例程四 按键中断

其实在上个例程就说那个中断的，但不是重点说，例程四就重点说下这个中断的设置，主要是针对外部中断，对于其他的中断，到时在相应的模块里面会说的。在 STM8S207RB 这个芯片里面有很多 IO 口都可以触发中断的。主要是 GPIO_A，GPIO_B，GPIO_C，GPIO_D，GPIO_E，这五组 IO 口都可以触发外部中断，所以大家以后要设计电路的话，必须先要查看先对应的文档来看下，了解清楚芯片的资料才好设置。其实大家学会调用库里面的函数的话，这些初始化相当来说就很容易的了。

10.6 外部中断

STM8S为外部中断事件专门分配了五个中断向量：

- Port A 口的5个引脚：PA[6:2]
- Port B 口的8个引脚：PB[7:0]
- Port C 口的8个引脚：PC[7:0]
- Port D 口的7个引脚：PD[6:0]
- Port E 口的8个引脚：PE[7:0]

PD7 是最高优先级的中断源 (TLI)。

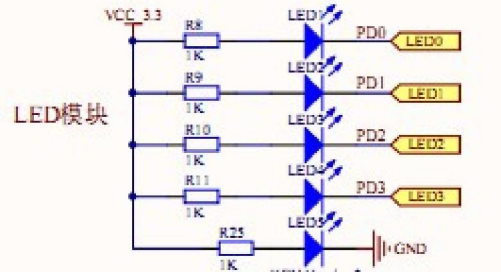
PD7 是最高优先级的中断源 (TLI)。

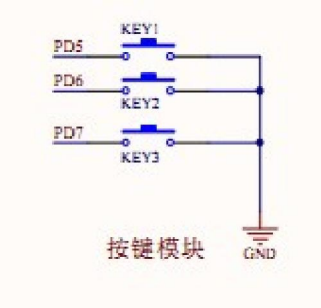
为了产生中断，相应的GPIO端口必须被配置为中断使能的输入口，详细内容请参考GPIO章节的寄存器描述部分。

中断的触发方式由外部中断控制寄存器1(EXTI_CR1)和外部中断控制寄存器2(EXTI_CR2)所配置(见 *10.9.3*和 *10.9.4*)

以上外部中断的设置来自“STM8 寄存器.pdf”文档第 74 页

下面看下电路图先吧，只要当你清楚电路具体的链接，才能完成相对应的初始化。





用到内部的资源

"stm8s_clk.h"

"stm8s_exti.h"

"stm8s_gpio.h"

"stm8s_uart1.h"

"stm8s_clk.c"

"stm8s_exti.c"

"stm8s_gpio.c"

"stm8s_uart1.c"

看完了电路图，照样是先看主函数

```
int main(void)
{

  /* Infinite loop */

  /*设置内部时钟16M为主时钟*/

   CLK_HSIPrescalerConfig(CLK_PRESCALER_HSIDIV1);
    /*!<Set High speed internal clock  */
   Buttom_Init();
   Uart_Init();
   LED_Init();
   __enable_interrupt();
   SetLedOFF();
   UART1_SendString("Key_Exti外部中断---STM8 Development Board of FengChi Electron",\
             sizeof("Key_Exti外部中断---STM8 Development Board of FengChi Electron"));
   Delay(0xffff);
   UART1_SendByte('\n');
   Delay(0xffff);
   while (1)
   {

   }
```

在主函数里面最重要的是 Buttom_Init();的初始化，其他的初始话上前几个例程已经有介绍过，相信大家也很清楚了。下面重点讲下 Buttom_Init()。
函数原型：

```
void Buttom_Init(void)
{
    GPIO_Init(GPIOD,Buttom2|Buttom1,GPIO_MODE_IN_PU_IT);
    EXTI_SetExtIntSensitivity(EXTI_PORT_GPIOD, EXTI_SENSITIVITY_FALL_ONLY);
}
```

