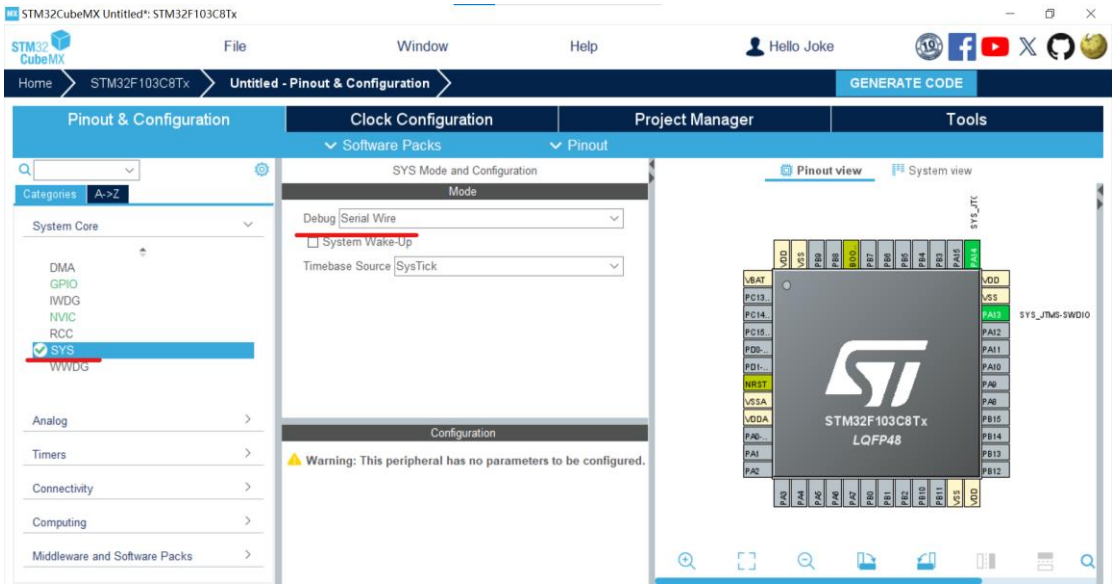
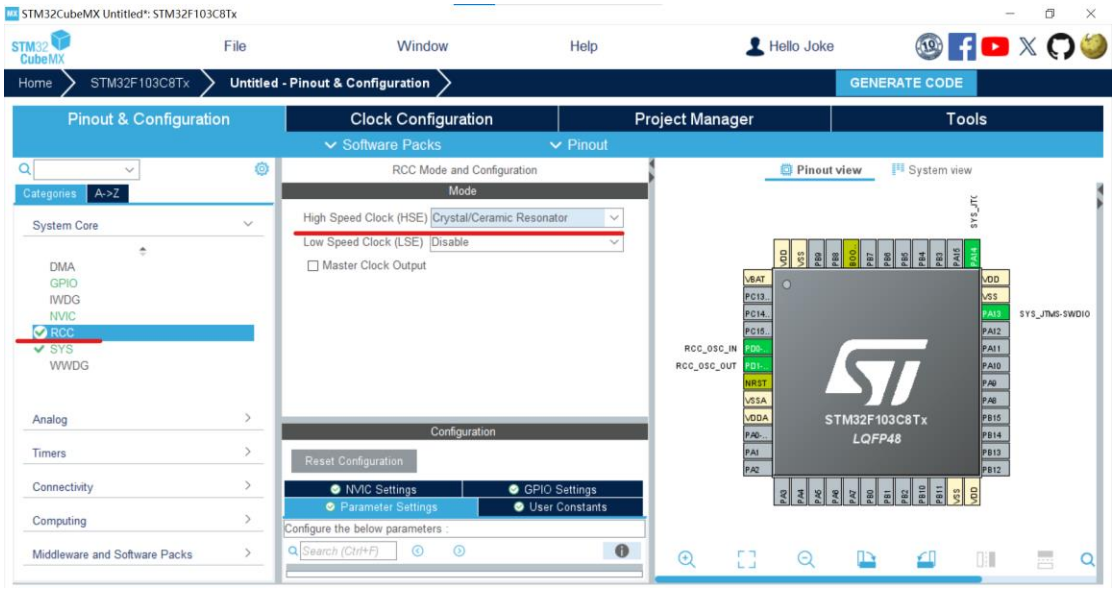
任务要求：使用串口轮询，输入字符串“123”时，PC13 亮，当输入字符串“456”时，PC13 灭，其他状态保持不变。

