第 5 章　MCS-51 单片机中断系统的原理及应用

中断系统是 MCS-51 单片机的重要组成部分，用于实时控制、故障自动处理以及与外围设备间的数据传送。MCS-51 单片机的中断系统能够提高 CPU 的多任务处理能力，大大提高了工作效率，主要体现在以下 3 个方面。

（1）并行操作。在启动某外设后，CPU 可以继续执行主程序而不必等待，二者处于并行工作状态；当某外设准备好后，向 CPU 发出中断请求，CPU 暂停主程序去处理中断服务程序，此时又启动了另外一个外设……此时 CPU 可与多个外设并行工作。

（2）实时处理。实时处理即及时处理（在规定的时间内处理）。有了中断功能后，当外设需要 CPU 处理时，向 CPU 发出中断请求，CPU 立即响应中断，使该请求被立即处理，提高了系统的实时性。当程序在查询方式下工作时不能保证实时性，这是因为当某外设需要处理时，CPU 可能正在查询其他外设。

（3）故障处理。系统设计人员把系统中可能出现的故障的处理程序设计为中断处理程序，当出现故障时，CPU 自动转入相应的中断处理程序对该故障进行处理，而不需要人工介入。

5.1　中断的基本概念

中断（Interrupt）是一个完整的过程。当 CPU 正在执行程序时，收到紧急求助信号，根据具体情况决定是否响应（处理），如果 CPU 决定响应（处理），则在执行完当前指令后，CPU 首先暂停执行当前的程序，自动保存下一条将要执行指令的地址，而后转入“处理服务程序”。当 CPU 将“处理服务程序”执行完后，CPU 重新返回到原来的程序处继续执行，这个完整的过程就叫做中断。整个中断过程如图 5-1 所示。

这个过程中包含的基本概念有以下几个。

（1）中断请求信号。当紧急事件发生时，外部设备或者单片机片上设备向 CPU 发出的紧急求助信号。

（2）主程序。当紧急求助信号来时，CPU 正在执行的程序。

（3）断点。主程序被暂停时，下一条将要执行指令的地址。

（4）中断源。引起中断的原因，或者能够发出中断请求信号的设备统称为中断源。

（5）中断响应。CPU 收到中断请求信号以后暂停当前正在执行的主程序，保存断点后

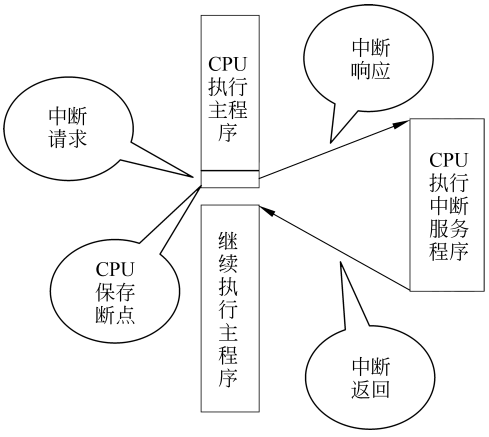

图 5-1　单片机中断过程的示意图

转去执行处理服务程序，这个过程就叫做中断响应。这个处理服务程序就被叫做中断服务程序。

(6) 中断返回。CPU 执行完中断服务程序后，执行中断返回指令 RETI，将断点的地址赋给 PC，重新回到主程序并继续执行主程序，这个过程叫做中断返回。

5.2 中断控制

中断系统是 MCS-51 单片机非常重要的组成部分，它可以使 MCS-51 单片机对内部或者外部随机发生的重要事件做出实时处理，大大提高了 MCS-51 单片机的实时处理能力和工作效率。中断的实现需要软件和硬件协同工作才能完成，硬件部分称为 MCS-51 单片机的中断控制装置，软件部分称为中断服务程序。

5.2.1 MCS-51 单片机的中断源

中断源就是指能够向 CPU 发送中断请求信号、引起中断的装置或事件。MCS-51 单片机共有 5 个中断源，其中两个为外部中断源，3 个内部中断源。每个中断源都有唯一的一个用于存放中断服务程序的地址空间，这个地址空间是固定不变的，也即中断服务程序的入口地址，如表 5-1 所示。

