CAPÍTULO VII

INTERRUPÇÕES

7.1. INTRODUÇÃO

O 8051 apresenta 5 tipos de interrupções:

- 2 externas
- 2 timers
- 1 serial

Alguns outros membros da família MCS-51 podem apresentar outras interrupções; por exemplo, o 8052 tem uma interrupção adicional dedicada ao timer 2.

7.2. REGISTROS ENVOLVIDOS

Os registros dedicados às interrupções permitem um controle total sobre as mesmas. Cada interrupção pode ser habilitada ou desabilitada individualmente. Também é possível desabilitar todas as interrupções de uma só vez. O registro **IE** (Interrupt Enable) controla a habilitação das interrupções e é ilustrado na figura 7.1.

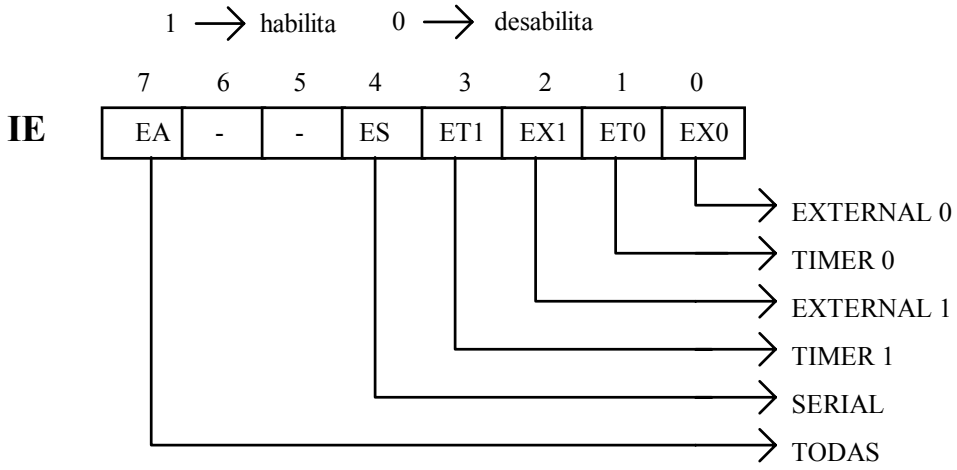


Figura 7.1. Registro IE - habilitação de interrupções.

Cada interrupção pode ter dois níveis de prioridade: prioridade alta ou prioridade baixa. Uma interrupção de alta prioridade pode interromper uma de baixa prioridade mas não acontece o contrário. Uma interrupção não pode interromper uma outra de mesma prioridade. Se forem

recebidas interrupções de diferentes prioridades, a de alta prioridade é atendida primeiro. Se forem recebidas duas interrupções de igual prioridade, determina-se por uma seqüência interna de polling (consulta) qual será atendida primeiro. Assim, dentro de um mesmo nível de prioridade, existe uma seqüência de atendimento (a seqüência do polling). O conteúdo do registro **IP** define as prioridades e é ilustrado na figura 7.2.

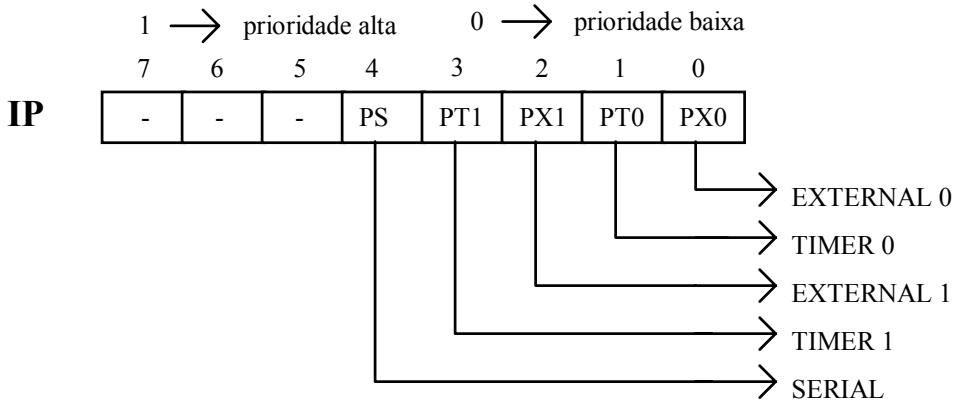


Figura 7.2. Registro IP - prioridade das interrupções.