一、STM32CubeMX 创建工程步骤

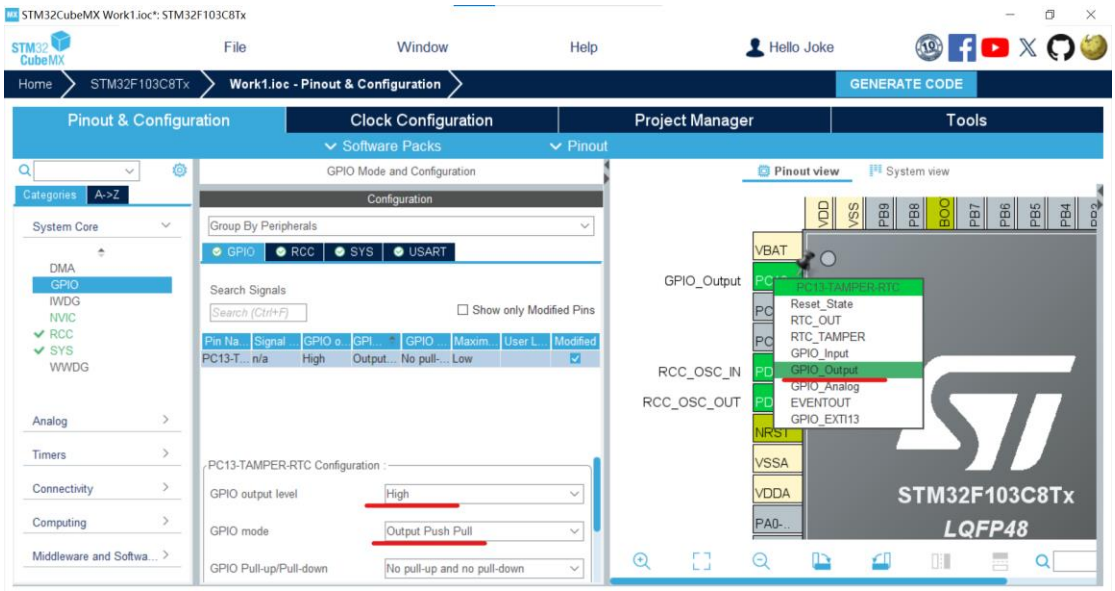
1、开启 SYS 系统时钟



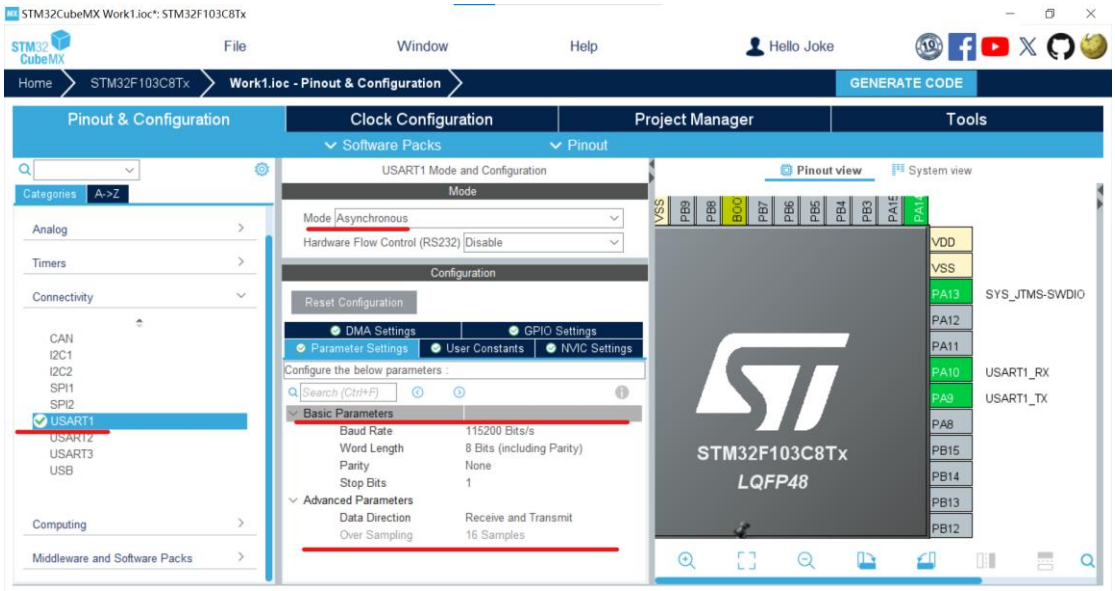
2、开启 RCC 晶振



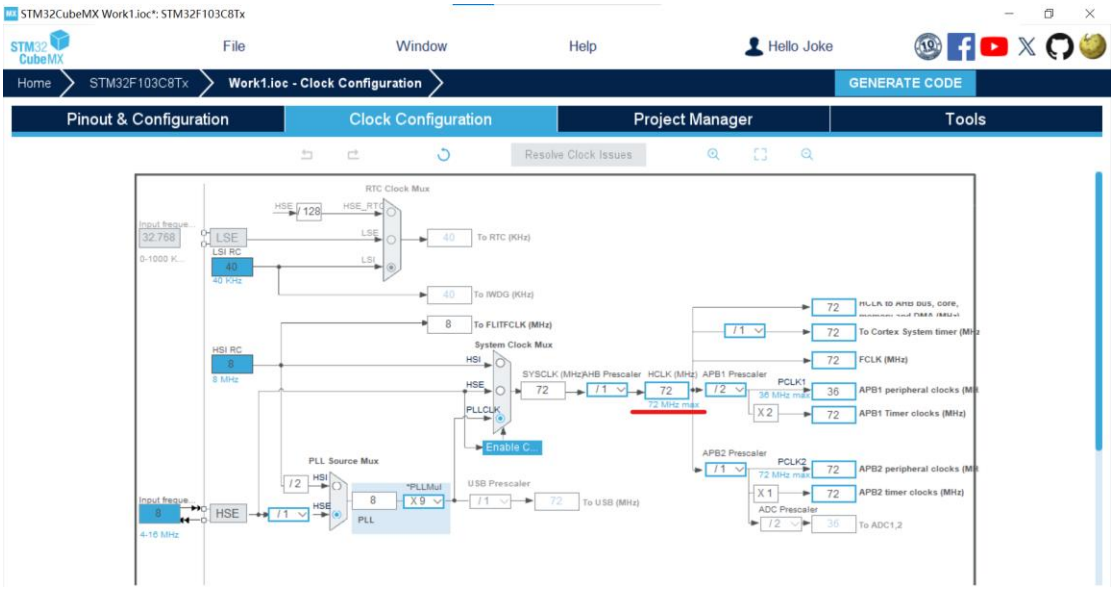
3、选择 PC13 端口设置为 GPIO_Output 模式，设置为推挽输出，并且初始化电平为高电平



