Arquitetura de Computadores

PROF. DR. ISAAC

Instruções do MSC-51

Tipo	Quantidade
Aritméticas	24
Lógicas	25
Transferência (cópia) de Dados	28
Booleanas	17
Saltos	17

Instruções aritméticas: envolvem operações do tipo soma, subtração, multiplicação, divisão, incremento e decremento.

Instruções lógicas: fazem operações bit a bit com registradores e também rotações.

Instruções do MSC-51

Instruções de transferência (cópia) de dados: copiam bytes entre os diversos registradores e a RAM interna.

Instruções booleanas: essas instruções são denominadas booleanas porque trabalham com variáveis lógicas (variável de 1 bit). Como o próprio nome sugere, elas são talhadas para resolver expressões booleanas.

Instruções de desvio: desviam o fluxo de execução do programa, chamam subrotinas, fazem desvios condicionais e executam laços de repetição.

Instruções de Desvio

Instruções de Desvio:

Chamadas de Subrotinas

Chamadas de Subrotinas

As **subrotinas** são úteis para evitar a repetição de trechos de programas. Antes de efetivar o desvio para a subrotina, o processador armazena na pilha o endereço de retorno, para que possa recuperá-lo por ocasião do regresso.

Esse empilhamento de endereços permite que, de dentro de uma subrotina, se faça chamada para uma outra subrotina.

Instrução - ACALL

Operação: ACALL

Função: Chamada absoluta dentro do bloco de 2K

Sintaxe: ACALL *endereço*

Descrição : ACALL chama uma subrotina localizada no endereço indicado. Nenhuma flag é afetada.

Exemplo:

- ACALL LABEL

Instrução - LCALL

Operação: LCALL

Função: Long Call

Sintaxe: LCALL *endereço_codigo*

Descrição : LCALL chama uma subrotina do programa. O endereço da próxima instrução a ser executada é inserido na pilha antes que o PC desvie para a subrotina.

Exemplo:

- LCALL SUB1

Instrução - LCALL

		Bytes	MC	Op1	Op2	Op3
LCALL	end16	3	2	12	MSB(end16)	LSB(end16)
ACALL	end11	2	2	[(MSB(end11))<<5]OU11H	LSB(end11)	-

Instruções de chamada de subrotinas.

Instruções de Desvio:

Retorno das Subrotinas

Instrução - RET

Operação: RET

Função: Retorna de uma subrotina

Sintaxe: RET

Descrição : RET é usado para retornar de uma subrotina chamada por LCALL ou ACALL. A execução do programa continua do endereço (2 bytes) restaurados da pilha. Primeiro o byte mais significativo é retirado, seguido pelo menos significativo.

Exemplo:

- RET

Instrução - RETI

Operação: RETI

Função: Retorna da Interrupção

Sintaxe: RETI

Descrição : RETI é usado para retornar de um serviço de interrupção.

Exemplo:

Instruções – RET e RETI

	Bytes	MC	Op
RET	1	2	22
RETI	1	2	32

Instruções de retorno de sub-rotinas.

Exemplo

Teste e verifique o que faz esse programa, observe o valor do SP e o que está sendo armazenado na Pilha.

```
Principal:
        MOV   A, #01BH
        ACALL FUNC01
        SJMP $


org 0150h

FUNC01:
        MOV   30H, #0FFH
        ACALL FUNC02
        NOP
        RET
```

```
org 0270h

FUNC02:
        MOV   40H, #0AAH
        ACALL FUNC03
        NOP
        RET

org 03B0h

FUNC03:
        MOV  40H, #11H
        RET
```

Mecanismos de Interrupção e Exceção

Interrupções

- Acontecem quando o controlador recebe um sinal requisitando a execução de uma sub-rotina específica;
- O controlador troca a execução do código principal pelo da interrupção e depois retorna ao código principal

Interrupções

Praticamente todos os computadores oferecem um mecanismo por meio do qual outros módulos (E/S, memória) podem interromper o processamento normal do computador.
A tabela abaixo lista as classes mais comuns de interrupção:

Programa	Gerada por alguma condição que ocorre como resultado da execução de uma instrução, como o *overflow* aritmético, divisão por zero, tentativa de executar uma instrução de máquina ilegal ou referência fora do espaço de memória permitido para o usuário.
Timer	Gerada por um timer dentro do processo. Isso permite que o sistema operacional realize certas funções regularmente.
E/S	Gerada por um controlador de E/S para sinalizar o término normal de uma operação ou para sinalizar uma série de condições de erro.
Falha de hardware	Gerada por uma falha como falta de energia ou erro de paridade de memória.

Interrupções

As interrupções são um mecanismo essencial nos sistemas computacionais modernos, permitindo que os processadores respondam a eventos externos ou internos de forma eficiente.

Uma interrupção é um sinal enviado para o processador que indica a ocorrência de um evento que requer atenção imediata. Quando o processador recebe uma interrupção, ele suspende temporariamente a execução do programa atual e passa a executar uma rotina de tratamento de interrupção (Interrupt Service Routine - ISR) associada ao evento específico.

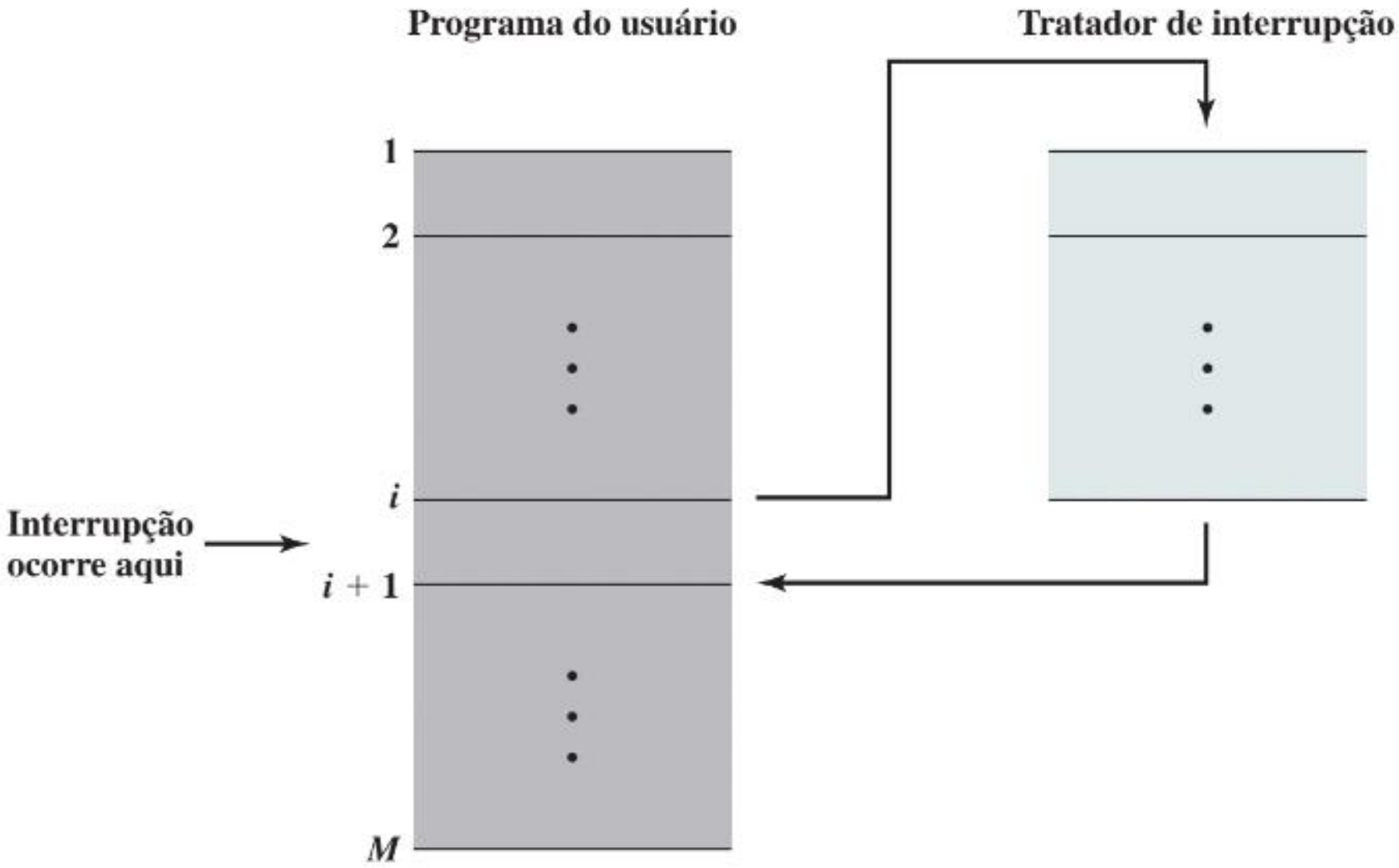
Interrupções

As interrupções podem ser categorizadas como:

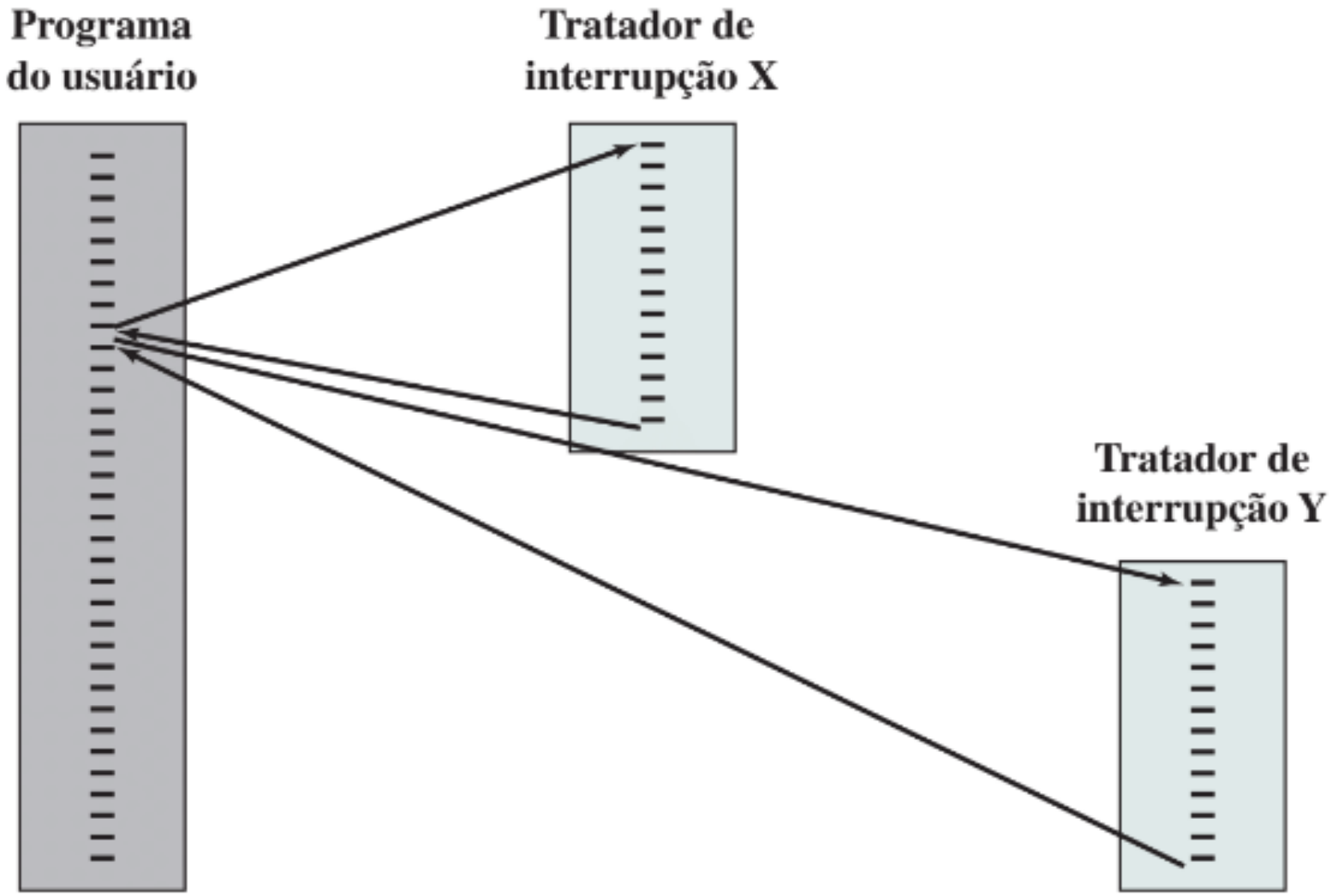
Interrupções de hardware (externas): Essas são geradas por dispositivos de hardware externos ao processador, como periféricos (teclado, mouse, disco rígido) ou controladores de interrupção. Geralmente, são usadas para sinalizar a conclusão de uma operação de E/S ou para solicitar atenção do processador em resposta a eventos externos.