As duas interrupções externas (INT0 e INT1) podem ser acionadas por nível ou por borda de descida (↓). Isto é definido através de dois bits do registro **TCON**. Este registro também tem outra finalidade pois cada interrupção indica sua ativação usando um bit do registro **TCON**.

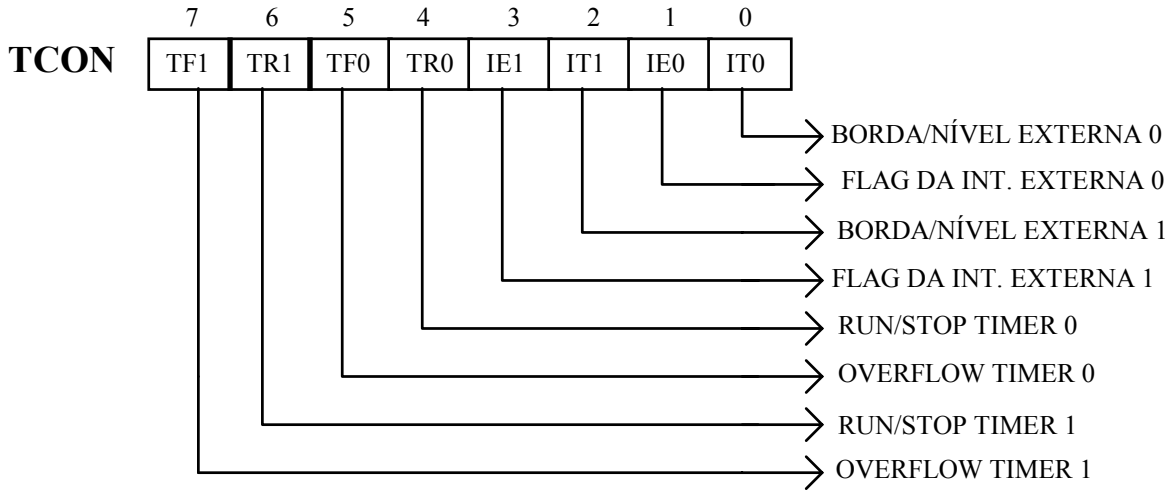


Figura 7.3. Registro TCON - diversas funções para interrupções e timers.

TF1 → flag de transbordamento (overflow) do contador/temporizador 1. Ativado por hardware quando há transbordamento no contador do timer 1 (timer/counter 1). É apagado por hardware quando o processador é desviado para a rotina de atendimento da interrupção.

TR1 → bit de partida/parada (run/stop) do contador/temporizador 1.

TF0 → flag de transbordamento (overflow) do contador/temporizador 0. Ativado por hardware quando há transbordamento no contador do timer 0 (timer/counter 0). É apagado por hardware quando o processador é desviado para a rotina de atendimento da interrupção.

TR0 → bit de partida/parada (run/stop) do contador/temporizador 0.

IE1 → flag da interrupção externa 1. É ativado (colocado em um) por hardware quando se detecta uma interrupção externa 1. É apagado (colocado em zero) por hardware (só no modo borda) quando o processador é desviado para a rotina de atendimento da interrupção.

IT1 → indica se a interrupção externa 1 opera por borda ou por nível:

1 → borda de descida (↓),

0 → nível baixo.

IE0 → flag da interrupção externa 0. É ativado (colocado um) por hardware quando se detecta uma interrupção externa 0. É apagado (colocado em zero) por hardware (só em modo borda) quando o processador é desviado para a rotina de atendimento da interrupção.

IT0 → indica se a interrupção externa 0 opera por borda ou por nível:

1 → borda de descida (↓),

0 → nível baixo.

O flag de interrupção da porta serial está em outro registro (SCON).