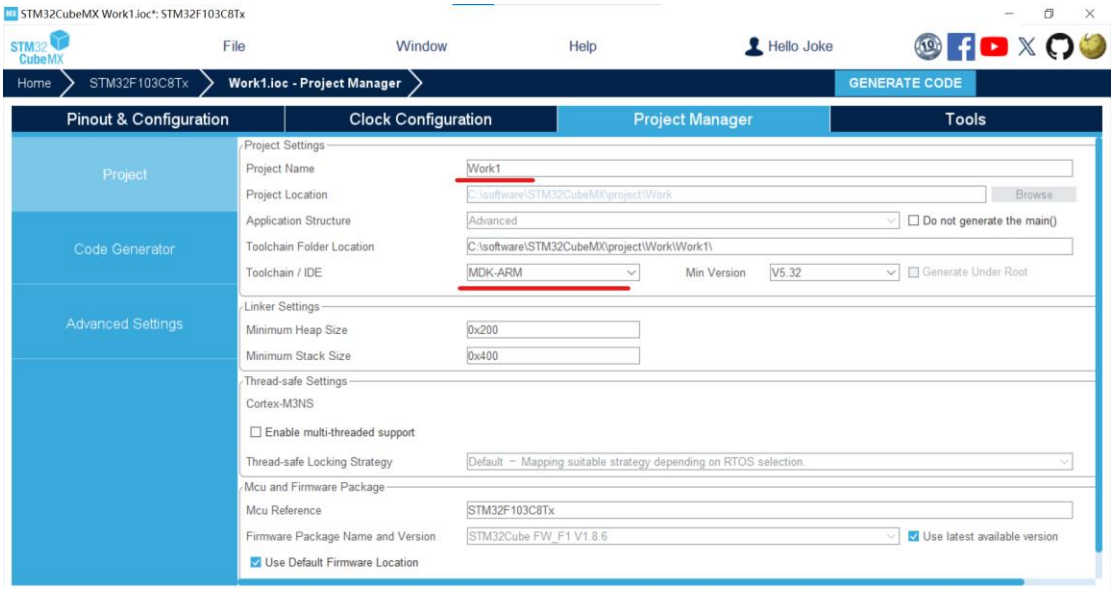
4、选择 USART1 作为串口通信，波特率设置为 115200， 8 位字节数据位，无校验位，1 位停止位，选择发送和接收模式



