第5章

CHAPTER 5

中断系统

5.1 中断的概念

视频讲解

单片机与外部设备之间的数据交换可以采用两种方式，即查询方式和中断方式。查询方式传送数据也称为条件传送，主要用于解决外部设备与CPU之间的速度匹配问题。在这种传送方式中，不论是输入还是输出，都是以计算机为主动的一方。为了保证数据传送的正确性，单片机在传送数据之前，首先要查询外部设备是否处于“准备好”状态，对于输入操作，需要知道外设是否已把要输入的数据准备好了；对于输出操作，需要知道外设是否已把上一次单片机输出的数据处理完毕。只有通过查询确信外设已处于“准备好”状态，单片机才能发出访问外设的指令，实现数据交换。查询方式的优点是通用性好，可以用于各类外部设备和CPU之间的数据传送；缺点是需要有一个等待查询过程，CPU在等待查询期间不能进行其他操作，从而导致单片机工作效率降低。

中断方式传送数据具有可以有效提高单片机工作效率，适合于实时控制系统等优点，因而更为常用。当CPU正在处理某件事情的时候，外部发生的某一事件（如电平的改变、脉冲边沿跳变、定时器/计数器溢出等）请求CPU迅速去处理，于是CPU暂时中断当前的工作，转去处理所发生的事件。处理完该事件以后，再回到原来被中断的地方，继续原来的工作。这样的过程称为中断。中断流程如图5.1所示。

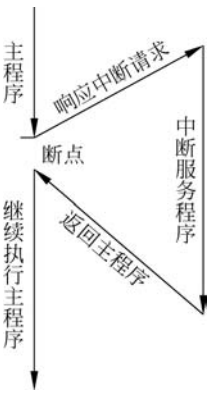

图5.1 中断流程

单片机中实现中断功能的部件称为中断系统，也就是中断管理系统。产生中断的请求源称为中断源，中断源向CPU发出的请求称为中断申请，CPU暂停当前的工作转去处理中断源事件称为中断响应，对整个事件的处理过程称为中断服务，事件处理完毕CPU返回被中断的地方称为中断返回。

与查询方式不同，中断方式是外设主动提出数据传送的请求，CPU在收到这个请求以前，一直在执行主程序，只是在收到外设希望进行数据传送的请求之后，才中断原有主程序的执行，暂时去与外设交换数据，数据交换完毕立即返回主程序继续执行。中断方式完全消除了CPU在查询方式中的等待现象，大大提高了CPU的工作效率。中断方式的一个重要应用领域是实时控制。将从现场采集到的数据通过中断方式及时传送给CPU，经过处理后就可立即做出响应，实现现场实时控制。

8051单片机可以接收的中断申请一般不止一个,对于这些不止一个的中断源进行管理,就是中断系统的任务。这些任务一般包括以下几方面。

1. 开中断或关中断

中断的开放或关闭可以通过指令对相关特殊功能寄存器的操作来实现,这是CPU能否接收中断申请的关键,只有在开中断的情况下,才有可能接收中断源的申请。

2. 中断排队

8051单片机是一个多中断源系统。在开中断的条件下,如果有若干中断申请同时发生,就需要决定先对哪一个中断申请进行响应,这就是中断排队的问题,也就是要对各个中断源进行优先级排序。单片机先响应优先级别高的中断申请。

3. 中断响应

单片机在响应了中断源的申请时,应使CPU从主程序转去执行中断服务子程序,同时要把断点地址送入堆栈进行保护,以便在执行完中断服务子程序后能返回到原来的断点继续执行主程序。断点地址入栈是由单片机内部硬件自动完成的,中断系统还要能确定各个被响应中断源的中断服务子程序的入口。

4. 中断撤除

在响应中断申请以后,返回主程序之前,中断申请应该撤除,否则就等于中断申请仍然存在,这将影响对其他中断申请的响应。8051单片机内部硬件只能对一部分中断申请在响应之后自动撤除,这一点在使用中一定要注意。

