**《中断与定时器》实验报告**

|  |  |  |
| --- | --- | --- |
| **课程名称**： **嵌入式系统设计** | **年级**：23级 | **上机实践成绩**： |
| **指导教师**：郭建 | **姓名**： **张建夫** |  |
| **上机实践名称**： **中断与定时器** | **学号：10235101477** | **上机实践日期**：2025/04/22 |
| **上机实践编号**： | **组号**： | **上机实践时间**： 14：50~16：30 |

1. **目的与要求**

**掌握如何对按键进行检测，且控制不同按键点亮不同的LED灯**

**了解中断的基本知识**

**掌握中断编程的具体流程，并能使用中断检测按键**

**掌握STM32定时器相关库函数的配置；**

**学会使用定时器进行定时并触发中断**

1. **内容与实验原理**
2. **实验内容：**

了解中断的基础知识

掌握中断编程过程，并能用中断检测按键

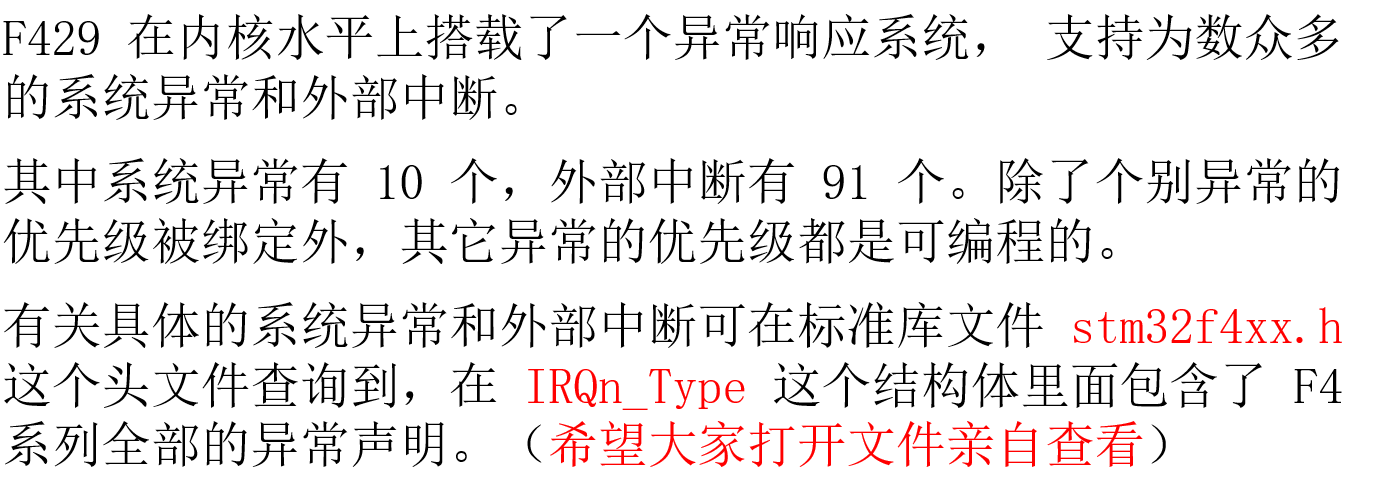