表 5-1 MCS-51 单片机的中断源

序号	中断源名称	中断标志位	中断入口地址
1	外部中断 0($\overline{\text{INT0}}$)(P3.2)	IE0(电平触发，软件清除)	0003H
		IE0(边沿触发，CPU 自动清除)	
2	定时器 0 溢出中断(T0)	TF0(CPU 自动清除)	000BH
3	外部中断 1($\overline{\text{INT1}}$)(P3.3)	IE1(同 IE0)	0013H
4	定时器 1 溢出中断(T1)	TF1(CPU 自动清除)	001BH
5	串行口中断(RX 和 TX)	RI 和 TI(软件清除)	0023H

(1) $\overline{\text{INT0}}$：外部中断源 0，中断请求信号由 MCS-51 单片机 P3.2 引脚输入。中断触发方式有两种：电平触发和边沿触发。由 TCON 寄存器中的 IT0 位选择采用哪种触发方式。当 IT0＝0 时，电平触发，低电平有效；当 IT0＝1 时，边沿触发，脉冲的下降沿有效。当中断控制装置中的硬件检测到有效的中断触发信号从 P3.2 引脚输入时，硬件自动将它的中断标志位 IE0 置“1”。它的中断服务程序入口地址为 ROM 0003H 单元，存放空间仅有 8B。

(2) $\overline{\text{INT1}}$：外部中断源 1，外部的中断请求信号由单片机的 P3.3 引脚送入。中断触发方式同$\overline{\text{INT0}}$，在此不再详述。当中断控制装置中的硬件检测到有效的中断请求信号从 P3.3 引脚输入时，硬件自动将它对应的中断标志位 IE1 置“1”。它的中断服务程序的入口地址为 0013H，存放空间仅有 8B。

(3) T0：定时器/计数器 0 溢出中断，当 MCS-51 单片机内部的加法计数器计数达到最大值时，再来一个脉冲则计数器溢出，中断控制装置自动将 TCON 寄存器中对应的溢出标志位 TF0 置“1”。它的中断服务程序入口地址为 000BH 单元，存放空间仅有 8B。

(4) T1：定时器/计数器 1 溢出中断，当 MCS-51 单片机内部的加法计数器计数达到最大值时，再来一个脉冲则计数器溢出，中断控制装置自动将 TCON 寄存器中对应的溢出标志位 TF1 置“1”。它的中断服务程序入口地址为 001BH 单元，存放空间仅有 8B。

(5) 全双工串行口中断：包括串行接收中断 RX 和串行发送中断 TX。当 MCS-51 单片机的串行口接收完一帧数据后，中断控制装置就自动将 SCON 寄存器中的串行接收标志位 RI 置“1”；同理，当 MCS-51 单片机的串行口发送完一帧数据后，中断控制装置也自动将 SCON 寄存器中的串行发送标志位 TI 置“1”；全双工串行口的接收和发送的中断服务程序共享同一入口，入口地址都为 0023H。

值得注意的是，每个中断服务程序的存储空间仅有 8B，存储空间非常少，所以稍微复杂一点的中断服务程序都无法容纳，唯一的解决办法为在此空间存放一条跳转指令，将中断服务程序存到其他地方，通过跳转指令找到中断服务程序即可。另外中断一般用来处理比较紧急且重要的事情，所以中断服务程序务必要短小精悍，不能冗长；否则很难保证中断服务程序的实时性。

5.2.2 MCS-51 单片机的中断控制寄存器

MCS-51 单片机的中断控制装置如图 5-2 所示，其中包括 4 个寄存器，它们是定时器/计数器控制寄存器 TCON、串行接口控制寄存器 SCON、中断允许控制寄存器 IE 和中断优先级控制寄存器 IP。

