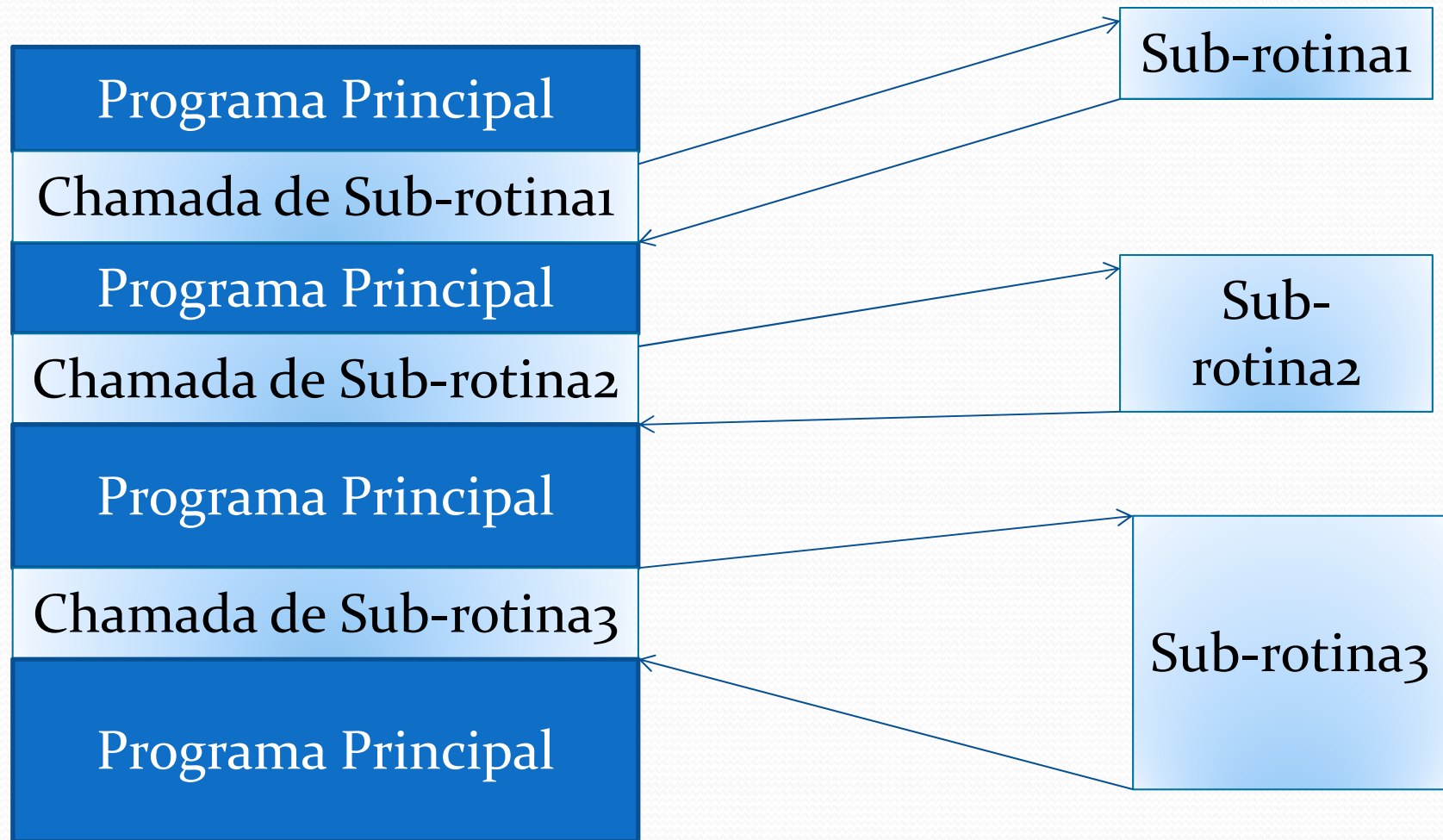


8051

AULA 3

Interrupção

Estrutura de Programação Assembly



Sub-Rotina

- É um procedimento estabelecido pelo programador que executa alguma tarefa específica.

- Cada chamada de dentro do Programa Principal causa um desvio automático para o endereço da sub-rotina chamada.

- Ao terminar a execução da sub-rotina o Contador de Programa (PC) retorna a execução para a próxima instrução dentro do Programa Principal, após a instrução de chamada.

- O endereço de retorno da Sub-rotina é armazenado na Pilha no endereço apontado pelo Stack Pointer (SP).

Exemplo:

```
ORG 0
MOV A, #0FFH
ACALL DELAY
MOV A, #00
SJMP $
; *****
;
; DELAY:
;
;     MOV R3, #005h
;     MOV R2, #0F6h
;     MOV R1, #003h
;     MOV R0, #085h
;     NOP
;     DJNZ R0, $
;     DJNZ R1, $-5
;     DJNZ R2, $-9
;     DJNZ R3, $-13
;     RET
; *****
;
; END
```

ACALL muda a execução do programa para o endereço da sub-rotina DELAY

RET retorna a execução do programa para a próxima instrução abaixo da chamada **ACALL**

O endereço de retorno (posição da instrução MOV A#00) é armazenado na Pilha definida por SP e recuperado pela instrução RET

Sub-rotinas re-entrantes

- De dentro de uma Sub-rotina pode ser chamada outra sub-rotina.

A cada chamada um endereço de retorno é armazenado na Pilha. A cada retorno o endereço é removido da Pilha.

