第七章作业

1、将基础任务中引脚 PC13 的外部中断触发方式修改为上升沿触发，完成按键检测程序的编写，观察指示灯状态变化的时刻，并分析原因。

实现代码（8 分）：

```
/* USER CODE BEGIN 4 */
/*外部中断回调函数*/
HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    /*按键引脚名称命名为BTN_USER，指示灯引脚命名为LD2*/
    if(GPIO_Pin == BTN_USER_Pin) /*判断是哪个按键产生的中断信号*/
    {
        HAL_GPIO_TogglePin(LD2_GPIO_Port,LD2_Pin);/*翻转指示灯状态*/
    }
}
/* USER CODE END 4 */
```

原因分析（2 分）：

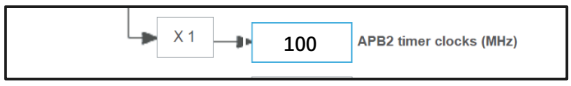
现象是指示灯在松手后才进行状态翻转。NUCLEO 开发板上的用户按键未按下时 MCU 上对应的引脚为高电平，反之则为低电平。用户从按下到释放按键，引脚的电平变化为：高→低→高。又因为中断触发模式设置为上升沿（低→高）变化时触发，故指示灯在按键释放时才进行状态翻转。

2、 使用一个 GPIO 引脚输出一个 10Hz 的方波来模拟外部中断信号。利用杜邦线将该引脚与 PC13 引脚连接，设置 PC13 引脚为双边沿触发，在中断中执行翻转指示灯 LD2 状态的操作。完成该程序的编写，并观察记录指示灯的变化情况，分析变化的原因。

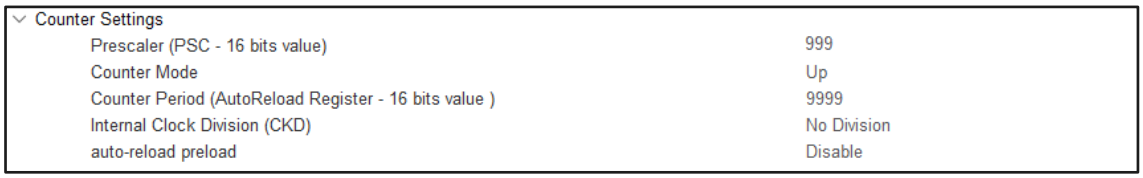
实现代码（8 分）：

外部中断代码与第 1 题一致，不做更改。产生 10Hz 方波的方式有多种，下面介绍一种作为参考。

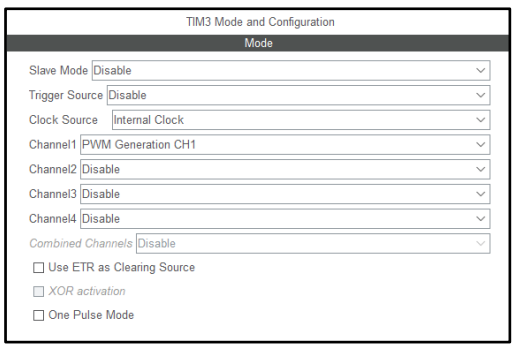
a） 已知定时器的时钟频率为：



b） 故为获取 10Hz 的方波，将计数器设定为如下参数（参数不唯一）



c） 这里用定时器输出 PWM 波来模拟方波，使用定时器 3 的通道一输出 PWM，对应引脚为（PA6）



d) 将 PWM 的比较值设置为计数器重载值的一半，以此产生占空比为 50%的 PWM 波，即所需的方波。

e) 在初始化代码的后面添加 PWM 启动代码。

```
/* USER CODE BEGIN 2 */
HAL_TIM_PWM_Start(&htim3, TIM_CHANNEL_1);/* 启动定时器3的通道一PWM输出 */
/* USER CODE END 2 */
```

原因分析（2 分）：

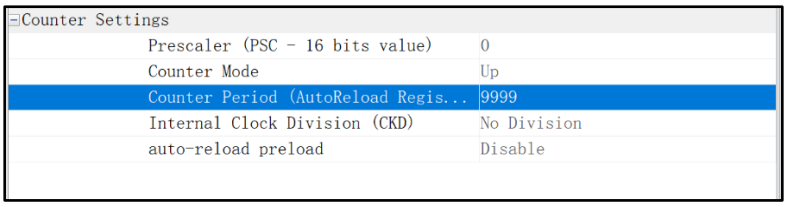
现象是指示灯快速闪烁，1 秒约闪烁 10 次。原因分析，方波一个完整周期中含有一个上升沿和一个下降沿。又因为设置为上升沿和下降沿都触发中断，故一个方波周期指示灯状态翻转 2 次，即闪烁 1 次。又因为该方波一个周期为 100ms，故每 1 秒闪烁 10 次。