Interrupções de software (internas): Essas são geradas internamente pelo próprio processador, geralmente como resultado de uma instrução de software específica (por exemplo, uma chamada de sistema) ou devido a condições excepcionais (como uma divisão por zero, acesso ilegal à memória ou estouro de pilha). As interrupções de software são usadas para implementar funcionalidades do sistema operacional, como chamadas de sistema e gerenciamento de exceções.

Interrupções



Interrupções



Interrupções básicas

Formas de Gerenciamento de Variáveis num Sistema Microcontrolado (Por varredura e por interrupção)

- **Por varredura**: é realizado dentro do programa principal;

- **Por interrupção**: quando um sinal elétrico conectado a um pino do microcontrolador dispara a execução de uma instrução de chamada a sub-rotina de atendimento a uma fonte de interrupção que deve ser armazenada num endereço pré-definido pelo fabricante;

Funcionamento da Interrupção no programa

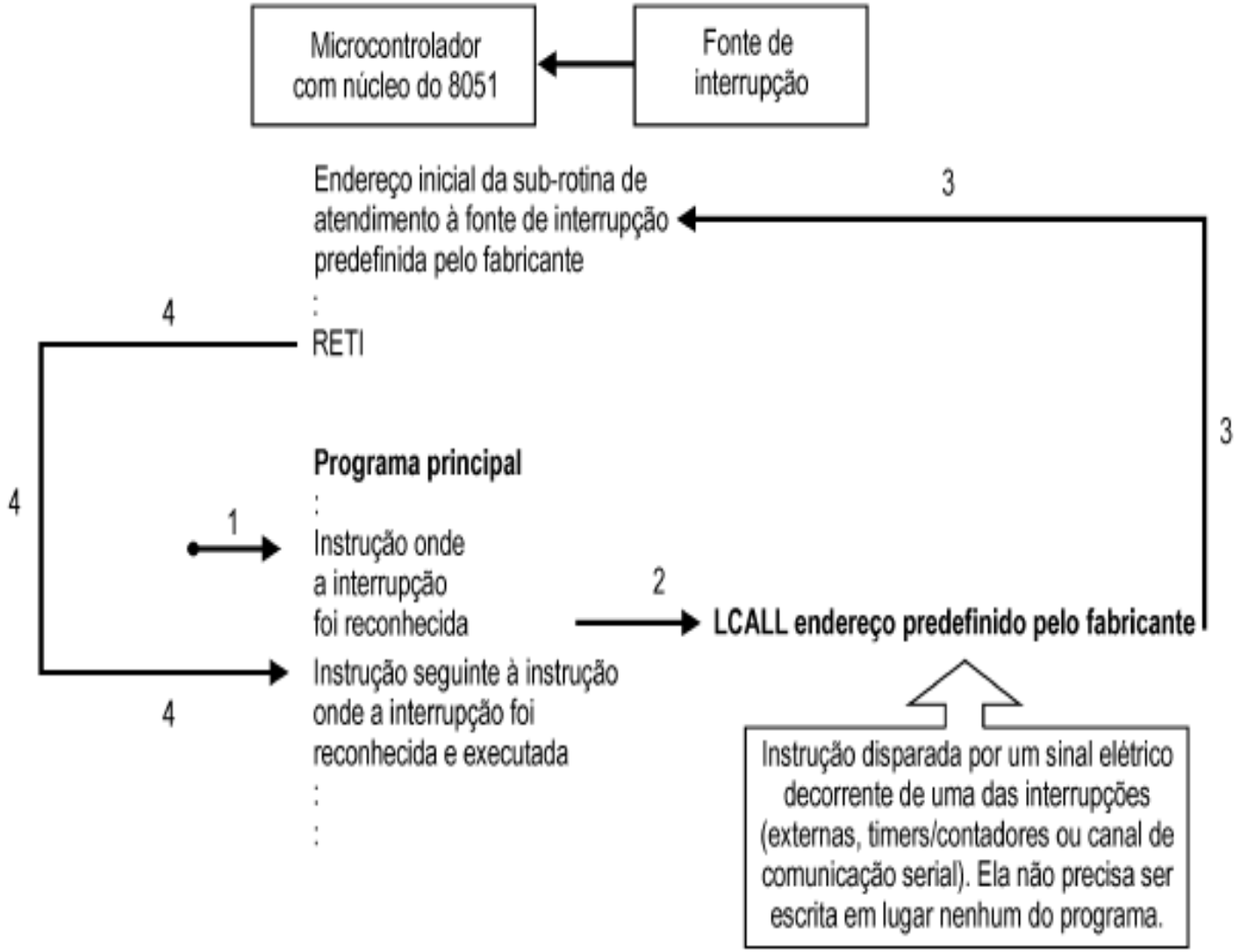
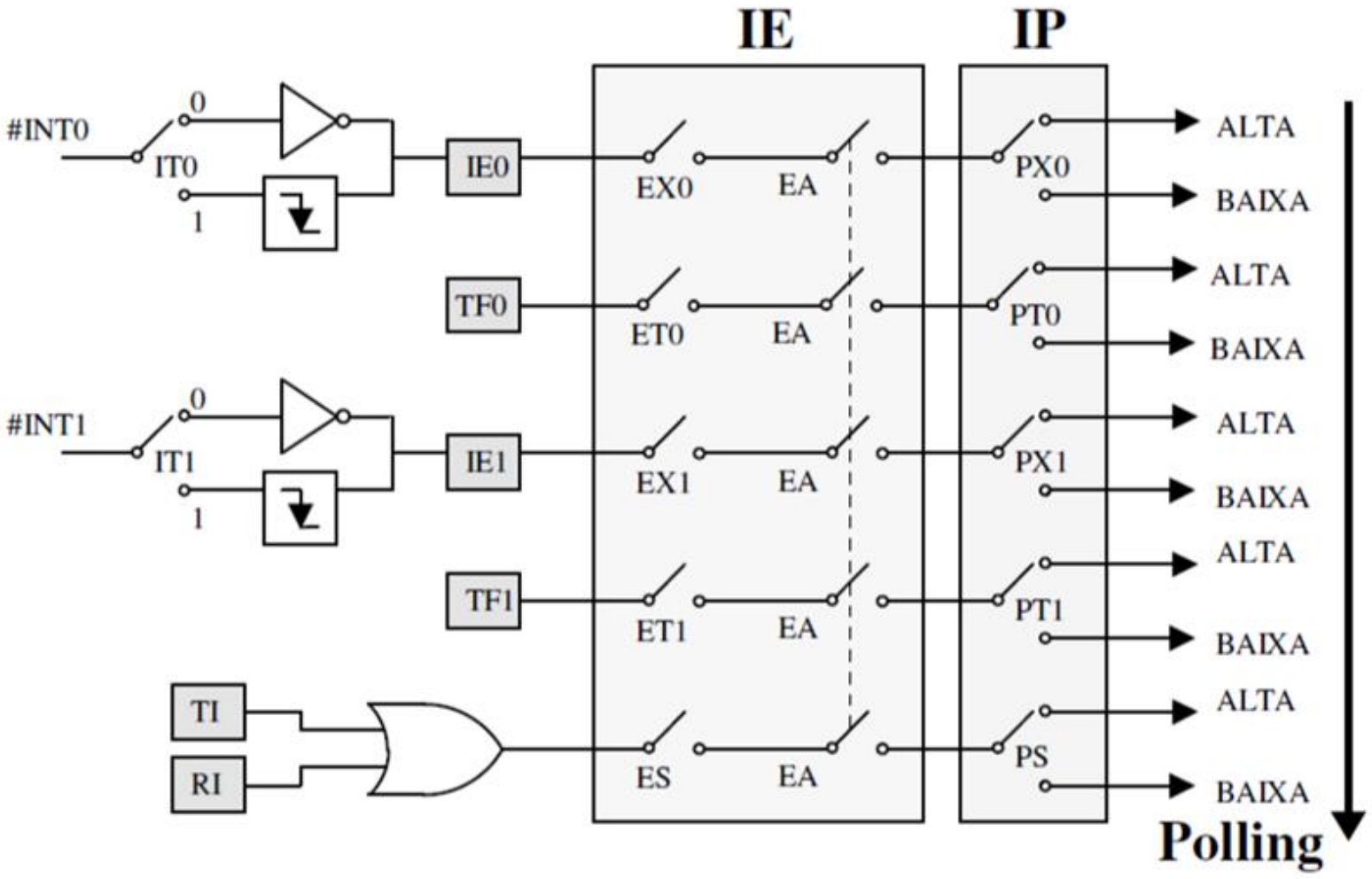


