项目3

外部中断

3.1 项目任务和指标

本项目将完成通过按键中断控制 LED 灯任务。

通过本项目的实施，读者应掌握中断的概述、中断屏蔽寄存器和中断的处理方法及应用。

3.2 项目的预备知识

3.2.1 中断概述

CC2530 有 18 个中断源，每个中断源都有它自己的、位于一系列特殊功能寄存器(Special Function Register，SFR)中的中断请求标志。每个中断可以分别使能或禁用。

中断是指计算机在执行期间，系统内发生任何非寻常的或非预期的急需处理的事件，使得 CPU 暂时中断当前正在执行的程序而转去执行相应的事件处理程序，待处理完毕后又返回原来被中断处继续执行的过程。

CC2530 中断描述如表 3.1 所示。

表 3.1 CC2530 中断概览

中断号码	描　　述	中断名称	中断向量	中断屏蔽，CPU	中断标志，CPU
0	RF 发送 FIFO 队列空或 RF 接收 FIFO 队列溢出	RFERR	03h	IEN0. RFERRIE	TCON. RFERRIE

续表

中断号码	描　　述	中断名称	中断向量	中断屏蔽,CPU	中断标志,CPU
1	ADC 转换结束	ADC	0Bh	IEN0. ADCIE	TCON. ADCIF
2	USART0 RX 完成	URX0	13h	IEN0. URX0IE	TCON. URX0IF
3	USART1 RX 完成	URX1	1Bh	IEN0. URX1IE	TCON. URX1IF
4	AES 加密/解密完成	ENC	23h	IEN0. ENCIE	S0CON. ENCIF
5	睡眠定时器比较	ST	2Bh	IEN0. STIE	IRCON. STIF
6	端口 2 输入/USB	P2INT	33h	IEN2. P2IE	IRCON2. P2IF
7	USART0 TX 完成	UTX0	3Bh	IEN2. UTX0IE	IRCON2. UTX0IF
8	DMA 传送完成	DMA	43h	IEN1. DMAIE	IRCON. DMAIF
9	定时器 1(16 位)捕获/比较/溢出	T1	4Bh	IEN1. T1IE	IRCON. T1IF
10	定时器 2	T2	53h	IEN1. T2IE	IRCON. T2IF
11	定时器 3(8 位)捕获/比较/溢出	T3	5Bh	IEN1. T3IE	IRCON. T3IF
12	定时器 4(8 位)捕获/比较/溢出	T4	63h	IEN1. T4IE	IRCON. T4IF
13	端口 0 输入	P0INT	6Bh	IEN1. P0IE	IRCON. P0IF
14	USART1 TX 完成	UTX1	73h	IEN2. UTX1IE	IRCON2. UTX1IF
15	端口 1 输入	P1INT	7Bh	IEN2. P1IE	IRCON2. P1IF
16	RF 通用中断	RF	83h	IEN2. RFIE	S1CON. RFIF
17	看门狗计时溢出	WDT	8Bh	IEN2. WDTIE	IRCON. WDTIF

3.2.2　中断屏蔽

1. 中断屏蔽寄存器

中断屏蔽是指在中断请求产生之后,系统用软件方式有选择地封锁部分中断而允许其余部分的中断仍能得到响应。

中断使能是让中断可以被触发,可以进入中断服务程序；如果是中断禁止状态的话,即使中断信号来了也不会触发中断,也就不会进入中断服务程序。中断使能寄存器有 IEN0、IEN1 和 IEN2,通常用 1 表示中断禁止,用 0 表示中断使能。

每个中断请求可以通过设置中断使能寄存器 IEN0、IEN1 或者 IEN2 的中断使能位使能或禁止。某些外部设备会因为若干中断时间产生中断请求。这些中断请求可以作用于 P0 端口、P1 端口、P2 端口、DMA、计数器或者 RF 上。对于每个内部中断源对应的特殊功能寄存器,这些外部设备都有中断屏蔽位。寄存器 IEN0、IEN1 和 IEN2 如表 3.2～表 3.4 所示。