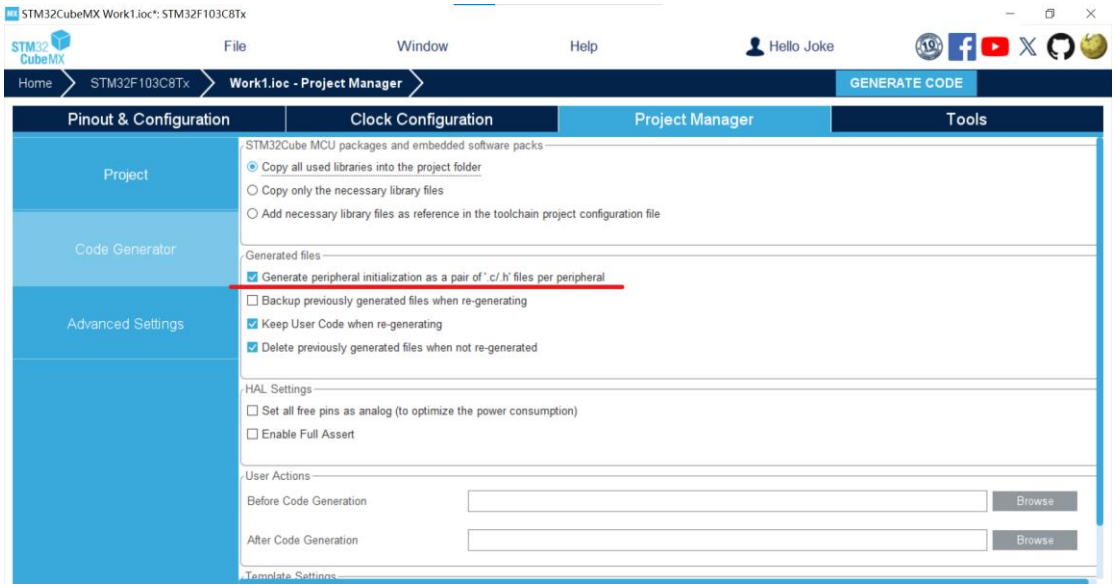
5、时钟树选择 72MHZ



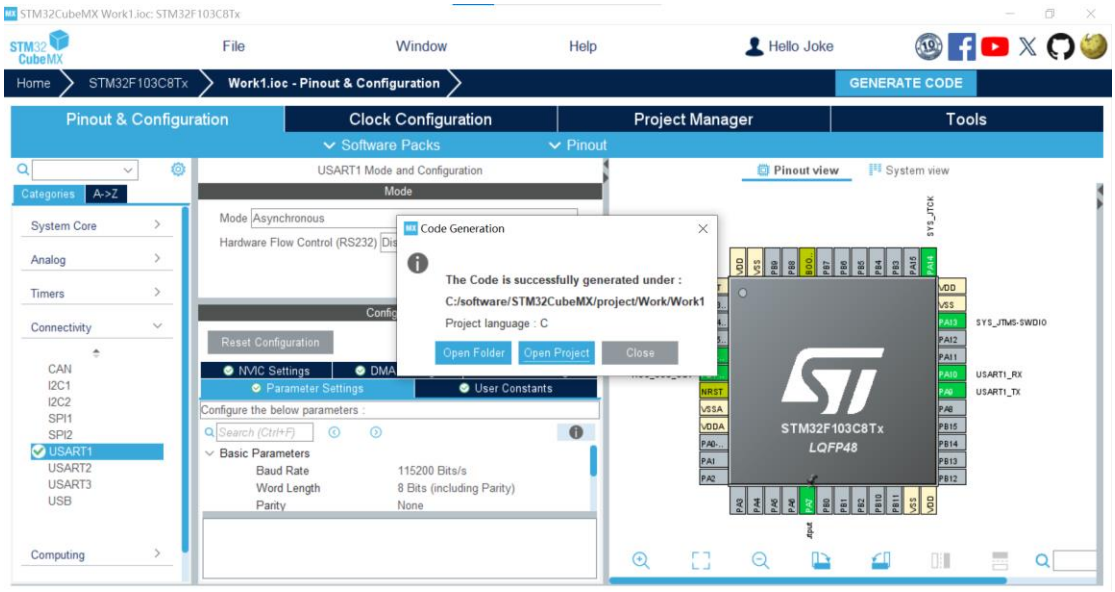
6、创建文件名，并存在对应的盘中，选择 MDK-ARM



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

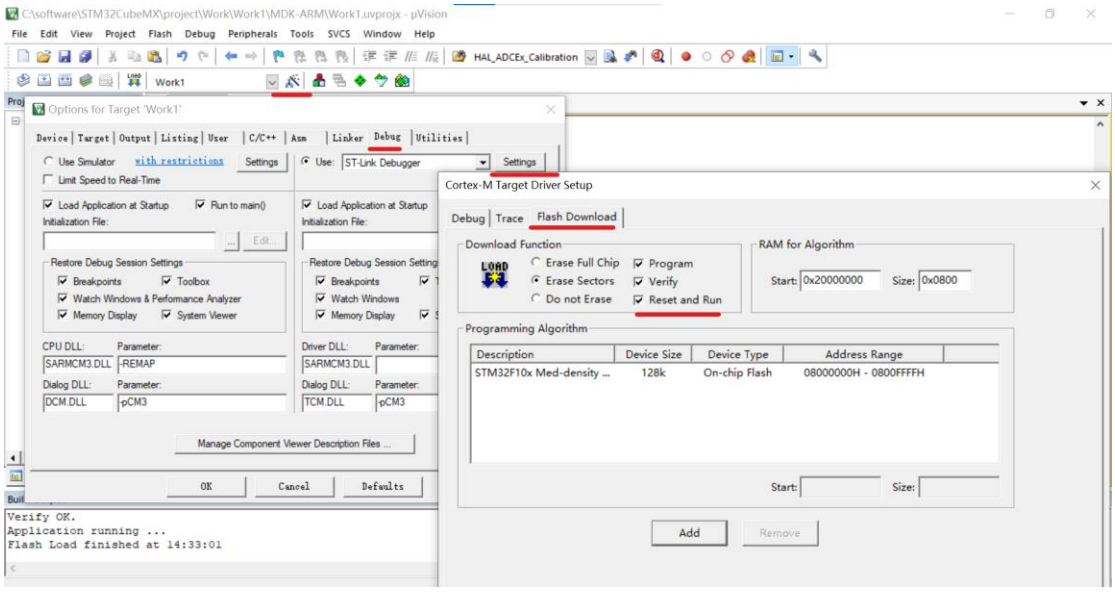


8、生成文件 GENERATE CODE，点击 CLOSE，并打开所生成的路径的 MDK 文件

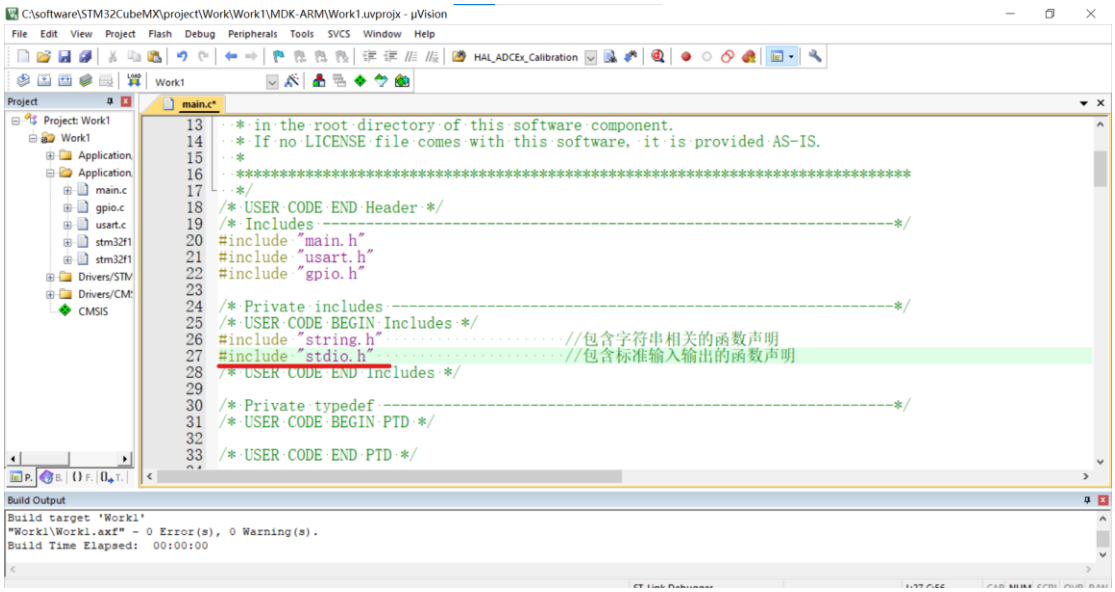


