EIE3810 Microprocessor System Design Laboratory

Laboratory Report #4

Name: 关嘉俊

Student ID: 116010060

Date: January 19th

The Chinese University of Hong Kong, Shenzhen

- Experiment A: e.g. learn to use set up the EXTI2 interrupt at the Key2 so that it can control the LED on and off.
- Experiment B: e.g. Similarly, set u the EXTI1 interrupt on board KeyUp
- Experiment C: e.g. Change the priority of the EXTI1 and observe the phenomenon of the LED, as well as the logic analyzer.
- Experiment D: e.g. Learn to set up the USART interrupt, and connect the LCD to show the register's bits condition.

I. Experiment A

(a) Basic procedure:

1) Type in the codes shown in the handout, then compile and download it to the project board.

3) Press Key2 once after reset. 4) Change falling trigger selection register to rising trigger selection register.

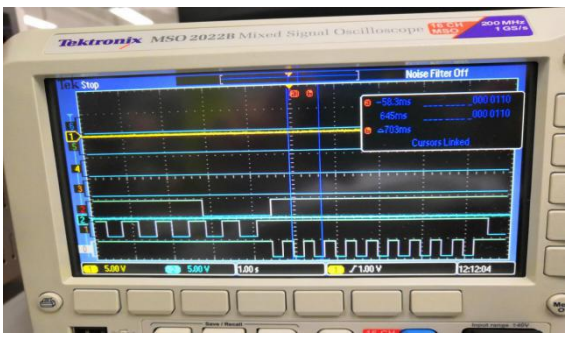
(b) Raw material:

In the figure we can see that, when the key2(Signal 2) is pressed, the LED1(Signal1) was interrupted, the LED0 handler was executed. This indicate that the interrupt work ideally.

(c)Questions:

1.If it has the same emption priority, it would be 0b01000101, 0b01010000,0b0111000, which is 0x45,0x55,0x75. If it has the same subpriority, then it can only be 0x65.

2.



From the logic analyzer we can see that, with the rising edge, the interrupt is triggered when the button is released(the voltage level is increasing).

3. The problems may occur when the touching of the key is not stable. That is , as soon as the key is pressed on, its voltage level change to high immediately.

(d) Experiment Code:

```
#include "stm32f10x.h"
#include "EIE3810_LED.h"

void Delay(u32);
void EIE3810_Key2_EXTIInit(void);
void EIE3810_KeyUp_EXTIInit(void);
void EIE3810_NVIC_SetPriorityGroup(u8 prigroup);


int main(void)
{
	u32 count;
	EIE3810_LED_Init();
	EIE3810_NVIC_SetPriorityGroup(5);//Set PRIGROUP
	//EIE3810_Key2_EXTIInit();//Initialize Key2 as an interrupt input
	EIE3810_KeyUp_EXTIInit();//Initialize KeyUp as an interrupt input
	DS0_OFF;
	DS1_OFF;
	while(1)
	{
		count++;//Just count
	}
}

void Delay(u32 count) //Use looping for delay
{
	u32 i;
	for (i=0;i<count;i++);
}

void EIE3810_Key2_EXTIInit(void)
{
	RCC->APB2ENR |= 1<<6;//enable GPIOE
	GPIOE->CRL &=0xFFFFF0FF;//reset PE2
```

```
    GPIOE->CRL |=0x00000800;//set PE2 to input pull-up mode
    GPIOE->ODR |=1<<2;//set the original status of PE2 to "set"
    RCC->APB2ENR |=0x01;//enable AFIO
    AFIO->EXTICR[0] &=0xFFFFF0FF;//reset EXTI2
    AFIO->EXTICR[0] |=0x00000400;//select PE2 pin as the input for EXTI2
    EXTI->IMR |= 1<<2;//demask the interrupt request from line 2
    EXTI->FTSR |= 1<<2;//enable falling trigger
    NVIC->IP[8] = 0x65; //set interrupt #8 with priority 0110
    NVIC->ISER[0] |= 1<<8;//enable interrupt #8
}


void EIE3810_NVIC_SetPriorityGroup(u8 prigroup)
{
    u32 temp1, temp2;
    temp2 = prigroup&0x00000007;//only 3 bits are valid
    temp2 <<=8;//3 bit are [10:8], start from bit-8
    temp1 = SCB->AIRCR;//read the data from AIRCR
    temp1 &= 0x0000F8FF;//reset bit [10:8]
    temp1 |= 0x05FA0000;//provide a rigister key as 0x5FA in the VECTKEY field.
                        //Otherwise the write value is ignored
    temp1 |= temp2;// set bit [10:8] as sepcified priority group number
    SCB->AIRCR = temp1;// write back to the AIRCR
}

void EXTI2_IRQHandler(void)
{
    u8 i;
    for (i=0;i<10;i++)
    {
        DS0_ON;
        Delay(3000000);
        DS0_OFF;
        Delay(3000000);
    }
    EXTI->PR = 1<<2; //clear the pending bit PR2 by writing a '1' into it
}
```

```
void EXTI0_IRQHandler(void)
{
    u8 i;
    for (i=0;i<10;i++)
    {
        DS1_ON;
        Delay(3000000);
        DS1_OFF;
        Delay(3000000);
    }
    EXTI->PR = 1<<0; //clear the pending bit PR0 by writing a '1' into it
}
```