第一条语句是设置 Buttom1 和 Buttom2 相对应的 IO 为上拉输入；
第二条语句是设置 GPIOD，也即是按键，为下降沿触发中断。
__enable_interrupt();这条语句是开总中断，在上一个例程里面说过了，以后凡是有触发中断的都要用上这条语句，所以说这条语句很重要的。
下面讲下外部中断常用的几个函数，这些函数都是库有的，可以直接调用的。

```
/**
  * @brief  Set the external interrupt sensitivity of the selected port.
  * @warning
  * - The modification of external interrupt sensitivity is only possible when the interrupts are disabled.
  * - The normal behavior is to disable the interrupts before calling this function, and re-enable them after
  * @param   Port The port number to access.
  * @param   SensitivityValue The external interrupt sensitivity value to set.
  * @retval None
  * @par Required preconditions:
  * Global interrupts must be disabled before calling this function.
  */
void EXTI_SetExtIntSensitivity(EXTI_Port_TypeDef Port, EXTI_Sensitivity_TypeDef SensitivityValue)
{

    /* Check function parameters */
    assert_param(IS_EXTI_PORT_OK(Port));
    assert_param(IS_EXTI_SENSITIVITY_OK(SensitivityValue));
```

```
  /* Set external interrupt sensitivity */
  switch (Port)
  {
  case EXTI_PORT_GPIOA:
      EXTI->CR1 &= (uint8_t)(~EXTI_CR1_PAIS);
      EXTI->CR1 |= (uint8_t)(SensitivityValue);
      break;
  case EXTI_PORT_GPIOB:
      EXTI->CR1 &= (uint8_t)(~EXTI_CR1_PBIS);
      EXTI->CR1 |= (uint8_t)((uint8_t)(SensitivityValue) << 2);
      break;
  case EXTI_PORT_GPIOC:
      EXTI->CR1 &= (uint8_t)(~EXTI_CR1_PCIS);
      EXTI->CR1 |= (uint8_t)((uint8_t)(SensitivityValue) << 4);
      break;
  case EXTI_PORT_GPIOD:
      EXTI->CR1 &= (uint8_t)(~EXTI_CR1_PDIS);
      EXTI->CR1 |= (uint8_t)((uint8_t)(SensitivityValue) << 6);
      break;
  case EXTI_PORT_GPIOE:
      EXTI->CR2 &= (uint8_t)(~EXTI_CR2_PEIS);
      EXTI->CR2 |= (uint8_t)(SensitivityValue);
      break;
```

```
    default:
        break;
    }
}
```

这个函数是设置哪组 GPIO 口为哪种方式触发中断的，触发方式有以下几种

```
/**
  * @brief  EXTI Sensitivity values for PORTA to PORTE
  */
typedef enum {
  EXTI_SENSITIVITY_FALL_LOW  = (uint8_t)0x00, /*!< Interrupt on Falling edge and Low level */
  EXTI_SENSITIVITY_RISE_ONLY = (uint8_t)0x01, /*!< Interrupt on Rising edge only */
  EXTI_SENSITIVITY_FALL_ONLY = (uint8_t)0x02, /*!< Interrupt on Falling edge only */
  EXTI_SENSITIVITY_RISE_FALL = (uint8_t)0x03  /*!< Interrupt on Rising and Falling edges */
} EXTI_Sensitivity_TypeDef;
```

下降沿和顶电平触发，只有上升沿触发，只有下降沿触发，上升沿和下降沿触发这 4 种。

```
/**
  * @brief  Get the external interrupt sensitivity of the selected port.
  * @param   Port The port number to access.
  * @retval EXTI_Sensitivity_TypeDef The external interrupt sensitivity of the selected port.
  */
EXTI_Sensitivity_TypeDef EXTI_GetExtIntSensitivity(EXTI_Port_TypeDef Port)
{
    uint8_t value = 0;

    /* Check function parameters */
    assert_param(IS_EXTI_PORT_OK(Port));


    switch (Port)
    {
    case EXTI_PORT_GPIOA:
        value = (uint8_t)(EXTI->CR1 & EXTI_CR1_PAIS);
        break;
    case EXTI_PORT_GPIOB:
        value = (uint8_t)((uint8_t)(EXTI->CR1 & EXTI_CR1_PBIS) >> 2);
        break;
    case EXTI_PORT_GPIOC:
        value = (uint8_t)((uint8_t)(EXTI->CR1 & EXTI_CR1_PCIS) >> 4);
        break;
    case EXTI_PORT_GPIOD:
        value = (uint8_t)((uint8_t)(EXTI->CR1 & EXTI_CR1_PDIS) >> 6);
        break;
    case EXTI_PORT_GPIOE:
        value = (uint8_t)(EXTI->CR2 & EXTI_CR2_PEIS);
        break;
    default:
        break;
    }

    return((EXTI_Sensitivity_TypeDef)value);
}
```

这个函数是获得哪个 IO 口是已哪种方式来触发中断的，这个在调试的时候经常要用的。

实验现象：

按一下 KEY1 或 KEY2，4 个 LED 亮，在按一下，4 个 LED 全部灭，如此循环，并在串口打印触发中断的相关信息

风驰电子祝您学习愉快～～～！！！！！