PreLAB: External Interrupt

Name:leejeayong

ID: 22000561

I. Introduction

In this tutorial, we will learn how to use External Interrupt. We will create functions that capture the falling edge trigger by pushing a button using an external interrupt.

The objectives of this tutorial are how to

- Configure External input (EXTI) interrupt with NVIC
- Create your own functions for configuration of interrupts

Hardware

- NUCLEO -F411RE

Software

- VS code, CMSIS, EC_HAL

Documentation

- STM32 Reference Manual

II.Basics of External Interrupt (EXTI)

A. Register List

List of external interrupt (EXTI) registers used in this tutorial [Reference Manual ch7, ch10.2]

B. Register Setting

(Digital Input Setting)

- Enable GPIO peripheral clock **RCC->AHB1ENR**
- Configure DigitalIn pin

(EXTI Setting)

- Enable SYSCFG peripheral clock. **RCC->APB2ENR**
- Connect the corresponding external line to GPIO **SYSCFG->EXTICR**
- Configure the trigger edge. **EXTI->FTSR/RTSR**
- Configure Interrupt mask **EXTI->IMR**
- Enable EXTI. **EXTI->IMR**

Type	Register Name	Description
SYSCFG	SYSCFG_EXTICRx	External Interrupt Configuration, x=1 to 4 EXTICR1: for pin0~pin3 , EXTICR2: for pin4~pin7, etc
EXTI_	EXTI_IMR	Interrupt Mask
	EXTI_FTSR EXTI_RTSR	Falling/Rising Trigger Selection

Schematic

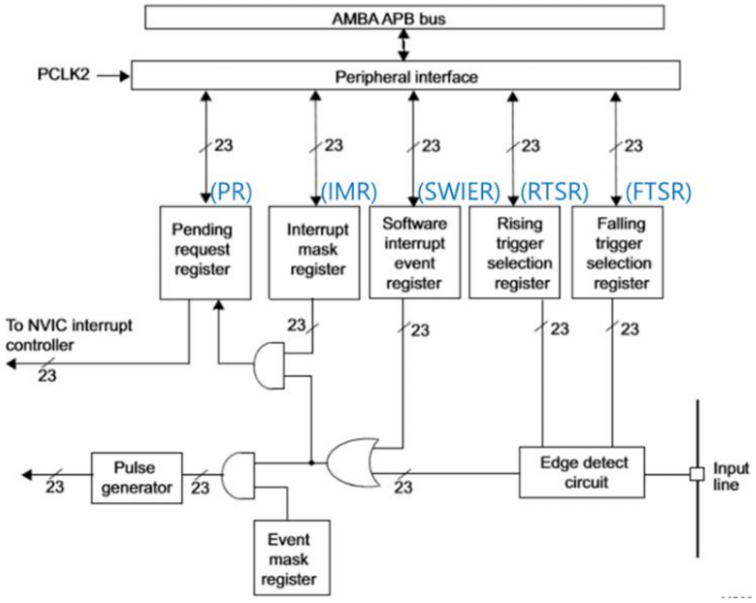

Figure 1: Register List

(NVIC Setting)

- Configure the priority of EXTI interrupt request. **NVIC_SetPriority()**
- Enable EXTI interrupt request. **NVIC_EnableIRQ()**

(EXTI Use)

- Create user codes in handler **EXTIx_IRQHandler()**
- Clear pending bit after interrupt call

III. Tutorial

A. Register Configuration

Fill in the blanks below

1. **Pin Initialization & Set LED and Push-button**

- LED Pin : Port B Pin 12 / Output / Push-Pull / No Pull-Up & No Pull-Down
- Push-Button: Port A Pin 4 / Input / No Pull-Up & No Pull-Down

```
// Use your library
GPIO_init(pin12, 1)
GPIO_otype(pin12, 0)
GPIO_pupd(pin12, 0)

GPIO_init(pin4, 0)
GPIO_pupd(pin4, 0)
```

2. **Enable Peripheral Clock:** SYSCFGEN

- **RCC_APB2ENR:** Enable SYSCFG

3. **EXTI Initialization & Connect Push-button to EXTI line**

- **SYSCFG_EXTICR2:** Connect PA_4(push-button) to EXTI4 line

Table 21. RCC register map and reset values for STM32F411xC/E (continued)

Addr. offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x40	RCC_APB1ENR	Reserved		PWREN	Reserved					I2C3EN	I2C2EN	I2C1EN	Reserved			USART2EN	Reserved	SPI3EN	SPI2EN	Reserved		WWDGEN	Reserved							TIM5EN	TIM4EN	TIM3EN	TIM2EN
0x44	RCC_APB2ENR	Reserved											SPI5EN	Reserved	TIM11EN	TIM10EN	TIM9EN	Reserved	SYSCFGEN	SPI4EN	SPI1EN	SDIOEN	Reserved		ADC1EN	Reserved		USART6EN	USART1EN	Reserved				TIM1EN
0x48	Reserved	Reserved																															
0x4C	Reserved	Reserved																															
0x50	RCC_AHB1LPENR	Reserved										DMA2LPEN	DMA1LPEN	Reserved			SRAM1LPEN	FLITFLPEN	Reserved	CRCLPEN	Reserved					GPIOHLPEN	Reserved		GPIOELPEN	GPIODLPEN	GPIOCLPEN	GPIOBLPEN	GPIOALPEN
0x54	RCC_AHB2LPENR	Reserved																								GFSLPEN	Reserved						

Figure 2: Image

7.2.8 SYSCFG register map

The following table gives the SYSCFG register map and the reset values.