表 3.2　IEN0——中断使能寄存器 0

位	名称	复位	R/W	描　　述
7	EA	0	R/W	禁用所有中断 0：无中断被禁用 1：通过设置对应的使能位将每个中断源分别使能和禁止
6	—	0	RO	不使用，读出来是 0
5	STIE	0	R/W	睡眠定时器中断使能 0：中断禁止 1：中断使能
4	ENCIE	0	R/W	AES 加密/解密中断使能 0：中断使能 1：中断禁止
3	URX1IE	0	R/W	USART 1 RX 中断使能 0：中断使能 1：中断禁止
2	URX0IE	0	R/W	USART 0 RX 中断使能 0：中断使能 1：中断禁止
1	ADCIE	0	R/W	ADC 中断使能 0：中断使能 1：中断禁止
0	RFERRIE	0	R/W	RF TX/RX FIFO 中断使能 0：中断使能 1：中断禁止

表 3.3　IEN1——中断使能寄存器 1

位	名称	复位	R/W	描　　述
7：6	—	00	RO	没有使用，读出来是 0
5	P0IE	0	R/W	端口 0 中断使能 0：中断禁止 1：中断使能
4	T4IE	0	R/W	定时器 4 中断使能 0：中断禁止 1：中断使能
3	T3IE	0	R/W	定时器 3 中断使能 0：中断禁止 1：中断使能
2	T2IE	0	R/W	定时器 2 中断使能 0：中断禁止 1：中断使能

续表

位	名称	复位	R/W	描　　述
1	T1IE	0	R/W	定时器 1 中断使能 0：中断禁止 1：中断使能
0	DMAIE	0	R/W	DMA 传输中断使能 0：中断禁止 1：中断使能

表 3.4　IEN2——中断使能寄存器 2

位	名称	复位	R/W	描　　述
7：6	—	00	RO	没有使用，读出来是 0
5	WDTIE	0	R/W	看门狗定时器中断使能 0：中断禁止 1：中断使能
4	P1IE	0	R/W	端口 1 中断使能 0：中断禁止 1：中断使能
3	UTX1IE	0	R/W	USART1 TX 中断使能 0：中断禁止 1：中断使能
2	UTX0IE	0	R/W	USART0 TX 中断使能 0：中断禁止 1：中断使能
1	P2IE	0	R/W	端口 2 中断使能 0：中断禁止 1：中断使能
0	RFIE	0	R/W	RF 一般中断使能 0：中断禁止 1：中断使能

在上面 3 个寄存器中，IEN0.EA 对总中断进行中断使能控制，其余部分对所有中断源进行中断使能控制（包括 P1、P2 和 P3 三个端口中断的使能及外设中断使能）。

寄存器 P0IEN、P11EN、P2IEN 为 P0、P1 和 P2 端口每个引脚设置中断使能，如表 3.5～表 3.7 所示。

表 3.5　P0IEN——端口 0 位中断屏蔽

位	名称	复位	R/W	描　　述
7：0	P0_[7：0]IEN	0x00	R/W	端口 P0.7～P0.0 中断使能 0：中断禁用 1：中断使能

表 3.6　P1IEN——端口 1 位中断屏蔽

位	名称	复位	R/W	描　　述
7：0	P1_[7：0]IEN	0x00	R/W	端口 P1.7～P1.0 中断使能 0：中断禁用 1：中断使能

表 3.7　P2IEN——端口 2 位中断屏蔽

位	名称	复位	R/W	描　　述
7：6	—	00	R/W	未使用
5	DPIEN	0	R/W	USB D+中断使能
4：0	P2_[4：0]IEN	0 0000	R/W	端口 P2.4～P2.0 中断使能 0：中断禁用 1：中断使能

2. 中断使能的步骤

按键中断控制的步骤如下：

(1) PxIEN：在引脚中断功能配置时，常需要设置 P0IEN，主要是开启/关闭引脚的中断功能。为 1 时开启，为 0 时关闭中断。例如，开启 S1 按钮中断为 P0IEN|＝BIT4。

(2) PICTL：可以控制 Px 口中断触发信号，上升沿触发、下降沿触发。由于按键在未按下时处于高电平，按下后为低电平，松开后又为高电平，所以会产生下降沿触发信号，配置为下降沿触发即可，PICTL|＝BITO。

