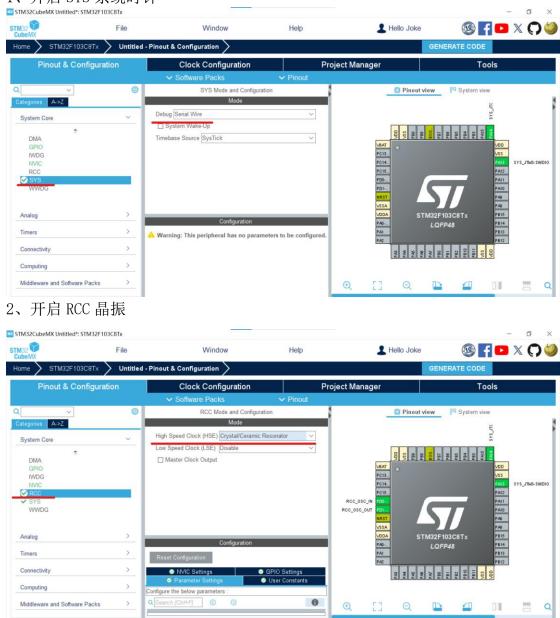
任务要求:使用**串口空闲**中断,输入字符串"123"时,PC13亮,当输入字符串"456"时,PC13灭,其他状态保持不变。

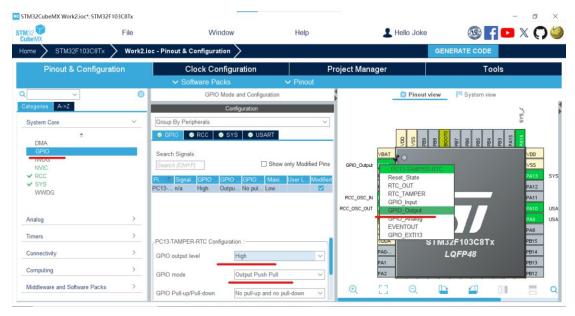
使用到的外设: GPIO、USART 串口

一、STM32CubeMX 创建工程步骤

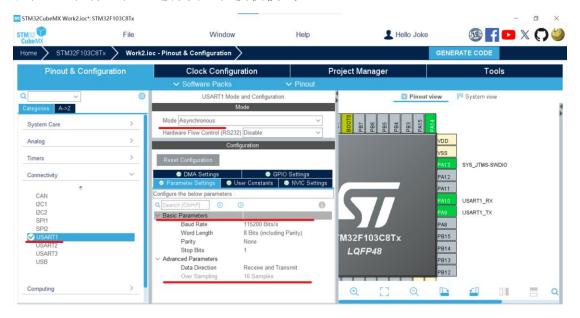
1、开启 SYS 系统时钟



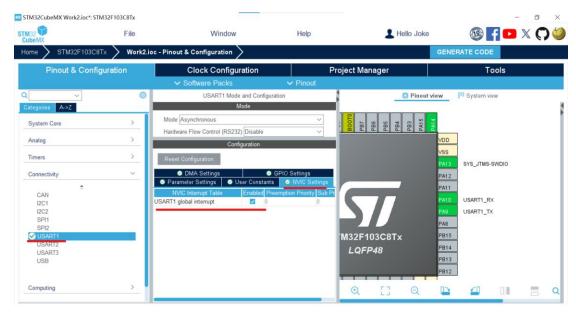
3、选择 PC13 端口设置为 GPI0_Output 模式,设置为推挽输出,并且初始化电平为高电平



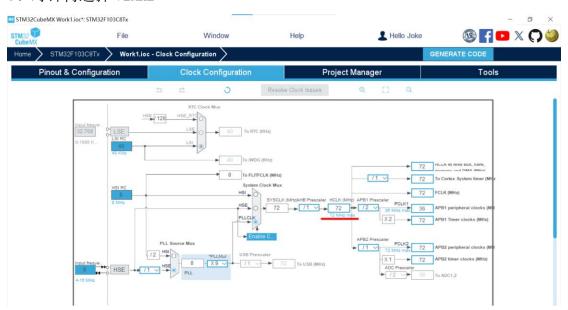
4、选择 USART1 作为串口通信,波特率设置为 115200, 8 位字节数据位,无校验位,1 位停止位,选择发送和接收模式



