按键输入

RT-Thread 评估板 RealTouch 裸机例程

版本号：1.0.0

日期：2012/8/7

修订记录

日期	作者	修订历史
2012/8/7	Heyuanjie87	添加例程说明

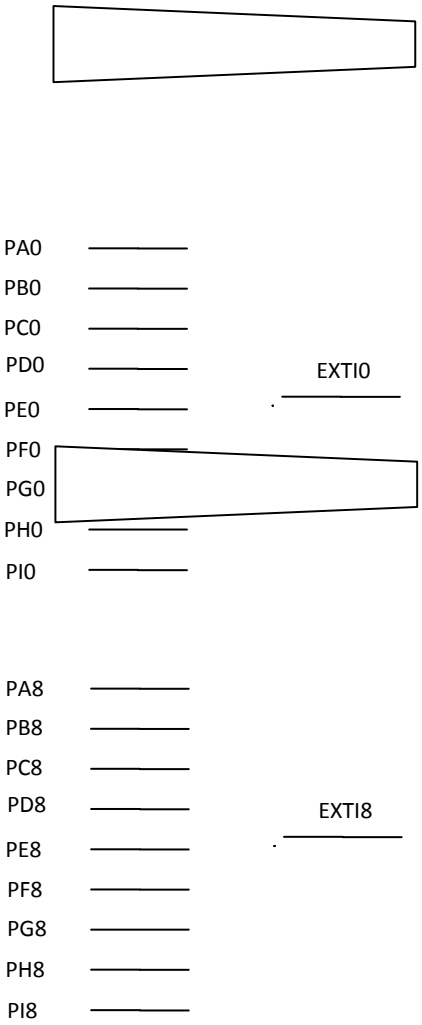
按键输入

本节将用 STM32 的 I/O 口以轮询和中断两种方式实现按键输入。这里通过板载的 7 个按键控制扩展板上的一个红色 LED 的亮灭，向大家展示如何使用按键输入。由于轮询方式涉及到的东西比较简单，下面只对外部中断给予简单介绍。

1. 外部中断简介

在 STM32 中有 16 个外部中是连接到 GPIO 上的。所有 GPIO 组的引脚 0 都连接到外部中断 0、引脚 1 就连接到了外部中断 1（PA0、PB0、PE0 等都连接到外部中断 0）。这样所有 GPIO 都可以作为中断输入使用。下面以简单的图示描述这种连接情况：

图一 GPIO 与外部中断的连接



要把 I/O 设置成外部中断输入需要经过如下几个设置步骤：

1） 初始化 I/O 为输入

选择 I/O 用途模式为通用输入，并且设置 I/O 口的输入状态。可以选择带上拉或下拉，也可以设置为悬空。为避免外部干扰引发不停地进入中断，当选择为悬空状态时应保证外部接有上拉或下拉电阻。

2） 连接 I/O 与中断线

I/O 连接到对应的中断线后才可以触发中断，这需要在外部中断配置寄存器（SYSCFG_EXTICR）中进行设置。

3） **设置触发条件**

在 STM32 中触发中断的条件可选择为上升沿或下降沿或者是任意电平变化。

4） **中断使能**

要让 CPU 能收到中断信号，必须得配置 NVIC，使能相应外部中断且还需要在中断屏蔽寄存器中把相应位置 1。

5） **编写中断服务函数**

如果开启了某个中断，一定要自己定义一个中断服务函数。否则就可能会引起程序崩溃。

2. 本例程所需硬件资源

本例程使用了板载的 7 个按键和扩展板上的一个红色 LED。其中 LED 接在 PD3 上，一个按键接在 PA0 上用中断方式使 LED 亮灭，其它 6 个按键分别接在 PF6 至 PF11 上用轮询方式使 LED 亮灭。