Exemplo:

```
ORG 0
MOV A, #0FFh
ACALL DELAY
MOV A, #00
SJMP $

; *****
DELAY:  ACALL VAL
        DJNZ R0, $
        DJNZ R1, $-5
        DJNZ R2, $-9
        DJNZ R3, $-13
        RET

; *****
VAL:    MOV R3, #005h
        MOV R2, #0F6h
        MOV R1, #003h
        MOV R0, #085h
        NOP
        RET

END
```

ERROS COMUNS!!!!!!

```

ORG 0
MOV A, #0FFH
SJMP DELAY
MOV A, #00
SJMP $
;
;*****
DELAY:  ACALL VAL
        DJNZ R0, $
        DJNZ R1, $-5
        DJNZ R2, $-9
        DJNZ R3, $-13
        RET
;*****
;
VAL:    MOV R3, #005h
        MOV R2, #0F6h
        MOV R1, #003h
        MOV R0, #085h
        NOP
        RET
;
END

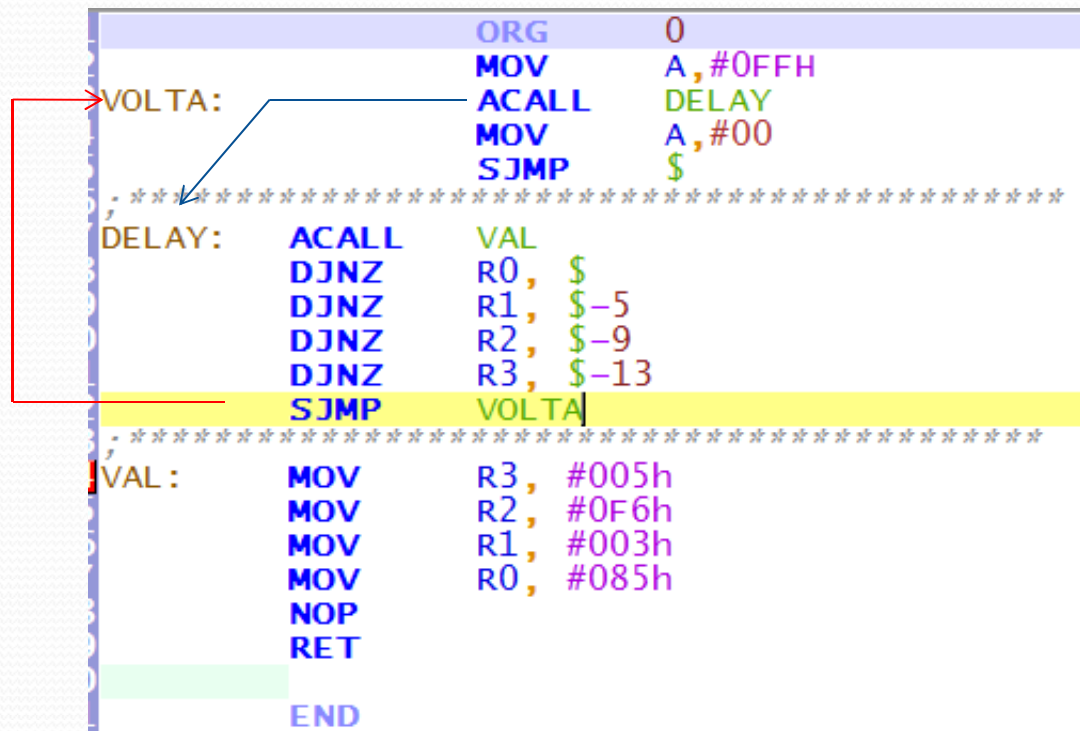
```

?????

Saltar com JUMP para dentro de Sub-rotinas.

Consequência: O PC será carregado com o último valor armazenado na PILHA (???). O programa se perderá.

ERROS COMUNS!!!!!!



```
ORG 0
MOV A, #0FFH
VOLTA: ACALL DELAY
MOV A, #00
SJMP $

; *****
DELAY: ACALL VAL
DJNZ R0, $
DJNZ R1, $-5
DJNZ R2, $-9
DJNZ R3, $-13
SJMP VOLTA

; *****
VAL: MOV R3, #005h
MOV R2, #0F6h
MOV R1, #003h
MOV R0, #085h
NOP
RET

END
```

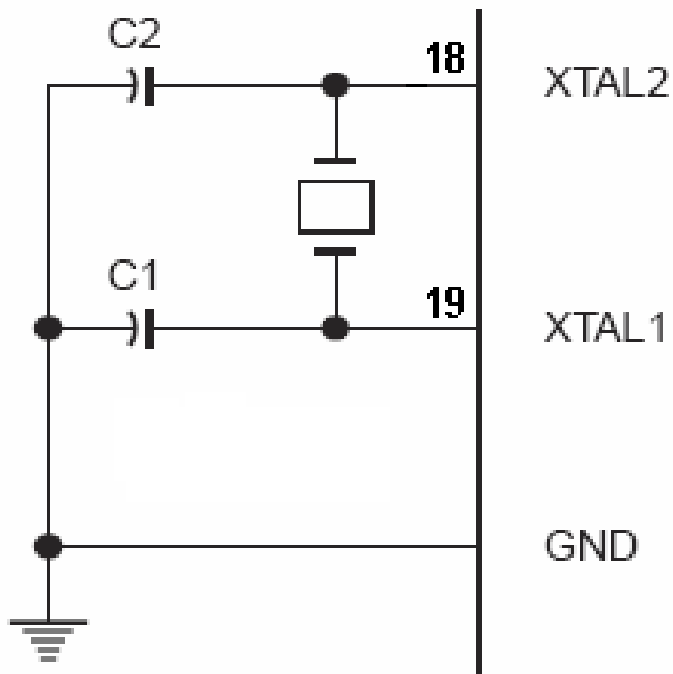
Saltar de Sub-rotinas com JUMP para dentro de Programas.