II. Experiment B

(a).Experiment procedure:

1) Modify main() to Fig. 21. Remove DS1 flashing behavior an then compose KeyUp_EXTIInit() for KeyUp. 2) Refer to EXTI2_IRQHandler(), compose a handler for KeyUp. 3) Set the priority of KeyUp to interrupt 0x75 and Compile and download your project to the board.

(b) Raw material:

Since keyup is high press voltage, we can see that when the keyup was pressed, the interrupt was triggered, and LED0 is on instead of LED1.

Figure 1 Logic Analyzer screen shot

(b) Question: In the last experiment we can see that, the effect of the falling edge is much better than the rising edge since it's more sensitive for the board to detect the press down signal. For the Keyup, since is press high, so here using the rising edge is better.

(c) Source Code:

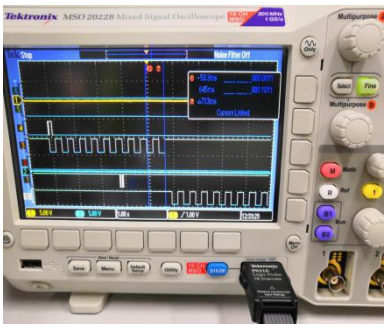
```
#include "stm32f10x.h"
#include "EIE3810_LED.h"

void Delay(u32);
void EIE3810_KeyUp_EXTIInit(void);
void EIE3810_NVIC_SetPriorityGroup(u8 prigroup);


int main(void)
{
	u32 count;
	EIE3810_LED_Init();
	EIE3810_NVIC_SetPriorityGroup(5);//Set PRIGROUP
	//EIE3810_Key2_EXTIInit();//Initialize Key2 as an interrupt input
	EIE3810_KeyUp_EXTIInit();//Initialize KeyUp as an interrupt input
	DS0_OFF;
	DS1_OFF;
	while(1)
	{
		count++;//Just count}}

void Delay(u32 count) //Use looping for delay
{
	u32 i;
	for (i=0;i<count;i++);}

OA->ODR &=0xE; //set the original status of PA0 to "reset" by & 1110
	RCC->APB2ENR |=0x01;//enable AFIO
	AFIO->EXTICR[0] &=0xFFFFFFF0;//reset EXTI0
	AFIO->EXTICR[0] |=0x00000000;//select PA0 pin as the input for EXTI0
	EXTI->IMR |= 1<<0; //demask the interrupt request from line 2
	EXTI->RTSR |= 1<<0;//enable rising trigger

	NVIC->IP[6] = 0x65;
	NVIC->ISER[0] |= 1<<6;//set Interrupt Set-Enable Register to enable interrupt #6}

void EIE3810_NVIC_SetPriorityGroup(u8 prigroup)
```

