按键程序(中断方式)

SAM4N 学习笔记

版本号：1.00

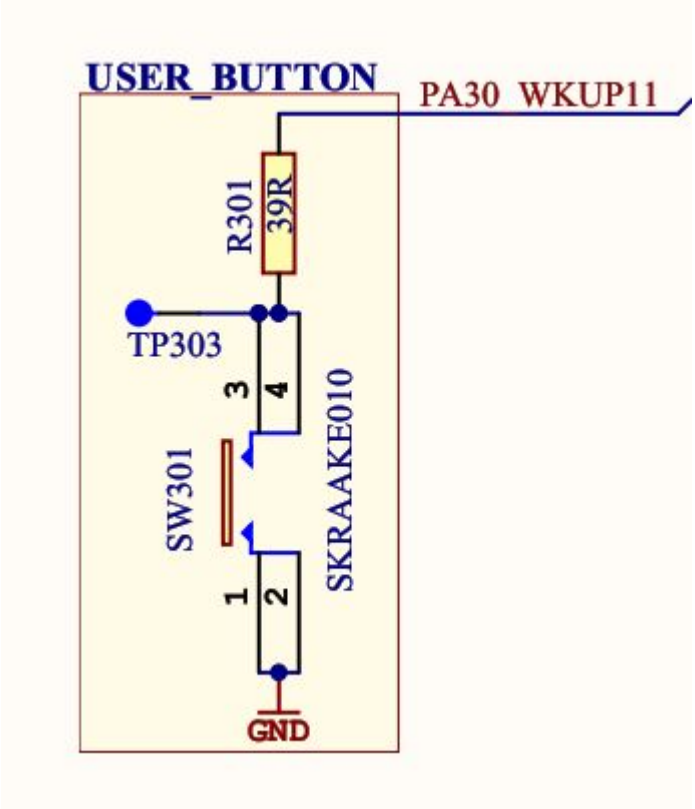
日期：2013/10/25

一、准备工作:

将上一节搭建的工程复制一份，命名为“6.key interrupt”。这一节主要讲如何使用 SAM4N 的 GPIO 中断功能，实现按键的中断输入。

二、程序编写:

这个程序主要是在上一个程序中进行改进，实现按键的中断输入。



上图可以看出按键一边连接在 PA30 上面，一边连接到 GND，当按下按键时，PA30 管脚的电平会被拉低，将按键管脚设置成上拉电阻方式，这样松开按键时 PA30 会被拉高。

所以要实现按键的输入中断可以捕获 PA30 的上升沿或是下降沿。

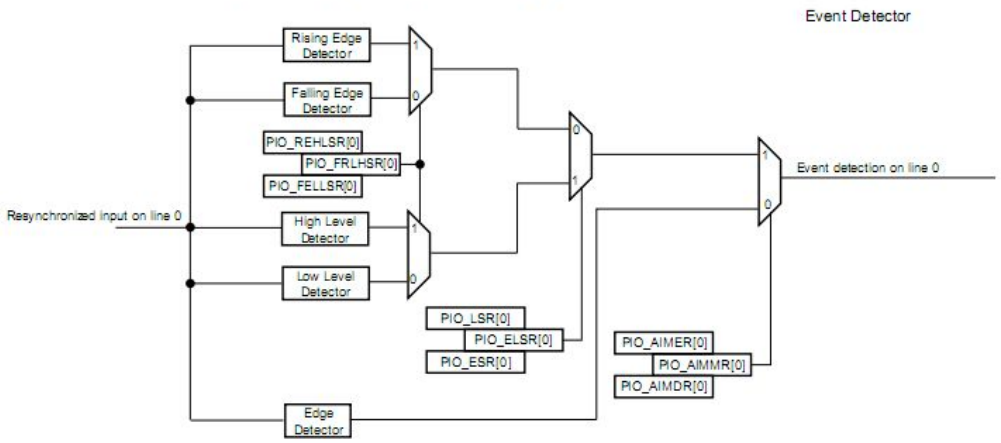
```
#define USER_BUTTON (0x01UL<<30)
/*************************************************************
*函数名： Key_GPIO_Config()
*参 数 : void
*返回值: void
*功 能 : 按键 GPIO 的初始化函数，使用按键前必须先调用此函数进行初始化
*************************************************************/
void Key_GPIO_Config(void)
{
            /*禁止外设管理控制寄存器(PMC)写保护*/
  PMC->PMC_WPMR = 0x504D4300;
    /*使能 PIOA 时钟*/
```

```
 PMC->PMC_PCER0 = (1UL << ID_PIOA);
    /*使能外设管理控制寄存器(PMC)写保护*/
 PMC->PMC_WPMR = 0x504D4301;
 /*使能 USER_BUTTON 管脚,对应为 PA30*/
PIOA->PIO_PER=(USER_BUTTON);
    /*禁止 USER_BUTTON 管脚*/
PIOA->PIO_ODR=(USER_BUTTON);
    /*使能 USER_BUTTON 管脚的上拉电阻，设置为上拉*/
PIOA->PIO_PUER=(USER_BUTTON);
    /*使能 USER_BUTTON 管脚滤波功能*/
    PIOA->PIO_IFER=USER_BUTTON;
    /*使能 USER_BUTTON 管脚中断功能*/
    PIOA->PIO_IER=USER_BUTTON;
   /*使能 USER_BUTTON 管脚中断为其他中断触发*/
    PIOA->PIO_AIMER=USER_BUTTON;
   /*使能 USER_BUTTON 管脚中断为边沿触发*/
    PIOA->PIO_ESR=USER_BUTTON;
    /*使能 USER_BUTTON 管脚中断为上降沿触发*/
    PIOA->PIO_REHLSR=USER_BUTTON;
    PIOA->PIO_ISR;

    /*配置 PIOA 的先占优先级为 1，从优先级为 1*/
  NVIC_SetPriority(PIOA_IRQn, ((0x01<<3)|0x01));
    /*使能 PIOA 的中断通道*/
   NVIC_EnableIRQ(PIOA_IRQn);
}
```