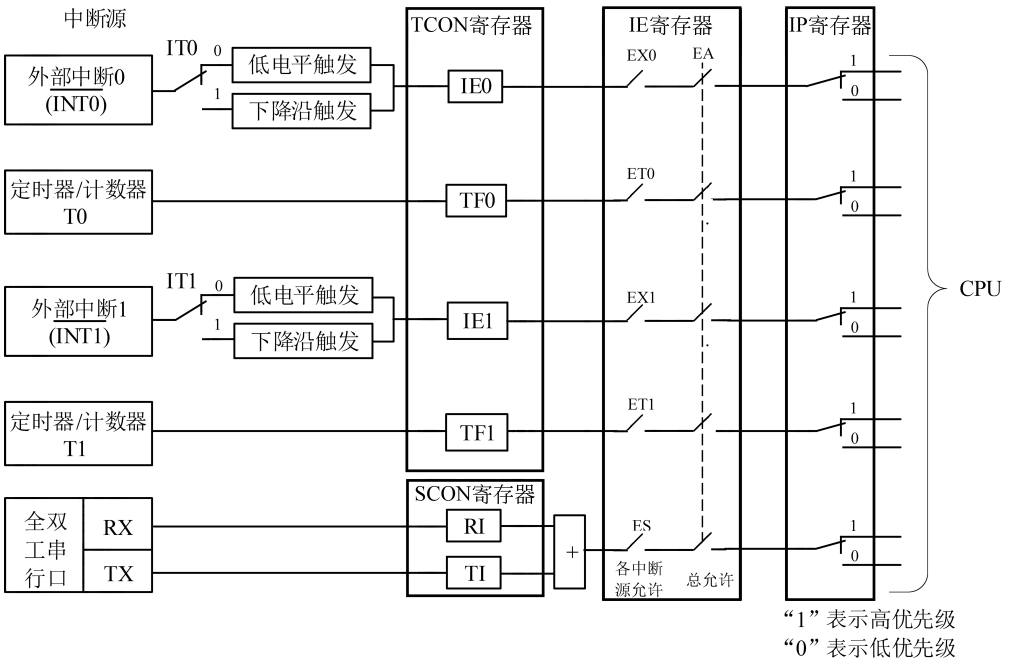

图 5-2 单片机的中断控制装置

1. 定时器/计数器控制寄存器 TCON

TCON 寄存器主要包括定时器/计数器 T0 和 T1 的标志位和运行控制位、外部中断源$\overline{\text{INT0}}$和$\overline{\text{INT1}}$的中断标志位以及外部中断源的触发方式选择位。TCON 寄存器的字节地址

为88H，它包含的位名称、位地址和功能如表5-2所示，定时器控制寄存器TCON既可以字节寻址也可以位寻址。

表5-2　定时器/计数器控制寄存器TCON

TCON	D7	D6	D5	D4	D3	D2	D1	D0
位名称	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
位地址	8FH	8EH	8DH	8CH	8BH	8AH	89H	88H
功能	T1的中断标志位	T1运行控制位	T0的中断标志位	T0运行控制位	$\overline{\text{INT1}}$的中断标志位	$\overline{\text{INT1}}$中断触发方式选择位	$\overline{\text{INT0}}$的中断标志位	$\overline{\text{INT0}}$中断触发方式选择位

(1) IT0位。外部中断0($\overline{\text{INT0}}$)的中断触发方式有两种：电平触发和边沿触发。由TCON寄存器中的IT0位选择采用哪种触发方式。当IT0＝0时，电平触发，低电平有效；当IT0＝1时，边沿触发，由1到0的下降沿有效。当采用电平触发方式时，P3.2引脚上的电平必须保持低电平有效，直到被CPU响应，同时在该中断服务程序返回前P3.2引脚上的低电平必须及时被清除，否则将会重复引起中断。

(2) IE0位。外部中断0($\overline{\text{INT0}}$)对应的中断标志位。CPU在每个机器周期的S_5P_2期间对$\overline{\text{INT0}}$对应的引脚(P3.2)进行电平采样。当中断控制装置中的硬件检测到P3.2引脚上的有效中断触发信号后，由硬件自动将中断标志位IE0置“1”，即外部中断0($\overline{\text{INT0}}$)向CPU发送了中断请求信号。

(3) IE1和IT1。IE1和IT1的功能与IE0和IT0的类似，在此不再赘述。

(4) TF0位。定时器/计数器T0的中断标志位。MCS-51单片机的定时器/计数器都是加法计数器，当T0计数产生计数溢出时，由硬件自动将TF0位置“1”，即定时器/计数器0向CPU发出了中断请求信号。当中断控制装置监测到TF0为“1”，且T0被允许中断，那么CPU响应定时器/计数器T0的中断请求，由硬件自动将TF0清“0”。当采用查询方式时，TF0可由软件清“0”。

(5) TF1位。定时器/计数器T1的计数溢出标志位。MCS-51单片机的定时器/计数器都是加法计数器，当T1计数产生计数溢出时，由硬件自动将TF1位置“1”，当中断控制装置监测到TF1为1且T1被允许中断，那么CPU响应定时器/计数器T1的中断请求，由硬件自动将TF1清“0”。当采用查询方式时，TF1由软件清“0”。

