МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

ІКНІ
Кафедра **ПЗ**

ЗВІТ

До лабораторної роботи № 2
З дисципліни: *“ Основи програмування вбудованих систем”*
На тему: *“ Робота з перериваннями на прикладі кнопки на платі STM32F4DISCOVERY ”*
Варіан 1

Лектор:
доц. каф. пз.
Марусенкова Т.А.

Виконав:
ст. гр. ПЗ-32
Бурець В.В.

Прийняв:
доц. каф. пз.
Крук О.Г.

« ____ » _____ 2021 р.

∑= _____ _____.

Львів-2021

Тема роботи: Робота з перериваннями на прикладі кнопки на платі STM32F4DISCOVERY.

Мета роботи: Ознайомитися з регістрами для конфігурації переривань, навчитися обробляти зовнішні переривання та читати технічну документацію

ТЕОРЕТИЧНІ ВІДОМОСТІ

1. Що таке переривання?

Це сигнал, який повідомляє МК про настання якоїсь події, що потребує його негайної реакції. Коли трапляється переривання, мікроконтролер «відволікається» від виконання основної програми і обробляє переривання, після чого продовжує виконання основної програми з тієї самої точки, з якої «відволікся». Обробити переривання – значить викликати відповідний обробник, написати який, звісно, є задачею самого програміста. Якщо програма-обробник відсутня, мікроконтролер використає обробник за замовчуванням.

Завдання

Варіант 1

1. При кожному натисненні користувацької кнопки збільшити частоту блимання світлодіодів на 10%, коли частота зросте удвічі, почати зворотний процес (зменшення частоти на 10%).

ХІД РОБОТИ

Для виконання встановленого завдання потрібно виконати наступні дії.
Підключити бібліотеки для роботи із відповідними функціями і структурами

```
#include "stm32f4xx.h"
#include <stm32f4xx_rcc.h>
#include <stm32f4xx_gpio.h>
#include <misc.h>
#include <stm32f4xx_syscfg.h>
#include <stm32f4xx_exti.h>  //contain EXTI_InitTypeDef

RCC_ClocksTypeDef RCC_Clocks;
```

Провести ініціалізацію структури GPIO_InitTypeDef для роботи із кнопкою, тобто для створення щовнішнього переривання

```
void ButtonInit() {

        GPIO_InitTypeDef GPIO_InitStructure;
        GPIO_StructInit(&GPIO_InitStructure);
```

```
        RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOD, ENABLE);

        RCC_GetClocksFreq(&RCC_Clocks);
 SysTick_Config(RCC_Clocks.HCLK_Frequency / 1000)
}
```

Створюємо константу в якій задаємо піни з якими я буду працювати

```
const uint16_t LEDS = GPIO_Pin_12 | GPIO_Pin_13 | GPIO_Pin_14 | GPIO_Pin_15;
```

Провести ініціалізацію структури GPIO_InitTypeDef для роботи із світлодіодами, яким відповідають піни 12, 13, 14, 15

```
void leds_init() {
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOD, ENABLE);
    GPIO_InitTypeDef gpio;   //set parameters for user leds
    GPIO_StructInit(&gpio);
    gpio.GPIO_OType = GPIO_OType_PP;
    gpio.GPIO_Mode = GPIO_Mode_OUT;
    gpio.GPIO_Pin = LEDS;
    GPIO_Init(GPIOD, &gpio);
}
```

Провести ініціалізацію структур для роботи із перериваннями
Створити та ініціалізувати структуру
EXTI_InitTypeDef яка відповідає за зовнішні переривання
Створити та ініціалізувати структуру
NVIC_InitTypeDef яка є контроллером вкладених переривань

```
void it_init() {
                //set iterrupt port    port   / out port
   SYSCFG_EXTILineConfig(EXTI_PortSourceGPIOA,EXTI_PinSource0); //associate PA0 with EXTI0


                //external iterrupt
   EXTI_InitTypeDef exti;   //create struct for working with EXTI_IMR & EXTI_RTSR registers
   exti.EXTI_Line = EXTI_Line0;
   exti.EXTI_Mode = EXTI_Mode_Interrupt;    //EXTI_IMR ? EXTI_EMR means event or interrapts
   exti.EXTI_Trigger =   EXTI_Trigger_Rising_Falling;//EXTI_Trigger_Rising(Falling/Rizing_Falling)
   exti.EXTI_LineCmd = ENABLE;    //set state
   EXTI_Init(&exti);


        //nested vector interrupt controller
   NVIC_InitTypeDef nvic; // set params for contoller vector interrupts
   nvic.NVIC_IRQChannel = EXTI0_IRQn;   // set Chanal IRG for activation/disactivation
   nvic.NVIC_IRQChannelPreemptionPriority = 13;
   nvic.NVIC_IRQChannelSubPriority = 13;
   nvic.NVIC_IRQChannelCmd = ENABLE;  //set activity
   NVIC_Init(&nvic);
}
```