Table 22. SYSCFG register map and reset values

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	SYSCFG_MEMRMP	Reserved																														MEM_MODE	
	Reset value																															x	x
0x04	SYSCFG_PMC	Reserved															ADC1DC2	Reserved															
	Reset value																0																
0x08	SYSCFG_EXTICR1	Reserved																EXTI3[3:0]				EXTI2[3:0]				EXTI1[3:0]				EXTI0[3:0]			
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	SYSCFG_EXTICR2	Reserved																EXTI7[3:0]				EXTI6[3:0]				EXTI5[3:0]				EXTI4[3:0]			
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	SYSCFG_EXTICR3	Reserved																EXTI11[3:0]				EXTI10[3:0]				EXTI9[3:0]				EXTI8[3:0]			
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	SYSCFG_EXTICR4	Reserved																EXTI15[3:0]				EXTI14[3:0]				EXTI13[3:0]				EXTI12[3:0]			
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **EXTI_FTSR:** Enable Falling Trigger

10.3.4 Falling trigger selection register (EXTI_FTSR)

Address offset: 0x0C
Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									TR22	TR21	Reserved		TR18	TR17	TR16
									rw	rw			rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TR15	TR14	TR13	TR12	TR11	TR10	TR9	TR8	TR7	TR6	TR5	TR4	TR3	TR2	TR1	TR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:23 Reserved, must be kept at reset value.

Bits 22:0 **TRx:** Falling trigger event configuration bit of line x
0: Falling trigger disabled (for Event and Interrupt) for input line
1: Falling trigger enabled (for Event and Interrupt) for input line.

)

- **EXTI_IMR:** Interrupt NOT masked (Enable)

Table 38. External interrupt/event controller register map and reset values

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	EXTI_IMR	Reserved									MR [22:21]		Reserved		MR[18:0]																			
	Reset value										0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	EXTI_EMR	Reserved									MR [22:21]		Reserved		MR[18:0]																			
	Reset value										0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	EXTI_RTSR	Reserved									TR [22:21]		Reserved		TR[18:0]																			
	Reset value										0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	EXTI_FTSR	Reserved									TR [22:21]		Reserved		TR[18:0]																			
	Reset value										0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	EXTI_SWIER	Reserved									SWIER [22:21]		Reserved		SWIER[18:0]																			
	Reset value										0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	EXTI_PR	Reserved									PR [22:21]		Reserved		PR[18:0]																			
	Reset value										0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 3: Image

B. Programming

This is an example code for toggling LED on/off with the button input trigger (EXTI)

Fill in the empty spaces in the code.

Procedure

- Name the project as `TU_EXTI` by creating a new folder as `tutorial\TU_EXTI`
- Download the template code
  - `TU_EXTI_student.c` Click here to download
- Fill in the empty spaces in the code.
- Run the program and check your result.
- Your tutorial report must be submitted to LMS

{% hint style="info" %} DO NOT use `ecEXTI2_student.h` for this tutorial. {% endhint %}