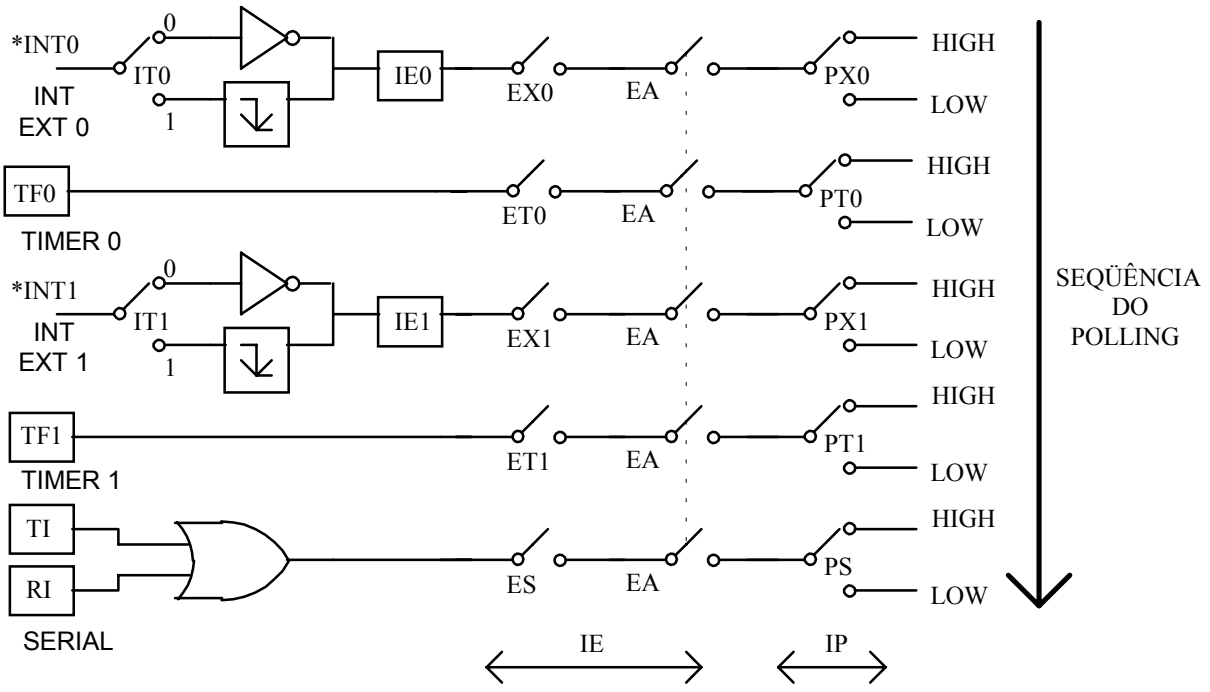


Figura 7.4. Esquema das interrupções na família MCS-51.

Todos os bits que geram interrupção podem ser ativados por software com os mesmos resultados como se tivessem sido ativados por hardware. Isto quer dizer que uma interrupção pode ser ativada por software e também que as interrupções pendentes podem ser canceladas.

Cada interrupção é vetorizada em um endereço pré-definido:

RESET		00H	
Externa 0	IE0	03H	
Timer 0	TF0	0BH	
Externa 1	IE1	13H	
Timer 1	TF1	1BH	
Serial	RI+TI	23H	(o + indica OU)

No endereço de vetorização das interrupções há pouco espaço, o suficiente para colocar umas poucas instruções. Portanto é normal colocar nestes endereços um desvio (jump) para um outro local, onde está a rotina que atende à interrupção.

7.3. MANEJO DE INTERRUPÇÕES

Os flags das interrupções são amostrados na segunda fase do quinto estado (S5P2) de cada ciclo de máquina. As amostras são submetidas a polling (consulta) durante o próximo ciclo de máquina. A figura 7.5 ilustra a seqüência para atendimento de uma interrupção.

Se um dos flags (de interrupção) está ativo durante S5P2, o próximo ciclo de polling vai detectar e identificar a interrupção a ser atendida e será gerada uma instrução LCALL sempre que não seja bloqueada por:

- uma interrupção de prioridade igual ou mais alta que a que está sendo atendida,
- não tenha finalizado a instrução que se está processando no ciclo de polling corrente,
- a instrução em progresso é um RETI ou uma escrita em IP ou IE.