Consequência: Estouro da PILHA

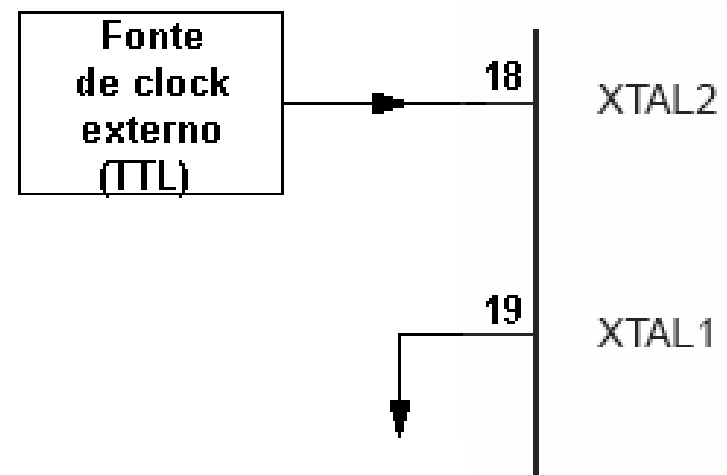
Temporização da CPU

Todos os Microcontroladores da família MCS-51 têm um oscilador interno.

- Para uso deste oscilador deve-se conectar um cristal entre os pinos Xtal1 e Xtal2 da CPU.



- Pode-se também utilizar um oscilador externo:

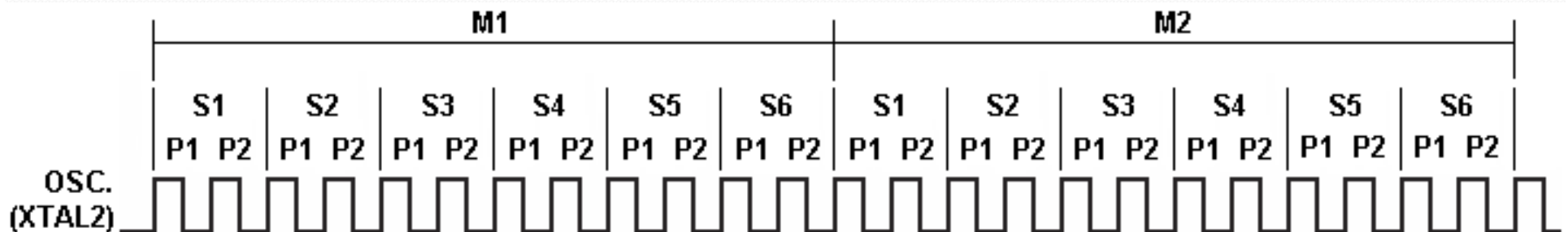


Ciclos de Máquina

- Um ciclo de máquina (M) consiste de uma seqüência de 6 estados (S1 a S6).
- Cada estado é formado por 2 períodos de clock (P1 e P2).

Logo :

1 ciclo de máquina (M) = 12 períodos de clock (P)



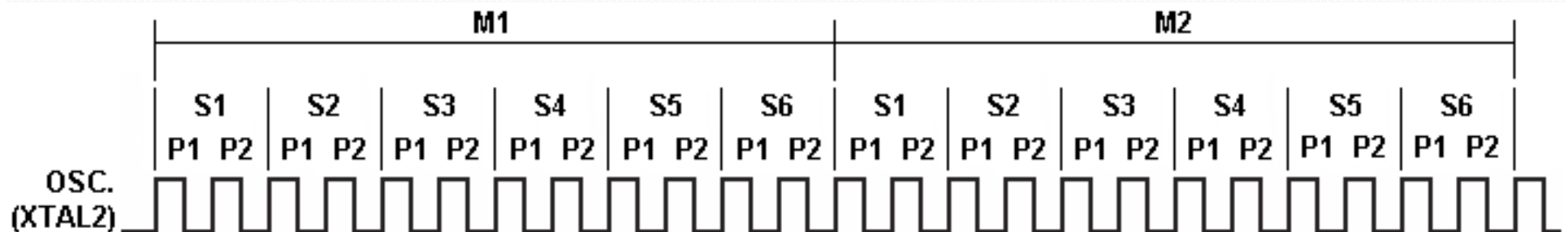
Ciclos de Máquina

Se o cristal é de 12 Mhz:

$$P = \frac{1}{12 \cdot 10^6} \quad (\text{Período do clock})$$

Ciclo de Máquina (M):