5.2 中断系统结构与中断控制

8051单片机的中断系统结构如图5.2所示。

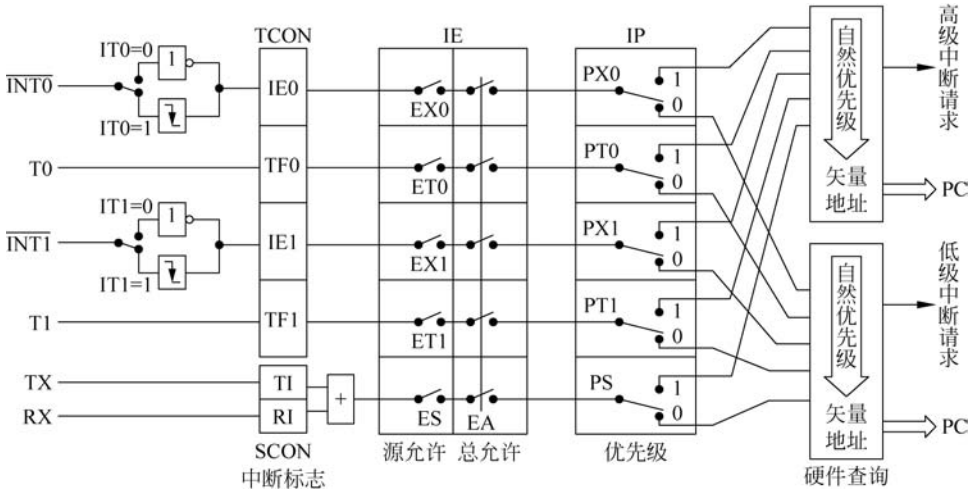

图5.2 中断系统结构

从图5.2中可见,8051单片机有5个中断请求源,4个用于中断控制的寄存器IE、IP、TCON和SCON,用来控制中断的类型、中断的开/关和各种中断源的优先级别。5个中断源有2个中断优先级,每个中断源可编程为高优先级或低优先级中断,实现2级中断服务程

序嵌套。

从面向用户的角度来看，8051 单片机的中断系统就是如下几个特殊功能寄存器：定时器控制寄存器 TCON；串行口控制寄存器 SCON；中断允许寄存器 IE；中断优先级寄存器 IP。

其中 TCON 和 SCON 只有一部分位用于中断控制。通过对以上各特殊功能寄存器中相应位的置“1”或清“0”，可实现各种中断控制功能。

8051 单片机是个多中断源系统，有 5 个中断源，即 2 个外部中断、2 个定时器/计数器中断和 1 个串行口中断。

两个外部中断源分别从 $\overline{\text{INT0}}$(P3.2)和 $\overline{\text{INT1}}$(P3.3)引脚输入，外部中断请求信号可以有两种方式，即电平触发方式和负边沿触发方式。若是电平触发方式，只要在 $\overline{\text{INT0}}$ 或 $\overline{\text{INT1}}$ 引脚上检测到低电平信号即为有效的中断申请。若是负边沿触发方式，则需在 $\overline{\text{INT0}}$ 或 $\overline{\text{INT1}}$ 引脚上检测到从“1”到“0”的负边沿跳变，才属于有效申请。

两个定时器/计数器中断是当 T0 或 T1 溢出(由全“1”进入全“0”)时发出的中断申请，属于内部中断。

串行口中断也属于内部中断，它是在串行口每接收或发送完一组串行数据后自动发出的中断申请。

CPU 在检测到有效的中断申请后，使某些相应的标志位置“1”，CPU 在下一个机器周期检测这些标志以决定是否要响应中断。这些标志位分别对应于特殊功能寄存器 TCON 和 SCON 的相应位。

1. 定时器控制寄存器 TCON

TCON 寄存器的地址为 88H，其中各位都可以位寻址，位地址为 88H～8FH。TCON 寄存器中与中断有关的各控制位分布如下：

D7	D6	D5	D4	D3	D2	D1	D0
TF1		TF0		IE1	IT1	IE0	IT0

其中各控制位的含义如下。

① IT0：选择外中断 $\overline{\text{INT0}}$ 的中断触发方式。IT0＝0 为电平触发方式，低电平有效。IT0＝1 为负边沿触发方式，$\overline{\text{INT0}}$ 脚上的负跳变有效。IT0 的状态可以用指令来置“1”或清“0”。