定时器及中断相关库函数的配置

使用定时器控制LED灯亮灭的间隔时间

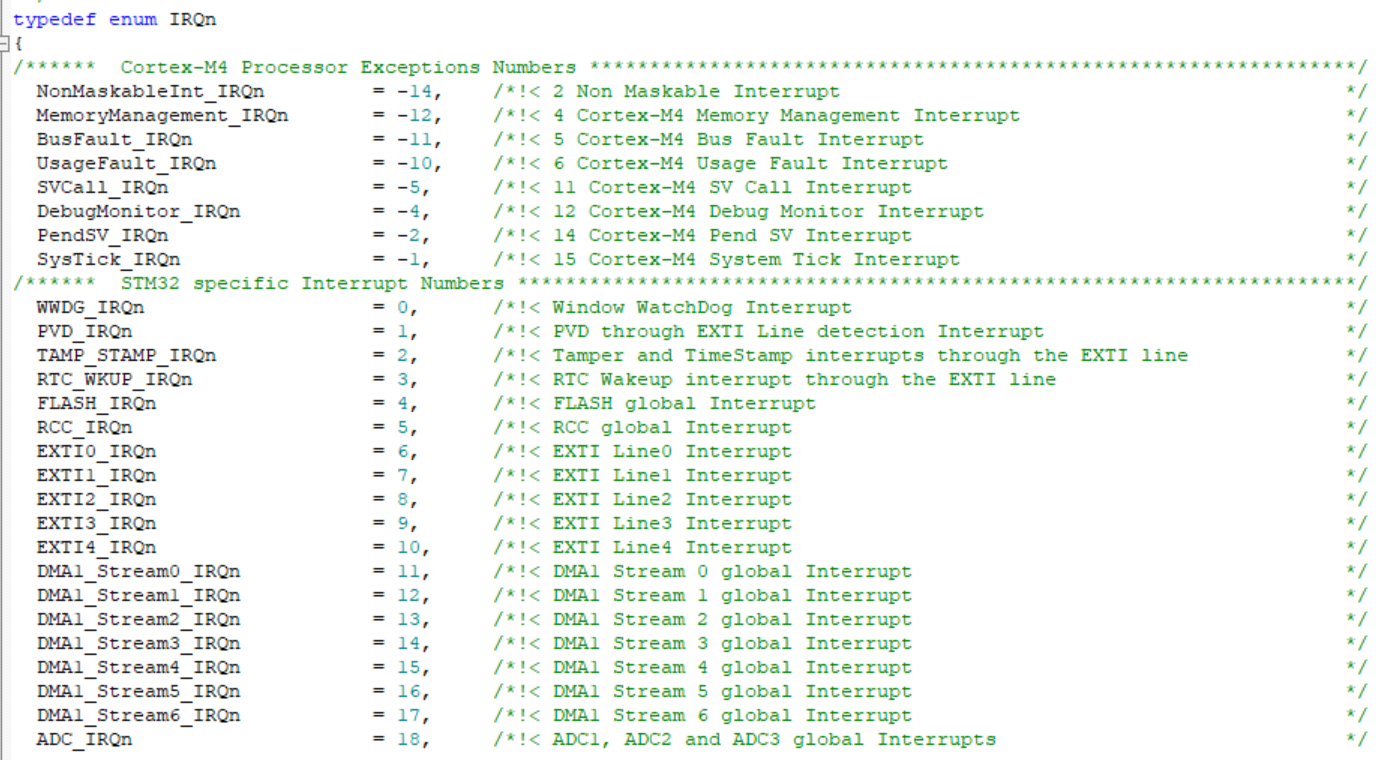
相关寄存器值，调整定时时间

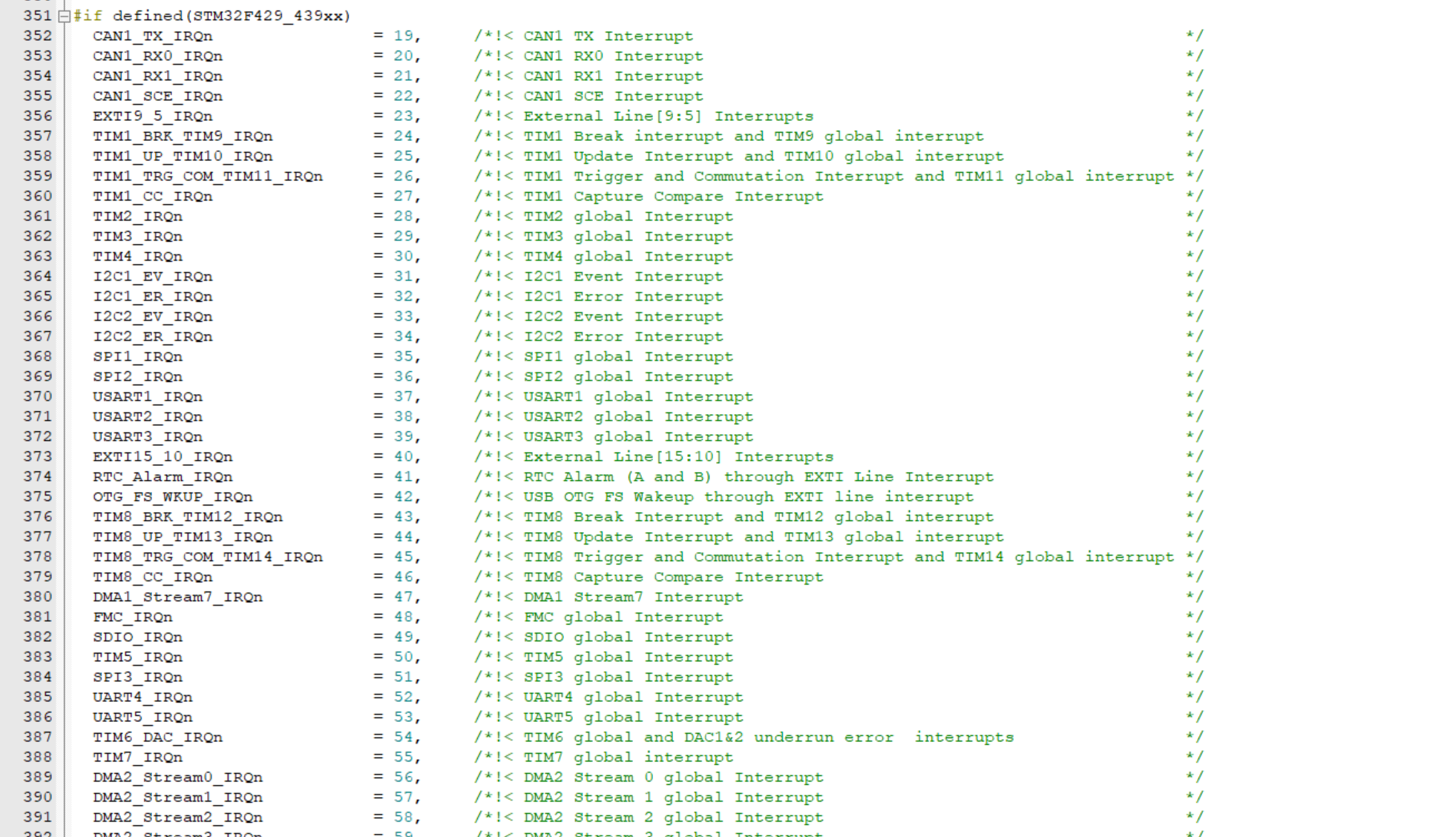
**2.实验原理：**

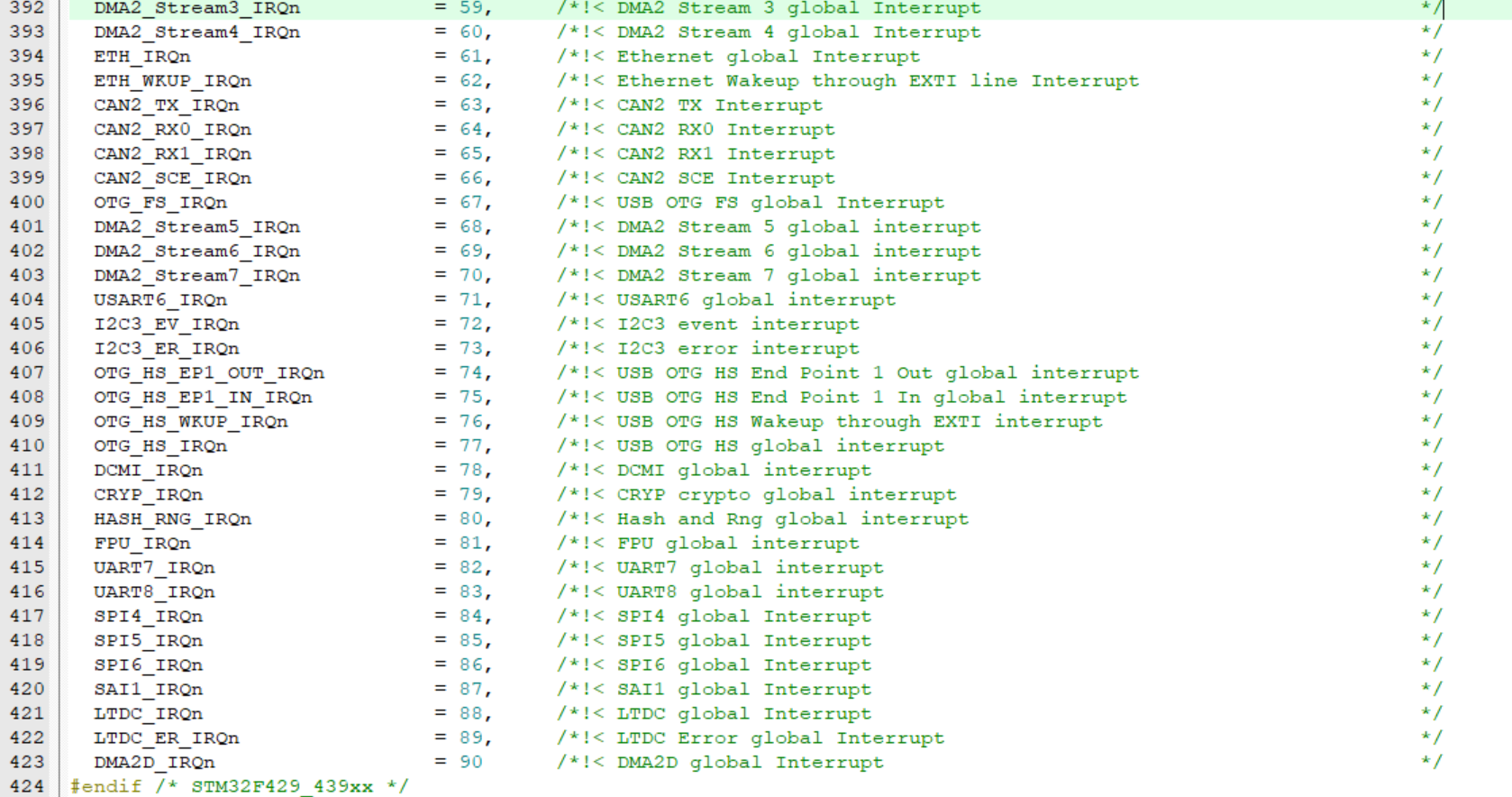
**（1）中断基本知识：**

****

其中F4系列的全部异常声明如下：

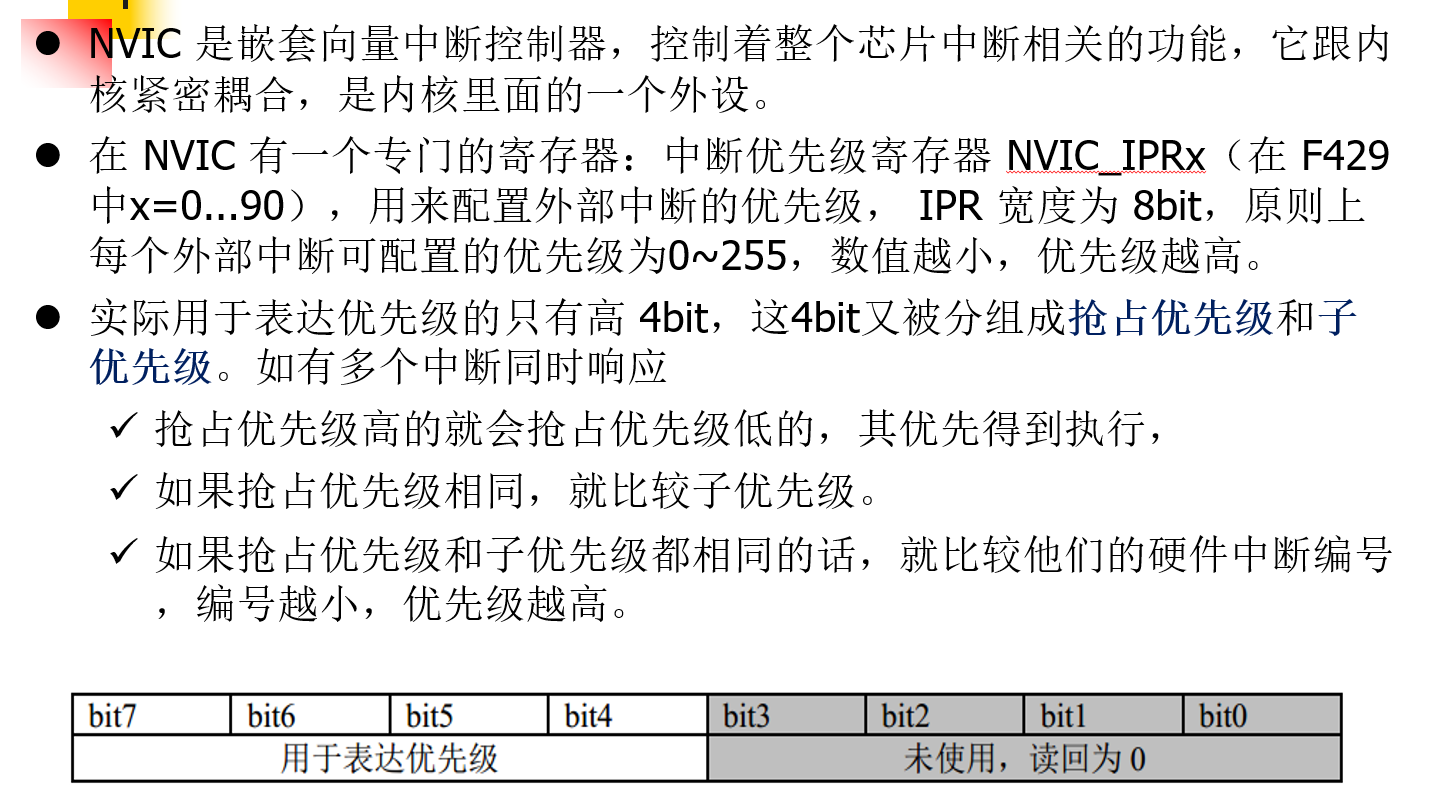






前八个为处理器异常，可以看出，这八个异常是所有异常中优先级最高的。

**（2）NVIC介绍：**

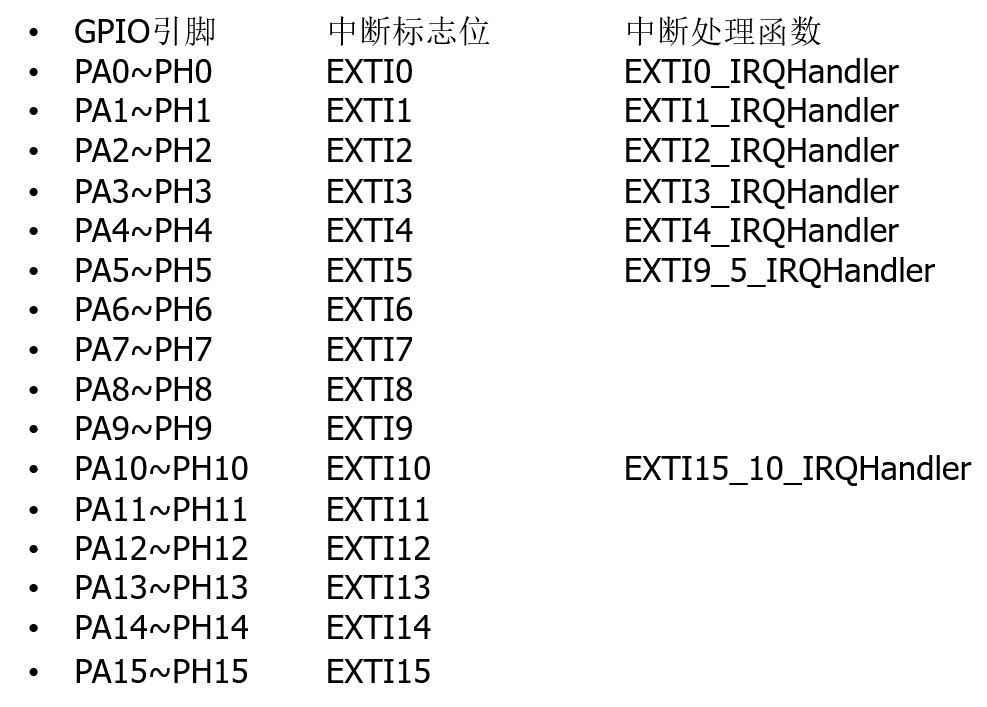
****

其中抢占优先级和子优先级需要在NVIC中先初始化才能设置中断，其中由于这两个中断共享四位，可以根据这两种中断又分成五组优先级：



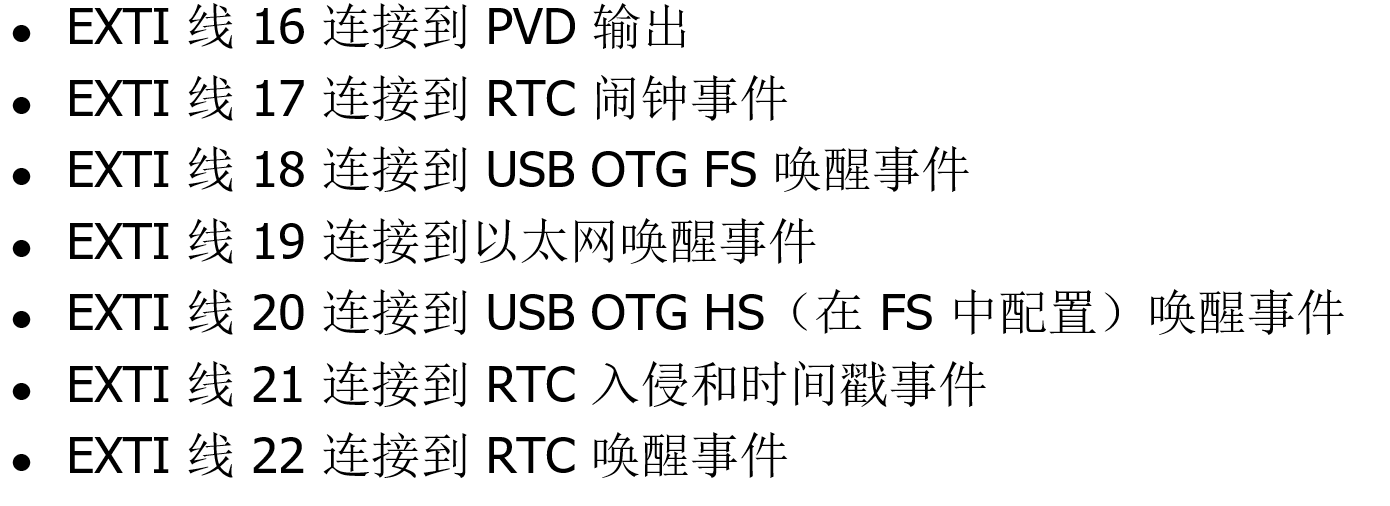