Diagrama das interrupções do 8051



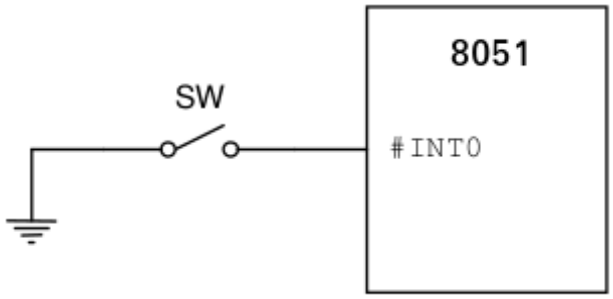
Endereço de desvio das interrupções do 8051

Fonte de interrupção	Nome da fonte de interrupção	Endereço vetor
RESET	*Reset*	0000h
IE0	Fonte de interrupção externa 0	0003h
TF0	Fonte de interrupção do *timer*/contador 0	000Bh
IE1	Fonte de interrupção externa 1	0013h
TF1	Fonte de interrupção do *timer*/contador 1	001Bh
RI + TI	Fonte de interrupção do canal de comunicação serial	0023h
TF2 + EXF2	Fonte de interrupção do *timer*/contador 2 + externa 2	002Bh

Exemplo de Interrupção Externa

Exemplo 1

Crie um programa com interrupção externa INT0 (pino P3.2) que realize o complemento do pino P1.0 cada vez que a interrupção for acionada.