注意：

(1) 当IT0(IT1)＝0时，外部中断设置为电平触发方式时，CPU响应中断后不能由硬件自动清除中断标志IE0(IE1)，只能由软件清除。所以在外部中断返回前，必须由硬件撤销$\overline{\text{INT0}}$或$\overline{\text{INT1}}$引脚上的低电平信号，否则将会导致重复中断。

(2) 当IT0＝1时，即下降沿触发中断，CPU在每个机器周期的S_5P_2期间对$\overline{\text{INT0}}$的引脚(P3.2)电平采样。若连续两个机器周期中采集到的电平由高到低变化，则由硬件自动将IE0置“1”，表明外部中断0($\overline{\text{INT0}}$)向CPU发出了中断请求，CPU响应中断后，由硬件自动清除IE0标志位使IE0＝0。所以在下降沿触发方式中，为了保证CPU能够检测到引脚上的负跳变，引脚上的高、低电平持续时间至少要保持1个机器周期。

2. 中断允许控制寄存器 IE

中断允许控制器寄存器 IE 用来屏蔽或者允许所有中断源的中断请求信号。在时 MCS-51 单片机进行中断控制时，所有的中断源都可以由软件设置为中断允许和中断屏蔽，此功能通过中断允许控制寄存器 IE 来实现。在中断允许控制寄存器 IE 中，每个中断源都有一个对应的位，还有一个中断允许总开关位 EA。只有通过软件将某中断源对应的位设为“1”且中断允许总开关位 EA 也设为“1”时，这个中断请求才能发送成功。中断允许控制器寄存器 IE 的结构如表 5-3 所示。

表 5-3　中断允许控制寄存器 IE

IE	D7	D6	D5	D4	D3	D2	D1	D0
位名称	EA	备用	备用	ES	ET1	EX1	ET0	EX0
位地址	AFH	AEH	ADH	ACH	ABH	AAH	A9H	A8H
功能	中断允许总开关	备用	备用	串口的中断允许位	T1 的中断允许位	$\overline{\text{INT1}}$ 中断允许位	T0 的中断允许位	$\overline{\text{INT0}}$中断允许位

(1) EX0：外部中断 0($\overline{\text{INT0}}$)的中断允许设置位。EX0 为“0”表示禁止中断，即屏蔽中断；EX0 为“1”表示允许中断。

(2) ET0：定时器/计数器 0(T0)的中断允许设置位。ET0 为“0”表示禁止中断，即屏蔽中断；ET0 为“1”表示允许中断。

(3) EX1：外部中断 1($\overline{\text{INT1}}$)的中断允许设置位。EX1 为“0”表示禁止中断，即屏蔽中断；EX1 为“1”表示允许中断。

(4) ET1：定时器/计数器 1(T1)的中断允许设置位。ET1 为“0”表示禁止中断，即屏蔽中断；ET1 为“1”表示允许中断。

(5) ES：全双工串行口的中断允许设置位。ES 为“0”表示禁止中断，即屏蔽中断；ES 为“1”表示允许中断。

(6) EA：中断允许总开关设置位。中断允许总开关 EA 是一个很重要的角色，只有当通过软件将某中断源对应的位设为“1”且中断允许总开关位 EA 也设为“1”时，这个中断请求信号才是有效的。其他的中断允许位也一样，必须与中断总开关 EA 配合使用。

【例 5-1】 要求设置外部中断 0 为边沿触发方式，外部中断 0 和定时器 0 允许中断，其他的中断源屏蔽。

分析：首先在 TCON 寄存器中设置外部中断 0 的触发方式，再将外部中断 0 以及定时器 0 的中断允许位都置“1”，最后将中断总开关 EA 置“1”就可以了。使用位操作指令实现如下：

```
...
SETB      IT0
SETB      EX0
SETB      ET0
SETB      EA
...
```

使用字节操作指令实现如下：

```
...
MOV     TCON,#01H     ;设置INT0的中断触发方式
MOV     IE,#83H       ;允许INT0和 T0 中断,并设置中断允许总开关
...
```

3. 中断优先级控制寄存器 IP

MCS-51 单片机有 5 个中断源，当有两个或两个以上的中断源同时向 CPU 发送中断请求时，由于在同一时刻 CPU 只能响应一个中断源，因此 CPU 将面临是否响应和首先响应哪个中断源的问题。为了避免中断控制引起混乱，MCS-51 单片机中断控制要求事先给每个中断源的中断请求赋予一个特定的优先级。每个中断源通过软件可以设置为高优先级和低优先级两个级别。CPU 先响应优先级高的中断请求，然后按照优先级的高低顺序依次响应中断优先级次高和次低的，最后响应优先级最低的中断请求。在中断优先级控制寄存器 IP 中可设置每个中断源的优先级。