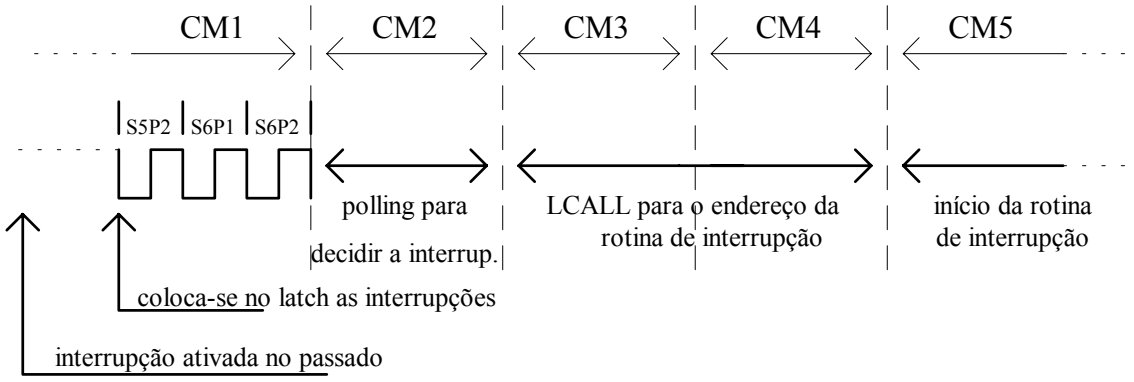


Figura 7.5. Seqüência para atendimento a uma interrupção.

Na figura 7.5 é ilustrada a resposta mais rápida a uma interrupção. Neste caso CM2 é o final de uma instrução e não é um RETI nem uma escrita em IP ou IE. Quando CM2 é uma instrução RETI ou uma escrita em IP ou IE, uma instrução a mais será executada antes que a interrupção seja vetorizada.

A seqüência de polling se repete a cada ciclo de máquina e os valores processados são aqueles que estarão presentes no instante S5P2 do ciclo de máquina anterior. Deve-se observar que se um flag de interrupção foi ativado mas não pôde ser atendido (por uma das condições de bloqueio) e é apagado antes da condição de bloqueio ser removida, a interrupção não será atendida. Quer dizer, o fato de um flag de interrupção estar ativo e não ser atendido, não será gravado.

O processador reconhece um pedido de interrupção através da execução de um LCALL gerado por hardware. Normalmente o flag que gerou o pedido é zerado por hardware, exceto para:

- TI, RI da porta serial,
- IE0, IE1 quando ativado por nível

A instrução RETI é usada para finalizar uma rotina de atendimento a interrupções. Uma instrução RET funciona mas o sistema de controle de interrupções não sabe que a rotina terminou, ou seja, serão bloqueadas todas as demais interrupções de prioridade igual ou inferior.

7.4. INTERRUPÇÕES EXTERNAS

As duas interrupções externas (0 e 1) podem ser programadas para funcionar por nível (ITX=0) ou por borda de descida (↓) (ITX=1). Os pinos das interrupções são amostrados em cada ciclo de máquina (1 ciclo de máquina=12 períodos de clock); assim, uma interrupção externa deve permanecer constante por pelo menos 12 períodos de clock, caso contrário ela pode ser ignorada.

Quando a interrupção opera por borda de descida (↓), o pino deve permanecer em alto por pelo menos 12 períodos de clock e depois em baixo por pelo menos 12 períodos de clock; isto garante a ativação do flag de interrupção. Os flags de interrupções externas (IEX0 ou IEX1) serão automaticamente zerados pela CPU quando uma rotina de serviço é chamada.

Quando a interrupção opera por nível, o pino deve permanecer ativado (em baixo) até que a interrupção seja atendida; depois disso, o pedido deve ser removido antes que a rotina de serviço de interrupção termine caso contrário uma nova interrupção será gerada.

O tempo de latência de uma interrupção depende, entre outras coisas, das condições de bloqueio já vistas. No caso destas condições de bloqueio existirem, leva-se entre 5 a 9 ciclos de máquina.