第八章作业一

1、使用定时器 10 产生 1s 的定时，定时器时钟 TIM_CLK 为 100M。要求不使用预分频寄存器，请编程实现。

实现代码：

a) TIM10 的自动重载寄存器设置如下：



b) 在初始化函数后调用以下函数，激活 TIM10 的中断模式。

```
/* USER CODE BEGIN 2 */
HAL_TIM_Base_Start_IT(&htim10);//激活定时器10中断模式
/* USER CODE END 2 */
```

c) 定义一个变量用于计数：

```
/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */
uint32_t Tim10Counter =0;//定时器计数变量
/* USER CODE END PM */
```

d) 添加中断回调函数，并在函数里面添加定时后要执行的操作，这里是翻转指示灯：

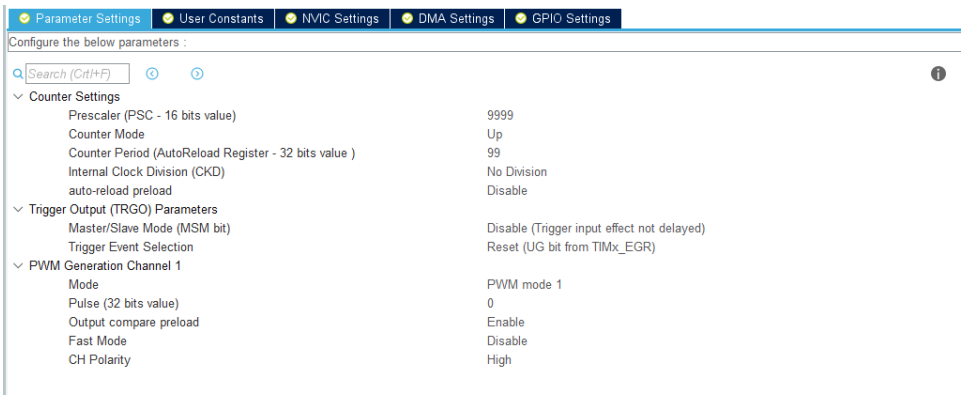
```
/* USER CODE BEGIN 4 */
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
    if(htim->Instance==TIM10)/*判断是否为定时器10产生的中断*/
    {
     Tim10Counter++;/*变量自加，向上计数*/
     if(Tim10Counter>10000)/*计数大于设置值，因为中断是每0.1ms进入，故当计数达10000次时刚好为1s*/
     {
         Tim10Counter=0;/*计数变量清零，下一周期开始*/
         HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin);/*翻转指示灯*/
     }
    }
}
/* USER CODE END 4 */
```

第八章作业二

1、修改呼吸灯程序，实现指示灯从暗到亮又从亮到暗的渐变，并重复该过程。

实现代码：

a） 时钟配置：



b） 代码：

```
/* USER CODE BEGIN WHILE */
while (1)
{
  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
  for(Duty=0;Duty<100;Duty++)
  {
      HAL_Delay(10);
      __HAL_TIM_SET_COMPARE(&htim2,TIM_CHANNEL_1,Duty);/*修改占空比 0% → 100%*/
  }
  for(Duty=0;Duty<100;Duty++)
  {
      HAL_Delay(10);
      __HAL_TIM_SET_COMPARE(&htim2,TIM_CHANNEL_1,100-Duty);/*修改占空比 100% → 0%*/
  }

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

第九章作业一

1、利用串口通信的轮询方式，实现对 Nucleo 开发板上指示灯 LD2 的控制。发送小写字符“o”，开启指示灯 LD2，发送小写字符“c”，关闭指示灯 LD2。

实现代码：

```
while (1)
{
  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
 HAL_UART_Receive(&huart2,&character,1u,HAL_MAX_DELAY);/*无限等待串口通信*/
 switch (character) {
  case 'o': HAL_GPIO_WritePin(LD2_GPIO_Port,LD2_Pin, GPIO_PIN_SET);/*GPIO置高，亮灯*/
      break;
  case 'c': HAL_GPIO_WritePin(LD2_GPIO_Port,LD2_Pin, GPIO_PIN_RESET);/*GPIO置低，灭灯*/
      break;
  default:
      break;
    }
 }
/* USER CODE END 3 */
```

第九章作业二

2、 在进阶任务“实现简单的帧格式通信”中增加一个设备码：0x02 表示按键，功能码：0x01 表示获取按键的状态，请编程实现。

实现代码：

a) 添加、定义一个 GPIO_PinState 类型的变量

```
GPIO_PinState pinState;
```

b) 在原有的代码基础上增加：

```
else if(RxBuffer[1]==0X02)/*判断设备码，按键*/
{
    if(RxBuffer[2]==0x01)/*判断功能码，获取按键状态*/
    {
        pinState=HAL_GPIO_ReadPin(BTN_USER_GPIO_Port, BTN_USER_Pin);/*获取某个用户按键的状态*/
    }
}
```