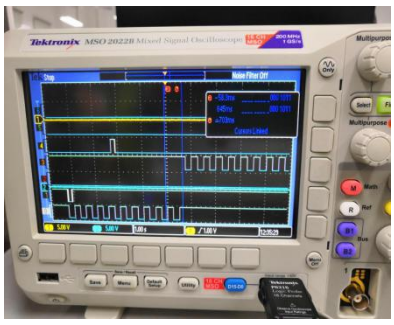
```
{
    u32 temp1, temp2;
    temp2 = prigroup&0x00000007;//only 3 bits are valid
    temp2 <<=8;//3 bit are [10:8], start from bit-8
    temp1 = SCB->AIRCR;//read the data from AIRCR
    temp1 &= 0x0000F8FF;//reset bit [10:8]
    temp1 |= 0x05FA0000;//provide a rigister key as 0x5FA in the VECTKEY field.
                        //Otherwise the write value is ignored
    temp1 |= temp2;// set bit [10:8] as sepcified priority group number
    SCB->AIRCR = temp1;// write back to the AIRCR}

void EXTI2_IRQHandler(void)
{
    u8 i;
    for (i=0;i<10;i++)
    {
        DS0_ON;
        Delay(3000000);
        DS0_OFF;
        Delay(3000000);
    }
    EXTI->PR = 1<<2; //clear the pending bit PR2 by writing a '1' into it
}

void EXTI0_IRQHandler(void)
{
    u8 i;
    for (i=0;i<10;i++)
    {
        DS1_ON;
        Delay(3000000);
        DS1_OFF;
        Delay(3000000);
    }
    EXTI->PR = 1<<0; //clear the pending bit PR0 by writing a '1' into it
}
```

III. Experiment C

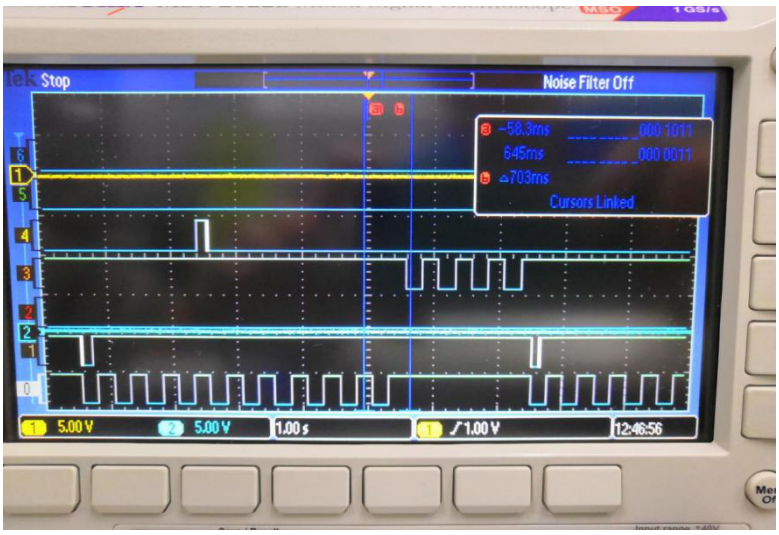
(a) **Procedure: 1.**In this experiment, we are going to enable interrupt Keyup EXTI1 and key2 EXTI at the same time and then 2.Change the priority of these two interrupt, then observe the phenomenon on the LED as well as the logic analyzer, and finally reach a conclusion.