$$M = 12 \cdot P = 12 \cdot \frac{1}{12 \cdot 10^6} = 1 \mu s$$

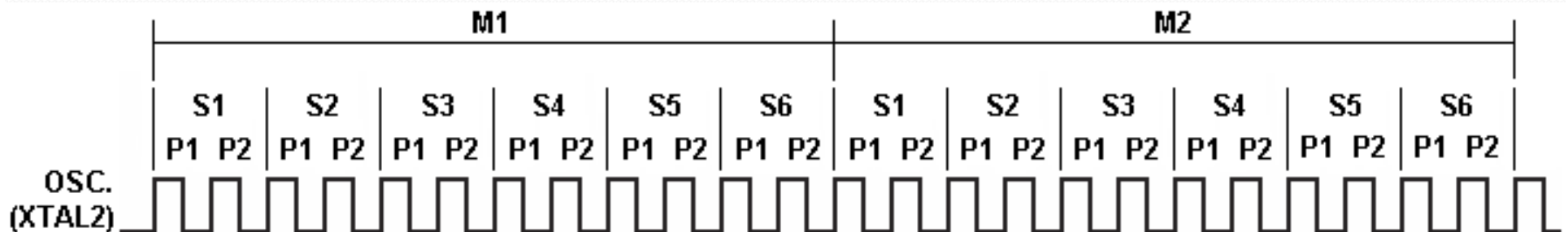


Ciclos de Máquina

- As instruções dos microcontroladores da família MCS-51 utilizam 12 ou 24 períodos de clock, com exceção das instruções **MUL AB** e **DIV AB** que utilizam 48 períodos .

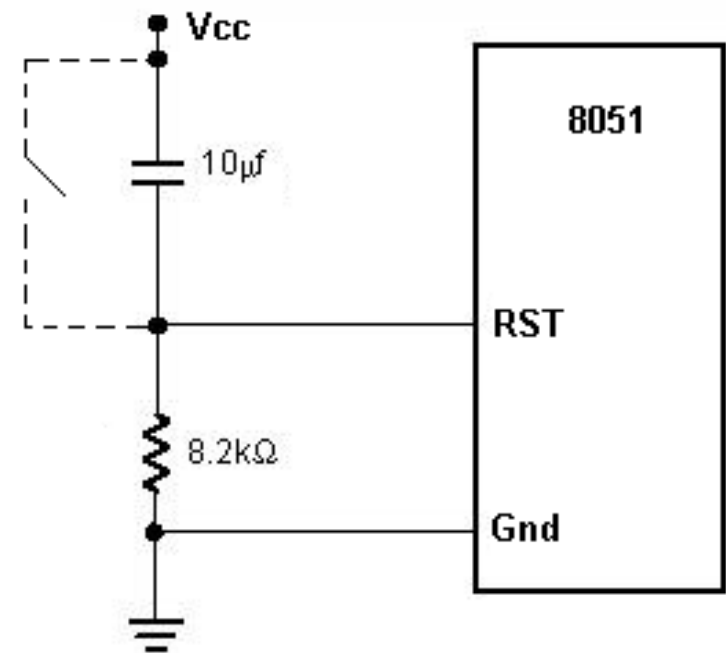
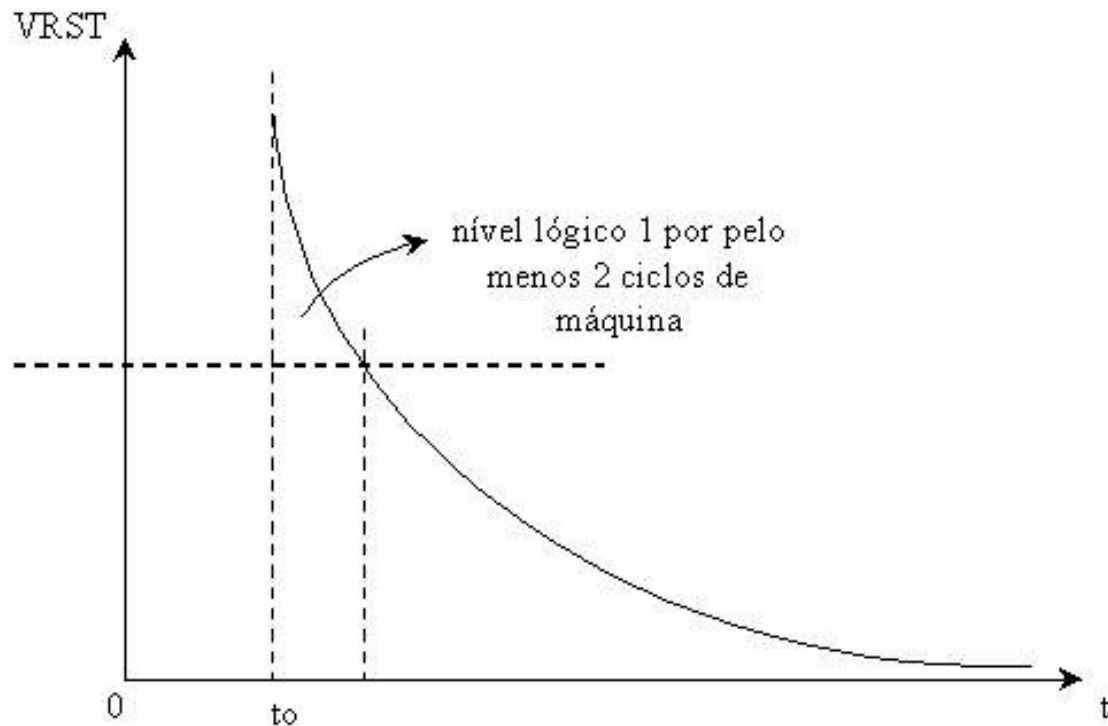
Exemplo : Com cristal de 12 Mhz .

MOV	R0,A	_____	12 P	_____	1 us
MOV	R0,#3Fh	_____	24 P	_____	2 us
SETB	P0.1	_____	12 P	_____	1 us
DJNZ	R1,loop	_____	24 P	_____	2 us



Reset de Power-on (reset automático)

Obs: Uma chave pode ser colocada em paralelo com o capacitor para que se possa realizar o **reset manual**.