② IE0：外中断 $\overline{\text{INT0}}$ 的中断申请标志。当检测到 $\overline{\text{INT0}}$ 上存在有效中断申请时，由内部硬件使 IE0 置“1”。当 CPU 转向中断服务时，由内部硬件将 IE0 清“0”。

③ IT1：选择外中断 $\overline{\text{INT1}}$ 的触发方式，功能与 IT0 类似。

④ IE1：外部中断 $\overline{\text{INT1}}$ 的中断申请标志，功能与 IE0 相同。

⑤ TF0：定时器/计数器 T0 溢出中断申请标志。当 T0 溢出时，由内部硬件将 TF0 置“1”，当 CPU 转向中断服务时，由内部硬件将 TF0 清“0”。

⑥ TF1：定时器 1 溢出中断申请标志，功能与 TF0 相同。

由此可见，外部中断和定时器/计数器溢出中断的申请标志，在 CPU 响应中断之后能够自动撤除。

2. 串行口控制寄存器 SCON

8051 单片机串行口的中断申请标志位于特殊功能寄存器 SCON 中，SCON 寄存器的地

址为98H,其中各位都可以位寻址,位地址为98H～9FH。串行口的中断申请标志只占用SCON中的两位,分布如下:

D7	D6	D5	D4	D3	D2	D1	D0
						TI	RI

其中各控制位的含义如下。

① RI:接收中断标志,当接收完一帧串行数据后置"1",必须由软件清"0"。

② TI:发送中断标志。当发送完一帧串行数据后置"1",必须由软件清"0"。

串行口的中断申请标志是由TI和RI相或以后产生的,并且串行口中断申请在得到CPU响应之后不会自动撤除,必须通过软件程序撤除。

3. 中断允许寄存器IE

8051单片机中断的开放和关闭是由特殊功能寄存器IE来实现两级控制的。所谓两级控制是指在寄存器IE中有一个总允许位EA,当EA=0时,就关闭所有的中断申请,CPU不响应任何中断申请。而当EA=1时,对各中断源的申请是否开放,还要看各中断源的中断允许位的状态。

中断允许寄存器IE的地址为A8H,其中各位都可以位寻址,位地址为A8H～AFH。总允许位EA和各中断源允许位在IE寄存器中的分布如下:

D7	D6	D5	D4	D3	D2	D1	D0
EA			ES	ET1	EX1	ET0	EX0

其中各控制位的含义如下。

① EX0:外部中断0($\overline{\text{INT0}}$)的中断允许位。EX0=1,允许中断;EX0=0,不允许中断。

② ET0:定时器/计数器T0的溢出中断允许位。ET0=1,允许中断;ET0=0,不允许中断。

③ EX1:外部中断1($\overline{\text{INT1}}$)的中断允许位。EX1=1,允许外部中断1申请中断;EX1=0则不允许中断。

④ ET1:定时器/计数器T1的溢出中断允许位。ET1=1,允许T1溢出中断;ET1=0,则不允许T1溢出中断。

⑤ ES:串行口中断源允许位。ES=1,串行口开中断;ES=0,串行口关中断。

⑥ EA:中断总允许位。EA=0时,CPU关闭所有的中断申请;只有EA=1时,才能允许各个中断源的中断申请,但还要取决于各中断源中断允许控制位的状态。

8051单片机在复位时,IE各位的状态都为0,所以CPU处于关中断的状态。对于串行口来说,其中断请求在被响应之后,CPU不能自动清除其中断标志,在这些情况下要注意用指令来实现中断的开放或关闭,以便进行各种中断处理。

4. 中断优先级寄存器IP

8051单片机的中断系统具有两个中断优先级,对于每一个中断请求源可编程为高优先级或低优先级中断,以实现两级中断嵌套。每个中断源的优先级别由特殊功能寄存器IP来管理。