(b) Raw Material:

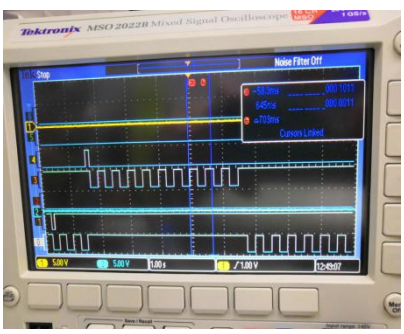


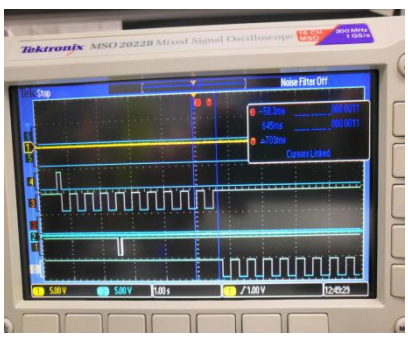


When the priority of the Keyup EXTI is 0x65, its preemption priority is 01 which is the same as the Key2 EXTI. So when the keyup handler is running, it won't be broken by the Key2 EXTI.



When the priority of the Keyup EXTI is 0x95, its preemption priority is 10 which is lower than that of the Key2 EXTI. So when the key2 interrupt handler is running, it won't be broken by the Keyup EXTI, but when the keyup interrupt handler is running, it would be paused by the key2 inteerupt.





When the priority of the Keyup EXTI is 0x35, its preemption priority is 00 which is the higher the Key2 EXTI. So when the key2 handler is running, it would be paused and pended by the Keyup EXTI.

（c）Source Code

```
#include "stm32f10x.h"
#include "EIE3810_LED.h"

void Delay(u32);
void EIE3810_Key2_EXTIInit(void);
void EIE3810_KeyUp_EXTIInit(void);
void EIE3810_NVIC_SetPriorityGroup(u8 prigroup);


int main(void)
{
	u32 count;
	EIE3810_LED_Init();
	EIE3810_NVIC_SetPriorityGroup(5);//Set PRIGROUP
	//EIE3810_Key2_EXTIInit();//Initialize Key2 as an interrupt input
	EIE3810_KeyUp_EXTIInit();//Initialize KeyUp as an interrupt input
	DS0_OFF;
	DS1_OFF;
	while(1)
	{
		count++;//Just count
	}
}

void Delay(u32 count) //Use looping for delay
{
	u32 i;
	for (i=0;i<count;i++);
}

void EIE3810_Key2_EXTIInit(void)
{
	RCC->APB2ENR |= 1<<6;//enable GPIOE
	GPIOE->CRL &=0xFFFFF0FF;//reset PE2
	GPIOE->CRL |=0x00000800;//set PE2 to input pull-up mode
	GPIOE->ODR |=1<<2;//set the original status of PE2 to "set"
	RCC->APB2ENR |=0x01;//enable AFIO
	AFIO->EXTICR[0] &=0xFFFFF0FF;//reset EXTI2
	AFIO->EXTICR[0] |=0x00000400;//select PE2 pin as the input for EXTI2
	EXTI->IMR |= 1<<2;//demask the interrupt request from line 2
	EXTI->FTSR |= 1<<2;//enable falling trigger
	NVIC->IP[8] = 0x65; //set interrupt #8 with priority 0110
	NVIC->ISER[0] |= 1<<8;//enable interrupt #8
}
```

