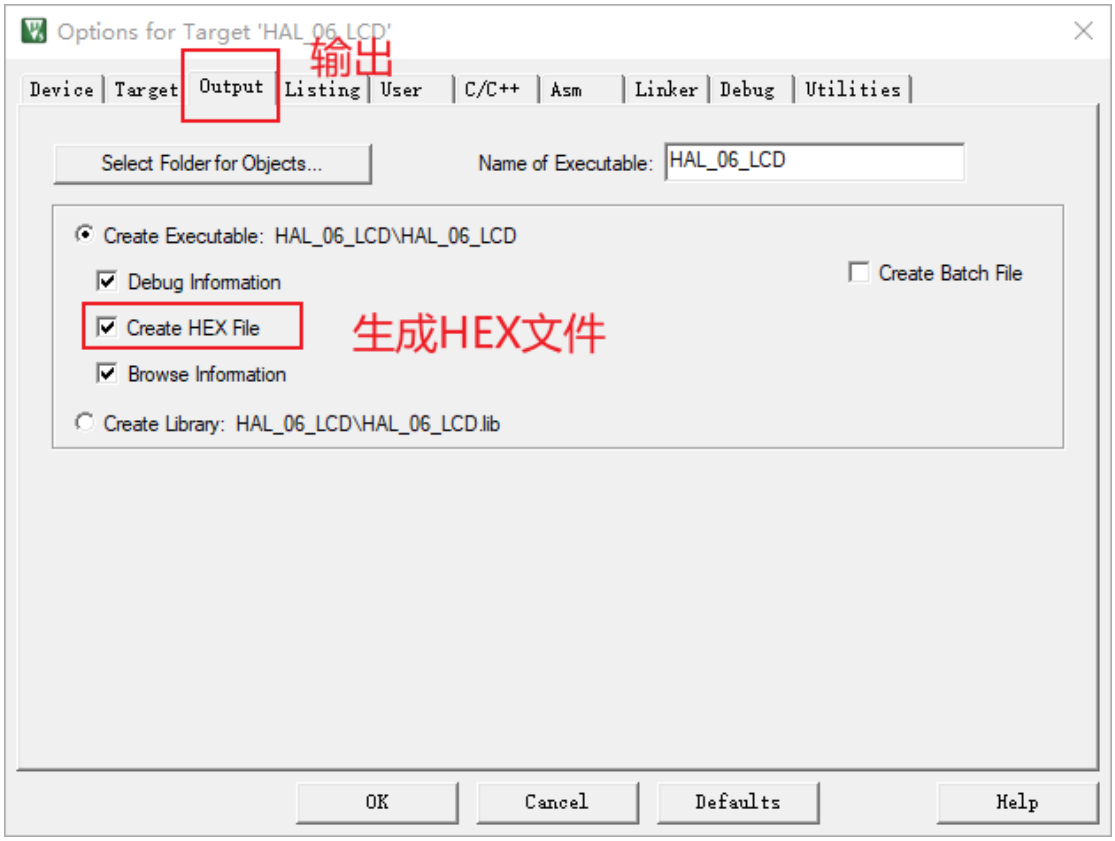


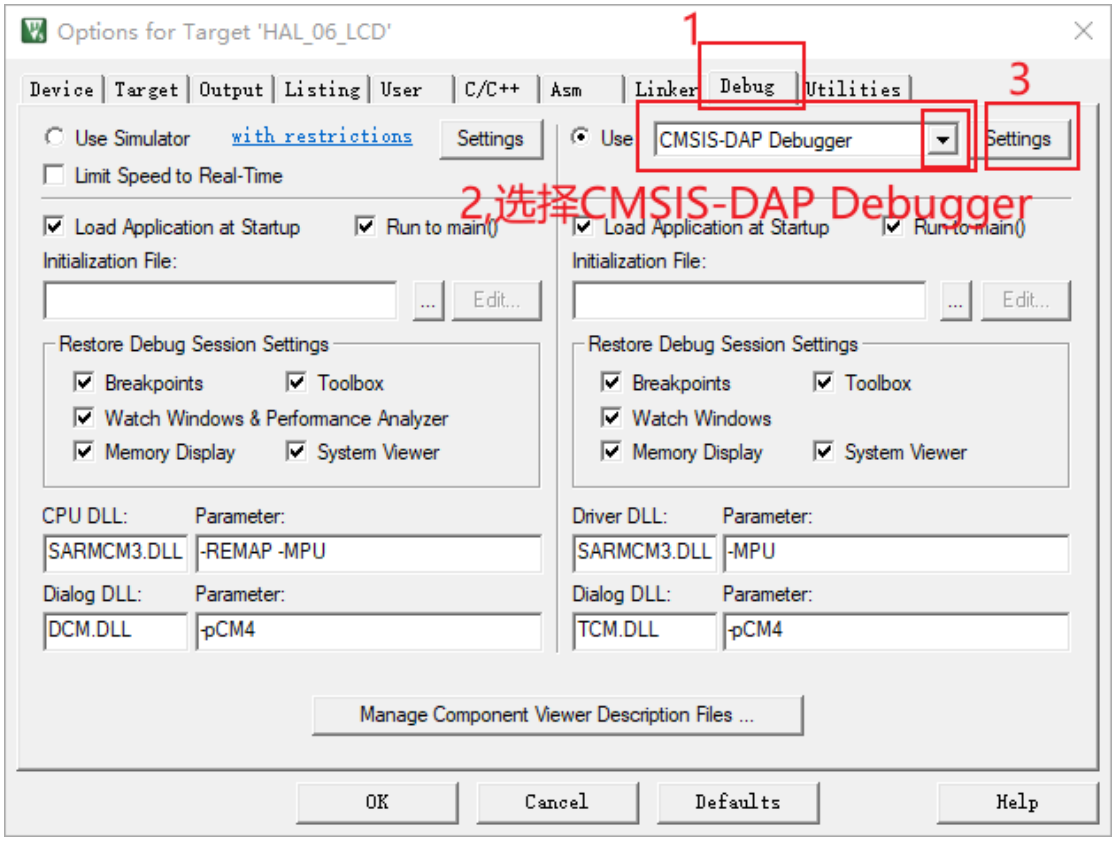


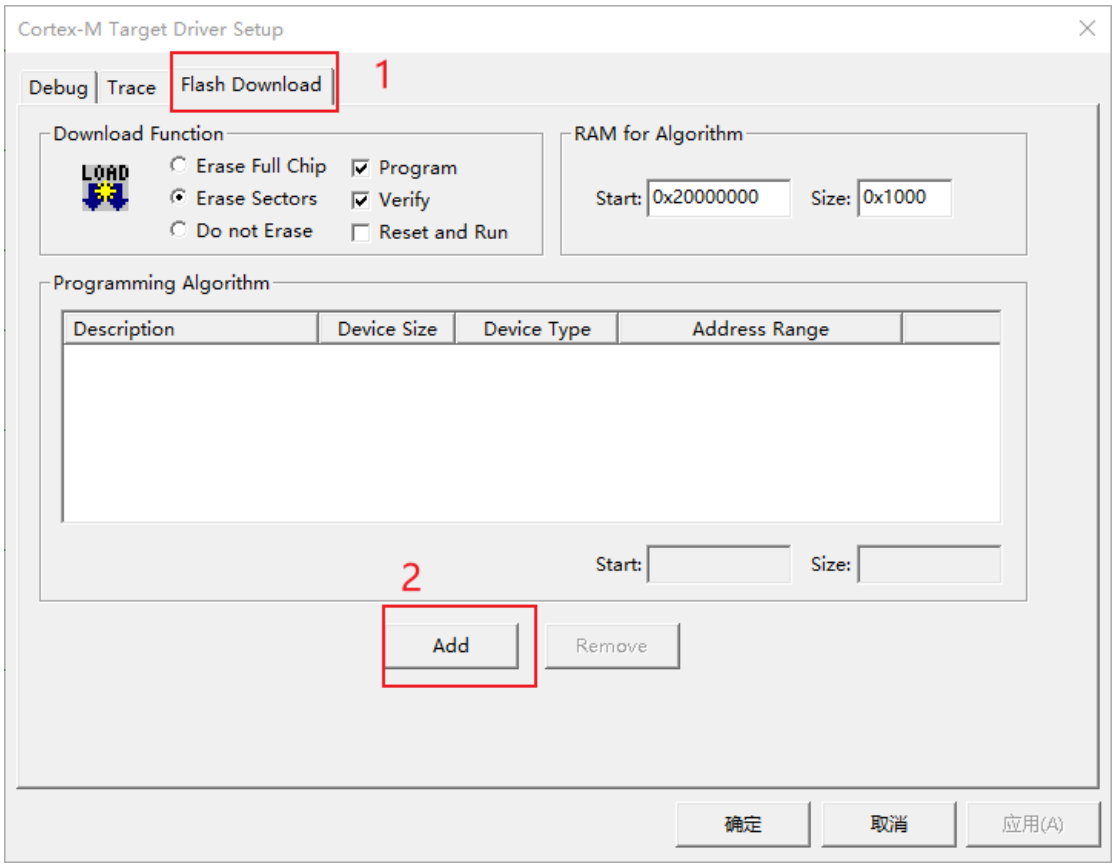

报了一个警告,但是不用管我们点继续就行,意思就是:这个项目用的固件版本是v1.2,最新的用v1.4问你要不要迁移
但是官方给的例程用的v1.2,我们就继续用1.2呗,无所谓啦
Continue
Migrate
Cancel

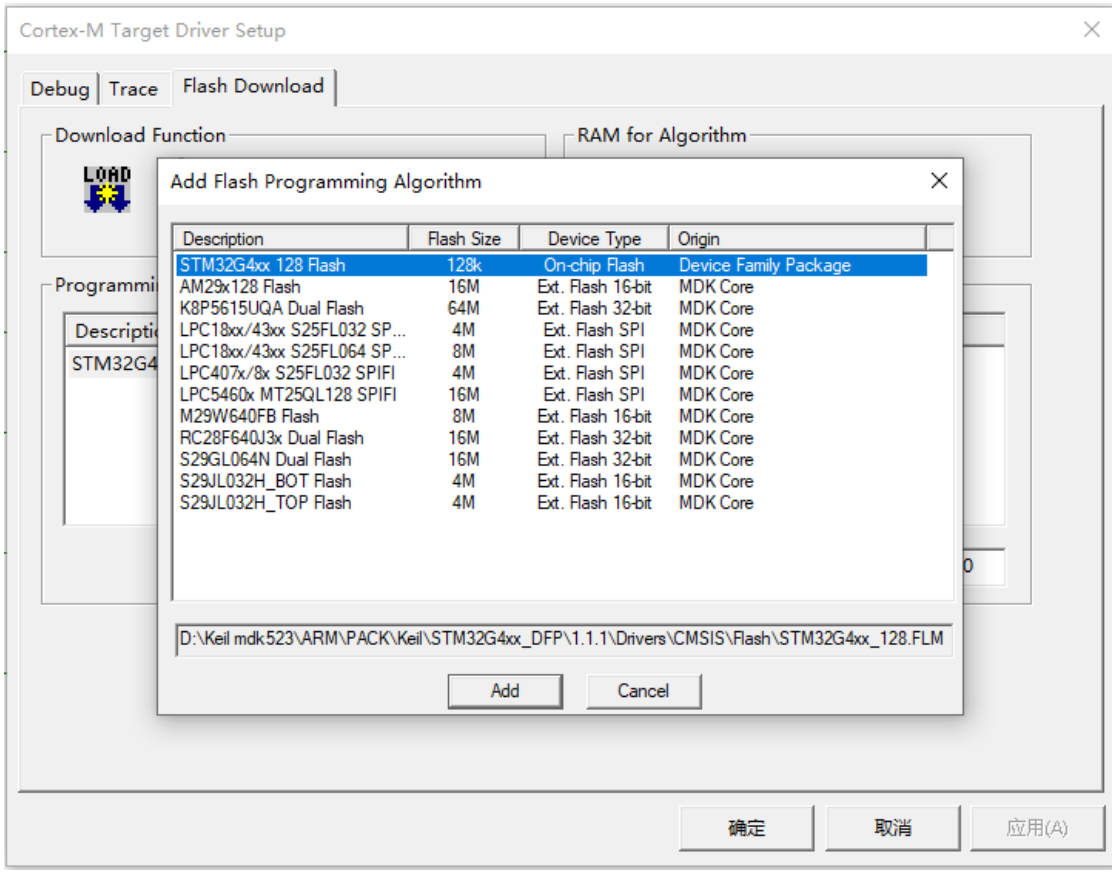

Pinout & Configuration
Clock Configuration
Project Manager
Tools
GENERATE CODE
Project
Code Generator
Advanced Settings
1
2
3

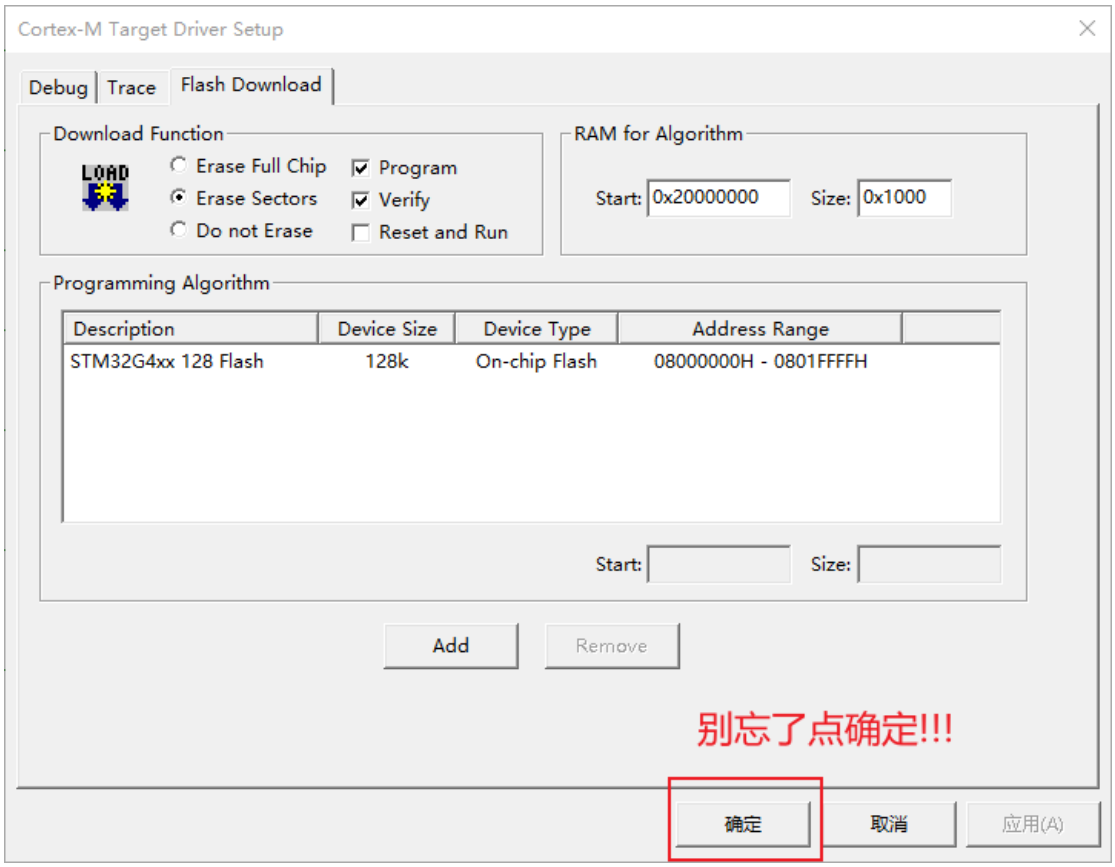

C:\Users\hp\Desktop\LQB\code\MDK5_LCD_HAL\MDK-ARM\HAL_06_LCD.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
TIM_CHANNEL_2
HAL_06_LCD
Project
Project: HAL_06_LCD
HAL_06_LCD
Application/MDK-ARM
Application/User
main.c
gpio.c
stm32g4xx_it.c
stm32g4xx_hal_msp.c
Drivers/STM32G4xx_HAL_Driver
Drivers/CMSIS
CMSIS
main.c
/* USER CODE BEGIN Header */
/**
* @file : main.c
* @brief : Main program body
* @attention
* <h2> © Copyright (c) 2021 STMicroelectronics.
* All rights reserved. </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 "gpio.h"