(3) IEN1：除了配置引脚，还需要开启端口引脚中断使能，如开启 P0 端口中断使能，IEN1|＝BIT5。

(4) PxIFG：在开启中断前需要先清除中断标志，以免误入中断造成系统混乱，如 P0IFG&＝～BIT4。

(5) 系统中断使能：在任何中断操作时，都需要开启系统中断，如 EA＝1。

中断使能的步骤如图 3.1 所示。

(1) 使 IEN0 中 IEN0.EA 位为 1，开中断。

(2) 设置寄存器 IEN0、IEN1 和 IEN2 中相应中断使能位为 1。

(3) 如果需要，则设置 P0、P1、P2 各引脚对应的各中断使能位为 1。

(4) 最后在寄存器 PICTL 中设置中断是上升沿还是下降沿触发。

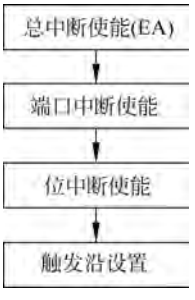

图 3.1　中断使能的步骤

3.2.3　中断处理

当中断发生时，无论该中断使能或禁止，CPU 都会在中断标志寄存器中设置终端

标志位，在程序中可以通过中断标志来判断是否发生了相应的中断。如果当设置中断标志时中断使能，那么在下一个指令周期，由硬件强行产生一个长调用指令 LCALL 到对应的向量地址，运行中断服务程序，中断的响应需要不同的时间，取决于该中断发生时 CPU 的状态。当 CPU 正在运行的中断服务程序，其优先级大于或等于新的中断时，新的中断暂不运行，直至新的中断的优先级高于正在运行的中断服务程序。

TCON、SCON、S1CON、IRCON、IRCON2 是 CC2530 的 5 个中断标志寄存器，如表 3.8～表 3.12 所示。

表 3.8　TCON——中断标志寄存器 1

位	名称	复位	R/W	描　　述
7	URX1IF	0	R/WH0	USART 1 RX 中断标志。当 USART 1 RX 中断发生时设为 1，当 CPU 指向中断向量服务例程时清除 0：无中断未决 1：中断未决
6	—	0	R/W	没有使用
5	ADCIF	0	R/WH0	ADC 中断标志。当 ADC 中断发生时设为 1，当 CPU 指向中断向量服务例程时清除 0：无中断未决 1：中断未决
4	—	0	R/W	没有使用
3	URX0IF	1	R/WH0	USART 0 RX 中断标志。当 USART0 中断发生时设为 1，当 CPU 指向中断向量服务例程时清除 0：无中断未决 1：中断未决
2	IT1	1	R/W	保留。必须一直设为 1。设置为 0 将使能低级别中断探测，几乎总是如此（启动中断请求时执行一次）
1	RFERRIF	0	R/WH0	RF TX、RX FIFO 中断标志。当 RFERR 中断发生时设为 1，当 CPU 指向中断向量服务例程时清除 0：无中断未决 1：中断未决
0	IT0	1	R/W	保留。必须一直设为 1。设置为 0 将使能低级别中断探测，几乎总是如此（启动中断请求时执行一次）

表 3.9　S0CON——中断标志寄存器 2

名称	复位	R/W	描　　述
—	0000 00	R/W	没有使用
ENCIF_1	0	R/W	AES 中断。ENC 有两个中断标志。当 AES 协处理器请求中断时两个标志都要设置 0：无中断未决 1：中断未决
ENCIF_0	0	R/W	AES 中断。ENC 有两个中断标志。当 AES 协处理器请求中断时两个标志都要设置 0：无中断未决 1：中断未决

表 3.10　S1CON——中断标志寄存器 3

位	名称	复位	R/W	描　　述
7：2	—	0000 00	R/W	没有使用
1	RFIF_1	0	R/W	RF 一般中断。RF 有两个中断标志，RFIF_1 和 RFIF_0，设置其中一个标志就会请求中断服务。当无线设备请求中断时两个标志都要设置 0：无中断未决 1：中断未决
0	RFIF_0	0	R/W	RF 一般中断。RF 有两个中断标志，RFIF_1 和 RFIF_0，设置其中一个标志就会请求中断服务。当无线电请求中断时两个标志都要设置 0：无中断未决 1：中断未决