二、MDK 程序编写

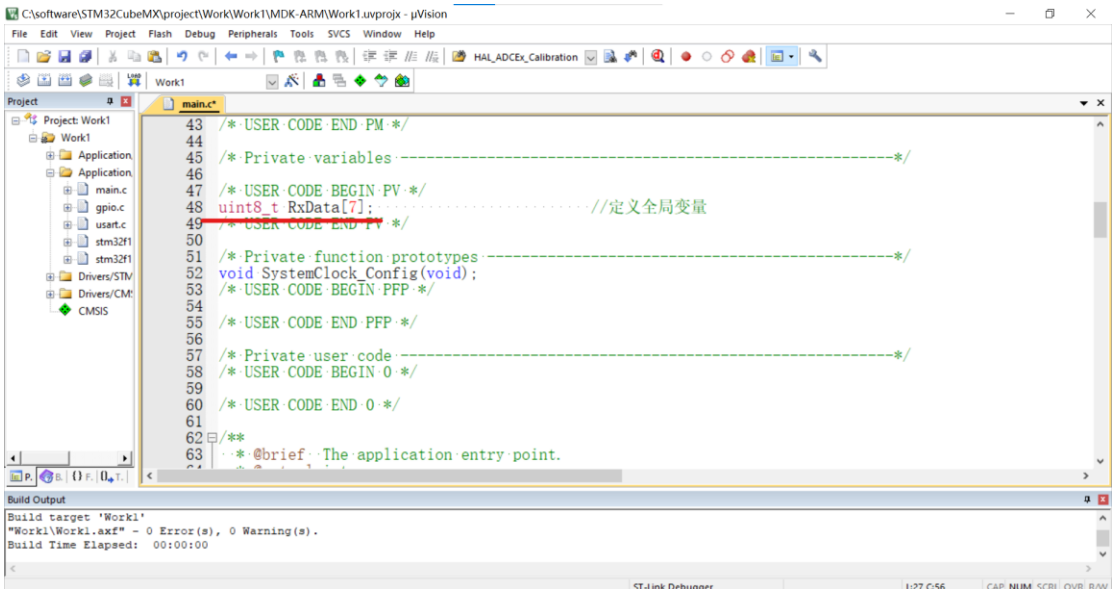
1、设置复位下载选项



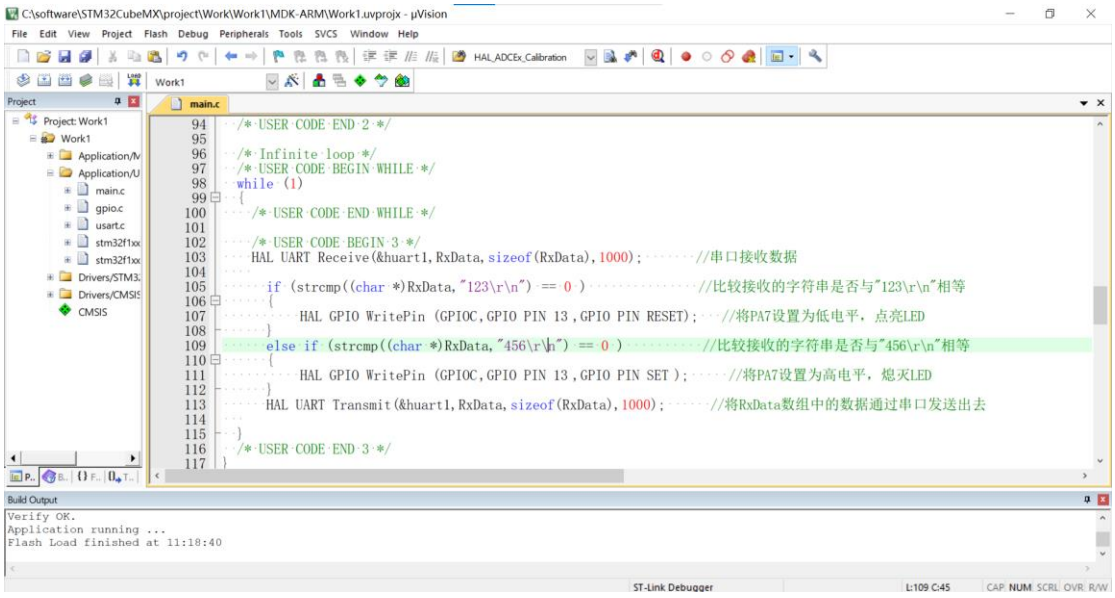
2、引用 C 库的头文件



3、定义全局变量

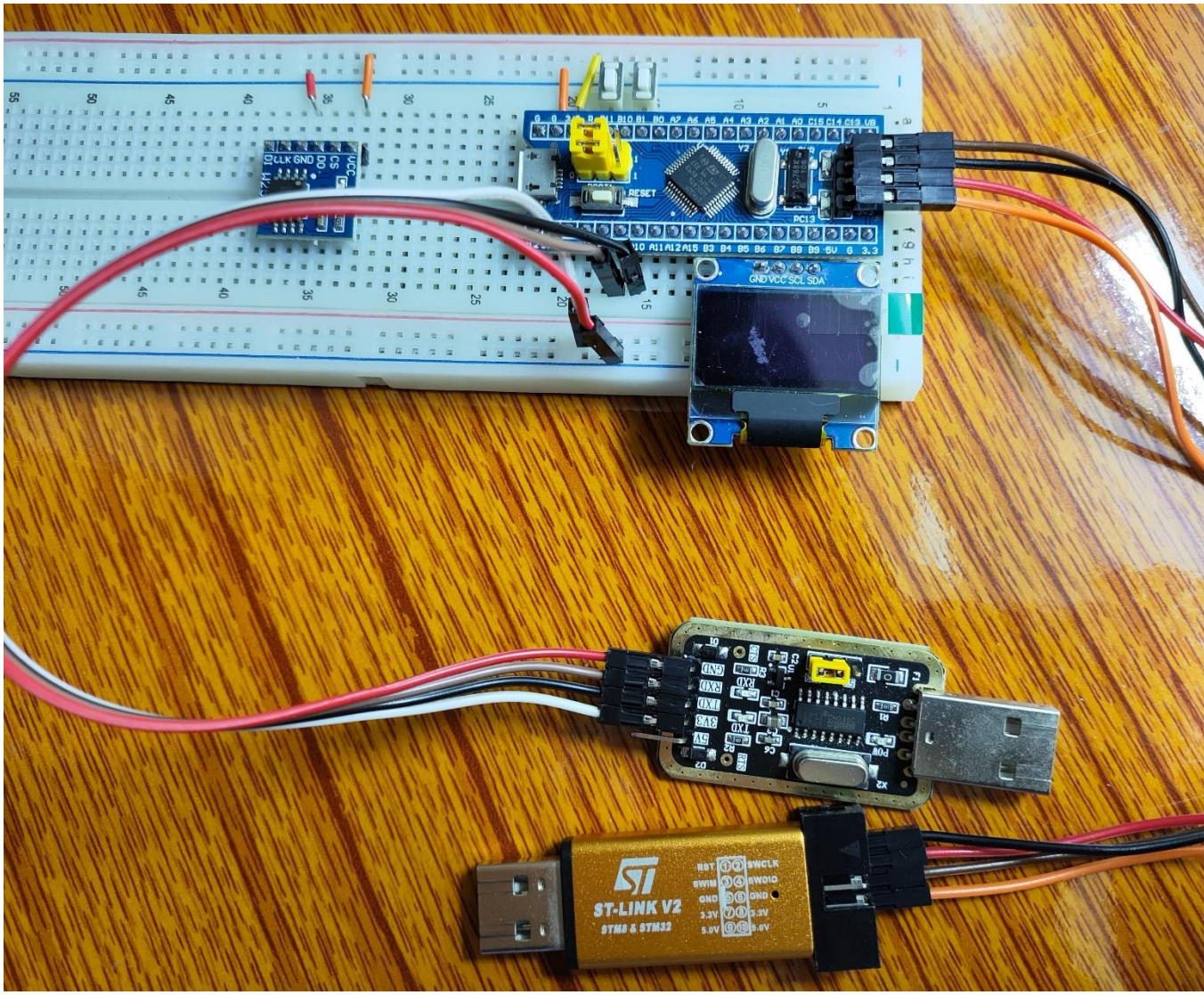


4、轮询方法实现接收数据，实现点亮和熄灭 LED



三、硬件连接

STlink	→	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

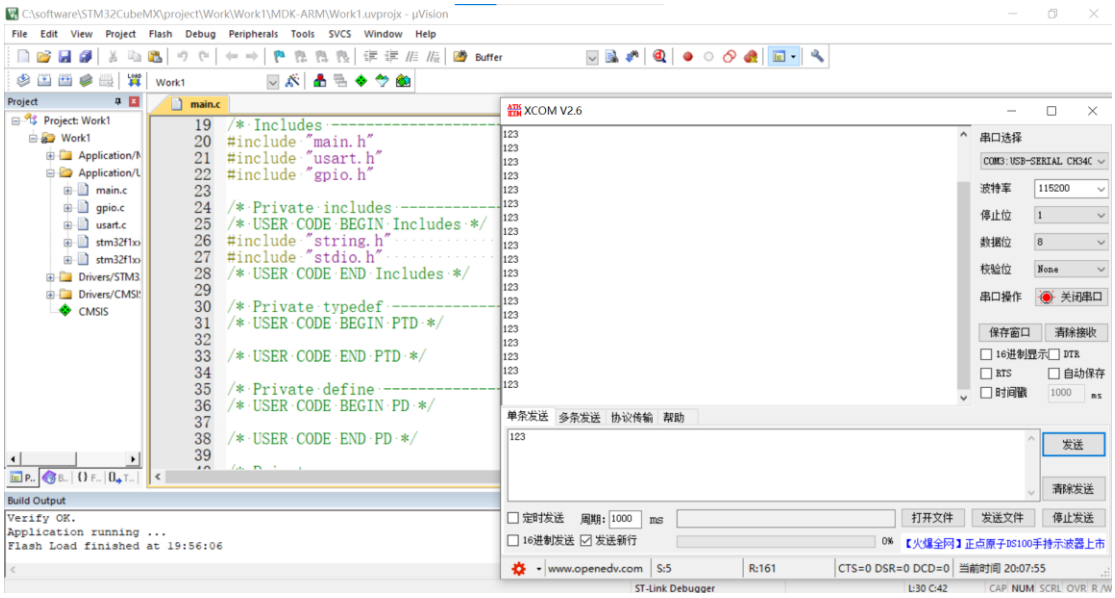
```
1.      /* Private includes ----------------------------------------------------------
--*/
2.      /* USER CODE BEGIN Includes */
3.      #include "string.h"                          //包含字符串相关的函数声明
4.      #include "stdio.h"                           //包含标准输入输出的函数声明
5.      /* USER CODE END Includes */
```