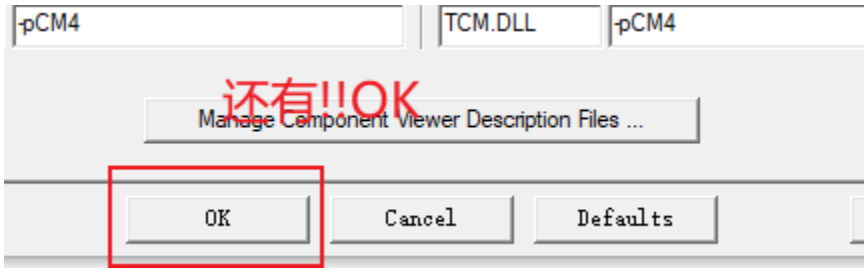

Options for Target 'HAL_06_LCD'
输出
Device
Target
Output
Listing
User
C/C++
Asm
Linker
Debug
Utilities
Select Folder for Objects...
Name of Executable: HAL_06_LCD
Create Executable: HAL_06_LCD\HAL_06_LCD
Debug Information
Create Batch File
Create HEX File
生成HEX文件
Browse Information
Create Library: HAL_06_LCD\HAL_06_LCD.lib
OK
Cancel
Defaults
Help


Options for Target 'HAL_06_LCD'
1
3
Device
Target
Output
Listing
User
C/C++
Asm
Linker
Debug
Utilities
Use Simulator
with restrictions
Settings
Limit Speed to Real-Time
Use
CMSIS-DAP Debugger
Settings
2,选择CMSIS-DAP Debugger
Load Application at Startup
Run to main()
Initialization File:
Edit...
Restore Debug Session Settings
Breakpoints
Toolbox
Watch Windows & Performance Analyzer
Watch Windows
Memory Display
System Viewer
CPU DLL:
Parameter:
SARMCM3.DLL
-REMAP -MPU
Dialog DLL:
DCM.DLL
-pCM4
Driver DLL:
-MPU
TCM.DLL
Manage Component Viewer Description Files ...
OK
Cancel
Defaults
Help


Cortex-M Target Driver Setup
Debug
Trace
Flash Download
1
Download Function
Erase Full Chip
Erase Sectors
Do not Erase
Program
Verify
Reset and Run
RAM for Algorithm
Start: 0x20000000
Size: 0x1000
Programming Algorithm
Description
Device Size
Device Type
Address Range
Start:
Size:
2
Add
Remove
确定
取消
应用(A)


Cortex-M Target Driver Setup
Debug
Trace
Flash Download
Download Function
RAM for Algorithm
Add Flash Programming Algorithm
Description
Flash Size
Device Type
Origin
STM32G4xx 128 Flash 128k On-chip Flash Device Family Package
AM29x128 Flash 16M Ext. Flash 16-bit MDK Core
K8P5615UQA Dual Flash 64M Ext. Flash 32-bit MDK Core
LPC18xx/43xx S25FL032 SP... 4M Ext. Flash SPI MDK Core
LPC18xx/43xx S25FL064 SP... 8M Ext. Flash SPI MDK Core
LPC407x/8x S25FL032 SPIFI 4M Ext. Flash SPI MDK Core
LPC5460x MT25QL128 SPIFI 16M Ext. Flash SPI MDK Core
M29W640FB Flash 8M Ext. Flash 16-bit MDK Core
RC28F640J3x Dual Flash 16M Ext. Flash 32-bit MDK Core
S29GL064N Dual Flash 16M Ext. Flash 32-bit MDK Core
S29JL032H_BOT Flash 4M Ext. Flash 16-bit MDK Core
S29JL032H_TOP Flash 4M Ext. Flash 16-bit MDK Core
D:\Keil mdk523\ARM\PACK\Keil\STM32G4xx_DFP\1.1.1\Drivers\CMSIS\Flash\STM32G4xx_128.FLM
Add
Cancel
确定
取消
应用(A)


Cortex-M Target Driver Setup
Debug
Trace
Flash Download
Download Function
Erase Full Chip
Erase Sectors
Do not Erase
Program
Verify
Reset and Run
RAM for Algorithm
Start: 0x20000000
Size: 0x1000
Programming Algorithm
Description
Device Size
Device Type
Address Range
STM32G4xx 128 Flash
128k
On-chip Flash
08000000H - 0801FFFFH
Start:
Size:
Add
Remove
别忘了点确定!!!
确定
取消
应用(A)

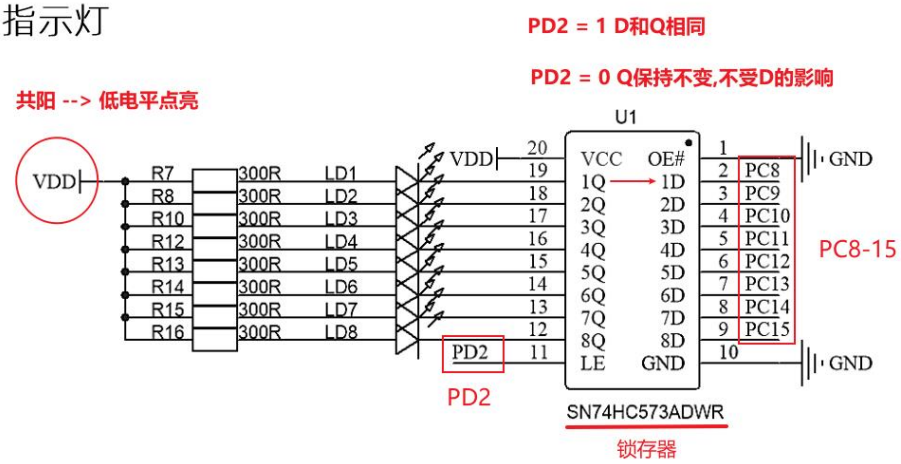

-pCM4
TCM.DLL
-pCM4
还有!!OK
Manage Component Viewer Description Files ...
OK
Cancel
Defaults

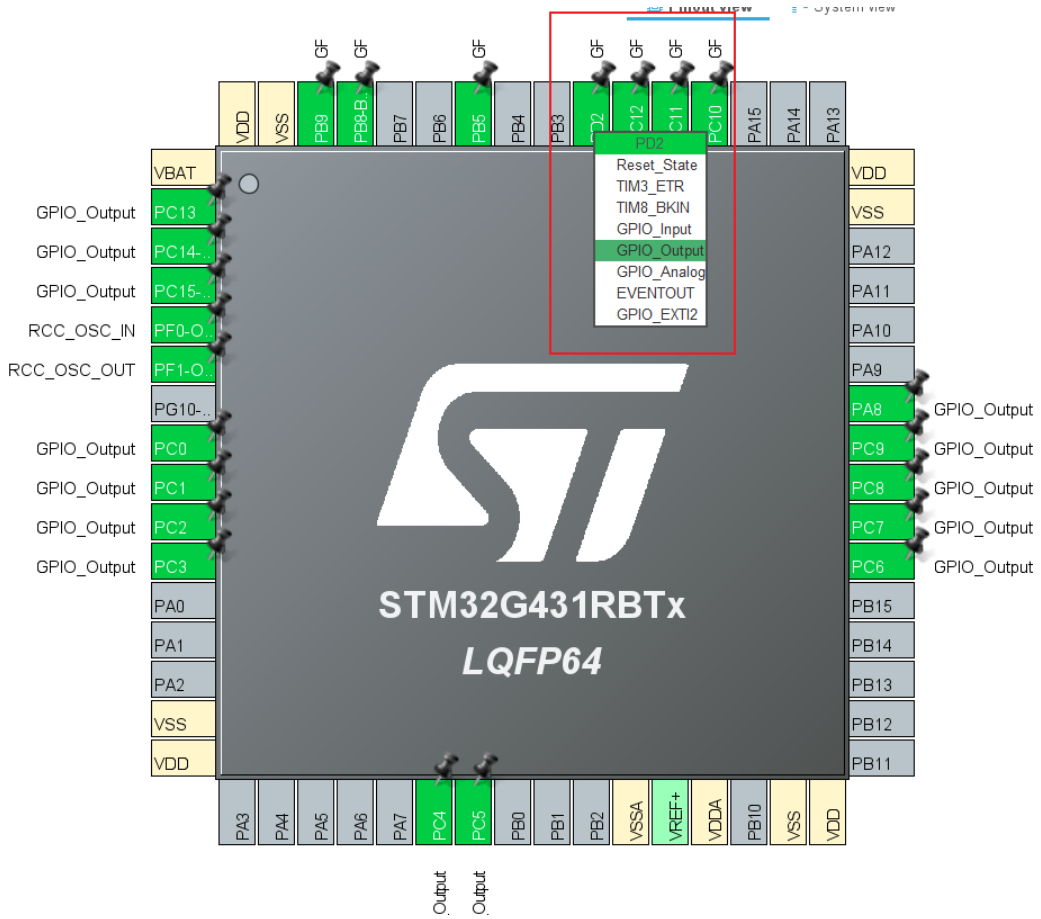

C:\Users\hp\Desktop\LQB\code\MDK5_LCD_HAL\MDK-ARM\HAL_06_LCD.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
1Build
2Load
TIM_CHANNEL_2
HAL_06_LCD
Project