IP 寄存器的地址为 B8H，其中各控制位是可以位寻址的，位地址为 B8H～BCH。IP 寄存器中各控制位分布如下：

D7	D6	D5	D4	D3	D2	D1	D0
			PS	PT1	PX1	PT0	PX0

其中各位的含义如下。

① PX0：外部中断 $\overline{\text{INT0}}$ 中断优先级控制位。

② PT0：定时器/计数器 T0 中断优先级控制位。

③ PX1：外部中断 $\overline{\text{INT1}}$ 中断优先级控制位。

④ PT1：定时器/计数器 T1 中断优级控制位。

⑤ PS：串行口中断优先级控制位。

IP 寄存器中若某一个控制位置“1”，则相应的中断源就规定为高优先级中断；反之，若某一个控制位为“0”，则相应的中断源就规定为低优先级中断。一个正在执行的低优先级中断服务程序能被高优先级中断源的中断申请所中断，形成中断嵌套，如图 5.3 所示。相同级别的中断源不能相互中断其服务程序，也不能被另一个低优先级的中断源所中断。若 CPU 正在执行高优先级的中断服务子程序，则不能被任何中断源所中断。

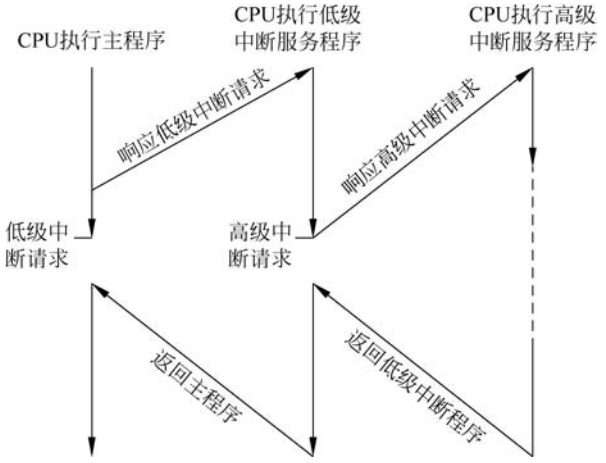

图 5.3　中断嵌套

5.3　中断响应

当有某个中断源请求中断，同时特殊功能寄存器 IE 中相应控制位处于置“1”状态，则 CPU 就可以响应中断。8051 单片机有 5 个中断源，但只有两个中断优先级，因此必然会有若干断源处于同样的中断优先级。当两个同样级别的中断申请同时到来时，CPU 应该如何响应呢？在这种情况下，8051 单片机内部有一个固定的查寻次序，当出现同级中断申请时，就按这个次序来处理中断响应。8051 单片机的 5 个中断源及其同级内的优先级次序如表 5.1 所示。

表 5.1 8051 单片机的中断源

中 断 源	入 口 地 址	同级内的优先级顺序	说 明
外部中断 0	0003H	最高	来自 P3.2 引脚($\overline{\text{INT0}}$)的外部中断请求
定时器/计数器 T0	000BH	↑	定时器/计数器 T0 溢出中断请求
外部中断 1	0013H		来自 P3.3 引脚($\overline{\text{INT1}}$)的外部中断请求
定时器/计数器 T1	001BH		定时器/计数器 T1 溢出中断请求
串行口	0023H	最低	串行口完成一帧数据的发送或接收中断

表 5.1 给出的只是 8051 单片机的 5 个最基本中断源,不同型号单片机除了这 5 个基本中断源之外还有它们各自专有的中断源,如 8052 就还有一个定时器/计数器 T2 溢出中断,T2 的中断入口地址为 002BH。

8051 单片机在接收到中断申请以后,先把这些申请锁定在各自的中断标志位中,然后在下一个机器周期按表 5.1 规定的内部优先顺序和中断优先级分别来查询这些标志,并在一个机器周期之内完成检测和优先排队。响应中断的条件有如下 3 个。

(1) 必须没有同级或更高级别的中断正在得到响应,如果有,则必须等 CPU 为它们服务完毕,返回主程序并执行一条指令之后才能响应新的中断申请。

(2) 必须要等当前正在执行的指令执行完毕以后,CPU 才能响应新的中断申请。

(3) 若正在执行的指令是 RETI(中断返回)或是任何访问 IE 寄存器或 IP 寄存器的指令,则必须要在执行完该指令以及紧随其后的另外一条指令之后,才可以响应新的中断申请。在这种情况之下,响应中断所需的时间就会加长,这个响应条件是 8051 单片机所特有的。