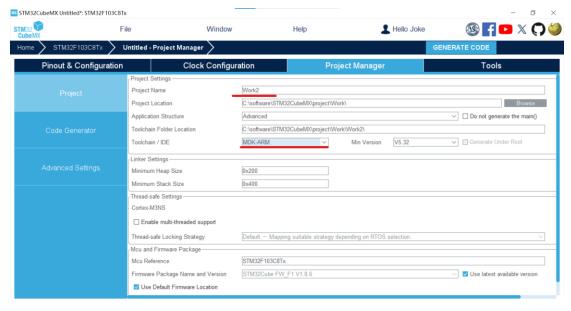
5、开启串口中断



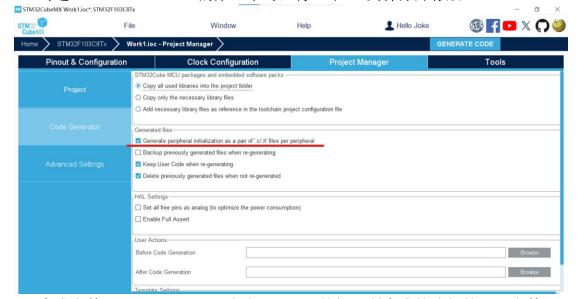
6、时钟树选择 72MHZ



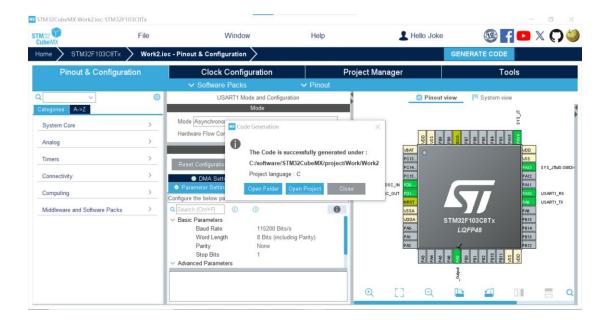
7、创建文件名,并存在对应的盘中,选择 MDK-ARM



8、勾选 Generated files 的第一个勾,将.c和.h文件分开存放

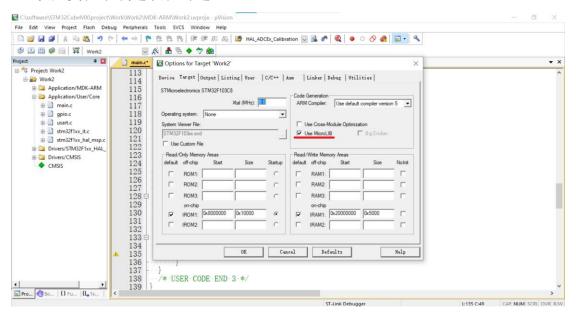


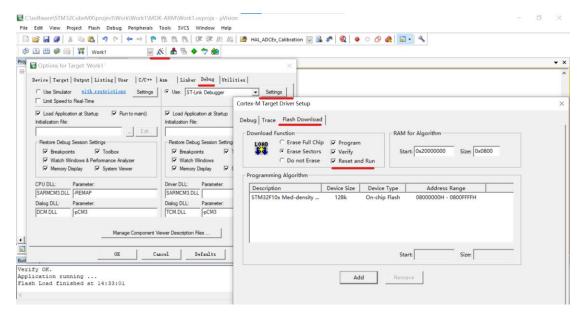
9、生成文件 GENERATE CODE,点击 CLOSE,并打开所生成的路径的 MDK 文件