LED 指示灯

- **STM32CUBEMX PC8~PC15,PD2 → GPIO_OUTPUT(推挽输出)**
- **写 led.c 和 led.h，并且添加进工程**
- **LedProcess()**

7.4 LED 指示灯





main.c led.c led.h

```c
#include "led.h"

void LED_Control(u8 led_Ctrl) {
  HAL_GPIO_WritePin(GPIOC, 0XFF00, 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, led_Ctrl << 8, GPIO_PIN_RESET);
  HAL_GPIO_WritePin(GPIOD, GPIO_PIN_2, GPIO_PIN_SET);
  HAL_GPIO_WritePin(GPIOD, GPIO_PIN_2, GPIO_PIN_RESET);
}
```

led.c

main.c led.c led.h

```c
#ifndef __LED_H
#define __LED_H

#include "main.h"

void LED_Control(u8 led_Ctrl);

#endif
```

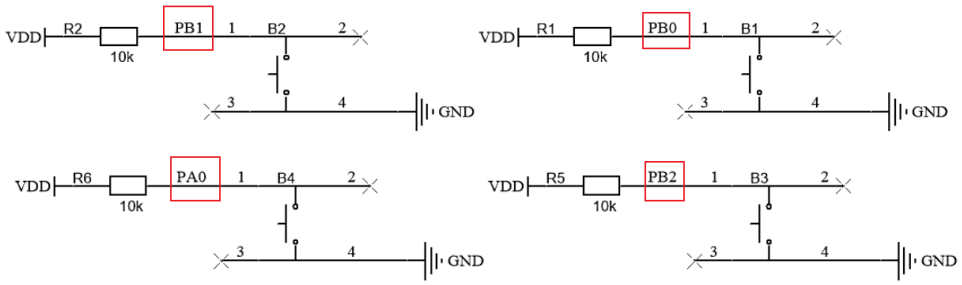
led.h

KEY 按键

- STM32CUBEMX PA0,PB0,PB1,PB2 → GPIO_INPUT[输入]
- 写 key.c 和 key.h，并且添加进工程
- KeyProcess()

7.7 按键

PB1 PB0 PB2 PA0



key.c

```c
#include "key.h"

#define KB1 HAL_GPIO_ReadPin(GPIOB,GPIO_PIN_0)
#define KB2 HAL_GPIO_ReadPin(GPIOB,GPIO_PIN_1)
#define KB3 HAL_GPIO_ReadPin(GPIOB,GPIO_PIN_2)
#define KB4 HAL_GPIO_ReadPin(GPIOA,GPIO_PIN_0)

#define KEYPORT KB1 | (KB2<<1) | (KB3<<2) | (KB4<<3)

u8 Trg;
u8 Cont;

void KeyRead(void) {
  u8 ReadData = (KEYPORT)^0xff;
  Trg = ReadData & (ReadData ^ Cont);
  Cont = ReadData;
}
```

key.h

```c
#ifndef __KEY_H
#define __KEY_H

#include "main.h"

extern u8 Trg;
extern u8 Cont;

void KeyRead(void);


#endif
```

KeyProcess

```c
// 按键执行程序
__IO uint32_t keyTick = 0;
void KeyProcess(void)
{
  if(uwTick - keyTick < 10) return ;
  keyTick = uwTick;

  KeyRead();
  if(Trg & 0x01)  //B1
  {
  }
  if(Trg & 0x02)  //B2
  {
  }
  if(Trg & 0x04)  //B3
  {
  }
  if(Trg & 0x08)  //B4
  {
  }
}
```

LCD 显示

因为官方给了我们 LCD 的例程,所以很 easy,我们照着套就行

- **LcdProcess()**

官方例程

```
LCD_Init();lcd初始化
/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */

LCD_Clear(Blue); 用蓝色清屏
LCD_SetBackColor(Blue);设置背景颜色
LCD_SetTextColor(White); 设置文字的颜色


LCD_DrawLine(120, 0, 320, Horizontal); 画水平线
LCD_DrawLine(0, 160, 240, Vertical); 画垂直线
HAL_Delay(1000);
LCD_Clear(Blue);

LCD_DrawRect(70, 210, 100, 100); 画矩形   <-- 画形状暂时还没考过
HAL_Delay(1000);
LCD_Clear(Blue);

LCD_DrawCircle(120, 160, 50); 画圆
HAL_Delay(1000);

LCD_Clear(Blue);                    考的较多        在屏幕上写字
LCD_DisplayStringLine(Line4, (unsigned char *)"     Hello,world.     ");
HAL_Delay(1000);

LCD_SetBackColor(White);
LCD_DisplayStringLine(Line0, (unsigned char *)"                      ");
LCD_SetBackColor(Black);
LCD_DisplayStringLine(Line1, (unsigned char *)"                      ");
LCD_SetBackColor(Grey);
LCD_DisplayStringLine(Line2, (unsigned char *)"                      ");
LCD_SetBackColor(Blue);
LCD_DisplayStringLine(Line3, (unsigned char *)"                      ");
LCD_SetBackColor(Blue2);
LCD_DisplayStringLine(Line4, (unsigned char *)"                      ");
LCD_SetBackColor(Red);
LCD_DisplayStringLine(Line5, (unsigned char *)"                      ");
LCD_SetBackColor(Magenta);
LCD_DisplayStringLine(Line6, (unsigned char *)"                      ");
LCD_SetBackColor(Green);   <-- 设置这行的背景颜色,之前考高亮的时候考到过
LCD_DisplayStringLine(Line7, (unsigned char *)"                      ");
LCD_SetBackColor(Cyan);
LCD_DisplayStringLine(Line8, (unsigned char *)"                      ");
LCD_SetBackColor(Yellow);
LCD_DisplayStringLine(Line9, (unsigned char *)"                      ");
```

```
void LcdProcess(void)
{
  u8 lcdbuf[20];
  LCD_DisplayStringLine(Line2, (unsigned char *)"     Hello,world.     ");
  sprintf((char*)lcdbuf,"      KEY: %d",10);
  LCD_DisplayStringLine(Line5,lcdbuf);
  sprintf((char*)lcdbuf,"      value: %3.1f",10.1);
  LCD_DisplayStringLine(Line7,lcdbuf);
}
```

用到这个函数记得添加头文件 "stdio.h"