O que contém os SFR's após um Power-on ou Reset?

Table 2. Contents of the SFRs after reset

Register	Value in Binary
*ACC	00000000
*B	00000000
*PSW	00000000
SP	00000111
DPTR	
DPH	00000000
DPL	00000000
*P0	11111111
*P1	11111111
*P2	11111111
*P3	11111111
*IP	8051 XXX00000, 8052 XX000000
*IE	8051 0XX00000, 8052 0X000000
TMOD	00000000
*TCON	00000000
*+T2CON	00000000
TH0	00000000
TL0	00000000
TH1	00000000
TL1	00000000
+TH2	00000000
+TL2	00000000
+RCAP2H	00000000
+RCAP2L	00000000
*SCON	00000000
SBUF	indeterminate
PCON	HMOS 0XXXXXXX CHMOS 0XXX0000

X = Undefined

* = Bit Addressable

+ = 8052 only

Estrutura de Interrupção

Interrupção é uma característica de um computador que permite ao mesmo parar a execução de um determinado programa e passar a executar uma sub-rotina, localizada em um endereço pré-determinado da memória.

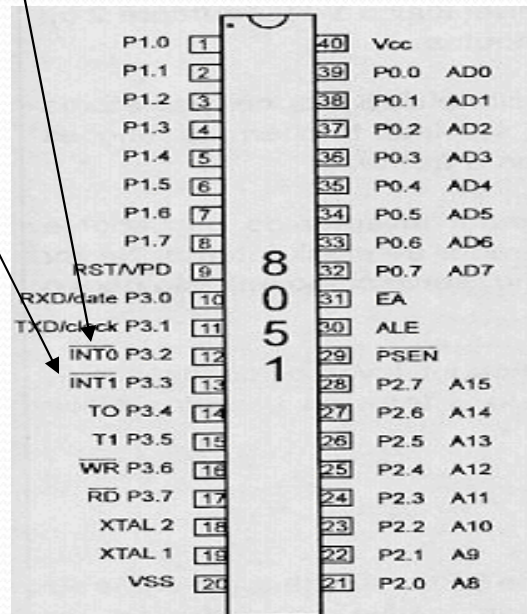
A sub-rotina a ser executada é denominada de
Sub-rotina de Atendimento de Interrupção.

- Ao terminar a execução desta sub-rotina o controle volta para o programa inicial no endereço imediatamente abaixo do ponto onde foi interrompido.

Estrutura de Interrupção

- O Microcontrolador 8051 possui 5 fontes de Interrupção :

1. Timer 0 Overflow.
2. Timer 1 Overflow.
3. Reception/Transmission of Serial Character.
4. External Event 0.
5. External Event 1.



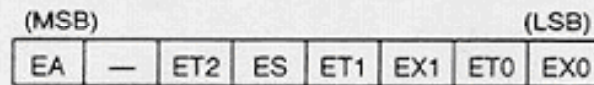
Endereço das interrupções (Memória de Programa)

Primeiro Endereço	0033h
Interrupção Extra	002Bh
Interrupção da Serial	0023h
Overflow do Timer 1	001Bh
Interrupção Externa 1	0013h
Overflow do Timer 0	000Bh
Interrupção Externa 0	0003h
Reset	0000h

Estrutura de Interrupção

Habilitação das interrupções

Registrador IE: (endereçável a Bit)



Bit = 1 --> Habilita a Interrupção
 Bit = 0 --> Desabilita a Interrupção

Símbolo	Posição	Função
EA	IE.7	Desabilita todas as Interrupções. Se EA=0, nenhuma Interrupção será reconhecida. Se EA=1, cada fonte de Interrupção será habilitada ou desabilitada individualmente.
—	IE.6	
ET2	IE.5	Bit de habilitação do Timer 2
ES	IE.4	Bit de habilitação da Porta Serial
ET1	IE.3	Bit de habilitação do Timer 1
EX1	IE.2	Bit de habilitação da Interrupção Externa 1
ET0	IE.1	Bit de habilitação do Timer 0
EX0	IE.0	Bit de habilitação da Interrupção Externa 0

Figure 5. SFR Memory Map

8 Bytes							
F8							FF
F0	B						F7
E8							EF
E0	ACC						E7
D8							DF
D0	PSW*						D7
C8	T2CON* [†]	T2MOD*	RCAP2L*	RCAP2H*	TL2*	TH2*	CF
C0							C7
B8	IP*						BF
B0	P3						B7
A8	IE*						AF
A0	P2						A7
98	SCON*	SBUF					9F
90	P1						97
88	TCON*	TMOD*	TL0	TL1	TH0	TH1	8F
80	P0	SP	DPL	DPH			87
							PCON*

↑
Bit Addressable

Exemplo:

SETB EX0 ; Habilita a Interrupção Externa 0

SETB EA ; Habilita para uso todas as Interrupções

Ou:

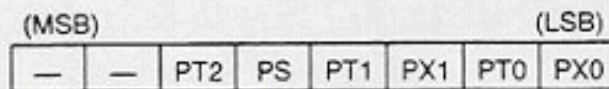
SETB IE.0

SETB IE.7

Estrutura de Interrupção

Prioridade de Interrupção

Registrador IP: (endereçável a Bit)