**（3）中断编程：**

由于对应的引脚需要使用对应的中断线进行处理，在编程前要搞清楚中断和引脚的对应关系，二者以及对应的中断处理函数如下：



其中EXIT9\_5\_TRQHandler对应中断线5到9的中断处理函数，EXIT15\_10同理。

除了上面的中断线以外，还有7根特殊的中断线：



1. **使用环境**

调用dxdiag工具：

Operating System: Windows 11 家庭中文版 64-bit (10.0, Build 22621) (22621.ni\_release.220506-1250)

Language: Chinese (Simplified) (Regional Setting: Chinese (Simplified))

System Manufacturer: HP

System Model: HP Pavilion Aero Laptop 13-be2xxx

BIOS: F.13 (type: UEFI)

Processor: AMD Ryzen 5 7535U with Radeon Graphics (12 CPUs), ~2.9GHz

Memory: 16384MB RAM

Available OS Memory: 15574MB RAM

Page File: 27604MB used, 5685MB available

Windows Dir: C:\WINDOWS

DirectX Version: DirectX 12

DX Setup Parameters: Not found

User DPI Setting: 144 DPI (150 percent)

System DPI Setting: 192 DPI (200 percent)

DWM DPI Scaling: UnKnown

Miracast: Available, with HDCP

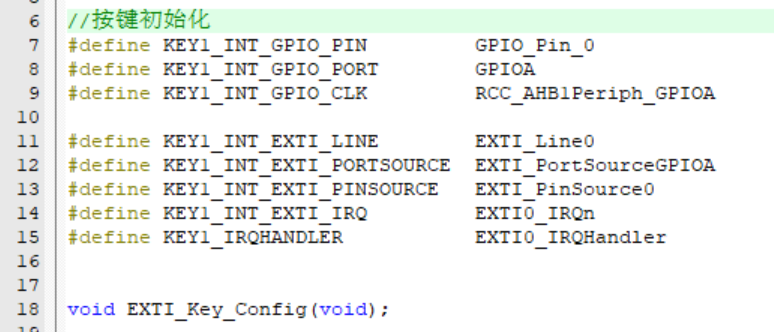
Microsoft Graphics Hybrid: Not Supported

1. **主要实验内容和结果展示**

本次实验使用的是上节课的模板，需要改动的地方较多，尤其是bsp\_key.c和bsp\_key.h, 在这个实验里我将这两个文件作为bsp\_exit.c和bsp\_exit.h进行修改，包括按键的初始化等操作。