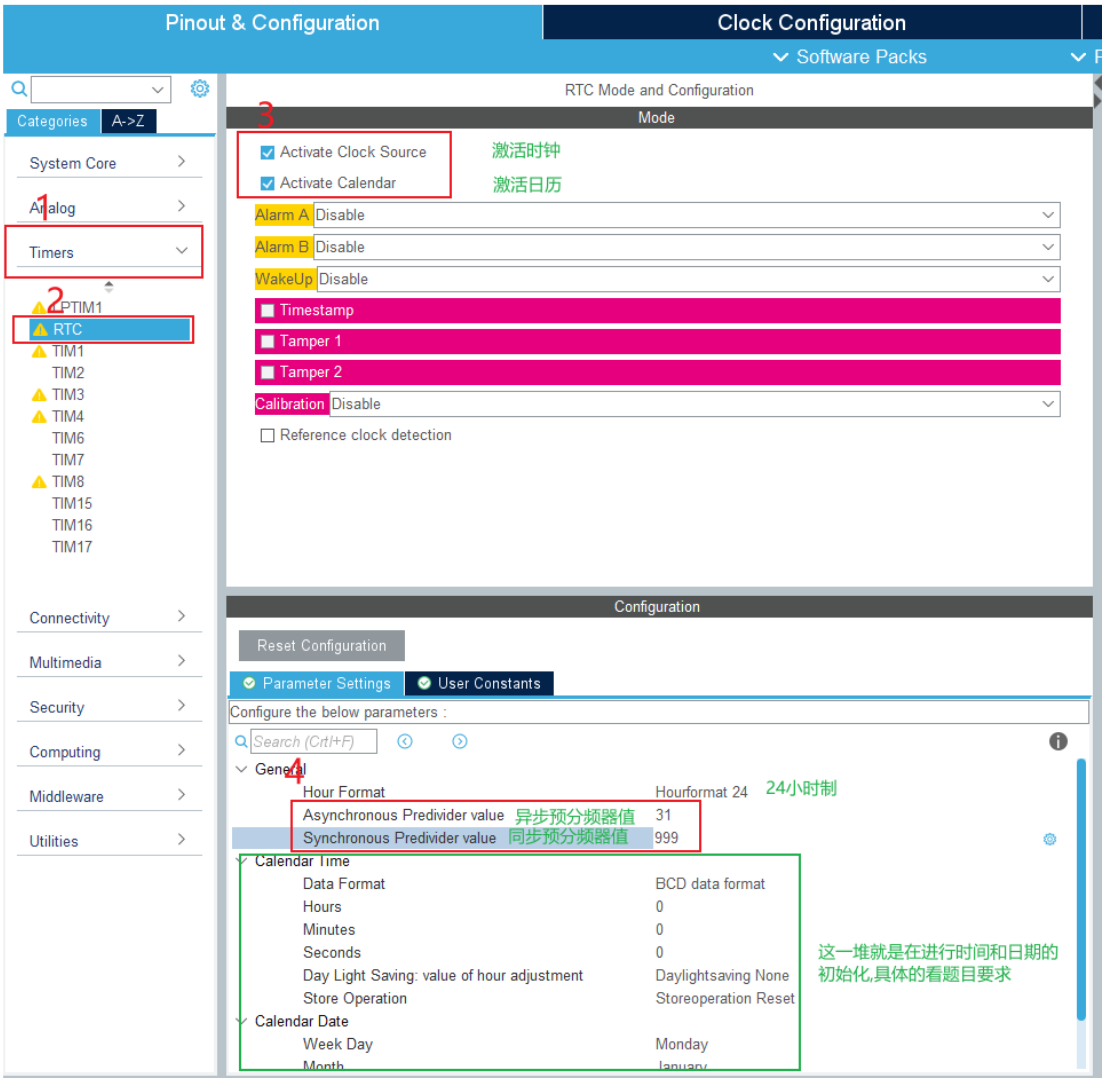
LcdProcess

RTC 时钟和日期

- STM32CUBEMX Timer→RTC → Activate Clock Source ; Activate Calendar

  →Asynchronous Predivider value=31 ; S ynchronous Predivider value=999

- **RtcProcess()**




```
//RTC
RTC_TimeTypeDef rtc_time;
RTC_DateTypeDef rtc_date;
void RtcProcess()
{
  HAL_RTC_GetTime(&hrtc, &rtc_time, RTC_FORMAT_BIN);
  HAL_RTC_GetDate(&hrtc, &rtc_date, RTC_FORMAT_BIN);
}
```

RtcProcess

EEPROM 电可擦可编程只读存储器

- 移植添加官方给的"i2c-hal,c"和"i2c-hal.h"文件 在竞赛平台→3-底层驱动代码参考→I2C_HAL
- EEPROMRead EEPROMWrite
- 在 main()中初始化 I2CInit();及按照比赛要求调用上面写的函数

```
//读24C02
u8 EEPROMRead(u8 address)
{
  u8 dat;
  I2CStart();
  I2CSendByte(0xa0);
  I2CWaitAck();
  I2CSendByte(address);
  I2CWaitAck();

  I2CStart();
  I2CSendByte(0xa1);
  I2CWaitAck();
  dat = I2CReceiveByte();
  I2CSendNotAck();
  I2CStop();

  return(dat);
}
```

EEPROMRead

```
//写24C02
void EEPROMWrite(u8 address,u8 data)
{
  I2CStart();
  I2CSendByte(0xa0);
  I2CWaitAck();

  I2CSendByte(address);
  I2CWaitAck();
  I2CSendByte(data);
  I2CWaitAck();
  I2CStop();
  HAL_Delay(5);
}
```

EEPROMWrite

MCP4017 数字电位器

- 移植添加官方给的"i2c-hal,c"和"i2c-hal.h"文件 在竞赛平台→3-底层驱动代码参考→I2C_HAL
- MCP4017Read MCP4017Write
- 在 main()中初始化 I2CInit();及按照比赛要求调用上面写的函数

```
//读MCP4017
u8 MCP4017Read(void)
{
  u8 value;
  I2CStart();
  I2CSendByte(0x5F);
  I2CWaitAck();

  value = I2CReceiveByte();
  I2CSendNotAck();
  I2CStop();

  return value;
}
```