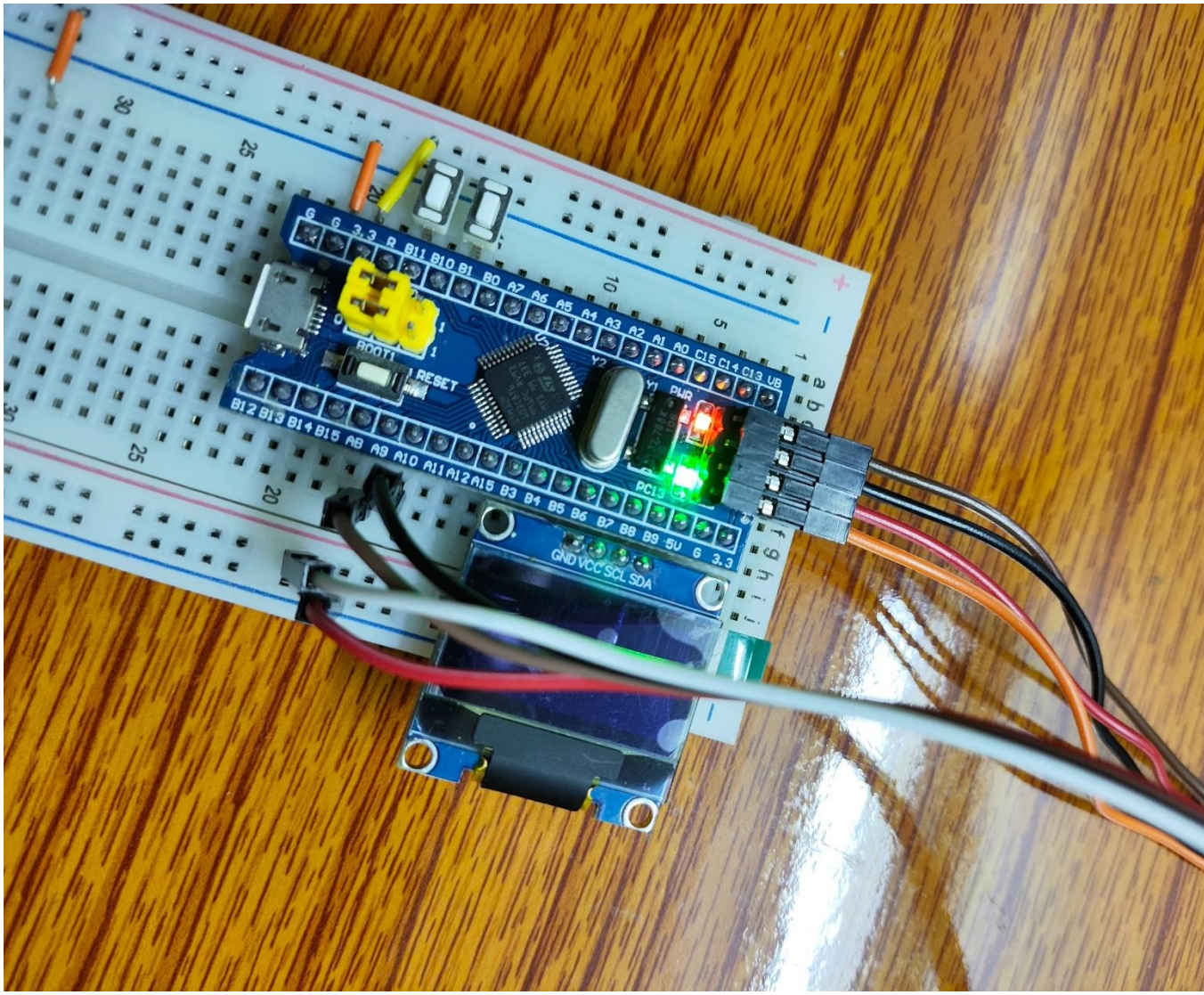
```
1.   /* USER CODE BEGIN PV */
2.   uint8_t RxData[7];                              //定义全局变量
3.   /* USER CODE END PV */
```

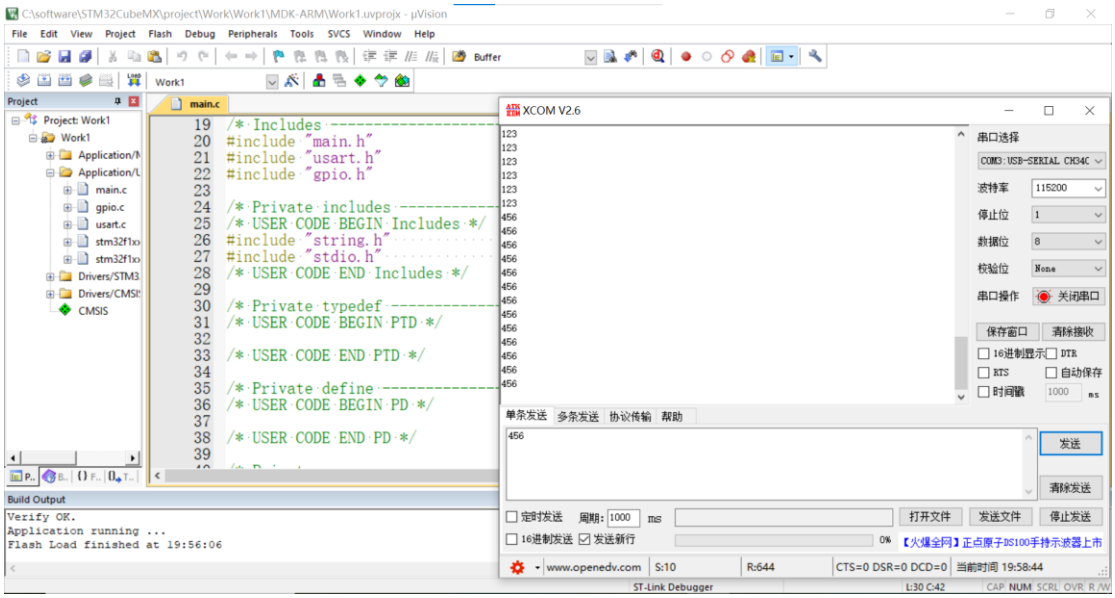
```
1.      /* Infinite loop */
2.      /* USER CODE BEGIN WHILE */
3.      while (1)
4.      {
```