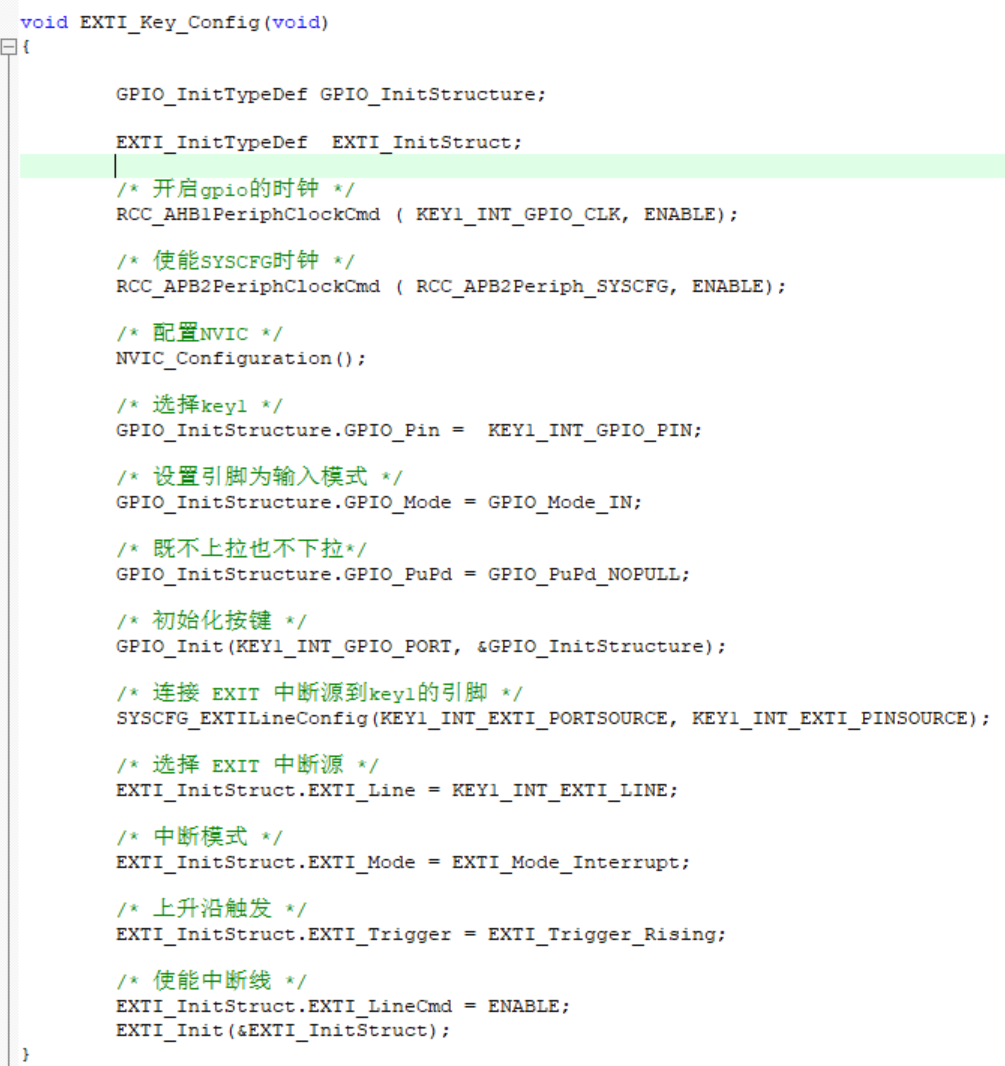
1. **示例实验：**

按照ppt上的教程，先要在头文件（bsp\_exit.h，我没有改上一次课的文件名，实际上是bsp\_key.h）对按键进行映射：

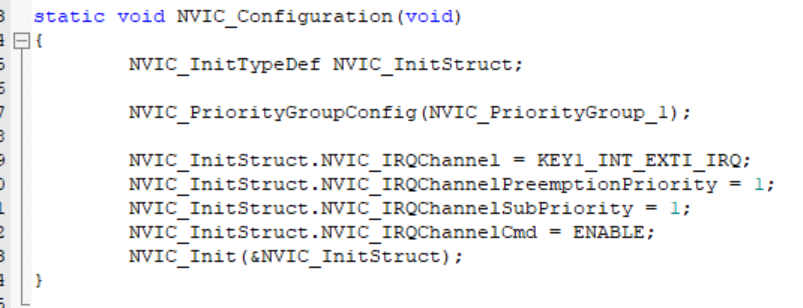


其中由于按键key1对应的GPIO引脚为0，端口为A，根据前一部分的引脚对应关系图，应选择中断线0，中断线的源端口应选择A，对应的中断处理函数为中断线0处理函数，另外，还需要写上中断和按键引脚的初始化函数的声明，在.c文件中写实现。

下面是该初始化函数的实现：



除了上节课对GPIO端口初始化外，我们加上了对中断线的初始化，以及让GPIO端口映射到中断线的代码，其中最需要关注的是中断控制器的初始化（NVIC）函数：



该函数对NVIC的四个要素（中断源IRQChannel， 抢占优先级Preemption，子优先级Sub，中断使能IRQChannelCmd）进行了初始化，当时看到ppt提供的函数时，我很疑惑为什么该函数加了个static的静态变量，便查了一下，发现该函数实际上只有该文件使用，如果其他文件初始化了别的NVIC（使用相同的函数名），不加static，就会链接错误。

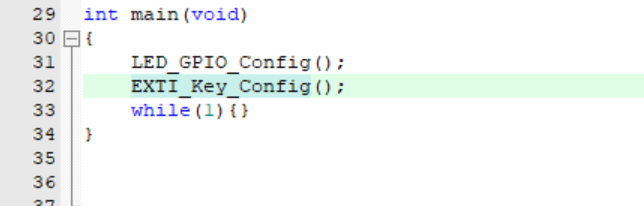
接着就需要编写中断处理函数：



该函数在stm32f4xx\_it.c中编写，虽然中断函数在startup\_stm32f429\_439xx.s中已有，但是是空实现，而自己实现的中断处理函数一般写在stm32f4xx\_it.c中。

这个函数中的EXTI\_GetITStatus用来获取EXTI 0的中断标志位状态，用于检查是否真的产生了中断，而EXTI\_ClearITPendingBit用于清除标志位，表示本次中断处理完毕。