表 3.11　IRCON——中断标志寄存器 4

位	名称	复位	R/W	描　　述
7	STIF	0	R/W	睡眠定时器中断标志 0：无中断未决 1：中断未决
6	—	0	R/W	必须写为 0。写入 1 总是使能中断源
5	P0IF	0	R/W	端口 0 中断标志 0：无中断未决 1：中断未决
4	T4IF	0	R/WH0	定时器 4 中断标志。当定时器 4 中断发生时设为 1，当 CPU 指向中断向量服务例程时清除 0：无中断未决 1：中断未决

续表

位	名称	复位	R/W	描　　述
3	T3IF	0	R/WH0	定时器 3 中断标志。当定时器 3 中断发生时设为 1，当 CPU 指向中断向量服务例程时清除 0：无中断未决 1：中断未决
2	T2IF	0	R/WH0	定时器 2 中断标志。当定时器 2 中断发生时设为 1，当 CPU 指向中断向量服务例程时清除 0：无中断未决 1：中断未决
1	T1IF	0	R/WH0	定时器 1 中断标志。当定时器 1 中断发生时设为 1，当 CPU 指向中断向量服务例程时清除 0：无中断未决 1：中断未决
0	DMAIF	0	R/W	DMA 完成中断未决 0：无中断未决 1：中断未决

表 3.12　IRCON2——中断标志寄存器 5

位	名称	复位	R/W	描　　述
7：5	—	000	R/W	没有使用
4	WDTIF	0	R/W	看门狗定时器中断标志 0：无中断未决 1：中断未决
3	P1IF	0	R/W	端口 1 中断标志 0：无中断未决 1：中断未决
2	UTX1IF	0	R/W	USART 1 TX 中断标志 0：无中断未决 1：中断未决
1	UTX0IF	0	R/W	USART 0 TX 中断标志 0：无中断未决 1：中断未决
0	P2IF	0	R/W	端口 2 中断标志 0：无中断未决 1：中断未决

P0IFG、P1IFG、P2IFG 是端口 0、端口 1、端口 2 每一位的中断标志寄存器，如表 3.13～表 3.15 所示。

表 3.13　P0IFG——端口 0 位中断标志位

位	名称	复位	R/W	描　　述
7：0	P0IF[7：0]	0x00	R/W0	端口 0，位 7～位 0 输入中断状态标志。当输入端口中断请求未决信号时，其相应的标志位将置 1

表 3.14　P1IFG——端口 1 位中断标志位

位	名称	复位	R/W	描　　述
7：0	P0IF[7：0]	0x00	R/W0	端口 1，位 7～位 0 输入中断状态标志。当输入端口中断请求未决信号时，其相应的标志位将置 1

表 3.15　P2IFG——端口 2 位中断标志位

位	名称	复位	R/W	描　　述
7：6	—	0	R0	不用
5	DPIF	0	R/W	USB D+中断状态标志。当 D+线有一个中断请求未决时设置该标志，用于检测 USB 挂起状态下的 USB 恢复事件。当 USB 控制器没有挂起时不设置该标志
4：0	P2IF[4：0]	0	R/W	端口 2，位 4～位 0 输入中断状态标志。当输入端口引脚有中断请求未决信号时，其相应的标志位将置 1

3.3　项目实施

1. 项目环境

（1）硬件：ZigBee(CC2530)模块、ZigBee 下载调试板、USB 仿真器和 PC。

（2）软件：IAR Embedded Workbench for MCS-51。

2. 项目原理

1）硬件接口原理

按键接口如图 3.2 所示。

CC2530 开发板有三个按键：一个复位按键，其余两个按键可以通过编程进行控制。当按键按下时，相应的管脚输出低电平。在此采用下降沿触发中断的方式检测是否有按键按下。