Exemplo 1

Solução:

```
org 0000h
        LJMP START              ;Pula incondicionalmente para START
org 0003h
INT_EXT0:
        CPL P1.0                ;complementa P1.0
        RETI                    ;Retorna da interrupção
org 0080h
START:
        SETB EA                 ;Habilita as interrupções
        SETB EX0                ;Habilita a interrupção 0
        SETB IT0                ;Trabalhando com borda de descida
        SJMP $                  ;Laço de repetição
```

Exemplo 1

Solução:

```
org 0000h
        LJMP START              ;Pula incondicionalmente para START
org 0003h
INT_0:
        CPL P1.0                ;complementa P1.0
        RETI                    ;Retorna da interrupção
org 0080h
START:
        SETB EA                 ;Habilita as interrupções
        SETB EX0                ;Habilita a interrupção 0
        SETB IT0                ;Trabalhando com borda de descida
        SJMP $                  ;Laço de repetição
```

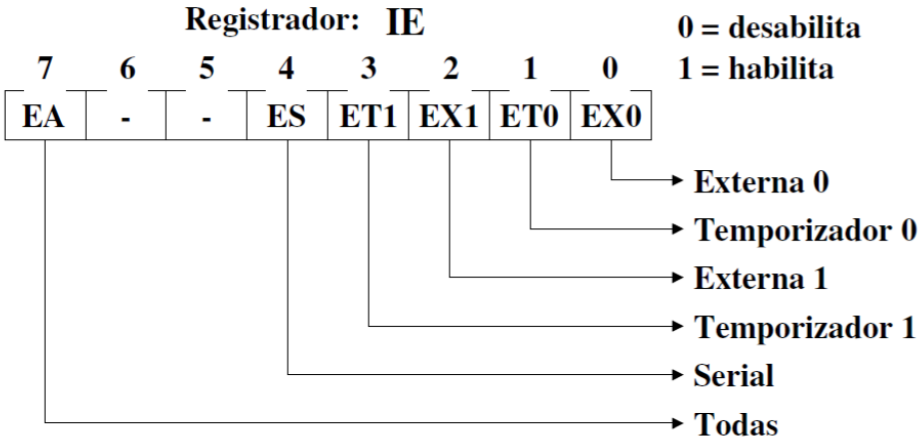
Exemplo 1

Pedido	Interrupção	Endereço
IE0	Externa 0	0003H
TF0	Temporizador 0	000BH
IE1	Externa 1	0013H
TF1	Temporizador 1	001BH
TI ou RI	Serial	0023H

Solução:

```
org 0000h
        LJMP START              ;Pula incondicionalmente para START
org 0003h
INT_0:
        CPL P1.0                ;complementa P1.0
        RETI                    ;Retorna da interrupção
org 0080h
START:
        SETB EA                 ;Habilita as interrupções
        SETB EX0                ;Habilita a interrupção 0
        SETB IT0                ;Trabalhando com borda de descida
        SJMP $                  ;Laço de repetição
```