> You MUST write your name on the source file inside the comment section

```
/**
********************************************************************************
* @author  SSSLAB
* @Mod       2025-9-25 by YKKIM
* @brief   Embedded Controller:  Tutorial ___
*leejeayong
********************************************************************************
*/
//#include "ecSTM32F4v2.h"
#include "ecRCC2.h"
#include "ecGPIO2.h"

#define LED_PIN   PB_12         //EVAL board JKIT
#define BUTTON_PIN PA_4         //EVAL board JKIT

void LED_toggle(PinName_t pinName);
void EXTI_init_tutorial(PinName_t pinName);

// Initialiization
void setup(void)
{
    RCC_PLL_init();                          // System Clock = 84MHz
    // Initialize GPIOB_12 for Output
    GPIO_init(LED_PIN, OUTPUT);    // LED for EVAL board
    // Initialize GPIOA_4 for Input Button
    GPIO_init(BUTTON_PIN, INPUT);  // OUTPUT for EVAL borad
    EXTI_init_tutorial(PA_4);
}
// MAIN  -----------------------------------
int main(void) {
```

```
    setup();
    while (1);
}
// EXTI Initialiization ----------------------------------------------------
void EXTI_init_tutorial(PinName_t pinName)
{
    GPIO_TypeDef *Port;
    unsigned int pin;
    ecPinmap(pinName, &Port, &pin);
    // SYSCFG peripheral clock enable
    RCC->APB2ENR |= RCC_APB2ENR_SYSCFGEN;
    // Connect External Line to the GPIO
    // Button: PA_4 -> EXTICR2(EXTI4)
    SYSCFG->EXTICR[1] &= ~SYSCFG_EXTICR2_EXTI4;
    SYSCFG->EXTICR[1] |= SYSCFG_EXTICR2_EXTI4_PA;
    // Falling trigger enable (Button: pull-up)
    EXTI->FTSR |= (1UL << 4);
    // Unmask (Enable) EXT interrupt
    EXTI->IMR |= (1UL << 4);
    // Interrupt IRQn, Priority
    NVIC_SetPriority(EXTI4_IRQn, 0);        // Set EXTI priority as 0
    NVIC_EnableIRQ(EXTI4_IRQn);             // Enable EXTI
}
void EXTI4_IRQHandler(void) {
    if ((EXTI->PR & EXTI_PR_PR4) == EXTI_PR_PR4) {
        LED_toggle(LED_PIN);
        EXTI->PR |= EXTI_PR_PR4; // cleared by writing '1'
    }
}

void LED_toggle(PinName_t pinName){
    GPIO_TypeDef *Port;
    unsigned int pin;
    ecPinmap(pinName,&Port,&pin);
    // YOUR CODE GOES HERE
    GPIO_write(pinName, !GPIO_read(pinName));
}
```

results

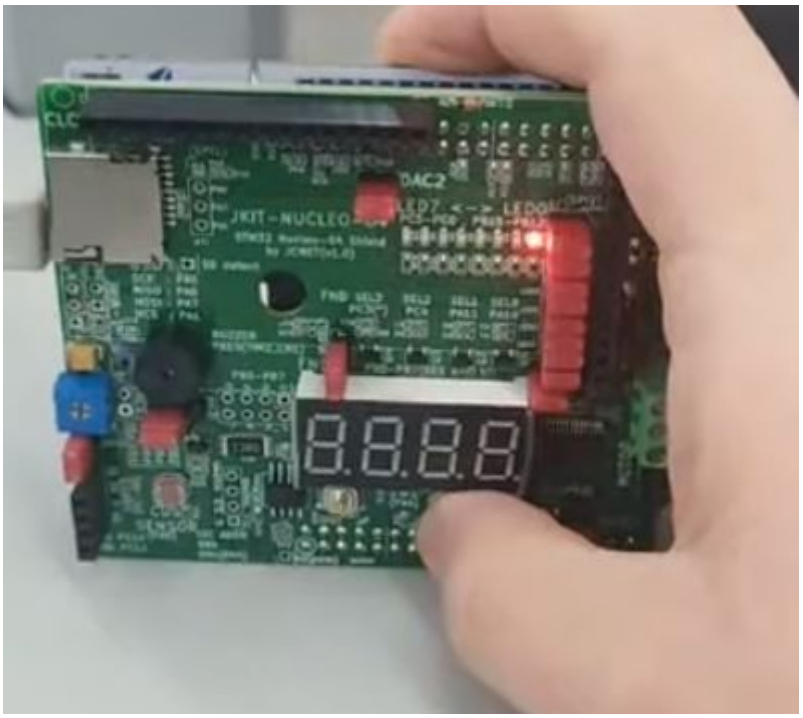

Figure 4: Image

Figure 5: Image

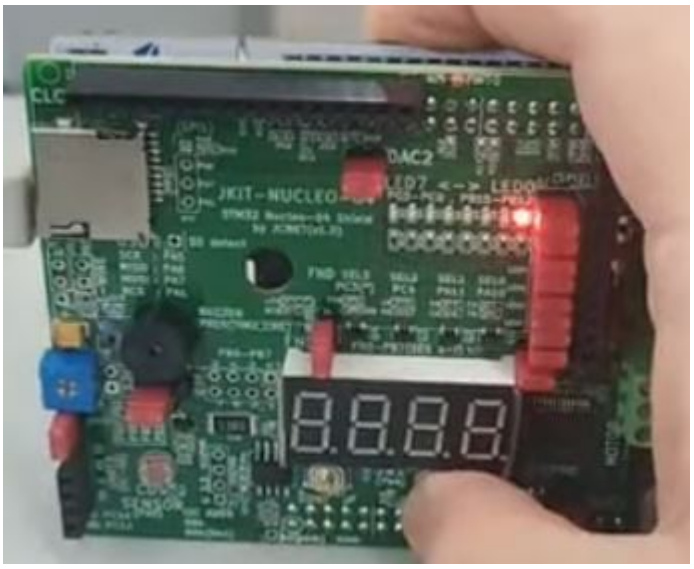

Figure 6: Image