中断优先级控制寄存器 IP 是一个既可位寻址也可字节寻址的寄存器，字节地址为 0B8H。在中断优先级控制寄存器 IP 中，通过软件可以为每个中断源设置优先级为“1”或者“0”。“1”表示“高优先级”，“0”表示“低优先级”。中断优先级控制寄存器 IP 的格式及各位的定义如表 5-4 所示。

(1) PX0：外部中断 0($\overline{\text{INT0}}$)的中断优先级设置位。PX0 为“0”表示低优先级，PX0 为“1”表示高优先级。

(2) PT0：定时器/计数器 0(T0)的中断优先级设置位。PT0 为“0”表示低优先级，PT0 为“1”表示高优先级。

(3) PX1：外部中断 1($\overline{\text{INT1}}$)的中断优先级设置位。PX1 为“0”表示低优先级，PX1 为“1”表示高优先级。

(4) PT1：定时器/计数器 1(T1)的中断优先级设置位。PT1 为“0”表示低优先级，PT1 为“1”表示高优先级。

(5) PS：全双工串行口的中断优先级设置位。PS 为“0”表示低优先级，PS 为“1”表示高优先级。

表 5-4　中断优先级控制寄存器 IP

IP	D7	D6	D5	D4	D3	D2	D1	D0
位名称	备用	备用	备用	PS	PT1	PX1	PT0	PX0
位地址	0BFH	0BEH	0BDH	0BCH	0BBH	0BAH	0B9H	0B8H
功能	备用	备用	备用	串口的中断优先级设置位	T1 的中断优先级设置位	$\overline{\text{INT1}}$ 中断优先级设置位	T0 的中断优先级设置位	$\overline{\text{INT0}}$中断优先级设置位

【例 5-2】 要求设置外部中断 1 为边沿触发方式，外部中断 1 和定时器 1 允许中断，且为高优先级，其他的中断源屏蔽。

分析：首先在 TCON 寄存器中设置外部中断 1 的触发方式，再将外部中断 1 以及定时

器 1 的中断允许位都设为“1”，最后将中断总开关 EA 设为“1”就可以了。用指令实现如下：

```
...
SETB    IT1       ;将外部中断 1 设为边沿触发
SETB    EX1       ;允许外部中断 1 中断
SETB    ET1       ;允许定时器 T1 中断
SETB    PX1       ;将外部中断 1 设为高优先级
SETB    PT1       ;将定时器 T1 设为高优先级
SETB    EA        ;将中断允许总开关设为 1
...
```

当 CPU 执行一个低优先级的中断服务程序时，如果又有一个高优先级的中断请求信号，那么此时 CPU 会暂停当前的中断服务程序，转去执行高级的中断服务程序，待处理完高优先级的中断服务程序后，再处理被暂停的低优先级中断，实现中断的嵌套。即高优先级中断服务程序能中断低优先级中断服务程序。MCS-51 单片机支持两层嵌套，单片机的中断嵌套示意图如图 5-3 所示。

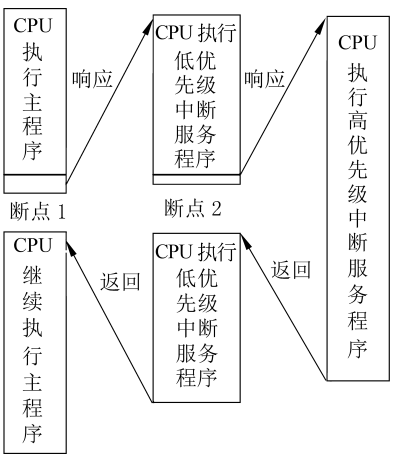

图 5-3　单片机的中断嵌套示意图

MCS-51 单片机系统中的中断源都可以通过软件在中断优先级寄存器 IP 中设定为高优先级中断和低优先级中断。那么，当 CPU 同时接收到几个相同优先级的中断请求信号时，首先响应同级内优先级最高的中断请求信号，称为“同级有安排”，即：外部中断 0 > 定时器/计数器 0 > 外部中断 1 > 定时器/计数器 1 > 全双工串行口。