Код для створення затримки у блиманні світлодіодів

```
static __IO uint32_t startDelay = 1000;
static __IO uint32_t TimingDelay = 1000;
static __IO uint32_t _tempTimingDelay;
RCC_ClocksTypeDef RCC_Clocks;
```

```
void Delay(__IO uint32_t nTime)
{
  _tempTimingDelay = nTime;

  while(_tempTimingDelay != 0);
}

void TimingDelay_Decrement(void){
  if (_tempTimingDelay != 0x00){
    _tempTimingDelay--;
  }
}
void SysTick_Handler(void)
{
  TimingDelay_Decrement();
}

 uint32_t flag = 0;
```

Реалізація обробника переривань

```
void EXTI0_IRQHandler(void){

                        if(GPIO_ReadInputDataBit(GPIOA,GPIO_Pin_0) && TimingDelay*2 < startDelay)
                                flag = 1;
                        else if(GPIO_ReadInputDataBit(GPIOA,GPIO_Pin_0) && TimingDelay >= startDelay)
                                flag = 0;

                        if(flag)
                                TimingDelay *= 1.1;
                        else
                                TimingDelay *= 0.9;

                        EXTI_ClearITPendingBit(EXTI_Line0);

}
```

Написана головна функція у якій міститься виклик вище згаданих функцій та цикл для блимання світлодіодами

```
int main(){
   ButtonInit();
   it_init();
   leds_init();
                        while(1)
                        {
                                GPIO_ToggleBits(GPIOD, GPIO_Pin_12);
                                GPIO_ToggleBits(GPIOD, GPIO_Pin_13);
                                GPIO_ToggleBits(GPIOD, GPIO_Pin_14);
                                GPIO_ToggleBits(GPIOD, GPIO_Pin_15);

                                Delay(TimingDelay);

                                GPIO_ToggleBits(GPIOD, GPIO_Pin_12);
                                GPIO_ToggleBits(GPIOD, GPIO_Pin_13);
                                GPIO_ToggleBits(GPIOD, GPIO_Pin_14);
                                GPIO_ToggleBits(GPIOD, GPIO_Pin_15);
                        };
}
```

