Homework of Lecture 4

1. LED turns and off alternately

```c
#include "main.h"

#include "led.h"

void Main() {
    LED_GPIO_Config();

    while (1) {
        LED_ON();
        for (int i = 0; i < 540000; i++);
        LED_OFF();
        for (int i = 0; i < 360000; i++);
    }
```

使用暴力延迟交替闪烁（

2. Press the key to control the change between LED on and off

```c
void Main() {
    LED_GPIO_Config();

    while(1)
    {
        KEY_Scan();
        if(flag == 1)
        {
            LED_OFF();
        }
        if(flag == 0)
        {
            LED_ON();
        }
    };
```

在main里感知按键。

```c
void KEY_Scan()
{
    if(PCin(13) == 0)
    {
        for (int i = 0; i < 720000; i++);
        if(PCin(13) == 0)
        {
            if(flag == 0)
            {
                flag = 1;
            }else if(flag == 1)
            {
                flag = 0;
            }

        }
        while(!PCin(13));
    }
```

设置按键扫描函数。

以下是按键外设的初始化：

```c
void KEY_GPIO_Config(void)
{
    GPIO_InitTypeDef GPIO_InitStruct;

    RCC_APB2PeriphClockCmd( RCC_APB2Periph: KEY_GPIO_CLK,  NewState: ENABLE);
    GPIO_InitStruct.GPIO_Pin = KEY_GPIO_PIN;
    GPIO_InitStruct.GPIO_Mode = GPIO_Mode_IPU;
    //GPIO_InitStruct.GPIO_Speed = GPIO_Speed_50MHz;

    GPIO_Init( GPIOx: KEY_GPIO_PORT,  GPIO_InitStruct: &GPIO_InitStruct);

}
```

Homework of Lecture 5

1. Sort out the process of *NVIC configuration*

步骤：

1. 设置NVIC 权限组，有0~4一共5个组；

   ```c
   NVIC_SetPriorityGrouping(uint32_t PriorityGroup);
   ```

2. 构造`NVIC_InitStruct`，选择频道；

```
NVIC_InitTypeDef NVIC_InitStruct;
NVIC_InitStruct.NVIC_IRQChannel = uint8_t NVIC_IRQChannel;
```

3. 设置抢占优先级；

```
NVIC_InitStruct.NVIC_IRQChannelPreemptionPriority = uint8_t
NVIC_IRQChannelPreemptionPriority;
```

4. 设置响应优先级；

```
NVIC_InitStruct.NVIC_IRQChannelSubPriority = uint8_t
NVIC_IRQChannelSubPriority;
```

5. 使能；

```
NVIC_InitStruct.NVIC_IRQChannelCmd = ENABLE;
```

6. 启动初始化。

```
NVIC_Init(&NVIC_InitStruct);
```

其中，分组关系如下：

组	AIRCR[10：8]	IP bit[7：4]分配情况	分配结果
0	111	0：4	0位抢占优先级，4位响应优先级
1	110	1：3	1位抢占优先级，3位响应优先级
2	101	2：2	2位抢占优先级，2位响应优先级
3	100	3：1	3位抢占优先级，1位响应优先级
4	011	4：0	4位抢占优先级，0位响应优先级

抢占优先级与响应优先级遵循的原则：

1. 高优先级的抢占优先级是可以打断正在进行的低抢占优先级中断的；
2. 抢占优先级相同的中断，高响应优先级不可以打断低响应优先级的中断；
3. 抢占优先级相同的中断，当两个中断同时发生的情况下，哪个响应优先级高，哪个先执行；
4. 如果两个中断的抢占优先级和响应优先级相同，则看哪个中断先发生就先执行；
5. 优先级数字越小，优先级越高，越先被执行。

2. Modify the project "定时器中断（需修改）" & Record what have been modified

Edited places:

1.

```
void KEY_Scan()
{
    if(PCin(13) == 0)
    {
        delay_ms(5);
        if(PCin(13) == 0)
        {
            if(flag == 0)
            {
                flag = 1;
            }else if(flag == 1)
            {
                flag = 0;
            }

        }
        while(!PCin(13));
    }

}
```

key 这里需要修改端口，虽然好像没用到.....

所以我们让它起点作用：

```
while(1)
{
    KEY_Scan();
    if(flag == 1)
    {
        TIM_Cmd(TIM3, DISABLE);
        GPIO_SetBits(LED_GPIO_PORT, LED_GPIO_PIN);
    }
    if(flag == 0)
    {
        TIM_Cmd(TIM3, ENABLE);
    }
};
```

添加main中对于scan的响应；

```
void KEY_GPIO_Config(void)
{
    GPIO_InitTypeDef GPIO_InitStruct;

    RCC_APB2PeriphClockCmd(KEY_GPIO_CLK, ENABLE);
    GPIO_InitStruct.GPIO_Pin = KEY_GPIO_PIN;
    GPIO_InitStruct.GPIO_Mode = GPIO_Mode_IPU;
    //GPIO_InitStruct.GPIO_Speed = GPIO_Speed_50MHz;

    GPIO_Init(KEY_GPIO_PORT, &GPIO_InitStruct);

}
```

发现这里的端口没有定义，文件树中没有key.h, 于是尝试定义一下，发现重复定义。点击跳转发现已经有key.h文件了，于是修改错误的端口：

```
#define KEY_GPIO_PIN   GPIO_Pin_13
#define KEY_GPIO_PORT  GPIOC
#define KEY_GPIO_CLK  RCC_APB2Periph_GPIOC
```

2.

```
NVIC_SetPriorityGrouping(NVIC_PriorityGroup_0);

NVIC_InitStruct.NVIC_IRQChannel = TIM3_IRQn;
NVIC_InitStruct.NVIC_IRQChannelPreemptionPriority = 0;
NVIC_InitStruct.NVIC_IRQChannelSubPriority = 2;
NVIC_InitStruct.NVIC_IRQChannelCmd = ENABLE;

NVIC_Init(&NVIC_InitStruct);
TIM_Cmd(TIM3, ENABLE);
```

给中断优先级分组，然后将主权限组调整至上限a0.

3.

```
/*
void LED_Change()
{

    static u8 i =0;
    j++;
    if(flag == 1)
    {
        flag = 0;
        switch (i)
        {
            case 0:PAout(8) = 1;i++;break;
            case 1:PAout(8) = 0;i=0;break;
        }
    }

}*/
```

这段代码的端口不对，但是没有用，直接写在main里了，所以注释掉了。

4.

```
#ifndef __LED_H__
#define __LED_H__

#include "sys.h"

#define LED_GPIO_PIN   GPIO_Pin_5
#define LED_GPIO_PORT  GPIOB
#define LED_GPIO_CLK   RCC_APB2Periph_GPIOB

void LED_GPIO_Config(void);
//void LED_Change(void);

#endif
```

端口不对，改掉了。

现在可以闪烁并使用按键2开关灯了。