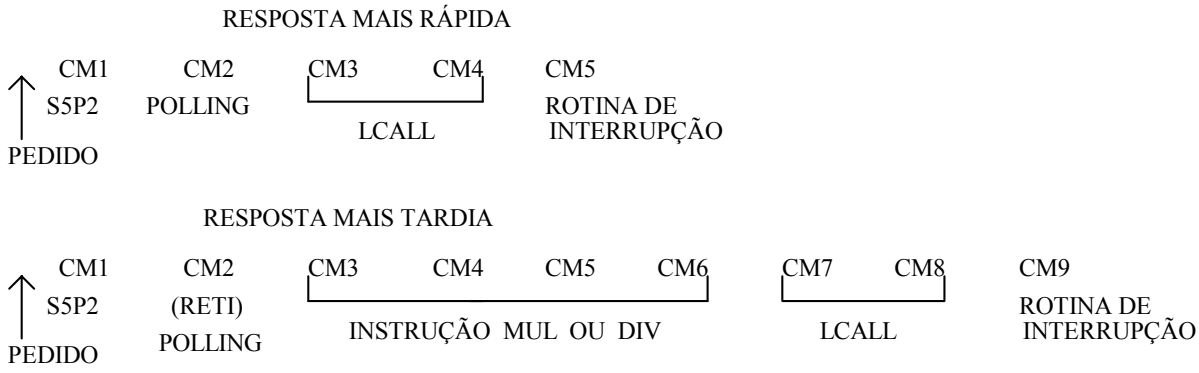
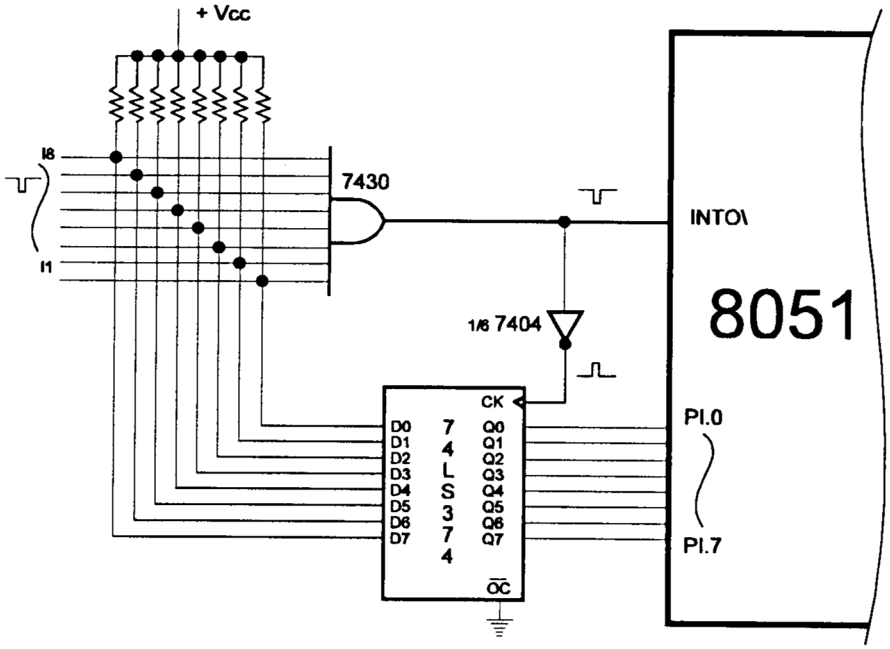


Figura 7.6. Latência para as interrupções.

Como os flags de interrupção serial e dos temporizadores/contadores são ativados em S5P2, o que foi estudado é válido para qualquer interrupção.

Caso seja necessário, pode-se usar um esquema como o da figura a seguir para expandir as interrupções externas:



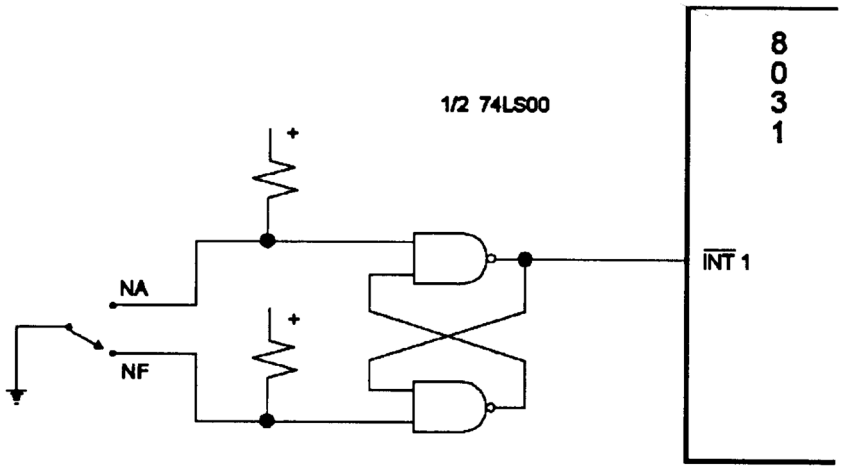
7.5. PASSO A PASSO

A estrutura de interrupção do MCS-51 permite, de forma muito simples, uma implementação para a execução de programas passo a passo (single step). Como já foi visto, uma interrupção não será atendida quando uma interrupção de igual prioridade esteja sendo atendida e, quando esta interrupção termina com um RETI, obrigatoriamente uma instrução será executada antes que se aceite a nova interrupção.

Uma interrupção não será re-atendida antes de terminar sua rotina. Essa rotina deve terminar com RETI e, depois dessa instrução, uma outra deverá ser executada antes de ser atendida a interrupção.

Imagine um sistema onde a interrupção 1 está programada para trabalhar por nível e que a entrada INT1 esteja conectada a terra, ou seja, a interrupção está sempre sendo invocada. Se forem adicionadas a esse sistema uma rotina principal e uma rotina de serviço para a interrupção 1, cria-se o ambiente para executar passo a passo o programa principal. Cada vez que voltar da rotina de interrupção (com um RETI), será executada uma única instrução do programa principal e, em seguida, desviará novamente para a rotina de interrupção. Pode-se usar uma chave para marcar o instante de retorno da rotina de interrupção. O exercício 7.4 ilustra a utilização do recurso de passo a passo.

A figura 11.5 (capítulo 11) mostra uma implementação possível para a execução passo a passo. Outra possibilidade, mais complexa mas sem bouncing na chave é mostrada na figura a seguir:



7.6. EXERCÍCIOS

EXERCÍCIO 7.1. LED_INT1, LED_INT2 E LED_INT3

No circuito de práticas existe um pushbutton (SW3) para a interrupção 1 que será usado para fazer acender os leds (vermelho, amarelo e verde) em seqüência. Cada vez que a chave for acionada, uma interrupção será gerada e a rotina de atendimento à interrupção deverá trocar o led aceso.

```
;LED_INT1.ASM
                DEFSEG PROG, CLASS=CODE, START=0
                SEG PROG
;
                ORG             RESET
                AJMP            INICIO
;
                ORG             EXTI1
                AJMP            EXT1
;
                ORG             50H
INICIO          MOV             A,#01001001B
                CLR             C
                MOV             P1,A
                SETB            EX1             ;HABILITAR INT. EXT. 1
                SETB            EA              ;HABILITAR FLAG GERAL
                SJMP            $               ;LOOP INFINITO
;
;ROTINA PARA ATENDER A INTERRUPCAO EXTERNA 1
EXT1            RLC             A               ;ACENDER LEDS EM SEQUENCIA
                MOV             P1,A
                RETI
                END
```

Nota-se um problema: os 3 leds se acendem quando a chave é acionada e só um led fica aceso quando se libera a chave, mas nunca se sabe qual. Este problema acontece porque a rotina de interrupção é muito mais rápida que a chave. Ao retornar da interrupção, a chave ainda está acionada e uma nova interrupção é gerada. Para solucionar isso, pode-se monitorar o sinal INT1 (P3.3) e só permitir o retorno depois de que a chave esteja liberada (vai para um).

Para ilustrar esta solução, o programa LED_INT2.ASM foi implementado. O programa é idêntico ao anterior exceto por uma instrução antes do retorno da interrupção.

```
;LED_INT2.ASM
                DEFSEG PROG, CLASS=CODE, START=0
                SEG PROG
SW3             EQU             P3.3
;
                ORG             RESET
                AJMP            INICIO
;
                ORG             EXTI1
                AJMP            EXT1
;
                ORG             50H
INICIO          MOV             A,#01001001B
                CLR             C
                MOV             P1,A
                SETB            EX1             ;HABILITAR INT. EXT. 1
                SETB            EA              ;HABILITAR FLAG GERAL
                SJMP            $               ;LOOP INFINITO
;
;ROTINA PARA ATENDER A INTERRUPCAO EXTERNA 1
EXT1            RLC             A               ;ACENDER LEDS EM SEQUENCIA
                MOV             P1,A
```

```
AQUI            JNB             SW3,AQUI        ;AGUARDAR LIBERAR SW2
                CLR             IE1
                RETI
                END
```