Bit = 1 --> Alta Prioridade
Bit = 0 --> Baixa Prioridade

Símbolo	Posição	Função
—	IP.7	
—	IP.6	
PT2	IP.5	Bit de Prioridade da Interrupção do Timer 2
PS	IP.4	Bit de Prioridade da Interrupção Serial
PT1	IP.3	Bit de Prioridade da Interrupção do Timer 1
PX1	IP.2	Bit de Prioridade da Interrupção Externa 1
PT0	IP.1	Bit de Prioridade da Interrupção do Timer 0
PX0	IP.0	Bit de Prioridade da Interrupção Externa 0

Figure 5. SFR Memory Map

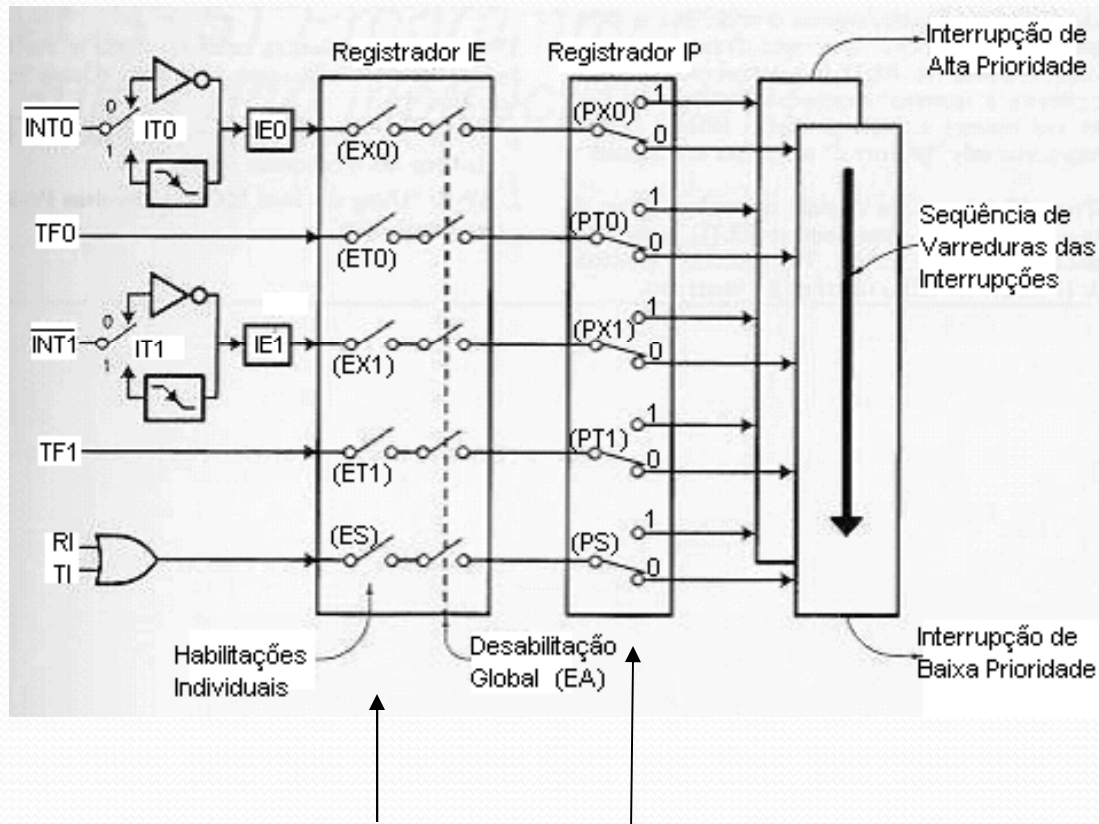
8 Bytes							
F8							FF
F0	B						F7
E8							EF
E0	ACC						E7
D8							DF
D0	PSW*						D7
C8	T2CON**	T2MOD*	RCAP2L*	RCAP2H*	TL2*	TH2*	CF
C0							C7
B8	IP*						BF
B0	P3						B7
A8	IE*						AF
A0	P2						A7
98	SCON*	SBUF					9F
90	P1						97
88	TCON*	TMOD*	TL0	TL1	TH0	TH1	8F
80	P0	SP	DPL	DPH			87
							PCON*

↑
Bit Addressable

Uma interrupção de baixa prioridade (bit 0) pode ser interrompida por uma de alta (bit 1), no entanto uma interrupção de alta prioridade não pode ser interrompida por qualquer outra fonte de interrupção.

Estrutura de Interrupção

Sistema Interno de Prioridade de Interrupção



(MSB) (LSB) (MSB) (LSB)

EA	...	ET2	ES	ET1	EX1	ET0	EX0
----	-----	-----	----	-----	-----	-----	-----

...	...	PT2	PS	PT1	PX1	PT0	PX0
-----	-----	-----	----	-----	-----	-----	-----

Exemplo:

Qual será a seqüência de atendimento de Interrupção no programa?

SETB EX0

SETB ET0

CLR PX0

CLR PT0

SETB EA

Estrutura de Interrupção

Para usar as interrupções do MCS-51 , seguir os seguintes passos

(Exemplo para a Interrupção Externa 1)

1. Setar o bit do registrador IE correspondente à interrupção utilizada →

SETB EX1



Estrutura de Interrupção

2. estabelecer para as interrupções externas o tipo de disparo, nível ou descida de borda; para isso deve-se programar os bits IT0 e/ou IT1 do registrador TCON → **CLR IT1**

Registrador TCON: (endereçoável a Bit)