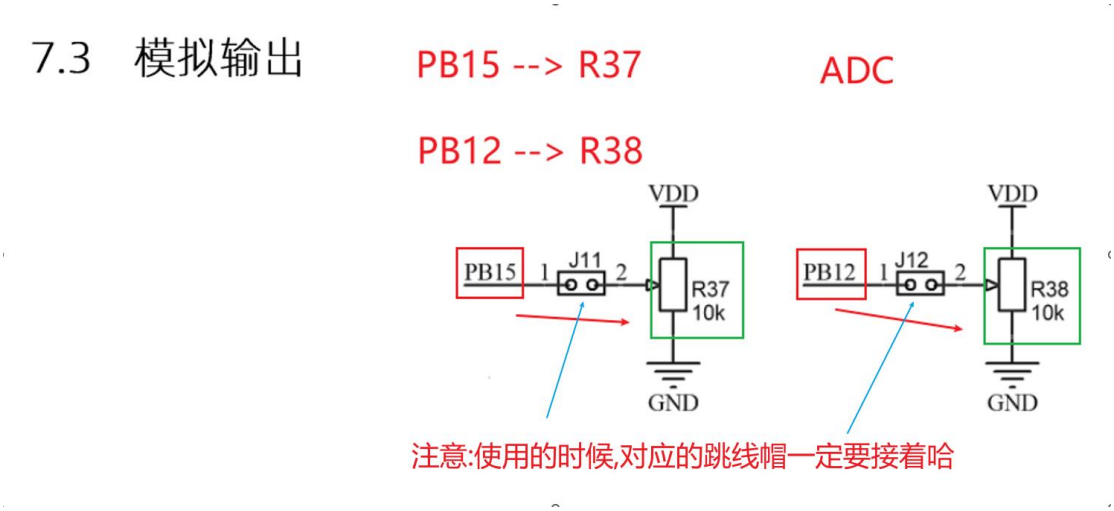
MCP4017Read

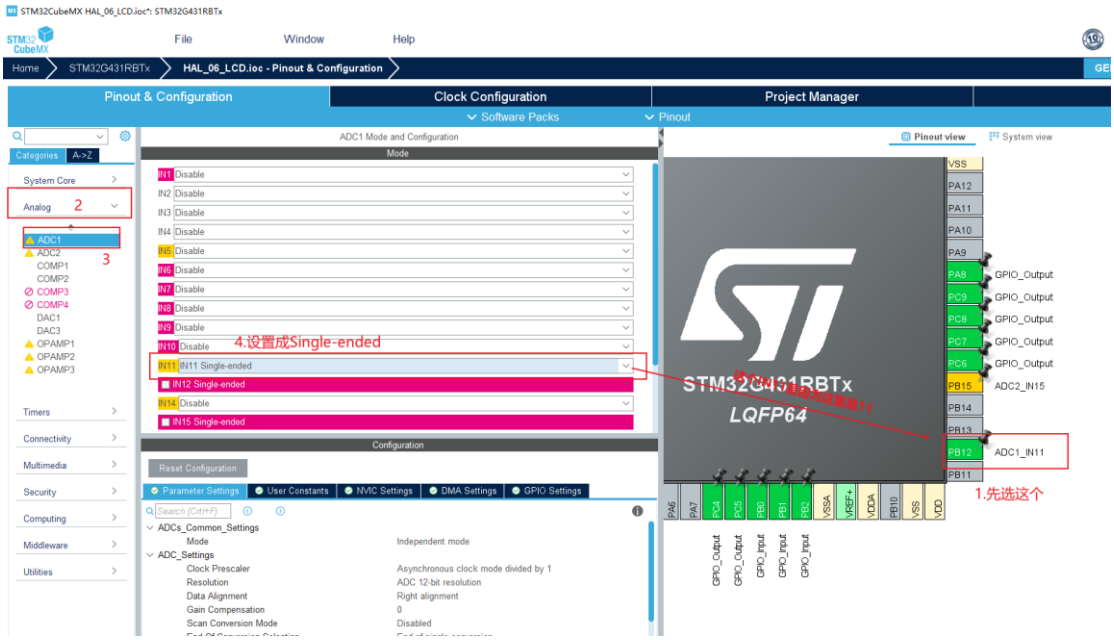
```
//写MCP4017
void MCP4017Write(u8 value)
{
  I2CStart();
  I2CSendByte(0x5E);
  I2CWaitAck();

  I2CSendByte(value);
  I2CWaitAck();
  I2CStop();
}
```

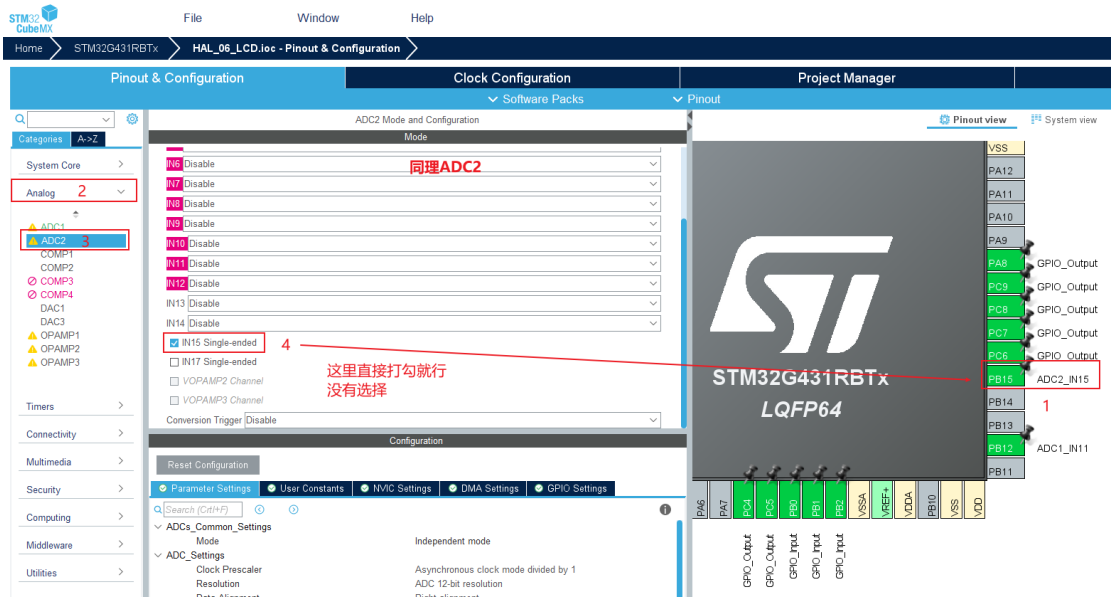
MCP4017Write

ADC 模数转换器

- **STM32CUBEMX PB15 →ADC2_IN15 ; PB12→ADC1_IN11**
- **ADCProcess**





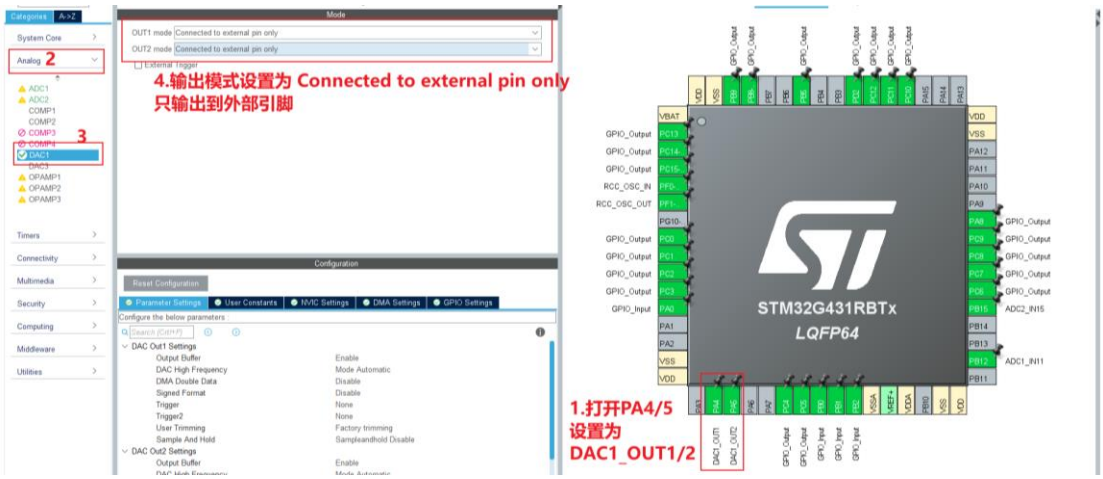



```
u32 adc1_val,adc2_val;
float volt_r37,volt_r38;
void ADCProcess(void)
{
  HAL_ADC_Start(&hadc1);
  adc1_val = HAL_ADC_GetValue(&hadc1);
  volt_r38= adc1_val/4095.0f*3.3f;
  HAL_ADC_Start(&hadc2);
  adc2_val = HAL_ADC_GetValue(&hadc2);
  volt_r37 = adc2_val/4095.0f*3.3f;

}
```

ADCProcess

DAC 数模转换器

- **STM32CUBEMX PA4 →DAC1_OUT1 ; PA5→ DAC1_OUT2**
- **DACProcess**



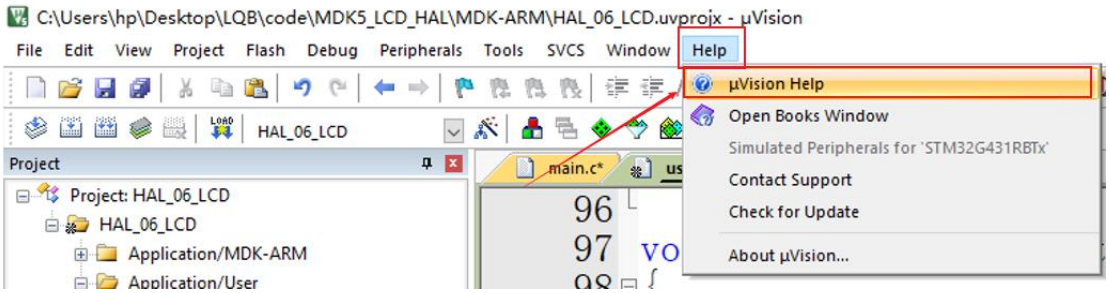

```
//0-3.3v    -->    0--4095
float dac_volt1 = 2.7;
float dac_volt2 = 1.0;
u16 dac_val1,dac_val2;
void DACProcess()
{
  dac_val1 = (dac_volt1/3.3f*4095);
  dac_val2 = (dac_volt1/3.3f*4095);

  HAL_DAC_SetValue(&hdac1, DAC_CHANNEL_1, DAC_ALIGN_12B_R, dac_val1);
  HAL_DAC_Start(&hdac1, DAC_CHANNEL_1);    //设完了值不要忘了Start

  HAL_DAC_SetValue(&hdac1, DAC_CHANNEL_2, DAC_ALIGN_12B_R, dac_val2);
  HAL_DAC_Start(&hdac1, DAC_CHANNEL_2);    //设完了值不要忘了Start
}
```

