**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

**TЕОРЕТИЧНІ 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);

};

}

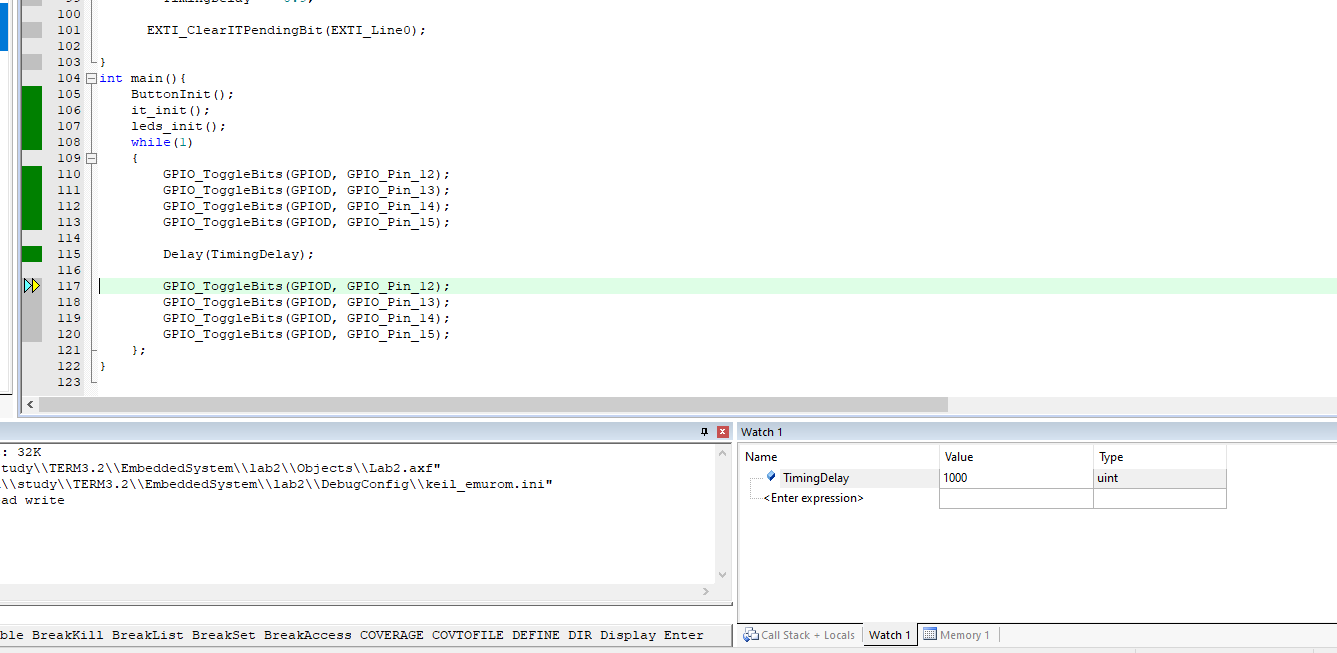


Рис.1 значення змінної TimingDalay(часу затримки) до переривання

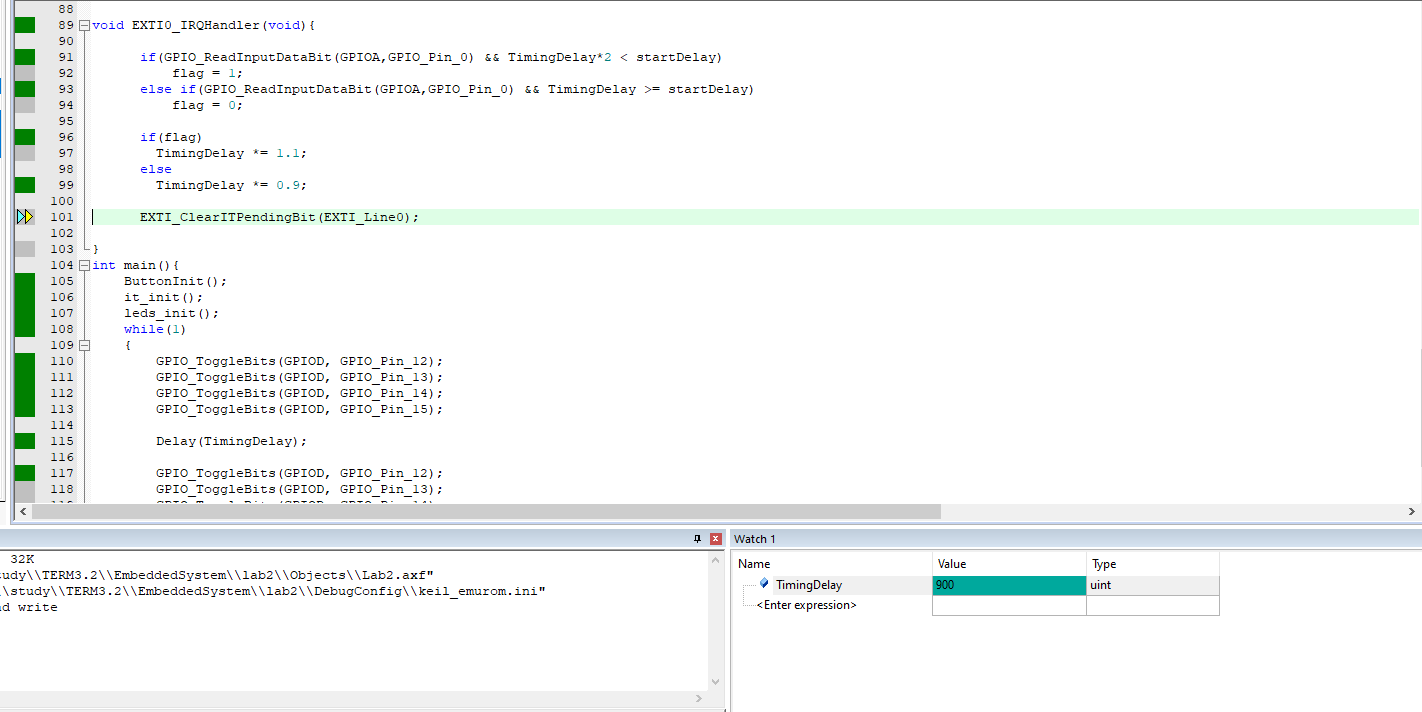


Рис.2 значення змінної TimingDalay(часу затримки) після переривання

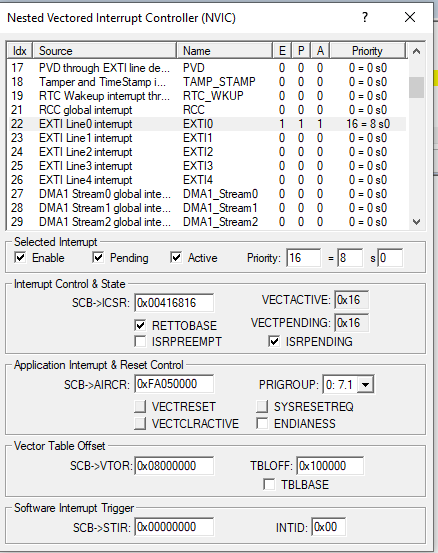


Рис.3 створення переривання

**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.