第一部分：GPIO部分

GPIO输出

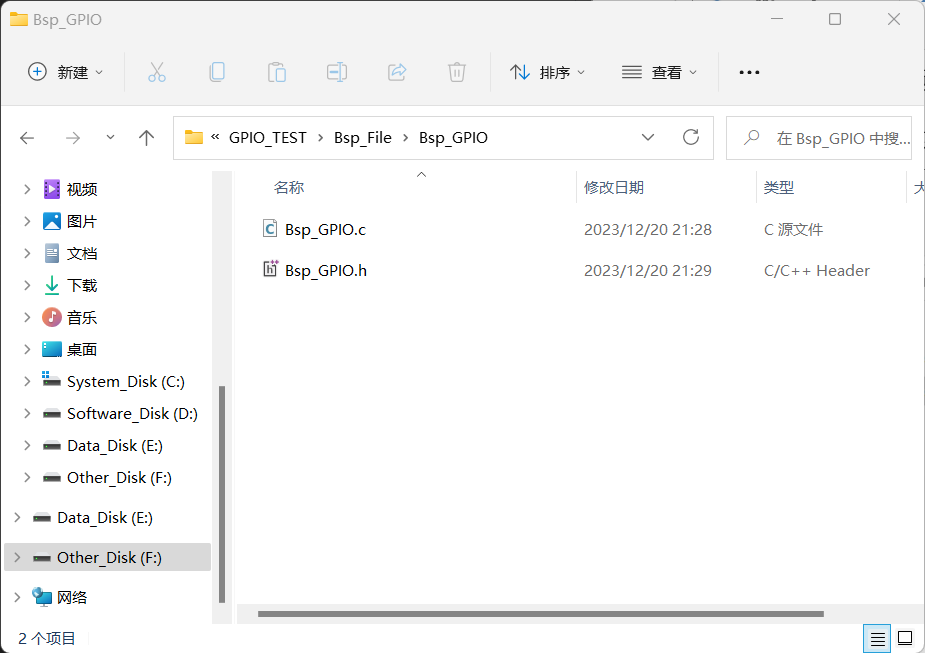
使用LED就是典型的GPIO输出控制

1.首先了解板载LED灯对应的GPIO引脚

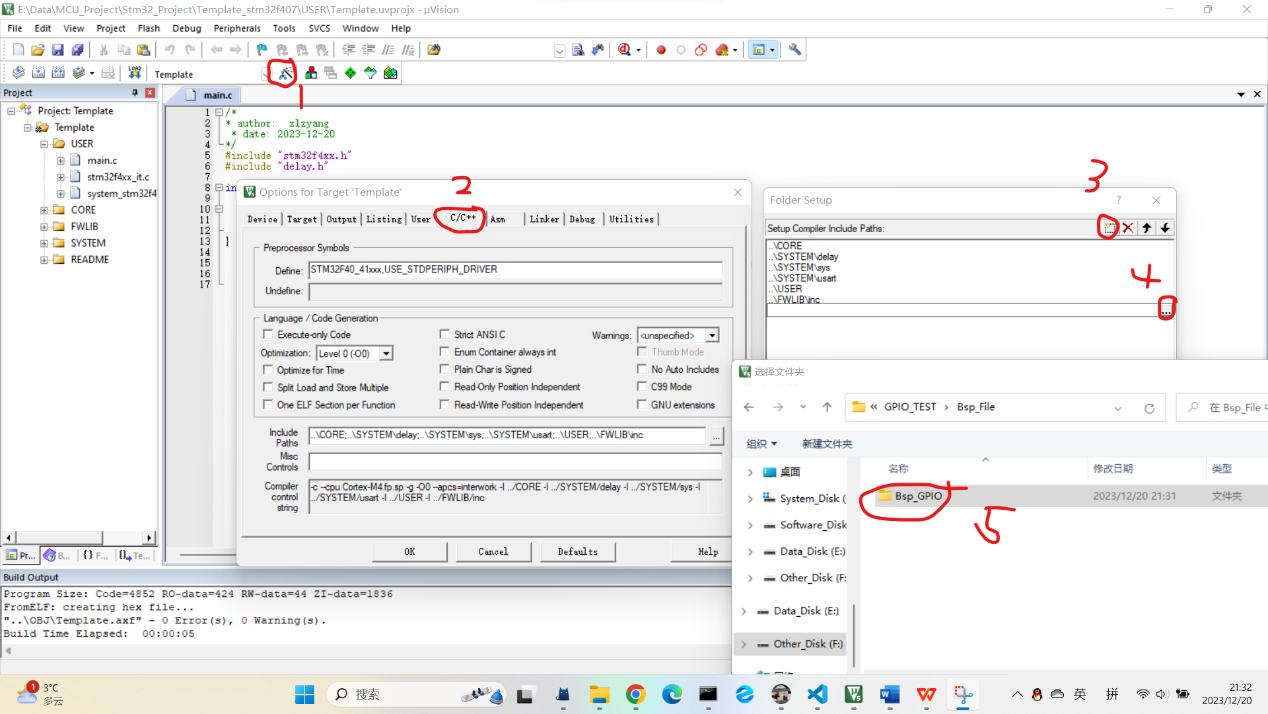


（由图可得LED灯绑定引脚为PF9，PF10，且低电平时点亮）

2.创建工程，将空白工程起名GPIO\_TEST，在Bsp\_File中创建文件夹Bsp\_GPIO在文件夹中创建Bsp\_GPIO.c和Bsp\_GPIO.h



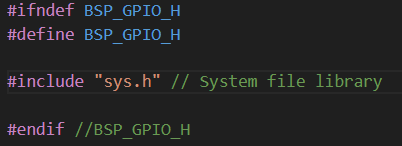
3.绑定文件夹（选择文件夹，ok ok）



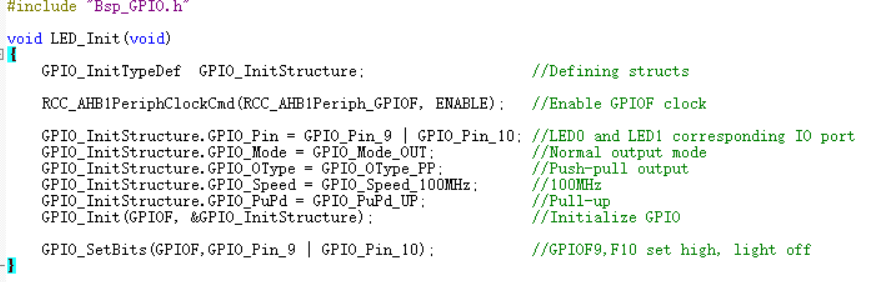
4.双击Bsp\_File，放入（2）添加文件

5.编写头文件

这个是防卫式声明，为了防止重复包含，必须有



6.函数编写



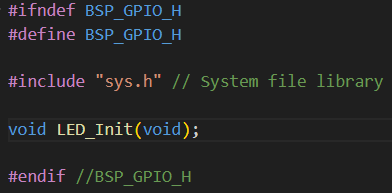
6.1包含头文件，为了调用下面函数各种定义好的结构体函数什么的