当 CPU 同时接收到几个不同优先级的中断请求时，首先响应高优先级中断服务程序，称为“高级优先”。

也就是说，当 CPU 正在执行一个低优先级中断服务程序时，又收到一个高优先级的中断请求信号，CPU 会暂停执行低优先级中断服务程序，转去执行高优先级中断服务程序，实现二级嵌套，称为“停低转高”。

当 CPU 正在执行高优先级中断服务程序时，又收到同级或者低级的中断请求信号时，CPU 继续执行高优先级中断，称为“高不睬低”。

5.3　中断服务程序的处理过程

5.3.1　中断服务程序的响应条件

CPU 在每一个机器周期的 S_5P_2 期间，按照事先设置好的优先级顺序查询每一个中断源，到本机器周期的 S_6 状态时将有效的中断请求信号按优先级高低顺序排好，接下来如果没有特殊情况，CPU 将响应。优先级最高的中断请求如果此刻有下列 3 种情况之一发生时，CPU 将不会响应查询到的优先级最高的中断请求信号。

(1) CPU 正在响应同级或更高优先级的中断服务程序。

(2) 当前指令未执行完。

(3) CPU 正在执行的指令是 RETI 或者访问特殊功能寄存器 IE 或 IP 的指令,执行这些指令后至少再执行一条指令,才会响应中断服务程序。

一个中断请求信号被 CPU 响应必须满足以下 3 个条件。

(1) 中断请求标志位为 1。

(2) 对应的中断允许位为 1,且中断允许总开关为 1(即 EA=1)。

(3) 没有同级或更高级的中断服务程序在执行。

5.3.2 中断服务程序的响应过程

CPU 响应一个中断源的中断请求信号的基本过程如下。

(1) 硬件自动将正在执行的程序的断点地址压入堆栈,即保存断点,以便从中断服务程序返回时能顺利返回到原来的程序。

(2) 硬件自动将对应的中断服务程序的入口地址装入 PC,转入中断服务程序。

(3) 保存一些特殊功能寄存器的内容,即保护现场(需要人为操作)。

(4) 中断处理(需要人为操作)。

(5) 恢复现场(需要人为操作)。

(6) 清除该中断标志(RI 和 TI 标志位需要软件清除)。

(7) 执行中断返回指令 RETI(自动弹出断点到 PC,使 CPU 能够返回到被中断的程序)。添加 RETI 是中断服务程序的最后一条指令。

5.4 中断服务程序举例

要使 CPU 在执行主程序的过程中能够响应中断请求并执行中断服务程序,就必须事先对中断系统进行初始化。下面以外部中断 0 为例,介绍中断程序的编写方法。

MCS-51 单片机中断系统初始化程序操作如下。

(1) 设置堆栈。堆栈区用来保存断点和一些重要的中间数据。

MCS-51 单片机系统复位或上电后,堆栈指针 SP 总是默认为 07H,堆栈区实际上是从 08H 单元开始的。但由于 08H～1FH 单元属于工作寄存器区 1～3,考虑到程序设计中经常要用到这些存储区,故常在主程序的初始化部分将堆栈指针 SP 的值重新设定。通常将 SP 的值设定在 30H 以上。

(2) 选择中断触发方式(对外部中断而言)、开中断允许及设置中断优先级等。

系统复位后,定时器控制寄存器 TCON、中断允许寄存器 IE 及中断优先级寄存器 IP 等均复位为 00H,需要根据题目的具体要求在主程序的初始化部分对这些寄存器进行设置。

中断服务程序编写中需要注意以下问题。

(1) 中断程序的实际存放地址。每个中断服务程序对应的入口地址只有 8B,而一般服务程序长度都会超过 8B。解决问题的唯一办法是在中断入口地址处安排一条跳转指令,将程序代码存放到别的存储空间。这样当 CPU 响应外部中断 0 后跳转到外部中断 0 中断服务程序的入口地址 0003H,执行事先存放的跳转指令即可跳转到中断服务程序并执行该

程序。

(2) 现场保护。CPU 在执行主程序时一般经常会用到工作寄存器 R0～R7 用来存放数据,而中断服务程序中也要使用工作寄存器 R0～R7 存放一些中间结果,这样很容易把原来的数据覆盖掉。因此,为了保护主程序中的数据,主程序和中断服务程序中用到的工作寄存器 R0～R7 不能为同一组,一般主程序中用工作寄存器 0 组(00H～07H),而中断服务程序中使用工作寄存器 1 组(08H～0FH)或者另外两组。