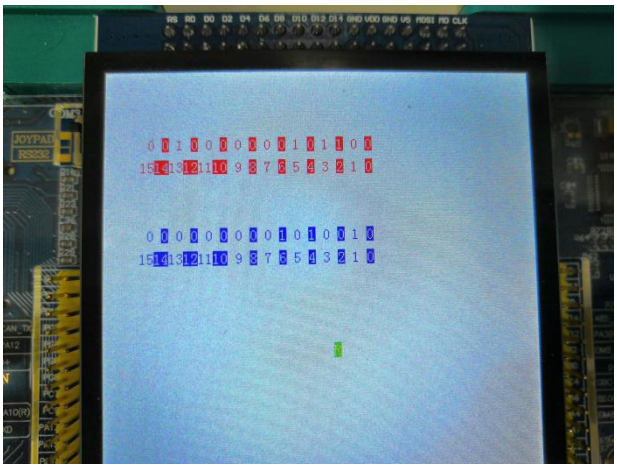
```
void EIE3810_KeyUp_EXTIInit(void)
{
    RCC->APB2ENR |= 1<<2;//enable GPIOA
    GPIOA->CRL &=0xFFFFFFF0;//reset PA0
    GPIOA->CRL |=0x00000008;//set PA0 to input pull-down mode
    GPIOA->ODR &=0xE; //set the original status of PA0 to "reset" by & 1110
    RCC->APB2ENR |=0x01;//enable AFIO
    AFIO->EXTICR[0] &=0xFFFFFFF0;//reset EXTI0
    AFIO->EXTICR[0] |=0x00000000;//select PA0 pin as the input for EXTI0
    EXTI->IMR |= 1<<0; //demask the interrupt request from line 2
    EXTI->RTSR |= 1<<0;//enable rising trigger

    //in exp.2, set interrupt #6 with priority 0111
    //NVIC->IP[6] = 0x75;
    //in exp.3, set interrupt #6 with priority 1001, preemption priority lower than EXTI2.
    //NVIC->IP[6] = 0x95;
    //in exp.4, set interrupt #6 with priority 0011, preemption priority higher than EXTI2.
    NVIC->IP[6] = 0x35;

    NVIC->ISER[0] |= 1<<6;//set Interrupt Set-Enable Register to enable interrupt #6
}

void EIE3810_NVIC_SetPriorityGroup(u8 prigroup)
{
    u32 temp1, temp2;
    temp2 = prigroup&0x00000007;//only 3 bits are valid
    temp2 <<=8;//3 bit are [10:8], start from bit-8
    temp1 = SCB->AIRCR;//read the data from AIRCR
    temp1 &= 0x0000F8FF;//reset bit [10:8]
    temp1 |= 0x05FA0000;//provide a rigister key as 0x5FA in the VECTKEY field.
                        //Otherwise the write value is ignored
    temp1 |= temp2;// set bit [10:8] as sepcified priority group number
    SCB->AIRCR = temp1;// write back to the AIRCR
}

void EXTI2_IRQHandler(void)
{
    u8 i;
    for (i=0;i<10;i++)
    {
        DS0_ON;
        Delay(3000000);
        DS0_OFF;
        Delay(3000000);
    }
```

```
    EXTI->PR = 1<<2; //clear the pending bit PR2 by writing a '1' into it
}

void EXTI0_IRQHandler(void)
{
    u8 i;
    for (i=0;i<10;i++)
    {
        DS1_ON;
        Delay(3000000);
        DS1_OFF;
        Delay(3000000);
    }
    EXTI->PR = 1<<0; //clear the pending bit PR0 by writing a '1' into it
}
```

IV. Experiment D

(a) **Procedure** 1.Set up the USART interrupt to if there is data from the RX pin 2.Modify the main.c in the example, which makes the LCD show the register digit when letter ‘R’ was received, and control the LCD

(b) Raw material:



From the figure we can see that, as the R was sent, the LCD screen shows the register lower 16bits successfully