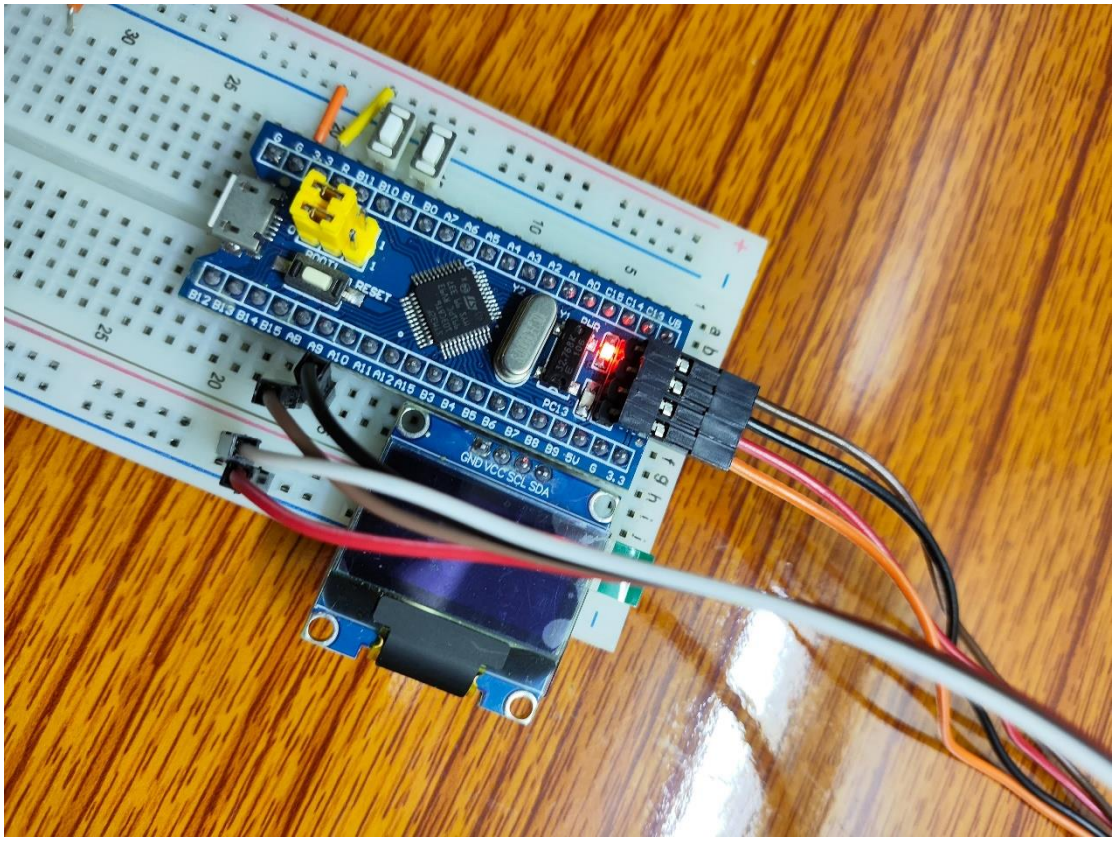
```
        /* USER CODE END WHILE */

        /* USER CODE BEGIN 3 */
        HAL_UART_Receive(&huart1,RxData,sizeof(RxData),1000);        //串口接收数据

        if (strcmp((char *)RxData,"123\r\n") == 0 )                  //比较接收的字符串是否与"123\r\n"相等
        {
            HAL_GPIO_WritePin(GPIOC,GPIO_PIN_13,GPIO_PIN_RESET);    //将 PA7 设置为低电平，点亮 LED
        }
        else if (strcmp((char *)RxData,"456\r\n") == 0 )             //比较接收的字符串是否与"456\r\n"相等
        {
            HAL_GPIO_WritePin(GPIOC,GPIO_PIN_13,GPIO_PIN_SET);      //将 PA7 设置为高电平，熄灭 LED
        }
        HAL_UART_Transmit(&huart1,RxData,sizeof(RxData),1000);       //将 RxData 数组中的数据通过串口发送出去

    }
    /* USER CODE END 3 */
```

五、实现效果




G G 3.3 B11 B10 B1 B0 A7 A6 A5 A4 A3 A2 A1 A0 C15 C14 C13 VB
B12 B13 B14 B15 A8 A9 A10 A11 A12 A15 B3 B4 B5 B6 B7 B8 B9 5V G 3.3
BOOT1
RESET
PWR
PC13
GND VCC SCL SDA


C:\software\STM32CubeMX\project\Work\Work1\MDK-ARM\Work1.uvprojx - µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Project: Work1
Work1
main.c
gpio.c
usart.c
CMSIS
/* Includes ------------------------------
#include "main.h"
#include "usart.h"
#include "gpio.h"
/* Private includes ----------
/* USER CODE BEGIN Includes */
#include "string.h"
#include "stdio.h"
/* USER CODE END Includes */
/* Private typedef -----------
/* USER CODE BEGIN PTD */
/* USER CODE END PTD */
/* Private define ------------
/* USER CODE BEGIN PD */
/* USER CODE END PD */
Build Output
Verify OK.
Application running ...
Flash Load finished at 19:56:06
XCOM V2.6
串口选择
COM3:USB-SERIAL CH340
波特率 115200
停止位 1
数据位 8
校验位 None
串口操作 关闭串口
保存窗口 清除接收
单条发送 多条发送 协议传输 帮助
发送
清除发送
定时发送 周期: 1000 ms
打开文件 发送文件 停止发送
16进制发送 发送新行
【火爆全网】正点原子DS100手持示波器上市
www.openedv.com S:10 R:644 CTS=0 DSR=0 DCD=0 当前时间 19:58:44
ST-Link Debugger



六、思考题

1、推挽输出

双向能力：推挽输出可以驱动高电平和低电平。
驱动能力强：能够提供较大的电流，驱动能力较强。
速度快：因为驱动能力强，所以信号的切换速度较快。
电平确定：输出电平由电路直接确定，不需要外部元件。
简单直接：输出逻辑电平 0 或 1 时，通常不需要外部电阻。

2、开漏输出

单向能力：开漏输出通常只能输出低电平，高电平需要外部上拉电阻来实现。
灵活的电平设定：可以通过外部电阻来设定不同的高电平电位。
线与：多个开漏输出可以连接在一起，实现逻辑与的功能。
低功耗：因为没有内部上拉，所以当输出低电平时功耗较低。
用于 I2C 通信接口：开漏输出常用于 I2C 等通信协议，因为它们需要多个设备共享同一总线。
内部设置上拉电阻：在配置开漏输出时，内部可能设置有上拉电阻，以减少外部干扰。