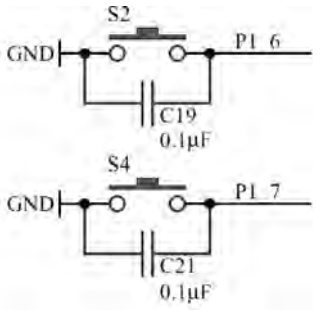

图 3.2　按键接口原理图

2）CC2530 相关寄存器

P1SEL 寄存器的详细信息如表 3.16 所示。

表 3.16 P1SEL 寄存器

位	名称	复位	R/W	描　　述
7：0	SELP1_[7：0]	0x00	R/W	P1_7～P1_0 功能选择 0：通用 I/O 1：外设功能

P1DIR 寄存器的详细信息如表 3.17 所示。

表 3.17 P1DIR 寄存器

位	名称	复位	R/W	描　　述
7：0	DIRP1_[7：0]	0x00	R/W	P1_7～P1_0 的 I/O 方向 0：输入 1：输出

P1INP 寄存器的详细信息如表 3.18 所示。

表 3.18 P1INP 寄存器

位	名称	R/W	描　　述
7：2	MDP1_[7：2]	R/W	P1_7～P1_2 的 I/O 输出模式 0：上拉/下拉 1：三态
1：0	MDP1_[1：0]	RO	未使用

P2INP 寄存器的详细信息如表 3.19 所示。

表 3.19 P2INP 寄存器

位	名称	R/W	描　　述
7	PDUP2	R/W	端口 2 上拉/下拉选择，对所有端口 2 引脚设置为上拉/下拉输入 0：上拉 1：下拉
6	PDUP1	R/W	端口 1 的设置 0：上拉 1：下拉
5	PDUP0	R/W	端口 0 的设置 0：上拉 1：下拉
4：0	MDP2_[4：0]	R/W	P2_4～P2_0 的输入模式 0：上拉/下拉 1：三态

PICTL 寄存器的详细信息如表 3.20 所示。

表 3.20　PICTL 寄存器

位	名称	复位	R/W	描　　述
7	PADSC	00	RO	控制 I/O 引脚在输出模式下的驱动能力，选择输出驱动能力补偿引脚 DVDD 的低 I/O 电压（为了确保在较低电压下的驱动能力和较高电压下的驱动能力相同） 0：最小驱动能力增强，DVDD1/2 等于或大于 2.6V 1：最大驱动能力增强，DVDD1/2 小于 2.6V
6：4	—	000	RO	保留
3	P2ICON	0	R/W	端口 2 的 P2.4～P2.0 输入模式下的中断配置，该位为所有端口 2 的输入 P2.4～P2.0 选择中断请求条件 0：输入的上升沿引起中断 1：输入的下降沿引起中断
2	P1ICONH	0	R/W	端口 1 的 P1.7～P1.4 输入模式下的中断配置，该位为所有端口 1 的输入 P1.7～P1.4 选择中断请求条件 0：输入的上升沿引起中断 1：输入的下降沿引起中断
1	P1ICONL	0	R/W	端口 1 的 P1.4～P1.0 输入模式下的中断配置，该位为所有端口 1 的输入 P1.4～P1.0 选择中断请求条件 0：输入的上升沿引起中断 1：输入的下降沿引起中断
0	P0ICON	0	R/W	端口 0 的 P0.7～P0.0 输入模式下的中断配置，该位为所有端口 0 的输入 P0.7～P0.0 选择中断请求条件 0：输入的上升沿引起中断 1：输入的下降沿引起中断