Na verdade, o que se fez foi obrigar que a interrupção que está programada para operar por nível opere por borda (aguarda-se a liberação da chave). O que acontece quando se usa o primeiro programa mas com a interrupção trabalhando por borda (↓) ? Supõe-se que os leds deverão acender na seqüência correta.

```
;LED_INT3.ASM
                DEFSEG PROG, CLASS=CODE, START=0
                SEG PROG
;
                ORG             RESET
                AJMP            INICIO
;
                ORG             EXTI1
                AJMP            EXT1
;
                ORG             50H
INICIO          MOV             A,#01001001B
                CLR             C
                MOV             P1,A
                SETB            IT1             ;EXT 1 OPERA POR BORDA
                SETB            EX1             ;HABILITAR INT. EXT. 1
                SETB            EA              ;HABILITAR FLAG GERAL
                SJMP            $               ;LOOP INFINITO
;
;ROTINA PARA ATENDER A INTERRUPCAO EXTERNA 1
EXT1            RLC             A               ;ACENDER LEDS EM SEQUENCIA
                MOV             P1,A
                RETI
                END
```

Talvez se note um acionamento indevido ao liberar a chave porque a chave é ruidosa e, ao ser liberada, aparece o bouncing (seqüências de zeros e uns). Uma solução definitiva é usar a rotina de eliminação de bouncing para controlar a transição de 1 para 0.

EXERCÍCIO 7.2. PASSO_EXE.ASM

Escrever um programa que faça acender os leds em seqüência (vermelho, amarelo e verde). Executar esse programa com um controle passo a passo usando a interrupção 1. A solução é muito simples. No programa principal estará somente uma seqüência de instruções que acende os leds segundo a ordem especificada (sem retardo). A interrupção externa 1 estará programada para operar por nível. A rotina de interrupção aguarda que a chave SW3 seja acionada (vá a zero); quando isto ocorre, retorna-se da rotina de interrupção. Ao regressar, a entrada INT1 estará em zero (pela chave SW3) e uma nova interrupção é gerada. Ou seja, a cada acionamento da chave SW3, é executada uma instrução do programa principal. A primeira interrupção, para que o programa entre em controle passo a passo, é provocada por software ao ativar o bit IE1.

```
;PASSO_EXE.ASM
;
; PROGRAMA PARA ILUSTRAR A EXECUCAO PASSO A PASSO
; SERA USADA A CHAVE SW3 PARA EXECUTAR PASSO A PASSO O LOOP PRINCIPAL
; CADA VEZ QUE SE ACIONA SW3, UMA INSTRUCAO É EXECUTADA
;
RTD             EQU             100
```

```
SW3             EQU             P3.3
;
                DEFSEG PROG, CLASS=CODE, START=0
                SEG             PROG
;
                ORG             RESET
                AJMP            INIC
;
                ORG             EXTI1
                AJMP            EXT1
;
                ORG             50H
INIC            MOV             A,#01001001B    ;CODIGO PARA ACENDER
                CLR             C               ;LEDS EM SEQUENCIA
                MOV             P1,A
                SETB            EX1             ;HABILITAR INT 1
                SETB            EA              ;HABILITAR GERAL
                SETB            IE1             ;PROVOCAR INT 1
                NOP                             ;AQUI O PROG INTERROMPE
PRINCIPAL       RLC             A               ;RODAR CODIGO
                MOV             P1,A            ;ACENDER LEDS
                SJMP            PRINCIPAL       ;RETORNAR
;
                ORG             100H
EXT1            ACALL           RBT_0_1         ;TRANSICAO DE 0 A 1
                ACALL           RBT_1_0         ;TRANSICAO DE 1 A 0
                RETI
;
;ELIMINAR BOUNCING NAS TRANSICOES DE 0 PARA 1
RBT_0_1         MOV             R7,#RTD
LB1             JNB             SW3,RBT_0_1
                DJNZ            R7,LB1
                RET
;
;ELIMINAR BOUNCING NAS TRANSICOES DE 1 PARA 0
RBT_1_0         MOV             R7,#RTD
LB2             JB              SW3,RBT_1_0
                DJNZ            R7,LB2
                RET
                END
```