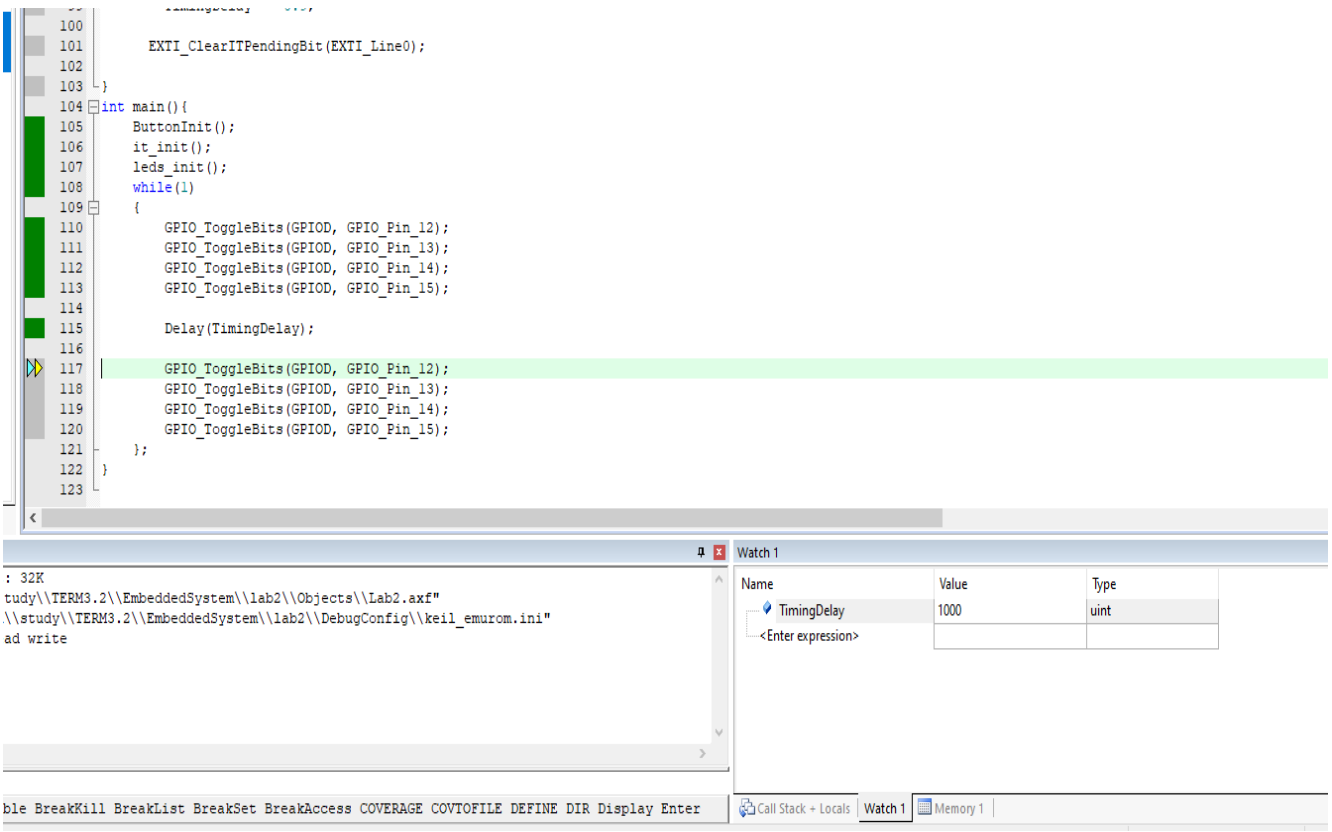

Рис.1 значення змінної TimingDalay(часу затримки) до переривання

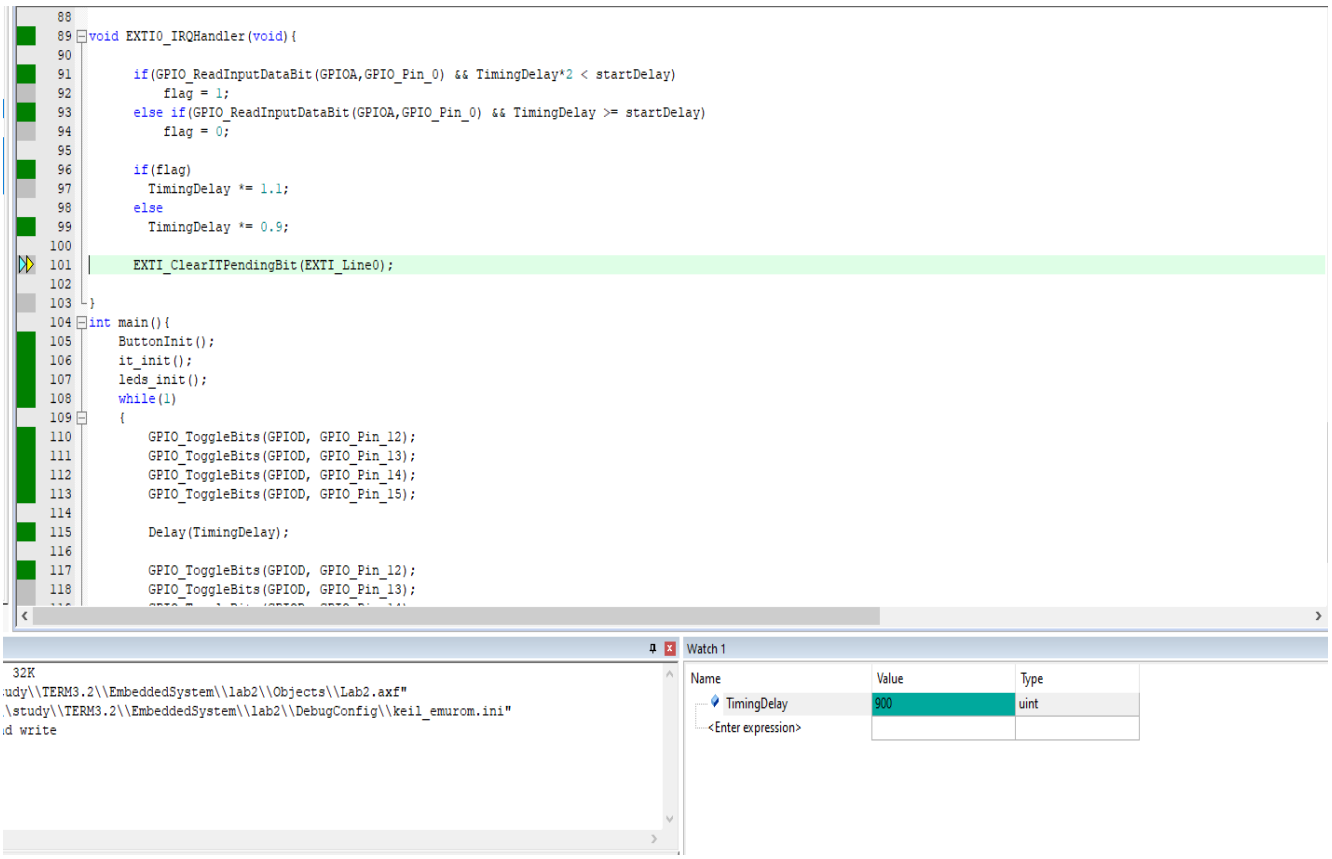

Рис.2 значення змінної TimingDalay(часу затримки) після переривання

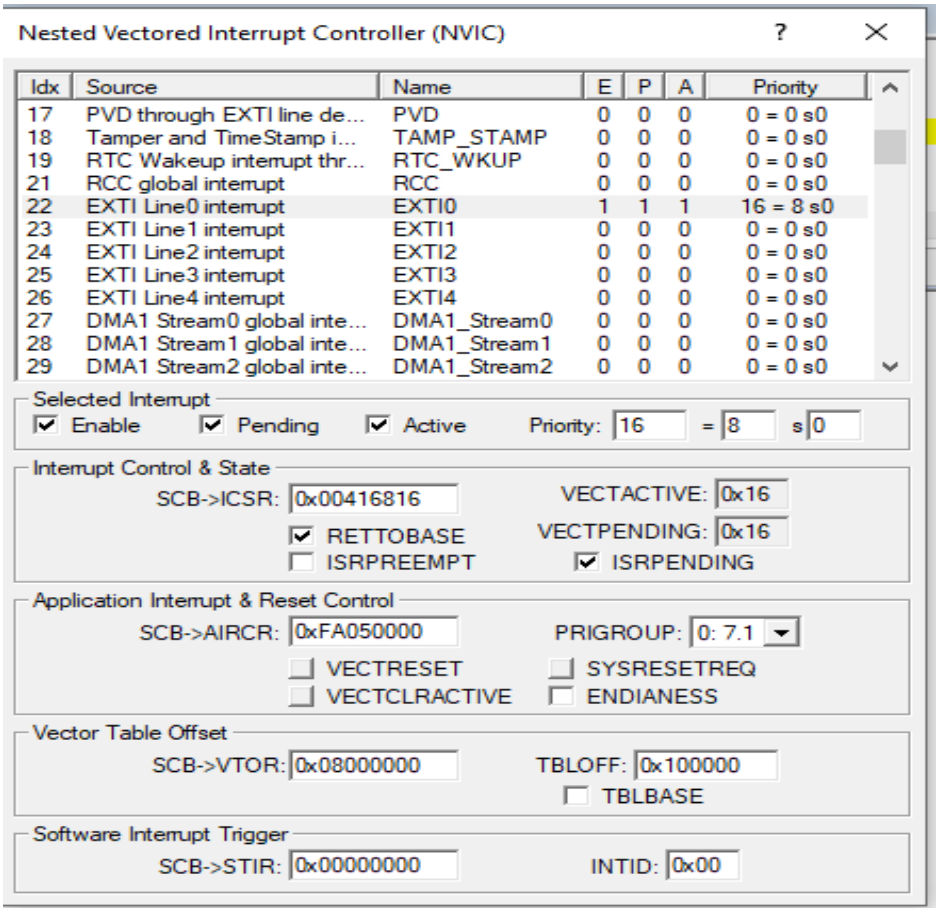

Рис.3 створення переривання

ВИСНОВКИ

На даній лабораторній роботі я ознайомився з регістрами для конфігурації переривань, навчився обробляти зовнішні переривання та читати технічну документацію. Написав програму для ініціалізації кнопки на ввід, та ініціалізації зовнішнього переривання, та виконав свій варіант. Створив переривання за допомогою інструментів середовища та відлагодив роботу програми.