6.2整个初始化配置其实就是对GPIO结构体的配置，所以要事先创建我们使用的配置结构体

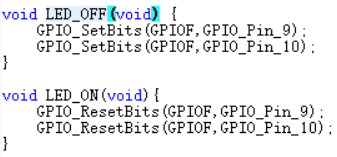
6.3 GPIOF在AHB1时钟总线上所以要使能

6.4 因为是输出电平控制LED，所以要输出模式

7. 声明LED\_Init函数在头文件中（方便主函数调用）

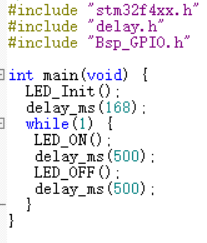


8. LED开关函数书写（本质上就是控制引脚电平）（写在Bsp\_GPIO.c记得声明）



显然GPIO\_SetBits是将引脚拉成高电GPIO\_ResetBits是拉低

9. 主函数编写



9.1 写在while(1)里的原因是因为让他死循环一直执行

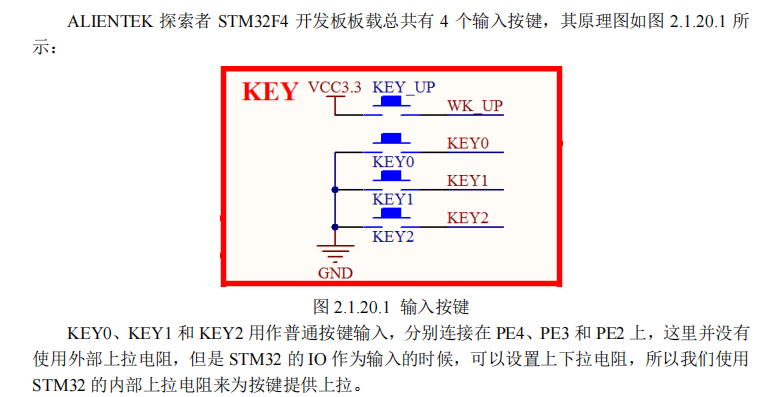
10． 运行（左上角，点第二个，再点六个）



GPIO输入

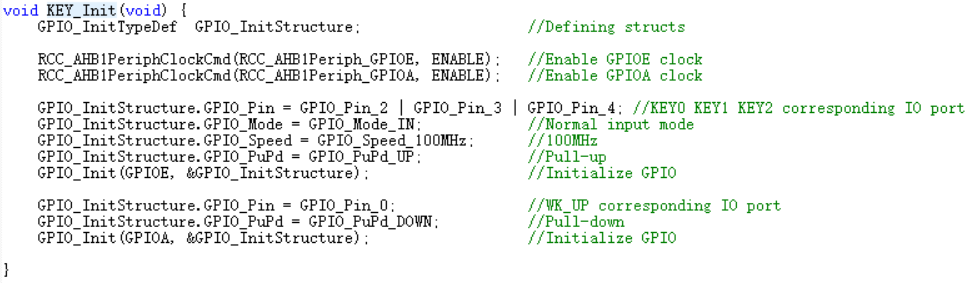
按键控制就是最典型的GPIO输入

1. 所有工程都是这个步骤，第一步硬件原理图分析