若上述条件满足,CPU 就在下一个机器周期响应中断,完成两项工作:一项是把中断点的地址,即当前程序计数器 PC 的内容送入堆栈保护;另一项是根据中断的不同来源把程序的执行转移到相应的中断服务子程序的入口。在 8051 单片机中,这种转移关系是固定的,对于每一种中断源,都有一个固定的中断服务子程序入口地址,如表 5.1 所示。

CPU 响应中断的时候,中断请求被锁存在 TCON 和 SCON 的标志位。当某个中断请求得到响应之后,相应的中断标志位应该予以清除(即清“0”),否则 CPU 又会继续查询这些标志位而认为又有新的中断申请来到,实际上这种中断申请并不存在。因此就存在一个中断请求的撤除问题。8051 单片机有 5 个中断源,对于其中的两种,在响应之后,系统能通过硬件自动使标志位清“0”(即撤除),它们是:

(1) 定时器 0 或 1 的中断请求标志 TF0 或 TF1;

(2) 外部中断 0 或 1 的中断请求标志 IE0 或 IE1。

在这里需要注意的是外部中断。由于外部中断有两种触发方式,即低电平方式和负边沿方式。对于边沿触发方式比较简单,因为在清除了 IE0 或 IE1 以后必须再来一个负边沿信号,才可能使标志位重新置“1”。对于低电平触发方式则不同,若仅是由硬件清除了 IE0 或 IE1 标志,而加在 $\overline{\text{INT0}}$ 或 $\overline{\text{INT1}}$ 引脚上的低电平不撤除,则在下一个机器周期 CPU 检测外中断申请时会发现又有低电平信号加在外中断输入上,又会使 IE0 或 IE1 置“1”,从而产生错误的结果。8051 单片机的中断系统没有对外的联络信号,即中断响应之后没有输出信号去通知外设结束中断申请,因此必须由用户自己来关心和处理这个问题。

对于串行口的中断请求标志 TI 和 RI,中断系统不予以自动撤除。在响应串行口中断之后要先测试这两个标志位,以决定是接收还是发送,故不能立即撤除。但在使用完毕之后应使

之清“0”,以结束这次中断申请。TI 和 RI 的清“0”操作可在中断服务子程序中用指令来实现。

8051 单片机在响应中断之前,必须对中断系统进行初始化,也就是对组成中断系统的若干特殊功能寄存器中的各控制位赋值。中断系统的初始化一般需要完成以下操作:

(1) 开中断;

(2) 确定各中断源的优先级;

(3) 若是外部中断,应规定是低电平触发还是负边沿触发。

CPU 响应中断后将转到中断源的入口地址开始执行中断服务程序。8051 单片机的每个中断源都有其固定的入口地址,它们的处理过程也有所区别。一般情况下,中断处理包括两部分:一是保护现场;二是为中断服务。

所谓保护现场,就是将需要在中断服程序中使用而又不希望破坏其中原来内容的工作寄存器压入堆栈中保护起来,等中断服务完成后再从堆栈中弹出以恢复原来的内容。通常需要保护的寄存器有 PSW、A 以及其他工作寄存器。

处理中断时要注意以下几点。

(1) 8051 各中断源的入口地址之间仅相隔 8 个单元,如果中断服务程序的长度超过 8 个地址单元时,应在中断入口地址处安排一条转移指令,转到其他有足够空余存储器单元的地址空间。

(2) 若在执行当前中断服务程序时需要禁止更高级中断源,则要用软件指令关闭中断,在中断返回之前再开放中断。

(3) 在保护和恢复现场时,为了不使现场信息受到破坏或造成混乱,保护现场之前应关中断,若需要允许高级中断,则应在保护现场之后再开中断。同样在恢复现场之前也应先关中断,恢复现场之后再开中断。

(4) 及时清除那些不能被硬件自动清“0”的中断请求标志,以免产生错误的中断。

(5) 编写中断服务函数。Keil C51 编译器中规定中断服务函数的格式如下:

```
void  函数名(void) [interrupt n] [using m]
```

其中:

① 关键字 interrupt 后面的 n 是中断号,对于 8051 单片机 n 的取值范围为 0～4,编译器根据中断号自动计算出对应中断源的入口地址。

② 关键字 using 后面的 m 是该中断函数所使用的工作寄存器区,延时为当前工作寄存器区。

③ 特别需要注意的是,中断服务函数既没有返回值,也没有调用参数,因此任何时候中断服务函数都不能被其他函数调用。

最后说明一下中断的响应时间问题,CPU 并不是在任何情况下都对中断请求立即响应,不同情况下中断响应的时间有所不同。下面以外部中断为例来进行说明。

外部中断请求在每个机器周期的 S5P2 期间,经过反向后锁存到 IE0 或 IE1 标志中,CPU 在下一个机器周期才会查询这些标志,这时如果满足响应中断的条件,CPU 响应中断时,需要执行一条两个机器周期的调用指令,以转到相应的中断服务程序入口。这样,从外部中断请求有效到开始执行中断服务程序的第一条指令,至少需要 3 个机器周期。

如果在申请中断时,CPU 正在执行最长的指令(如乘、除指令),则额外等待时间增加 3 个机器周期;若正在执行中断返回(RETI)或访问 IE、IP 寄存器的指令,则额外等待时间

又要增加两个机器周期。综合估算,若系统中只有一个中断源,则中断响应时间为 3～8 个机器周期。


视频讲解

5.4 中断系统应用举例

5.4.1 中断源扩展

8051 单片机只有两个外部中断源 $\overline{\text{INT0}}$ 和 $\overline{\text{INT1}}$,当实际应用中需要多个外部中断源时,可采用硬件请求和软件查询相结合的办法进行扩展,把多个中断源通过“或非”门接到外部中断输入端,同时又连到某个 I/O 端口,这样每个中断源都能引起中断,然后在中断服务程序中通过查询 I/O 端口的状态来区分是哪个中断源引起的中断。若有多个中断源同时发出中断请求,则查询的次序就决定了同一优先级中断里的优先级。

利用中断加查询扩展中断源的 Proteus 仿真电路如图 5.4 所示,3 个转换开关 SW1～SW3 通过一个或非门连到 8051 的外中断输入引脚 $\overline{\text{INT0}}$,按键 B1 连到 8051 的外中断输入引脚 $\overline{\text{INT1}}$。SW1～SW3 的初始位置接地,当 SW1～SW3 中无论哪个转换到高电平时都会使 $\overline{\text{INT0}}$ 引脚电平变低,向 CPU 提出中断申请,究竟是哪个转换开关提出中断申请,可以在 $\overline{\text{INT0}}$ 中断服务程序中通过查询 P1.0、P1.2、P1.4 的逻辑电平获知,同时单片机通过 P1.1、P1.3、P1.5 输出高电平点亮相应的 LED 指示灯。当按键 B1 按下(接地)时,将触发外部中断 $\overline{\text{INT1}}$,在 $\overline{\text{INT1}}$ 中断服务程序中向 P1 口输出低电平,熄灭所有 LED 指示灯。