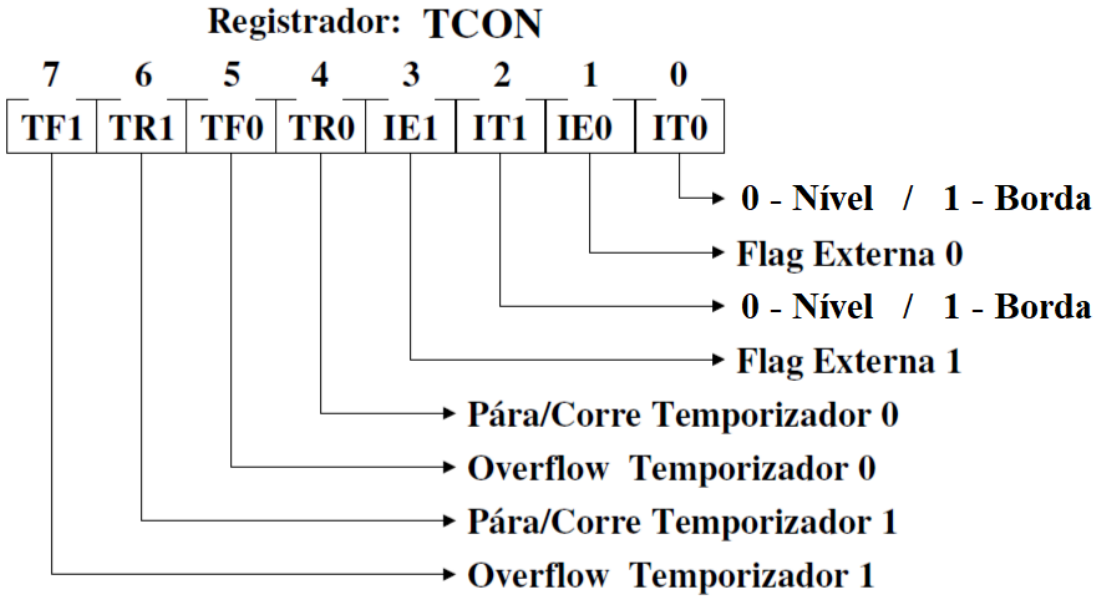
Exemplo 1



Solução:

```
org 0000h
        LJMP START              ;Pula incondicionalmente para START
org 0003h
INT_0:
        CPL P1.0                ;complementa P1.0
        RETI                    ;Retorna da interrupção
org 0080h
START:
        SETB EA                 ;Habilita as interrupções
        SETB EX0                ;Habilita a interrupção 0
        SETB IT0                ;Trabalhando com borda de descida
        SJMP $                  ;Laço de repetição
```

Exemplo 1



Solução:

```
org 0000h
        LJMP START              ;Pula incondicionalmente para START
org 0003h
INT_0:
        CPL P1.0                ;complementa P1.0
        RETI                    ;Retorna da interrupção
org 0080h
START:
        SETB EA                 ;Habilita as interrupções
        SETB EX0                ;Habilita a interrupção 0
        SETB IT0                ;Trabalhando com borda de descida
        SJMP $                  ;Laço de repetição
```

Exemplo 1

Solução:

```
org 0000h
        LJMP START              ;Pula incondicionalmente para START
org 0003h
INT_EXT0:
        CPL P1.0                ;complementa P1.0
        RETI                    ;Retorna da interrupção
org 0080h
START:
        SETB EA                 ;Habilita as interrupções
        SETB EX0                ;Habilita a interrupção 0
        SETB IT0                ;Trabalhando com borda de descida
        SJMP $                  ;Laço de repetição
```

Instruções de Desvio:

Laços de Repetição

Instrução - DJNZ

Operação: DJNZ (decrement and jump if not zero**).**

Função: Decrementa e desvia se não for zero

Sintaxe: DJNZ *registrador*, *endereço*

Descrição : DJNZ decrementa o valor do *registrador*. Se o novo valor do registrador não for zero, o programa desviará para o endereço especificado em *endereço*. Essa instrução é análoga ao comando for que existe em C.

Exemplo:

- DJNZ 40h, LABEL
- DJNZ R6, LABEL

Instrução - DJNZ

			Bytes	MC	Op1	Op2	Op3
DJNZ	Rn	,rel	2	2	D8+n	rel	-
	end8		3	2	D5	end8	rel

Instruções para construir laços de programa.

Exercícios

Exercício 2

Exercício 1:

Construir e testar programa-fonte em linguagem assembly que invoca uma **sub-rotina** que deve carregar (alocar) o valor EEh em 80 bytes consecutivos da RAM interna iniciando no endereço 20h. O programa deve iniciar no endereço 0080h.

Exercício 3

Exercício 2:

Crie um programa que fique alternando a rotação a esquerda e rotação a direita na porta P1, faça com que essa rotação seja alternada a cada 3 voltas. Faça com que seu programa chame **duas sub-rotinas**, uma para a rotação a esquerda e a outra para a rotação a direita. O programa deve iniciar no endereço 0080h.

Bibliografia

ZELENOVSKY, R.; MENDONÇA, A. Microcontroladores Programação e Projeto com a Família 8051. MZ Editora, RJ, 2005.

Gimenez, Salvador P. Microcontroladores 8051 - Teoria e Prática, Editora Érica, 2010.