CPU 在执行主程序时也经常会用到寄存器 A、B 和 DPTR 等用来存放数据,而中断服务程序中也要使用这些寄存器进行运算或者数据处理,这样原有数据同样容易被覆盖从而造成错误结果。因此,为了保护这些寄存器中的数据,可在中断服务程序中首先将这些数据压入堆栈保存,在中断返回前再把这些数据从堆栈中弹出来。一般把这个过程叫做现场保护。常用的中断服务程序结构如下:

```
IRV:    PUSH    PSW       ;保护程序状态字和工作寄存器组
        PUSH    ACC       ;保护累加器 A
        PUSH    B         ;保护寄存器 B
        PUSH    DPL       ;保护数据指针低字节
        PUSH    DPH       ;保护数据指针高字节
        SETB    RS0       ;选择寄存器组 1
        CLR     RS1
        …                 ;中断处理核心程序
        POP     DPH       ;恢复现场
        POP     DPL
        POP     B
        POP     ACC
        POP     PSW
        RETI              ;中断返回
```

注意:

(1) 在保护现场时,要保证堆栈操作的"先进后出"原则;

(2) PUSH 和 POP 指令必须成对使用;

(3) PUSH 或者 POP 指令后面是累加器 A 时,应该写成 ACC。以外部中断 0 为例,比较完整的程序如下所示。

```
        ORG     0000H             ;主程序入口地址
        AJMP    MAIN              ;跳转到 MAIN 程序
        ORG     0003H             ;外部中断 0 的中断服务程序入口地址
        LJMP    EX00              ;跳转到 EX00 服务程序
        ORG     0030H
MAIN:   MOV     SP,#60H           ;设置堆栈指针
        SETB    IT0               ;设置边沿触发
        SETB    EX0               ;允许外部中断 0 中断
        SETB    EA                ;开中断总开关
        …                         ;主程序处理部分
```

```
LOOP: AJMP    LOOP              ;原地等待
EX00:                           ;外部中断服务程序
      PUSH    PSW               ;保护现场
      PUSH    ACC
      …                         ;中断服务程序的处理部分
      POP     ACC               ;恢复现场
      POP     PSW
      RETI                      ;中断返回
      …                         ;其他程序
      END
```

【例 5-3】 在图 5-4 中，正常情况下 P1 口所接的发光二极管依次循环被点亮（每次只有一个被点亮）。当 S0 按下时，产生中断，此时 8 只发光管“全亮-全暗”交替出现 8 次，然后恢复正常，用外部中断 0 的中断来实现。

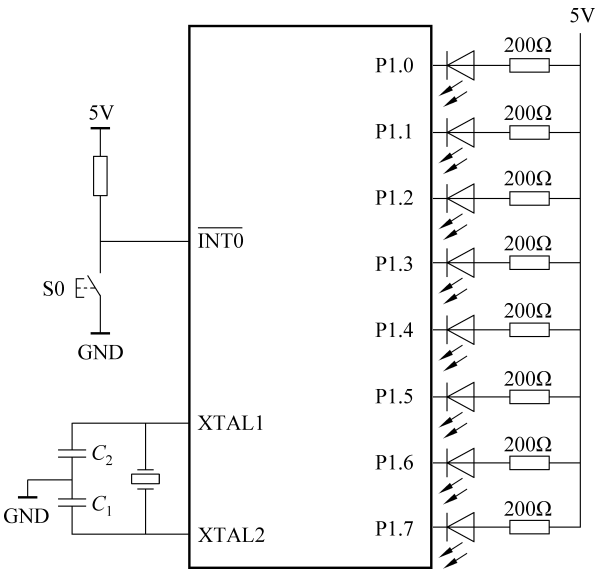

图 5-4　单片机驱动发光二极管

```
        ORG     0000H             ;主程序入口地址
        AJMP    MAIN              ;跳转到 MAIN 程序
        ORG     0003H             ;外部中断 0 的中断服务程序入口地址
        LJMP    EX00              ;跳转到 EX00 服务程序
        ORG     0030H
MAIN:   MOV     SP,#60H           ;设置堆栈指针
        SETB    IT0               ;设置沿触发
        SETB    EX0               ;允许外部中断 0 中断
        SETB    EA                ;开中断允许总开关
```