MCS51 系列单片机软件抗干扰技术中的误区

摘要：文章指出了一种广泛流传的误解：在 MCS-51 系列单片机中，只要用指令使程序从起始地址开始执行，就可以复位单片机，摆脱干扰。通过一个简单的实验，揭示了软件复位的可靠方法。

有的单片机（如 8098）有专门的复位指令，某些增强型 MCS-51 系统单片机虽然没有复位指令，但片内集成了 WATCHDOG 电路，故抗干扰也不成问题。而普及型 MCS-51 系列单片机（如 8031 和 8032）既然无复位指令，又不带硬件 WATCHDOS，如果没有外接硬件 WATCHDOG 电路，就必须采用软件抗干扰技术。常用的软件抗干扰技术有：软件陷阱、指令冗余、软件 WATCHDOG 等，它们的作用是在系统受干扰时能及时发现，再用软件的方法使系统复位。所谓软件复位就是用一系列指令来模仿复位操作，这就是 MCS-51 系列单片机所特有的软件复位技术。

现用一简单的实验说明，实验电路如附图所示。接于仿真插座 P1.0 的发光二极管 LED0 用来表示主程序的工作情况，接于 P1。1 的发光二极管 LED1 用于表示低级中断子程序的工作情况，接于 P1。2 的发光二极管 LED2 用来表示高级中断子程序的工作情况，接于 P3。2 口的按钮用来设立干扰标志，程序检测到干扰标志后故意进入死循环或掉进陷井，模仿受干扰的情况，从而检验各种复位方法的实际效果。寮验初始化程序如下：

```
        ORG 0000H
STAT:   LJMP MAIN       复位入口地址
        LJMP PX0        按钮中断向量（低级中断）
        ORG 000BH
        LJMP PT0        t0 中断向量（低级中断）
        ORG 001BH
        LJMP PT1        T1 中断向量（高级中断）
        ORG 0030H
MAIN:   CLR EA
        MOV SP，#7
        MOV P1，#0FFH
        MOV P3，#0FFH
        MOV TMOD，#11H
        CLR 00H         干扰标志初始化
        SETB ET0
        SETB ET1
        SETB EX0
        SETB PT1
        SETB TR0
        SETB TR1
        SETB EA
LOOP:   CPL P1.0        主程序发光二极管 LED 闪烁
        MOV R6,#80H
```

```
        MOV R7,#0
TT1:    DJNZ R7,TT1
        DJNZ R6,TT1
        SJMP LOOP
PX0:    SETB 00H        设立干扰标志，模拟发生干扰
PT0:    CPL P1.1        低级中断程序发光二极管 LED1 闪烁
        RETI
PT1:    CPL P1.2        高级中断程序发光二极管 LED2 闪烁
        RETI
END
```

实验步骤如下：

1. 按上述程序启动执行，三个发光二极管都应闪烁（否则应先排除故障），表示主程序和各中断子程序正常。因模拟干扰标志未加检测，故不受按钮影响。

2. 修改主程序如下，按下按钮后主程序即掉入死循环中。

```
LOOP:   CPL P1.0
        MOV R6,#80H
        MOV R7,#0H
TT1:    DJNZ R7,TT1
        DJNZ R6,TT1
        JNB 00H,LOOP    受干扰否？
STOP:   LJMP STOP       掉入死循环。
```

这时可以看到，主程序停止工作（LED0 停止闪烁），而两个中断子程序继续运行（LED1 和 LED2 继续闪烁）。

3. 将定时器 T1 妆作软件 WATCHDOG，将 30H 单元用作软件 WATCHDOG 计数器。主程序中加入一条复位软件 WATCHDOG 的指令。

```
LOOP:   CPL P1.0
        MOV 30H,#0      复位软件 WATCHDOG 计数器
LOOP:   CPL P1.0
        MOV R6,#80H
        MOV R7,#0H
TT1:    DJNZ R7,TT1
        DJNZ R6,TT1
        JNB 00H,LOOP    受干扰否？
STOP:   LJMP STOP       掉入死循环
```

T1 中断子程序修改如下：

```
PT1:    CPL P1.2        高级中断程序发光二极管闪烁
        INC 30H
        MOV A,30H
```

```
        ADD A,#0FDH
        JC ERR          达到3次否?
        RETI
ERR:    LJMP STAT       软件WATCHDOG动作
```