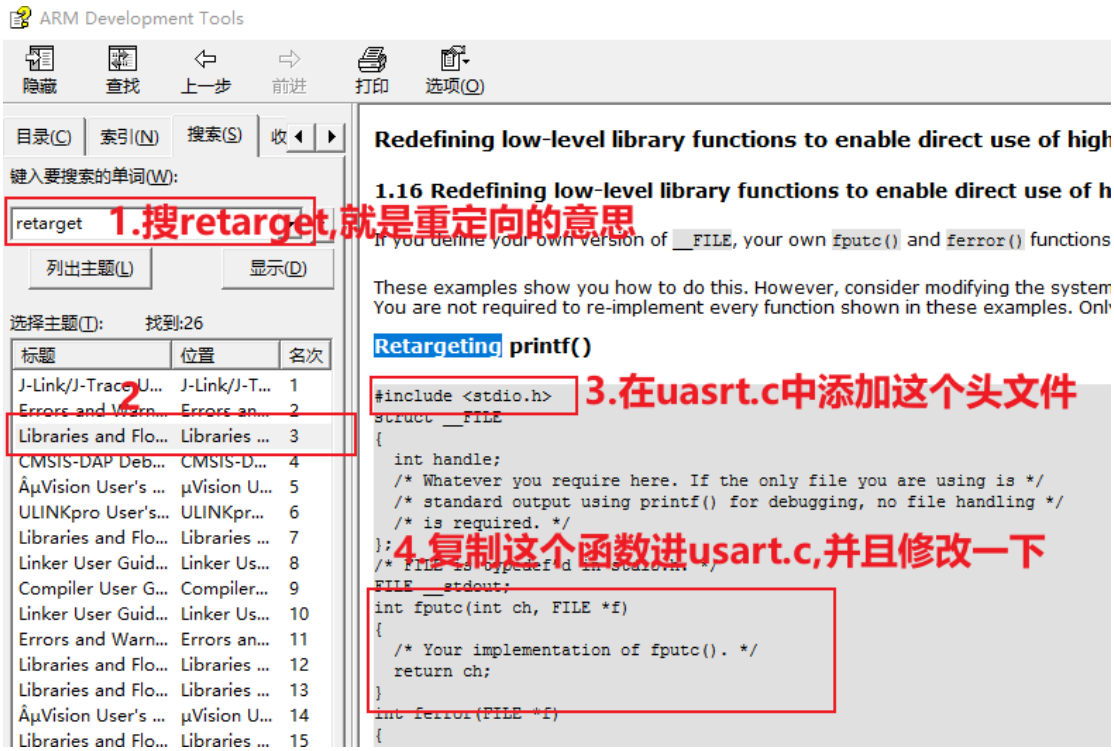
DACProcess

USART 串行通信

- **STM32CUBEMX PA9 →USART1_TX ; PA10→ USART1_RX,修改波特率,开启中断**
- **串口发送:在 usart.c 中将 printf 重定向,在 mian.c 中直接用 `printf` 发送**
- **串口接收: 初始化别忘了开启串口接收中断 HAL_UART_Receive_IT(&huart1,uart_buf,1);**
- **串口接收: 写一个串口接收回调函数 HAL_UART_RxCpltCallback**
-

串口发送:在 usart.c 中将 printf 重定向





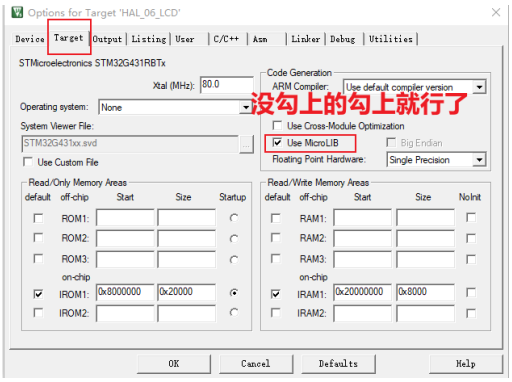

```c
/* Includes ------------------------------------------------------------------*/
#include "usart.h"

/* USER CODE BEGIN 0 */
#include <stdio.h>
/* USER CODE END 0 */

UART_HandleTypeDef huart1;
```

```c
/* USER CODE BEGIN 1 */

int fputc(int ch, FILE *f)
{
  HAL_UART_Transmit(&huart1, (unsigned char*)&ch, 1, 50);
  return ch;
}
```

fputc

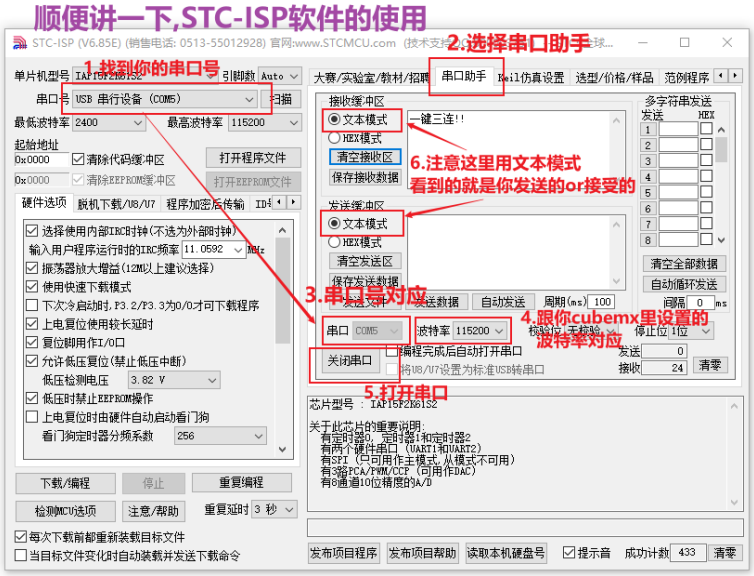
添加好之后我们就可以在 `main.c` 文件中直接写 `printf` 来进行 usart 的发送了

此外,提示,如果不能使用,可以查看是不是微库没有勾选上



```
/* Infinite loop */
/* USER CODE BEGIN WHILE */

  LCD_Clear(Blue);
  LCD_SetBackColor(Blue);
  LCD_SetTextColor(White);
  LED_Control(0x00);
  keyInum = EEPROMRead(0x20);
  //MCP4017
  MCP4017Write(0x08);
  val_mcp = MCP4017Read();

  printf("一键三连!!")
  直接像printf一样使用就可以了
  while (1)
  {
  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
    KeyProcess();
    LcdProcess();
    RtcProcess();
    LcdProcess();
    ADCProcess();
    DACProcess();
  }
/* USER CODE END 3 */
```