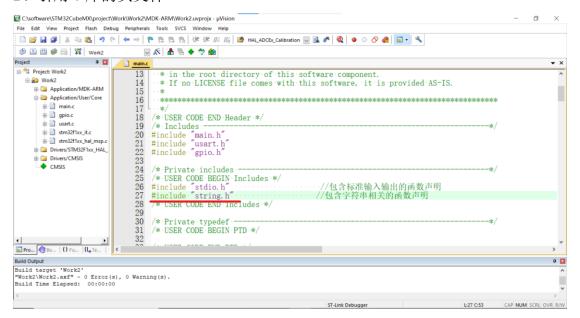
二、MDK 程序编写

1、设置复位下载选项和勾选 Use MicroLIB





2、引用 C 库的头文件



3、定义全局变量

```
☑ C:\software\STM32CubeMX\project\Work\Work2\MDK-ARM\Work2.uvprojx - μVision
                                                                                                                                                 f X
 File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
 👿 🔊 🗂 🖥 💠 🥎 🚵
                    t 🖪
 ☐ 🎁 Project: Work2
                                      /* · USER · CODE · END · PM · */

    Application/MDK-ARM

                                      /* Private variables
     46
                                  47 /* USER CODE BEGIN PV */
48 uint8_t Buffer[50];
49 uint8_t RxFlag = 0;
       ⊕ ☐ gpio.c
⊕ ☐ usart.c
⊕ ☐ stm32f1xx_it.c
⊕ ☐ stm32f1xx_hal_msp.c
                                                                                    //定义全局数组Buffer
//定义接收标志位,初值为0
                                  51
     Drivers/STM32F1xx_HAL_
Drivers/CMSIS
CMSIS
                                 /* Private function prototypes
void SystemClock Config(void);
/* USER CODE BEGIN PFP */
55 void HAL_UARTEX_RXEVentCallbac
                                      void HAL_UARTEx_RxEventCallback(UART_HandleTypeDef *huart, uint16_t Size)
                                  57
58 □
59
                                            if (huart == &huart1) -----//检查回调函数是否是由huart1 (即UART1接口) 触发的
                                                    flag = 1; //如果为1,表示已经接收到数据
(strcmp((char *)Buffer, "123\r\n") == 0) //接收到的数据与字符串"123\r\n"进行比较
                                                RxFlag = 1;
                                  60
                                                          HAL GPIO WritePin(GPIOA, GPIO PIN 7, GPIO PIN RESET): //点亮PC13
                                  62
Build Output
                                                                                                                                                        4 🖪
Build target 'Work2'
"Work2\Work2.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed: 00:00:00
```

4、重写中断回调函数,创建发送字节函数和 fputc 重定向

```
đ
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
S & 3 5 9 8
                          Project

Project: Work2
                 4 🗵

□ Macation/User/Core

                                 if (huart == &huart1) ·····//检查回调函数是否是由huart1 (即UART1接口) 触发的
     Application/User/Lore

iii main.c

iii gpio.c

iii usert.c

iii stm32f1xc_it.c

iii stm32f1xc_hal_msp.c
                                     RxFlag = 1; //如果为1,表示已经接收到数据
if (strcmp((char *)Buffer , ~123\r\n") = 0) //接收到的数据与字符串"123\r\n"进行比较
                                           HAL GPIO WritePin(GPIOC, GPIO PIN 13, GPIO PIN RESET); //点亮PC13

■ Drivers/STM32F1xx_HAL_C

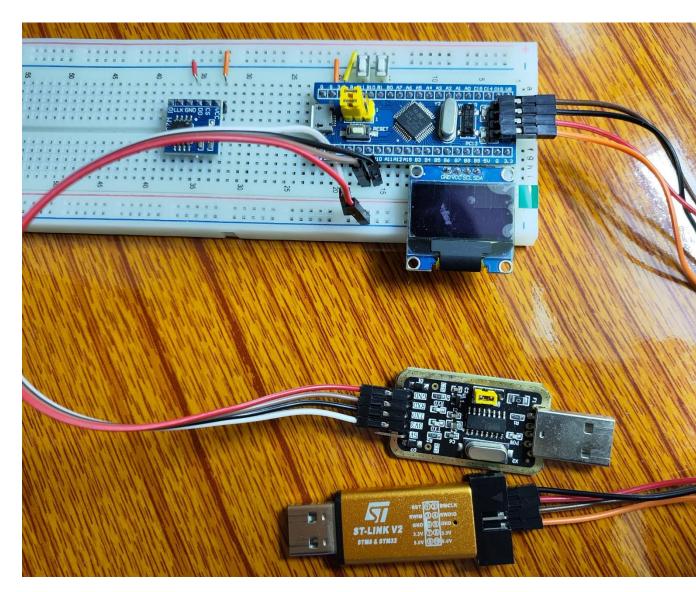
                                          if(strcmp((char *)Buffer , "456\r\n") == 0)//接收到的数据与字符串"456\r\n"进行比较
    # Drivers/CMSIS
                                           HAL GPIO WritePin(GPIOC, GPIO PIN 13, GPIO PIN SET); //熄灭PC13
                                 HAL UARTEX ReceiveToIdle IT(&huartl, Buffer, sizeof(Buffer)); //重新启动UART接收
                                  /* 将字节数据写入数据寄存器,写入后USART自动生成时序波形 */
HAL UART Transmit (&huartl, &byte, 1, 1000);
while( HAL UART GET FLAG(&huartl, UART FLAG TXE) = RESET); //等待发送完成
1:63 C:14 CAP NUM SCRU OVE BA
```