3. 例程

本例程涉及到了 GPIO 和外部中断的初始化以及对 NVIC 的配置过程。

代码 1 按键初始化

```
/* key.c */

void key_init(void)
{
    GPIO_Configuration();
    EXTI_Configuration();
    NVIC_Configuration();
}


void GPIO_Configuration(void)
{
    GPIO_InitTypeDef GPIO_InitStructure;

    /* 使能时钟 */
```

```
    RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA | \
                           RCC_AHB1Periph_GPIOF | \
                           RCC_AHB1Periph_GPIOD,
                           ENABLE);

    GPIO_InitStructure.GPIO_Mode  = GPIO_Mode_IN;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_2MHz;
    GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
    GPIO_InitStructure.GPIO_PuPd  = GPIO_PuPd_DOWN;

       /* 初始化中断按键 */
    GPIO_InitStructure.GPIO_Pin   = GPIO_Pin_0;
    GPIO_Init(GPIOA, &GPIO_InitStructure);

       /* 初始化轮询按键 */
    GPIO_InitStructure.GPIO_Pin   = GPIO_Pin_6 | GPIO_Pin_7 | \
                                    GPIO_Pin_8 | GPIO_Pin_9 | \
                                    GPIO_Pin_10 | GPIO_Pin_11;
    GPIO_Init(GPIOF, &GPIO_InitStructure);

       /* 初始化 LED 引脚 */
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_3;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;

    GPIO_Init(GPIOD, &GPIO_InitStructure);
  }

  void EXTI_Configuration(void)
  {
    EXTI_InitTypeDef   EXTI_InitStructure;

    /* 连接 PA0 到外部中断线 0 */
    SYSCFG_EXTILineConfig(EXTI_PortSourceGPIOA,
                          EXTI_PinSource0);
    /* 配置外部中断线 0 */
    EXTI_InitStructure.EXTI_Line = EXTI_Line0;
    EXTI_InitStructure.EXTI_Mode = EXTI_Mode_Interrupt;
    EXTI_InitStructure.EXTI_Trigger =
EXTI_Trigger_Rising_Falling;
    EXTI_InitStructure.EXTI_LineCmd = ENABLE;
    EXTI_Init(&EXTI_InitStructure);
  }

  void NVIC_Configuration(void)
```

```
{
    NVIC_InitTypeDef NVIC_InitStructure;

    /* 使能外部中断 0 */
    NVIC_InitStructure.NVIC_IRQChannel = EXTI0_IRQn;
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0x0F;
    NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0x0F;
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);
}
```

在初始化 GPIO 前需要先打开各个 I/O 组的时钟，LED 的引脚需要设置为输出，按键的引脚则需要设置为输入。PA0 对应的外部中断 0，要让它作为中断输入还需要将它连接到外部中断线 0 上。最后就是在 NVIC 中使能外部中断 0 了。当然还需要一个中断处理函数，如下：

代码 2 外部中断 0 处理函数

```
 /* stm32f4xx_it.c */
void EXTI0_IRQHandler(void)
{
    /* 清除中断标志 */
    EXTI_ClearITPendingBit(EXTI_Line0);
    /* 闪灯 */
    blink();
}
```

中断处理函数退出前需要清除中断标志。这里进入中断后的效果是扩展板上的红色LED会闪烁。其它6个按键是采用轮询方式不断读取 I/O 状态，读到高电平就会闪烁 LED，代码如下：

代码 3 轮询读取按键

```
/* key.c */
 void key_scan(void)
{
    if ( GPIO_ReadInputDataBit(GPIOF, GPIO_Pin_6) == 1)
    {
        blink();
    }

    if ( GPIO_ReadInputDataBit(GPIOF, GPIO_Pin_7) == 1)
    {
        blink();
    }
```

```
    if ( GPIO_ReadInputDataBit(GPIOF, GPIO_Pin_8) == 1)
    {
       blink();
    }

    if ( GPIO_ReadInputDataBit(GPIOF, GPIO_Pin_9) == 1)
    {
       blink();
    }

    if( GPIO_ReadInputDataBit(GPIOF, GPIO_Pin_10) == 1)
    {
       blink();
    }

    if(GPIO_ReadInputDataBit(GPIOF, GPIO_Pin_11) == 1)
    {
       blink();
    }
}
```