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
TF1	TCON. 7	Flag de Overflow do Timer 1. Ativado por hardware quando o Timer 1 transborda. Zerado por hardware assim que o processador salta para a sub-rotina de atendimento da interrupção.					
TR1	TCON. 6	Bit de controle do Timer 1. Ativado/zerado por software para Ligar/Desligar o Timer 1.					
TF0	TCON. 5	Flag de Overflow do Timer 0. Ativado por hardware quando o Timer 0 transborda. Zerado por hardware assim que o processador salta para a sub-rotina de atendimento da interrupção.					
TR0	TCON. 4	Bit de controle do Timer 0. Ativado/zerado por software para Ligar/Desligar o Timer 0.					
IE1	TCON. 3	Flag de borda da Interrupção Externa 1. Ativado por hardware quando uma borda na Interrupção Externa 1 é detectada. Zerado por hardware quando a Interrupção é processada.					
IT1	TCON. 2	Bit de controle da Interrupção Externa 1. Ativado/zerado por software para especificar se a Interrupção Externa 1 é sensível à descida de borda/nível baixo.					
IE0	TCON. 1	Flag de borda da Interrupção Externa 0. Ativado por hardware quando uma borda na Interrupção Externa 0 é detectada. Zerado por hardware quando a Interrupção é processada.					
IT0	TCON. 0	Bit de controle da Interrupção Externa 0. Ativado/zerado por software para especificar se a Interrupção Externa 0 é sensível à descida de borda/nível baixo.					

Estrutura de Interrupção

3. Setar o bit EA (Enable All) do registrador IE → **SETB EA**



4. Escrever a sub-rotina de atendimento de interrupção no endereço correspondente.

Interrupt Source	Vector Address
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
RI & TI	0023H
TF2 & EXF2*	002BH



ATENÇÃO

Não existe instrução de chamada para uma Sub-rotina de Atendimento de Interrupção !!!!

A interrupção é atendida, ou seja, o programa desvia para o endereço de Interrupção ao ocorrer um evento ligado ao Hardware do Microcontrolador:

Externa_0 e Externa_1 : Pulso negativo nos pinos P3.2(INT0) ou P3.3(INT1) da CPU

Timer_0 e Timer_1 : Overflow dos contadores internos

Serial: Transmissão ou Recepção de um caractere pela interface Serial

O programa retorna da sub-rotina de atendimento quando executar a instrução RETI

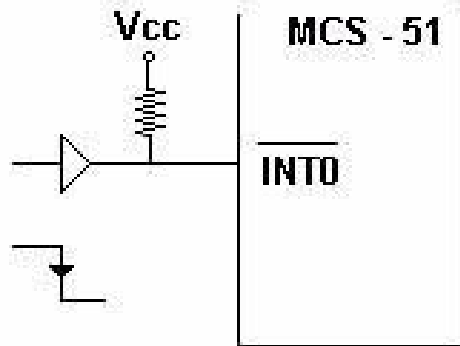
Exemplo de estruturação do código Assembly ao usar Interrupções

```
ORG 0
SJMP PROG
;
; *****
;
ORG 0003
SJMP EXTERNA_0
;
; *****
;
ORG 000BH
SJMP TIMER_0
;
; *****
;
ORG 0013H
SJMP EXTERNA_1
;
; *****
;
ORG 001BH
SJMP TIMER_1
;
; *****
;
ORG 0023H
SJMP SERIAL
;
; *****
;
PROG:
    SETB    EA
    ; instruções
    SJMP    $
;
; *****
;
EXTERNA_0:
    CLR     EA
    ; instruções
    SETB    EA
    RETI
;
; *****
;
TIMER_0:
    CLR     EA
    ; instruções
    SETB    EA
    RETI
;
; *****
;
END
```

Como o espaço reservado de memória para atendimento de cada interrupção é de apenas 8 Bytes, usa-se um JUMP para outro local após o Programa Principal com mais espaço para escrita do código da sub-rotina de atendimento.

Exemplo

Programação da Interrupção Externa 0 sensível à descida de borda.



Interrupt Source	Vector Address
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
RI & TI	0023H
TF2 & EXF2*	002BH

```
ORG    00H      ; Origem do Programa
SJMP   PROG     ; Salte para o Programa Principal
ORG    0003H    ; Local da sub-rotina de Interrupção Externa 0
.....
.....
RETI          ; Fim da sub-rotina de Interrupção
PROG:        ; Início do Programa Principal
    SETB EA   ; Habilita o uso de Interrupções
    SETB IE.0 ; Habilita a Interrupção Externa 0
    SETB IT0  ; Sensível a descida de borda
    ...
    ...      ; Comandos do Programa Principal
    ...
END
```