5、开启接收中断,打印接收到的数据发回串口软件

```
| Clasifiware/STM32cubeMXtproject/WorkWorkZwDK-ARM/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZwproject/WorkZw
```

三、硬件连接

ST1ink	→	核心板	CH340	→	核心板
3. 3V	→	3. 3V	3. 3V	→	3. 3V
GND	→	GND	GND	→	GND
SWDIO	→	SWIO	TXD	→	PA10
SWCLK	→	SWCLK	RXD	→	PA9



四、代码部分

```
/* Private includes -----
--*/
2. /* USER CODE BEGIN Includes */
      #include "stdio.h"
                                       //包含标准输入输出的函数声明
     #include "string.h"
                                      //包含字符串相关的函数声明
      /* USER CODE END Includes */
5.
      /* USER CODE BEGIN PV */
1.
      uint8_t Buffer[50];
                                     //定义全局数组 Buffer
      uint8_t RxFlag = 0;
                                     //定义接收标志位,初值为 0
      /* USER CODE END PV */
     /* Private function prototypes -----
1.
--*/
```

```
2.
     void SystemClock_Config(void);
3.
       /* USER CODE BEGIN PFP */
       void HAL_UARTEx_RxEventCallback(UART_HandleTypeDef *huart, uint16_t Size)
4.
5.
          if (huart == &huart1)
                                     //检查回调函数是否是由 huart1(即 UART1接口)触
6.
发的
7.
          {
                                     //如果为1,表示已经接收到数据
8.
              RxFlag = 1;
              if (strcmp((char *)Buffer , "123\r\n") == 0) //接收到的数据与字符串
9.
"123\r\n"进行比较
10.
11.
                      HAL_GPIO_WritePin(GPIOC,GPIO_PIN_13,GPIO_PIN_RESET); //点亮
PC13
12.
13.
              else if (strcmp((char *)Buffer , "456\r\n") == 0)//接收到的数据与字符
串"456\r\n"进行比较
14.
                  {
15.
                      HAL GPIO WritePin(GPIOC,GPIO PIN 13,GPIO PIN SET);
                                                                           //熄
灭 PC13
16.
17.
           }
18.
          HAL_UARTEx_ReceiveToIdle_IT(&huart1,Buffer,sizeof(Buffer)); //重新启动
UART 接收
19.
       }
20.
21.
       void SendByte(uint8_t Byte)
22.
23.
           /* 将字节数据写入数据寄存器,写入后 USART 自动生成时序波形 */
24.
          HAL_UART_Transmit(&huart1,&Byte,1,1000);
25.
          while( HAL UART GET FLAG(&huart1, UART FLAG TXE) == RESET); //等待发送完
成
26.
       }
27.
28.
       int fputc(int ch ,FILE *f)
29.
          SendByte(ch);
                            //将 printf 的底层重定向到自己的发送字节函数
30.
31.
          return ch;
32.
       /* USER CODE END PFP */
33.
1.
       /* USER CODE BEGIN 2 */
2.
       /* 启动了一个中断驱动的 UART 接收操作*/
3.
       HAL_UARTEx_ReceiveToIdle_IT(&huart1,Buffer,sizeof(Buffer));
4.
```

```
5.
     /* USER CODE END 2 */
6.
7.
      /* Infinite loop */
    /* USER CODE BEGIN WHILE */
8.
9.
      while (1)
10.
        /* USER CODE END WHILE */
11.
12.
        /* USER CODE BEGIN 3 */
13.
          if (RxFlag == 1) //如果接收到数据
14.
15.
            RxFlag = 0; //清空接收标志位
16.
17.
            printf((char *)Buffer); //打印 Buffer 数组里面的值
18.
19.
       }
      /* USER CODE END 3 */
20.
```

五、实现效果