（可得按键1，按键2，按键3对应引脚为PE4 PE3 PE2，补充一个最上面的按键是复位按键PA0）

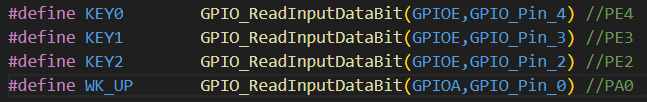
1. 创建头文件（注意防卫式声明），这里已经创建
2. 第二步，编写程序初始化文件，继续在Bsp\_GPIO.c中编写程序（注意声明）



3.1 这里配置模式是输入模式

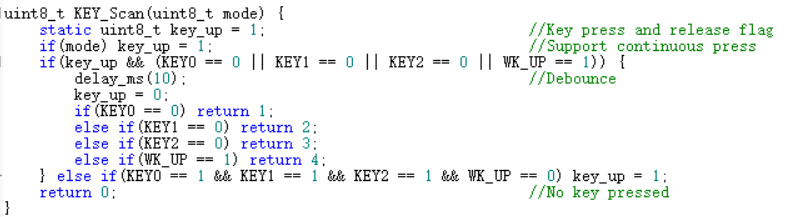
3.2 按键123默认上拉，因为导通低电平，按键生效。复位按键默认下拉，导通高电平

4. 第三步，编写程序文件，普通编写方式如LED灯，对于这种简单程序，可以采用宏定义的方法



（函数如名，读取GPIO输入电平）

按键函数还有一个地方需要注意，就是需要按键消抖



（通过延时的方法来消抖）函数的用法：

//按键处理函数

//返回按键值

//mode:0,不支持连续按;1,支持连续按;

//0，没有任何按键按下

//1，KEY0按下

//2，KEY1按下

//3，KEY2按下

//4，WKUP按下 WK\_UP

//注意此函数有响应优先级,KEY0>KEY1>KEY2>WK\_UP!!