3. 软件设计

```
/* 包含头文件 */
/*********************************************************************/
#include "ioCC2530.h"   // CC2530 的头文件，包含对 CC2530 的寄存器、中断向量等的定义
/*********************************************************************/
/*********************************************************************
 * 函数名称：delay
 * 功 能：软件延时
 * 入口参数：t 延时参数，值越大延时时间越长
 * 出口参数：无
```

```
 * 返 回 值: 无
 ********************************************************************/
void delay(unsigned short t)
{
  unsigned char i,j;
  while( -- t)
  {
    j = 200;
    while( -- j)
      while( -- i);
  }
}
/*********************************************************************
 * 函数名称: EINT_ISR
 * 功 能: 外部中断服务函数
 * 入口参数: 无
 * 出口参数: 无
 * 返 回 值: 无
 ********************************************************************/
#pragma vector = P1INT_VECTOR
__interrupt void EINT_ISR(void)
{
  EA = 0;                    // 关闭全局中断
  /* 若是 P2.0 产生的中断 */
  if(P1IFG & 0x40)
  {
    /* 切换 LED1(绿色)的亮灭状态 */
    if(P1_0 == 0)            // 若之前是控制 LED1(绿色)点亮,则现在熄灭 LED1
    {
      P1_0 = 1;
    }
    else                     // 若之前是控制 LED1(绿色)熄灭,则现在点亮 LED1
    {
      P1_0 = 0;
    }
    /* 切换 LED2(红色)的亮灭状态 */
    if(P1_1 == 0)            // 若之前是控制 LED2(红色)点亮,则现在熄灭 LED2
    {
      P1_1 = 1;
    }
    else                     // 若之前是控制 LED2(红色)熄灭,则现在点亮 LED2
    {
      P1_1 = 0;
    }
    /* 切换 LED3(黄色)的亮灭状态 */
    /* 等待用户释放按键,并消抖 */
    while(P1_6 & 0x40);
    delay(10);
```

```
    while(P1_6 & 0x40);
    /* 清除中断标志 */
    P1IFG &= ~0x40;          // 清除 P1.6 中断标志
    IRCON2 &= ~0x08;         // 清除 P1 端口中断标志
  }
    if(P1IFG & 0x80)
  {
    /* 切换 LED1(绿色)的亮灭状态 */
    if(P1_0 == 0)            // 若之前是控制 LED1(绿色)点亮,则现在熄灭 LED1
    {
      P1_0 = 1;
    }
    else                     // 若之前是控制 LED1(绿色)熄灭,则现在点亮 LED1
    {
      P1_0 = 0;
    }
    /* 切换 LED2(红色)的亮灭状态 */
    if(P1_1 == 0)            // 若之前是控制 LED2(红色)点亮,则现在熄灭 LED2
    {
      P1_1 = 1;
    }
    else                     // 若之前是控制 LED2(红色)熄灭,则现在点亮 LED2
    {
      P1_1 = 0;
    }
    /* 切换 LED3(黄色)的亮灭状态 */
    /* 等待用户释放按键,并消抖 */
    while(P1_7 & 0x80);
    delay(10);
    while(P1_7& 0x80);
    /* 清除中断标志 */
    P1IFG &= ~0x80;          // 清除 P1.7 中断标志
    IRCON2 &= ~0x08;         // 清除 P1.0 端口中断标志
  }
  EA = 1;                    // 使能全局中断
}
/*********************************************************************
* 函数名称: main
* 功 能: main 函数入口
* 入口参数: 无
* 出口参数: 无
* 返 回 值: 无
*********************************************************************/
void main(void)
{
  /*
```

由于 CC253x 系列片上系统上电复位后,所有 21 个数字 I/O 均默认为具有上拉的通用输入 I/O,因此本实验只需要改变作为 LED 控制信号的 P1.0、P1.1 和 P1.4 方向为输出即可.另外

还需要将 P2.0 设置为输入下拉模式.

在用户的实际应用开发中,我们建议用户采用如下步骤来配置数字 I/O:

(1) 设置数字 I/O 为通用 I/O;

(2) 设置通用 I/O 的方向;

(3) 若通用 I/O 的方向被配置为输入,可配置上拉/下拉/三态模式,在此实验中不需要配置;

(4) 若通用 I/O 的方向被配置为输出,可设置其输出高/低电平.

```
   */
  P1SEL = 0;
  /* 配置 P1.0、P1.1 和 P1.4 的方向为输出 */
  P1DIR |= 0x03;          // 0x13 = 0B00010011
  P1_0 = 1;               // P1.0 输出低电平熄灭其所控制的 LED1(绿色)
  P1_1 = 1;               // P1.1 输出低电平熄灭其所控制的 LED2(红色)
  /* 配置 P1 端口的中断边沿下降沿产生中断 */
  PICTL |= 0x02;
  /* 使能 P1.6 和 P1.7 中断 */
  P1IEN |= 0xC0;
  /* 使能 P1 端口中断 */
  IEN2 |= 0x10;
  /* 使能全局中断 */
  EA = 1;
  while(1);
}
```

4. 实施步骤

(1) 启动 IAR 开发环境,新建工程。

(2) 在 IAR 开发环境中编译、运行、下载程序。

(3) 通过两个按键来控制两个 LED 的亮灭。