第一步，打开 PIOA 的时钟，接着使能 PIOA30 的 GPIO 功能，然后禁止 PIOA30 的输出，只作为输入功能，接着使能 PIOA30 的上拉电阻。第二步，打开 PIOA30 管脚的滤波功能,这样可以起到一个硬件消抖的作用，然后通过 PIO_IER 寄存器使能 PIOA30 的中断功能。默认情况下中断会被设置成边沿触发，这明显不是我们要的，我们需要的是下降沿触发或上升沿触发。SAM4N 的 GPIO 中断提供边沿触发、下降沿触发、上升沿触发、低电平触发、高电平触发五种类型。除了边沿触发为默认方式外，其他方式类型需要通过配置。首先配置 PIO_AIMER 寄存器，使能其他中断模式，接着配置 PIO_ESR 寄存器，使能上升/下降沿触发方式，最后配置 PIO_REHLSR 寄存器，配置成上升沿触发方式，下面是结构图：

Figure 28-7. Event Detector on Input Lines (Figure represents line 0)



接着需要设置 PIOA 向量中断优先级，最后使能 PIOA 的向量中断。在 PIOA 的中断函数中写中断处理程序：

```
/*************************************************************
  * 函数名：PIOA_Handler()
  * 参数  ：void
  * 返回值：void
  * 描述  ：PIOA 管脚中断服务函数
  *************************************************************/
  void PIOA_Handler(void)
  {
       /*检测是否为 USER_BUTTON 引发的中断*/
    if((PIOA->PIO_ISR&USER_BUTTON)!=0)
              {
      printf("USER_BUTTON 按键被按下\r\n");
     }
  }
```

这里我们需要读取 PIOA 的 PIO_ISR 寄存器，判断是不是 PIOA30 中断，读取中断寄存器以后，中断标志位会自动清除，如果在这里不读去这个 PIO_ISR，中断不清楚将会连续触发，这点需要注意。

在 PIOA30 中断后，也是打印按键被按下的信息到串口。

在 main 函数中只要去初始化按键即可：

```
  int main(void)
  {
   systick_hw_init();
   led_hw_init();
   UART0_Init(115200);
   Key_GPIO_Config();
```

```
  UART0_SendString("this is a key test demo!\r\n");
  while(1){
   PIOB->PIO_CODR=(0x01<<LED0_PIN);
       delay_ms(200);
   PIOB->PIO_SODR=(0x01<<LED0_PIN);
    delay_ms(200);
 }
 }
```

和上一个程序一样，下载运行，按下按键串口会打印出如下信息：