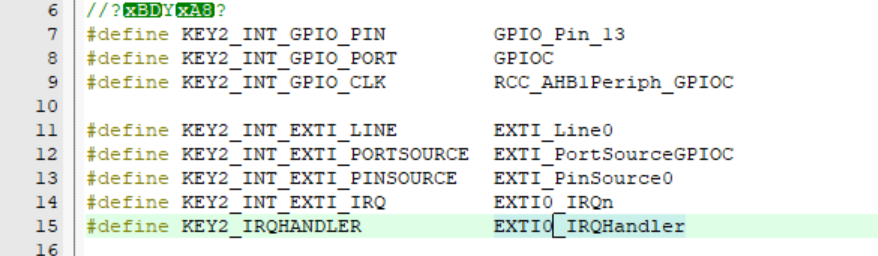
最后就是主函数的编写：



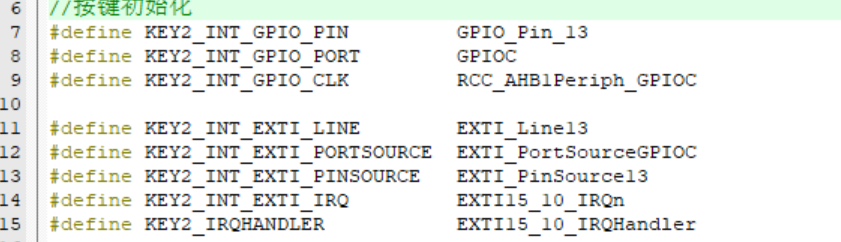
显然，主函数只需要在所有东西初始化（led灯和中断）后忙等中断发生即可。

演示视频见提交文件。

1. 利用中断服务，通过key2按键，来控制红灯、绿灯、蓝灯的轮流亮。即按一次按键2，红灯亮，在按一次，绿灯亮、在按一次，蓝灯亮，在按一次红灯亮……**：**

该题要求使用key2按键，首先要先改变bsp\_key.h中的宏定义，刚开始做时，我并没有意识到中断线和引脚端口的强对应性，没有改中断线编号：

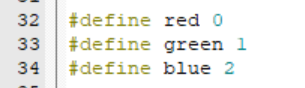
使我不论怎么debug都无法使用key2控制灯的亮灭，后面又仔细看了一遍ppt，才发现引脚和中断线是有对应关系的，改了过来：



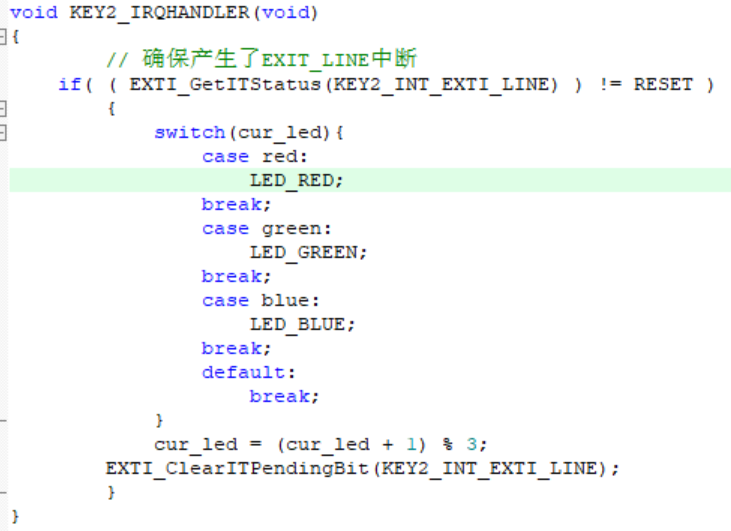
EXIT15\_10\_TRQHandler表示的是中断线10到15的中断处理函数。

另外，为了使红绿蓝交替亮，而所有关于灯的变量的定义都位于bsp\_led.c中，我便在其中设置了一个int变量cur\_led表示当前led的颜色，同时软编码了led的红绿蓝颜色，方便后面写代码：



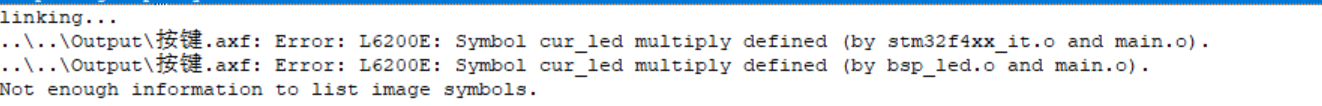
（这三条语句写在bsp\_led.h中）

接着修改中断服务函数如下：



通过每次中断发生时改变为cur\_led的颜色（switch语句），并增加cur\_led（后面的赋值语句）的值实现红绿蓝的循环亮。

在实现的过程中由于对c语言引用外部变量不熟悉，在第一次编译时得到如下报错：



（我第一次使用的时候是将cur\_led的定义写在了bsp\_key.h中）

后面查了相关资料发现还要在中断函数所在文件（stm32f4xx\_it.c）里加上



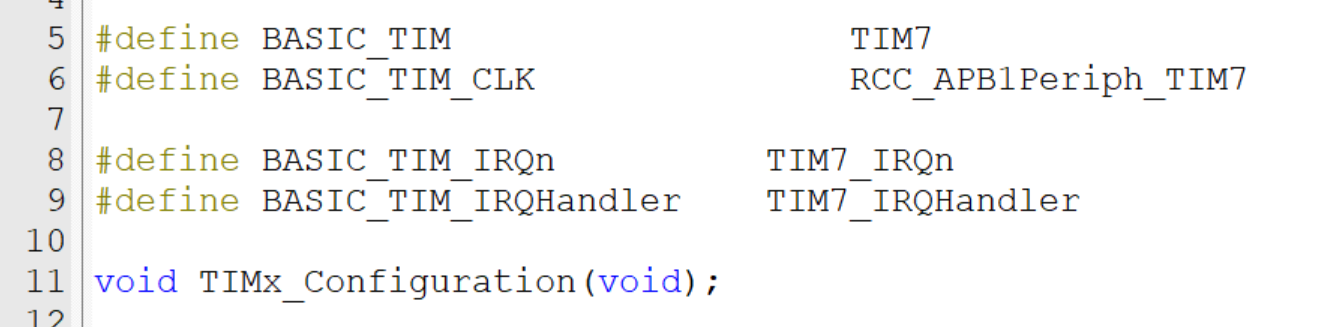
并将头文件的定义挪到源文件里，至此，实验成功。

演示视频见提交文件。

1. 利用中断机制，使得红灯0.5s翻转一次：

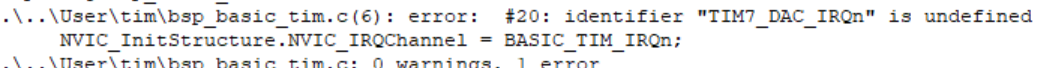
此处需要注意的是，ppt所给代码实际上是tim6的中断实现，应改为tim7的中断实现。