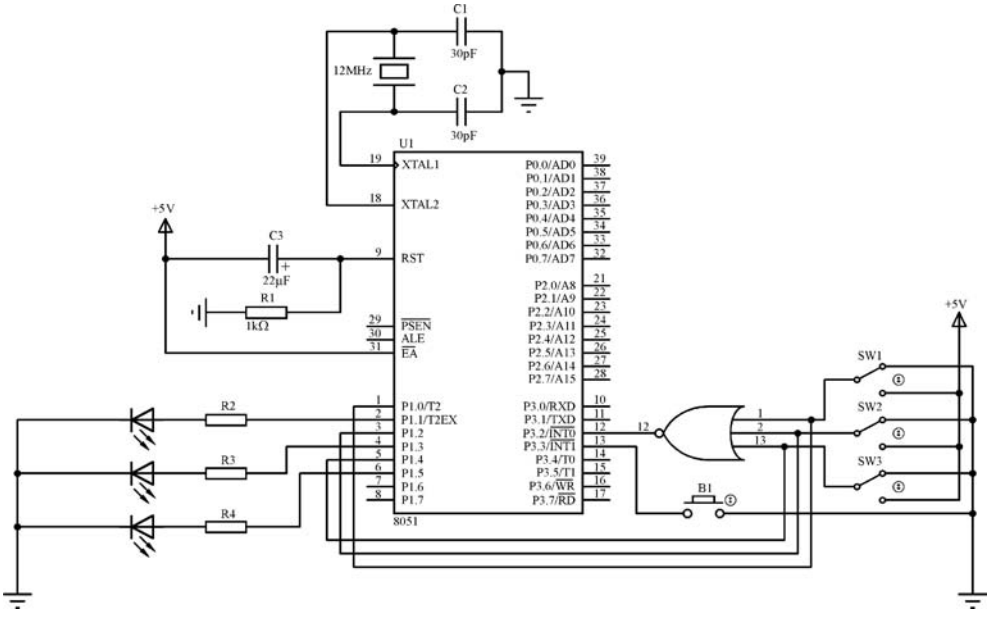

图 5.4 利用中断加查询扩展中断源的 Proteus 仿真电路

【例 5-1】 中断源扩展的 C51 源程序清单。

```
#include<reg52.h>
#define uchar unsigned char
#define uint unsigned int
```

```
sbit K1 = P1^0;
sbit K2 = P1^2;
sbit K3 = P1^4;
sbit L1 = P1^1;
sbit L2 = P1^3;
sbit L3 = P1^5;
/***** INT0 中断服务函数 *****/
void int0() interrupt 0 {
    if(K1 == 1) L1 = 1;
    if(K2 == 1) L2 = 1;
    if(K3 == 1) L3 = 1;
}
/***** INT1 中断服务函数 *****/
void int1() interrupt 2 {
    P1& = 0x55;
}
/***** 主函数 *****/
void main(){
    P1& = 0x55;
    IE = 0x85;TCON = 0x05;
    while(1);
}
```

上面这个例子比较简单，不需要保护现场，在实际应用时如果中断服务程序较复杂，需要采用多个工作寄存器时，一定要注意现场的保护和恢复。

5.4.2　中断嵌套


视频讲解

8051 单片机的中断系统具有两个优先级，每个中断源都可以设置为高、低优先级，多个中断同时发生时，CPU 根据优先级别的高低分先后进行响应，并执行相应的中断服务程序。一个正在执行的低优先级中断服务程序能被高优先级中断源的中断申请所中断，形成中断嵌套。相同级别的中断源不能相互中断，也不能被另一个低优先级的中断源所中断。若 CPU 正在执行高优先级的中断服务子程序，则不能被任何中断源所中断。

高、低优先级中断服务程序嵌套，Proteus 仿真电路如图 5.5 所示。在 8051 单片机外部中断 INT0、INT1 端分别通过两个按键接地，单片机的 P0、P1、P2 口分别接 3 个共阳极 LED 数码管。将 INT1 设置为高优先级，INT0 设置为低优先级，负边沿触发。

主程序在开中断后进入循环状态，通过 P0 口循环显示 1～8 字符，此时无论按下“低优先级”或“高优先级”按键，主程序都会被中断，进入中断服务程序，通过 P2 或 P1 口显示 1～8 字符。

如果先按下“低优先级”按键，则 P0 口的显示将停在某一数字，进入低优先级中断服务程序，通过 P2 口显示 1～8 字符；在 P2 口显示结束前按下“高优先级”按键，则 P2 口的显示将停在某一数字，进入高优先级中断服务程序，通过 P1 口显示 1～8 字符，高优先级中断服务程序结束后，先返回到低优先级中断服务程序继续执行，即 P2 口从刚才暂停的数字继续显示，P2 口显示结束后返回到主程序执行，即 P0 口从刚才暂停的数字继续循环显示。

【例 5-2】 中断服务嵌套的 C51 源程序清单。

```
#include <reg52.h>
#define uchar unsigned char
#define uint unsigned int
```