顺便讲一下,STC-ISP软件的使用

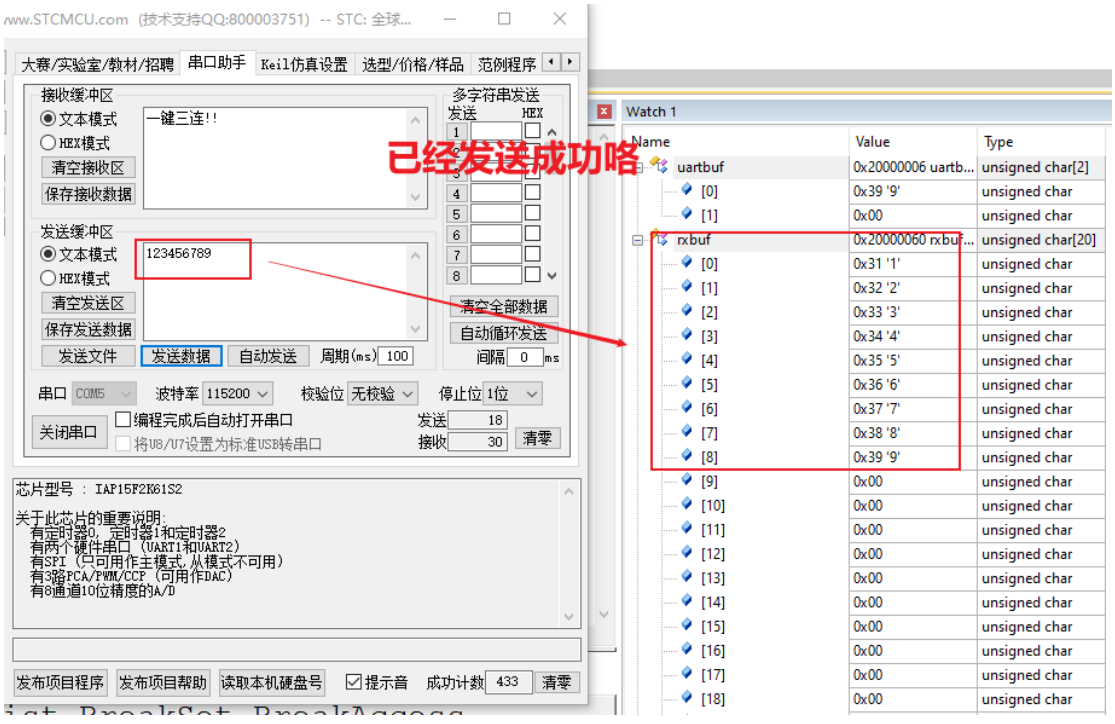


串口接收回调函数:

```
/* USER CODE BEGIN 4 */
//串口接收回调函数
int n = 0;
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{
  rxbuf[n++] = uartbuf[0];
  HAL_UART_Receive_IT(&huart1,uartbuf,1);
}
/* USER CODE END 4 */
```

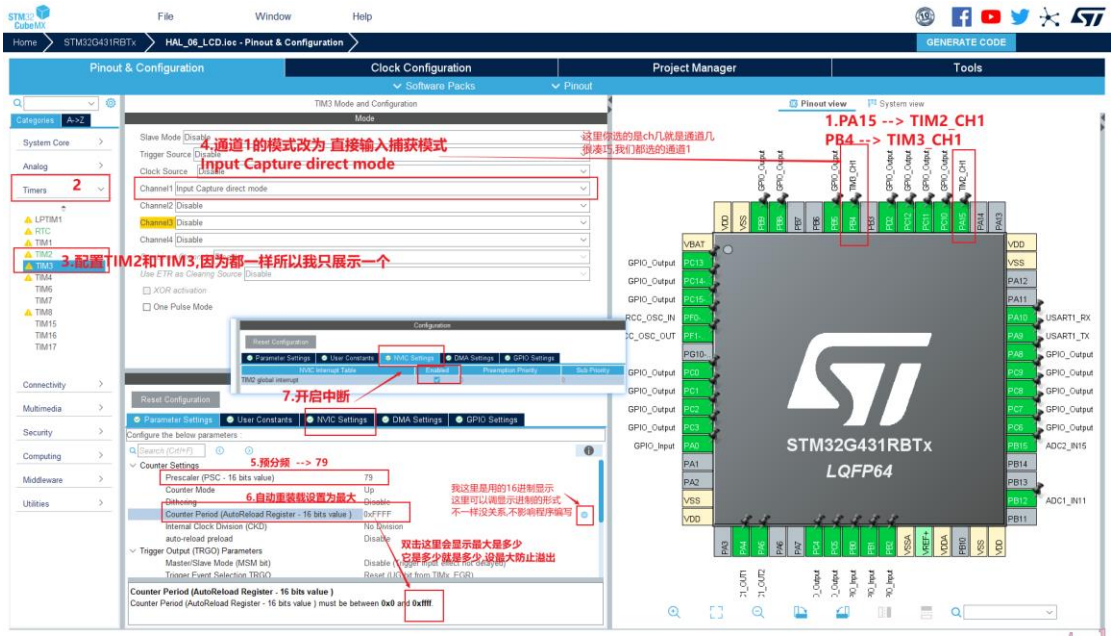
HAL_UART_RxCpltCallback

因为要main里初始化,所以这俩变量定义在前面
u8 uartbuf[2]; u8 rxbuf[20];



PWM 捕获

- STM32CUBEMX PA15 →TIM2_CH1; PB4→ TIM3_CH1 预分频设置为 79,自动重装载设置最大,开启中断
- PWM 捕获: HAL_TIM_IC_CatureCallback,
- PWM 捕获: 对了别忘了初始化
  HAL_TIM_IC_Start_IT(&htim2,TIM_CHANNEL_1);
  HAL_TIM_IC_Start_IT(&htim3,TIM_CHANNEL_1);

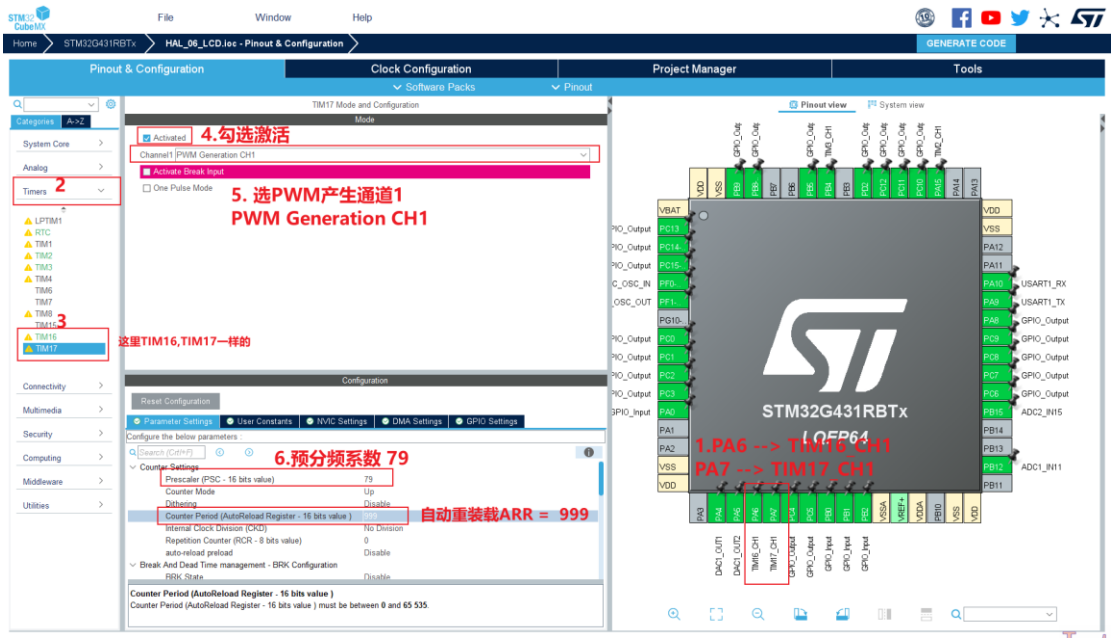


PWM 捕获:

```
//PWM捕获                                    HAL_TIM_IC_CatureCallback
u32 num2 = 0, num3 = 0;
u32 f40 = 0, f39 = 0;
void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim)
{
  if(htim == &htim2)
  {
    num2 = __HAL_TIM_GetCounter(&htim2);
    __HAL_TIM_SetCounter(&htim2,0);
    f40 = 1000000/num2;
    HAL_TIM_IC_Start_IT(&htim2,TIM_CHANNEL_1);
  }

  if(htim == &htim3)
  {
    num3 = __HAL_TIM_GetCounter(&htim3);
    __HAL_TIM_SetCounter(&htim3,0);
    f39 = 1000000/num3;
    HAL_TIM_IC_Start_IT(&htim3,TIM_CHANNEL_1);
  }
}
```

PWM 输出

- STM32CUBEMX PA6 →TIM16_CH1; PA7→ TIM17_CH1 预分频设置为 79,自动重装载设置 999
- PWM 输出就是设置一下对应的 ARR CCR1 寄存器,也可以直接写成 PWMOutputProcess 函数



PWMOutputProcess

```
//PWM输出
void PWMOutputProcess() {
    HAL_TIM_PWM_Start(&htim16,TIM_CHANNEL_1);
    TIM16->ARR = 49;  // 周期 = 50us，频率 = 20kHz    ARR = T-1   f = 1/T
    TIM16->CCR1 = 10; // 占空比 = 20% 10/(50)      占空比 = CCR1/T
    HAL_TIM_PWM_Start(&htim17,TIM_CHANNEL_1);
    TIM17->ARR = 499; // 周期=500us，频率 = 2kHz
    TIM17->CCR1 = 400; // 占空比 = 80%
}
```