首先是定时器的头文件定义：



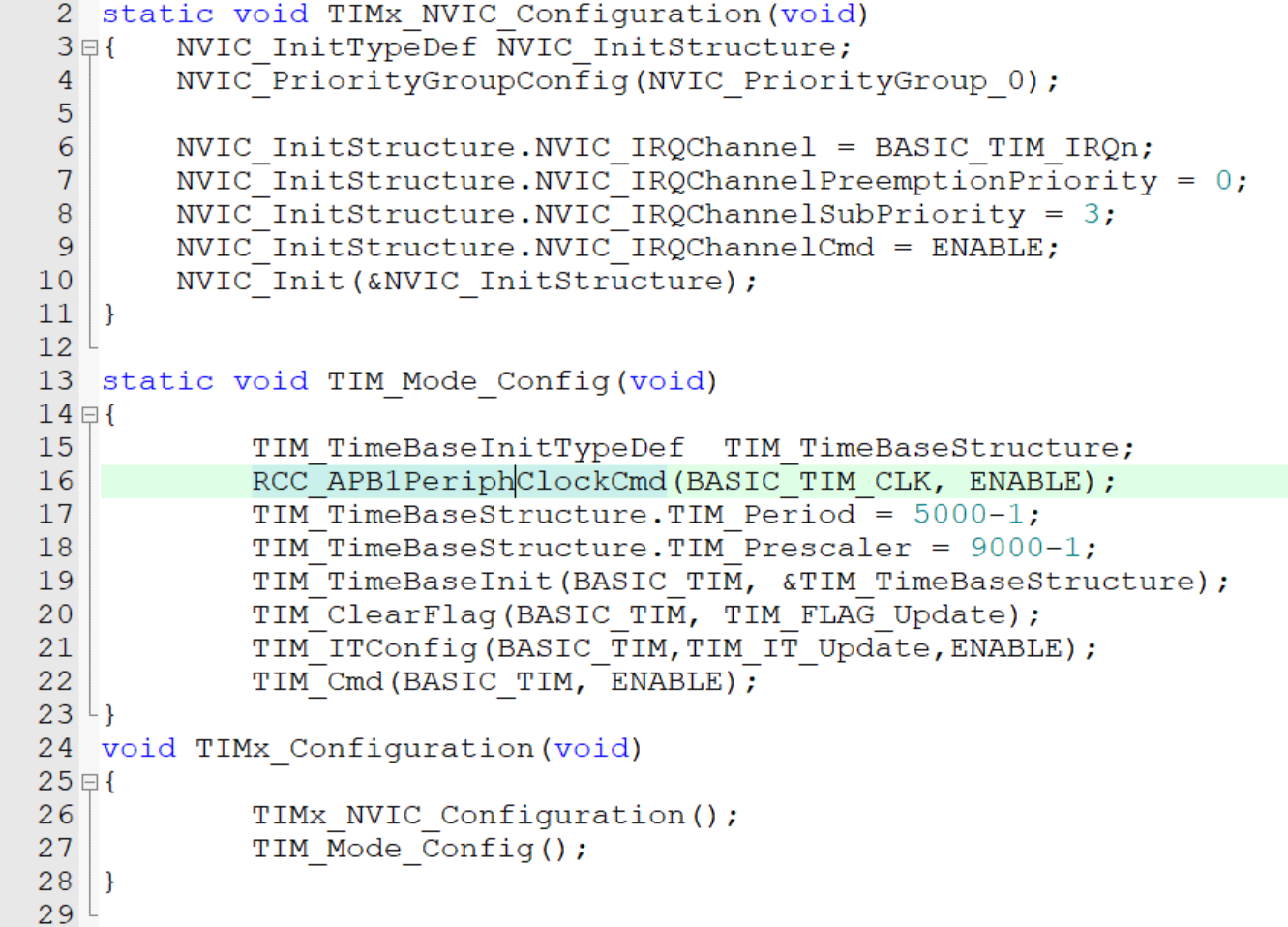
相比于按键的头文件定义要简洁不少，第一行表示使用的是基本定时器tim7（基本定时器只有两个，tim6和tim7），第二行表示的是tim7使用的时钟源， 第三行则是定时器的中断源，第四行是定时器的中断处理函数，最后一行是定时器中断的初始化函数声明。

在改成tim7时，由于对tim7中断函数的名字不熟悉，产生如下报错：



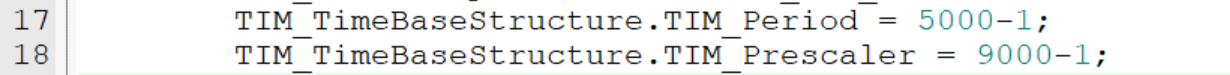
后面查看库函数定义才发现tim7的中断函数名字里并没有DAC，改后即可运行。

接着是具体的初始化部分：



由于定时器中断和按键中断都是中断，二者都初始化了中断控制NVIC，与按键中断不同的是，定时器中断使用的中断优先组号为0，表示不能进行中断的嵌套。

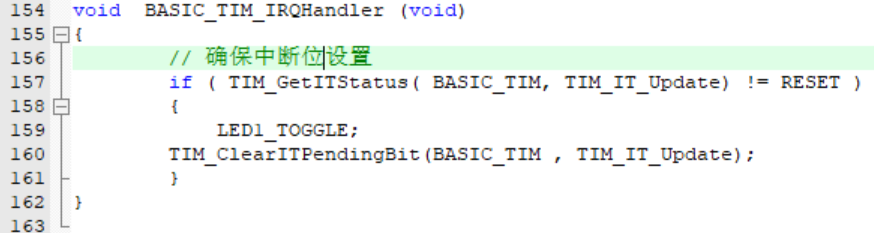
其他需要解释的是定时器的预分频概念，用来决定定时器隔多久发出一次中断，对应到代码中的这两行：