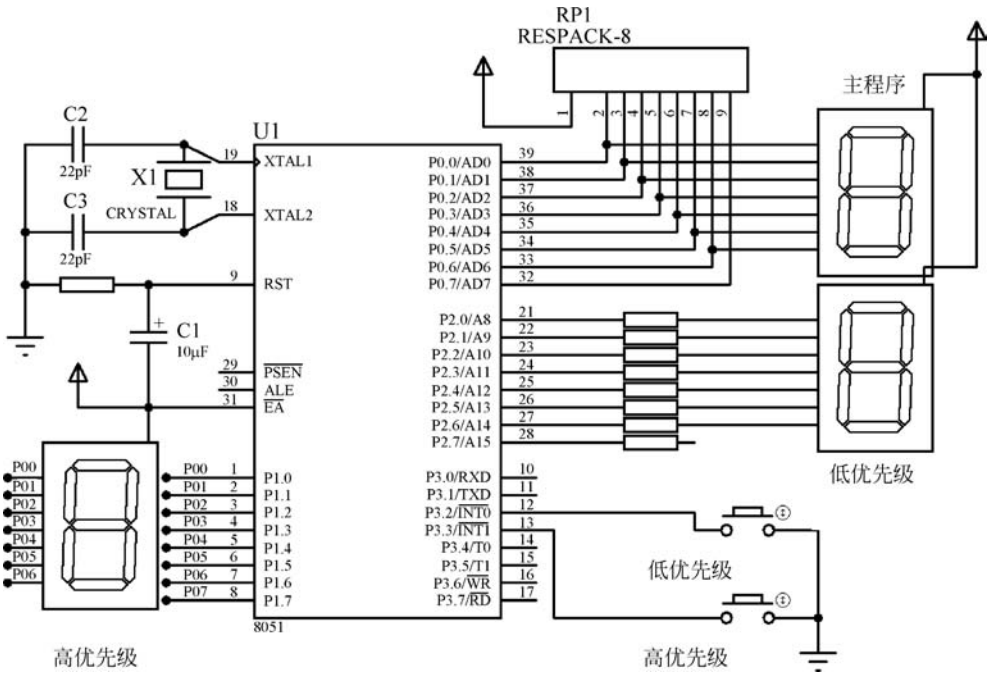

图 5.5 高、低优先级中断服务程序嵌套

```
uchar seg[ ] = {0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0x80};//LED 段码表

sbit K1 = P3^2;                                          //定义按键
sbit K2 = P3^3;
/********************************* 延时函数 *****************************/
void delay(){
    uint j;
    for(j = 0;j < 31000;j++);
}
/************************* INT0 中断服务函数 ************************/
void int0() interrupt 0 using 1{
    uchar i;
    for(i = 1;i < 9;i++){
        P2 = seg[i];delay();                             //循环显示 1～8
    }
    P2 = 0xFF;
}
/************************* INT1 中断服务函数 ************************/
void int1() interrupt 2  using 2{
    uchar i;
    for(i = 1;i < 9;i++){
        P1 = seg[i];delay();                             //循环显示 1～8
    }
    P1 = 0xFF;
}
/****************************** 主函数 ******************************/
void main(){
    uchar i;
    IE = 0x85;TCON = 0x05;PX1 = 1;                       //开中断,设置 INT1 为高优先级
    while(1){
```

```
            for(i = 1;i < 9;i++){
                P0 = seg[i];delay();                          //循环显示 1～8
            }
        }
    }
```

复习思考题

1. 什么叫中断？常见的中断类型有哪几种？单片机的中断系统要完成哪些任务？

2. 8051 单片机的中断系统由哪几个特殊功能寄存器组成？

3. 8051 单片机有几个中断源？试写出它们的内部优先级顺序以及各自的中断服务子程序入口地址。

4. 8051 单片机有哪些中断标志位？它们位于哪些特殊功能寄存器中？各中断标志是怎样产生的？

5. 简述 8051 单片机中断响应全过程。

6. 用适当指令实现将 $\overline{\text{INT1}}$ 设为脉冲下降沿触发的高优先级中断源。

7. 试编程实现将 $\overline{\text{INT1}}$ 设为高优先级中断，且为电平触发方式，T0 设为低优先级中断计数器，串行口中断为高优先级中断，其余中断源设为禁止状态。

8. 8051 单片机中，哪些中断标志可以在响应后自动撤除？哪些需要用户撤除？如何撤除？

9. 用中断加查询方式对 8051 单片机的外部中断源 $\overline{\text{INT0}}$ 进行扩展，使之能分别对 4 个按键输入的低电平信号做出响应。