(c) Source Code:

```
/*********************************************************************/

USART1->CR1=0x202C; //enable USART1,set word length(1 start bit, 8 data bit, n stop bit);
    //disable parity, enable transmitter by assigning 0b0010 0001 0000 1100 to CR1

/*********************************************************************/
```

```
#include "stm32f10x.h"
#include "EIE3810_LED.h"
#include "stm32f10x.h"
#include "EIE3810_USART.h"
#include "EIE3810_Clock.h"
#include "EIE3810_TFTLCD.h"

void Delay(u32);
void EIE3810_NVIC_SetPriorityGroup(u8 prigroup);
void EIE3810_USART1_EXTIInit(void);
void USART1_IRQHandler(void);
u32 count = 0;
int main(void)
{
  EIE3810_clock_tree_init();
 EIE3810_LED_Init();
 EIE3810_TFTLCD_Init();
 Delay(500000);
 EIE3810_TFTLCD_DrawAll(0,0,WHITE);
 EIE3810_NVIC_SetPriorityGroup(5);
 EIE3810_USART1_init(72,9600);
 EIE3810_USART1_EXTIInit();
 USART_print(1,"1234567890");

 while(1)
 {
  USART_print(1,"EIE3810_Lab4");
  while (!((USART1->SR>>7)&0x1));
  Delay(1000000);
 }
}
void Delay(u32 count){
 u32 i;
 for (i=0;i<count;i++);
}

void EIE3810_USART1_EXTIInit(void)
{
 NVIC->IP[37] = 0x65;
 NVIC->ISER[1] |= 1<<5;
}

void EIE3810_NVIC_SetPriorityGroup(u8 prigroup)
{
```

```
 u32 temp1, temp2;
 temp2 = prigroup&0x00000007;//only 3 bits are valid
 temp2 <<=8;//3 bit are [10:8], start from bit-8
 temp1 = SCB->AIRCR;//read the data from AIRCR
 temp1 &= 0x0000F8FF;//reset bit [10:8]
 temp1 |= 0x05FA0000;//provide a rigister key as 0x5FA in the VECTKEY field.
      //Otherwise the write value is ignored
 temp1 |= temp2;// set bit [10:8] as sepcified priority group number
 SCB->AIRCR = temp1;// write back to the AIRCR
}
void EXTI2_IRQHandler(void)
{
 u8 i;
 for (i=0;i<10;i++)
 {
  DS0_ON;
  Delay(3000000);
  DS0_OFF;
  Delay(3000000);
 }
 EXTI->PR = 1<<2; //clear the pending bit PR2 by writing a '1' into it
}
void USART1_IRQHandler(void)
{
 u32 buffer;
 if (USART1->SR &(1<<5))
 {
  buffer = USART1->DR;
  if (buffer == 'Q'){ TurnOnLED0(); EIE3810_TFTLCD_ShowChar(250,300,'Q',YELLOW,GREEN);}
  else if(buffer == 'H'){TurnOffLED0(); EIE3810_TFTLCD_ShowChar(250,300,'H',YELLOW,GREEN);}
  else if(buffer == 'R'){
   EIE3810_TFTLCD_ShowChar(250,300,'R',YELLOW,GREEN);
   for (int i=15;i>=0;i--){
    if (i%2 == 1){
     if (i>9){
      EIE3810_TFTLCD_ShowChar(40+16*(15-i),100,'1',RED,WHITE);
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),100, i+38,RED,WHITE);
      EIE3810_TFTLCD_ShowChar(40+16*(15-i),200,'1',BLUE,WHITE);
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),200, i+38,BLUE,WHITE);

      EIE3810_TFTLCD_ShowChar(48+16*(15-i),75,((USART1->CR1 >> i)&0x01)+48,RED,WHITE);
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),175,((USART1->DR >> i)&0x01)+48,BLUE,WHITE);
     }

     else{
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),100,i+48,RED,WHITE);
```

```
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),200,i+48,BLUE,WHITE);
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),75,((USART1->CR1 >> i)&0x01)+48,RED,WHITE);
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),175,((USART1->DR >> i)&0x01)+48,BLUE,WHITE);
     }

    }

    else if (i%2 == 0){
     if (i>9){
      EIE3810_TFTLCD_ShowChar(40+16*(15-i),100,'1',WHITE,RED);
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),100, i+38,WHITE,RED);
      EIE3810_TFTLCD_ShowChar(40+16*(15-i),200,'1',WHITE,BLUE);
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),200, i+38,WHITE,BLUE);
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),75,((USART1->CR1 >> i)&0x01)+48,WHITE,RED);
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),175,((USART1->DR >> i)&0x01)+48,WHITE,BLUE);
     }

     else{
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),100,i+48,WHITE,RED);
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),200,i+48,WHITE,BLUE);
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),75,((USART1->CR1 >> i)&0x01)+48,WHITE,RED);
      EIE3810_TFTLCD_ShowChar(48+16*(15-i),175,((USART1->DR >> i)&0x01)+48,WHITE,BLUE);
     }
    }
   }









  }
 }
}
```