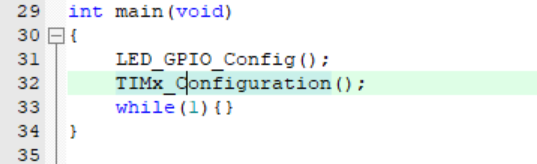
由于时钟源频率为90MHz，即一个时钟持续1/90M秒，为了得到先得到100微秒（ppt中为100微秒，便沿用）的间隔（即预分频，先分一次频率，再根据这个频率得到具体的时间间隔），（100微秒） 除以 （1/90M秒） 得到 9000，即需要多少个时钟周期得到100微秒，我们的预分频就为9000，同时由于时钟是从0开始计时，实际的初始化值应为9000-1（当时看到代码时感觉非常奇怪为什么是8999，便查了一下资料），而为了得到0.5s的周期，需要5000\*100微秒，所以分频的大小为5000-1（减1的原因同上），这样就得到了0.5s的中断周期。

这个部分是我花时间最多的试图搞懂的一个部分，由于这个TIM\_Period和TIM\_Prescaler的值设置为0~65535(都是16位寄存器)，这个分频不是随便设置的，当时我就在想预分频为什么不设置为1ms，这样不就更方便设置秒级的数据了吗，后来发现这样预分频就得是90000，超出了寄存器限制。

而中断函数部分与按键中断大同小异，只不过调用的检查中断位是否设置的函数以及清除标志位的函数不同：



接下来是主函数：

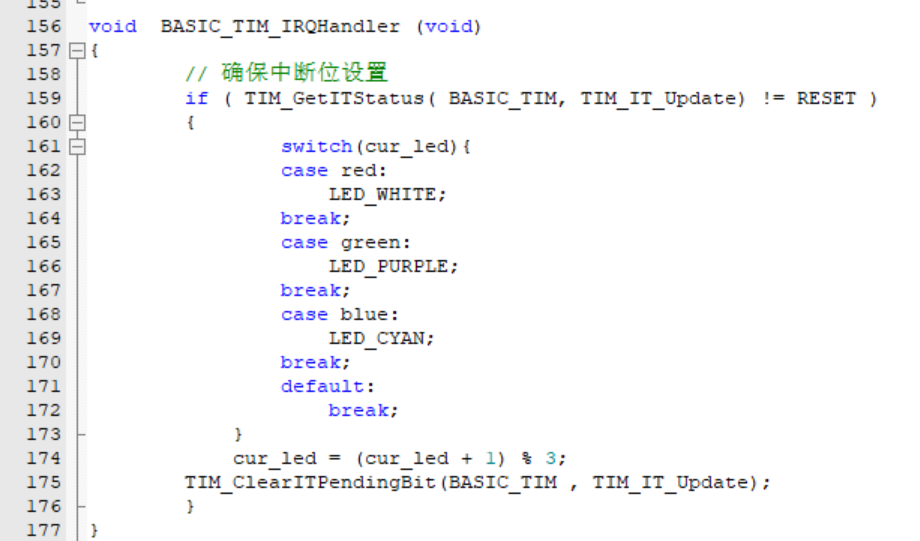


将按键中断初始化换为定时器中断初始化即可。

演示视频见提交文件。

4. 参考所给项目，让白灯先亮6s，后熄灭，此时紫灯亮6s，后熄灭，在青灯亮6s后熄灭，如此往复。(自己测试是否正确可以拿出自己手机的秒表做一个近似估计)

这个实验和之前的实验2很像，有了实验2的经验，直接修改定时器的中断处理函数就行：



思路与实验2的完全一样，通过一个全局变量cur\_led控制灯的颜色，在每 次中断发生后更改cur\_led的值，使下一次产生的颜色发生改变。

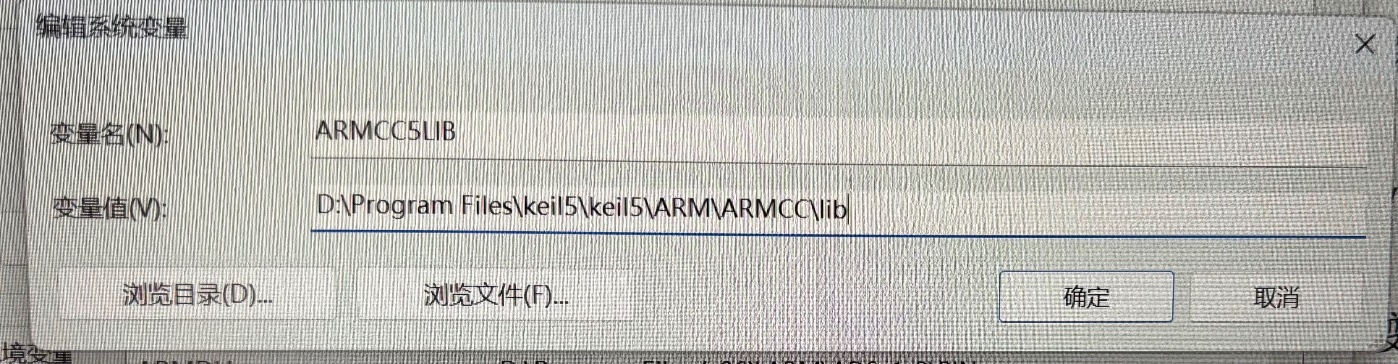
演示视频见提交文件（视频正好18秒，一个周期，白紫青）。

1. 其他

在上面的实验完成前，由于主机安装了ads环境，与keil5环境冲突，编译时有如下报错：

..\..\Output\按键.axf: Error: L6411E: No compatible library exists with a definition of startup symbol \_\_main.

通过查找资料，在系统环境变量中将ads的系统路径删除并在系统变量中加上如下语句：



（打开高级系统设置后截图键的中断好像失效了，于是使用手机拍的）

即可解决该环境冲突错误。

**五、实验总结**

本次实验让我对硬件中断有了一个初步但较为全面的的认识，尤其是做实验的过程带来的各种问题，有一些是自己看不懂向网上找答案的，也有一些是对语言知识点的不熟悉导致的编程错误。我上次的实验试图做出一个中断相关的项目，但是没有做出来，这次算是理解了上次实验中自己的错误。除中断之外，我还学会了定时器的使用，以及定时器的预分频和设置时间间隔的原理，在反复研读ppt的过程中，也对NVIC这个中断控制器有了一定的了解，每个中断，无论是按键的还是定时器的，都需要初始化这个来实现中断，而且中断线和GPIO端口是有对应关系的，不能随便使用。