当按下按钮前，程序正常运行（三个 LED 全闪）。按下按钮后，主程序能迅速恢复工作，但两个中断子程序被封锁，不再工作。过程如下：主程序检测到干扰后进入死循环，不能执行复位 30H 单元的操作，T1 中断使 30H 不断增值，计数到 3 时，软件 WATCHDOG 执行动作，执行一条 LJMP 指令，使程序从头执行。MAIN 过程中清除了干扰标志（表示干扰已经过去），使主程序迅速恢复工作。按理说 MAIN 过程中也重新设定了各个中断，并开放了它们，为什么中断不能恢复工作呢？这是因为中断激活标志的复位工作被遗忘了，因为它没有明确的位地址可供编程，直接转向 0000H 地址并不能完成真正的复位。软件复位是使用软件陷井和软件 WATCHDOG 后必须进行的工作，这时程序出错完全有可能发生中断子程序中，中断激活标志已置位，它将阻止同级中断响应。由于软件 WATCHDOG 是高级中断，它将阻止所有中断响应。由此可见，清除中断激活标志的得要性，很多文献的作者回为没有认识到这一点进入误区。

4. 在所有指令中，只有 RETI 指令能清除中断激活标志。出错处理程序 ERR 主要是完成这一功能，其它的善后工作交由复位后的系统去完成。为此，我们重新设计 T1 中断子程序如下所示：

```
PT1:    CPL P1.2        高级中断程序发光二极管闪烁
        INC 30H         软件WATCHDOG计数器增值
        MOV A,30H
        ADD A,#0FD
        JC ERR          达到3次否?
        RETI
ERR:    CLR EA          关中断
        CLR A           准备复位地址（0000H）
        PUSH ACC
        PUSH ACC
        RETI            清除中断激活标志并复位
```

这段程序先关中断，以便后续处理能顺利进行，然后用 RETI 指令替代 LJMP 指令，从而既清除了中断激活标志又完成了转向 0000H 的任务。按这样改好后程序再运行，结果仍不理想：按下按钮后，有时只有主程序和高级中断子程序能迅速恢复正常，而低级中断仍有被关闭的可能。如果按如下方法把干扰转移到低级中断中，则按下按钮后低级中断必然被关闭：

```
LOOP:   CPL P1.0
        MOV R6,#80H
        MOV R7,#0H
TT1:    DJNZ R7,TT1
        DJNZ R6,TT1
        SJMP LOOP
PT0:    CPL P1.1
        JB 00H,STOP
```

```
        RETI
STOP:   LJMP STOP 掉入死循环。
```

仔细分析后可能得出结论：当软件 WATCHDOG 是嵌套在低级中断中起作用时，复位后只清除了高级中断激活标志，低级中断标志仍然被置位，从而使低级中断一直被关闭。

5. 修改出错处理如下：

```
ERR:    CLR EA              正确的软件复位入口
        MOV 66H,#0AAH       重建上电标志
        MOV 67H,#55H
        MOV DPTR,#ERR1      准备第一次返回地址
        PUSH DPL
        PUSH DPH
        RETI                清除高级中断激活标志
ERR1:   CLR A
        PUSH ACC
        PUSH ACC
        RETI 清除低级中断激活标志
```

这时，必须执行两次 RETI，才能到达 0000H，以保证清除全部中断激活标志，达到和硬件复位相同的效果。同样，软件陷井也必须由下列三条指令

```
NOP
NOP
LJMP STAT
```

改成：

```
NOP
NOP
LJMP ERR
```

才能达到目的。

当主程序受到干扰被软件陷阱捕获时，中断标志并未置位，执行 ERR 过程中，RETI 指令等效于 RET 指令，同样可以达到软件复位的目的。有兴趣的读者可以将软件陷阱代替死循环，分别用 LJMP STAT 和 LJMP ERR1 来替代 LJMP ERR，再将干扰检测分别设在低级中断和主程序中，实验结果必然证明同：只有 LJMP ERR 才能万无一失地实现软件复位，使系统摆脱干扰同，恢复正常。在 MCS-51 单片机的软件复位过程中，必须连续执行两次中断返回指令 RETI 才能确保系统恢复正常