Exemplo

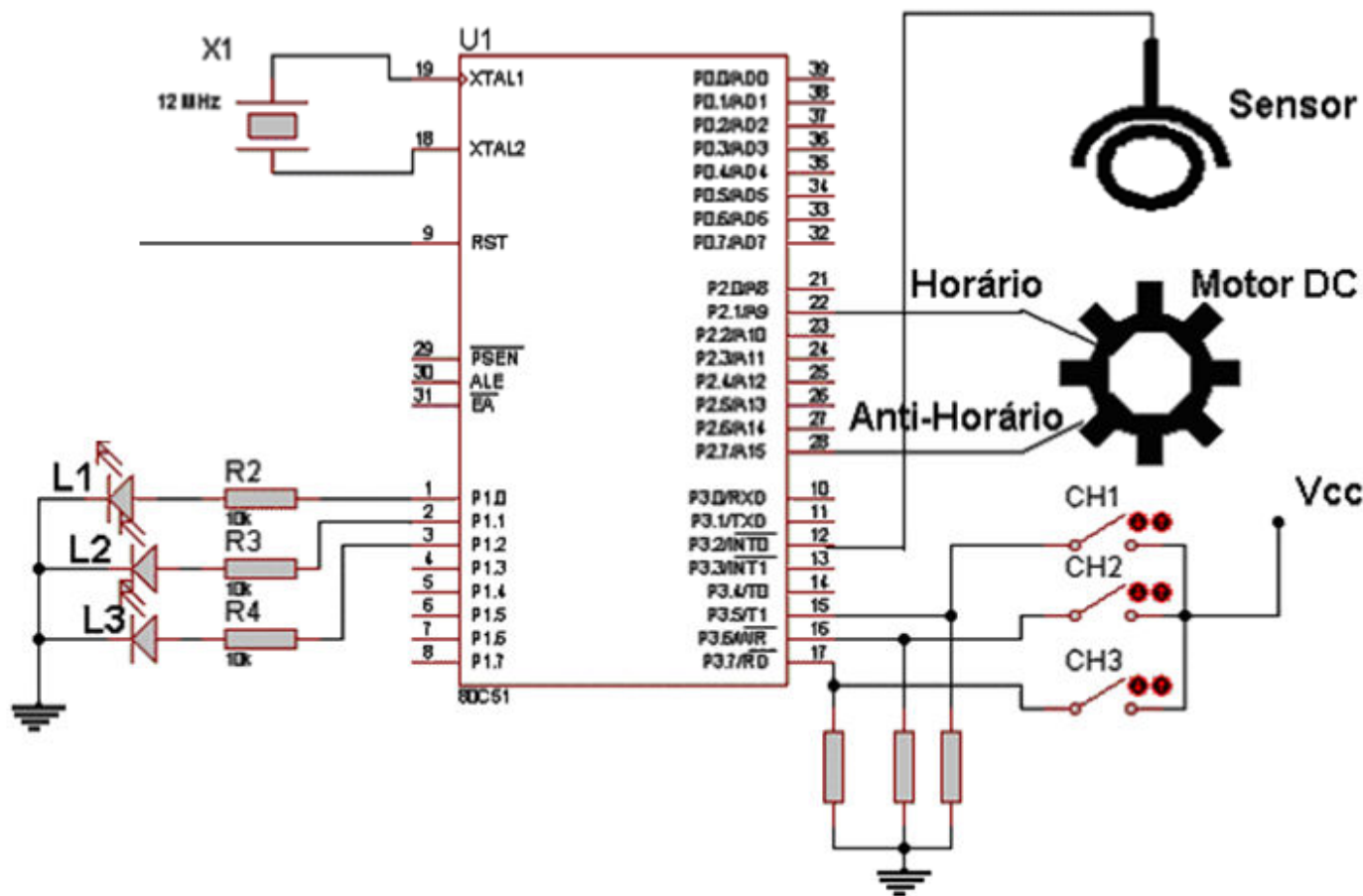
Programação da Interrupção Externa 0 sensível à descida de borda.

Sub-rotina de Atendimento da Interrupção:

ORG 0003h	; Sub-rotina de Atendimento da Interrupção Externa 0.
CLR EA	; Desabilita as Interrupções para evitar Interrupção da ; Interrupção
PUSH PSW	; Salva os Flags do Programa Principal na pilha
...	
...	; Comandos da Sub-rotina de Atendimento da Interrupção
...	
POP PSW	; Recupera os Flags do Programa Principal
SETB EA	; Re-habilita as interrupções antes de voltar ao Programa ; Principal
RETI	; Volta para o Programa Principal

Exercícios

Para os exercícios(1 ,2, e 3) considerar o esquema com o Microcontrolador 8051 da Figura abaixo. Cada programa, de cada exercício, é independente do outro.



Exercício 1

1) Escrever um programa em Assembly do 8051 que ao ligar qualquer das chaves acende o Led correspondente.

CH1(P3.5)	fechada	→	acende	LED	L1(P1.0)
CH2(P3.6)	fechada	→	acende	LED	L2(P1.1)
CH3(P3.7)	fechada	→	acende	LED	L3(P1.2)

O programa deve ficar em Loop para que a qualquer instante o operador possa repetir a operação.

Solução_1 para o Exercício 1

```
ORG 0
MOV P1,#00      ; Apagando todos os LED's
LOOP1: JB P3.5,L1  ; Verificando se CH1 está fechada
        CLR P1.0   ; Apagar o LED L1 pois CH1 está aberta
LOOP2: JB P3.6,L2  ; Verificando se CH2 está fechada
        CLR P1.1   ; Apagar o LED L2 pois CH2 está aberta
LOOP3: JB P3.7,L3  ; Verificando se CH3 está fechada
        CLR P1.2   ; Apagar o LED L3 pois CH3 está aberta
        SJMP LOOP1 ; Todas as chaves estão abertas, verificar novamente
L1:      SETB P1.0  ; Acender o LED L1
        SJMP LOOP2 ; Verificar CH2
L2:      SETB P1.1  ; Acender o LED L2
        SJMP LOOP3 ; Verificar CH3
L3:      SETB P1.2  ; Acender o LED L3
        SJMP LOOP1 ; Verificar CH1
END
```


Solução_2 para o Exercício 1

	ORG 0	
	MOV P1,#00	; Apagando todos os LED's
LOOP:	MOV C,P3.5	; Colocando o Status da CH1 no Carry
	MOV P1.0,C	; Transportando para o LED 1
	MOV C,P3.6	; Colocando o Status da CH2 no Carry
	MOV P1.1,C	; Transportando para o LED 2
	MOV C,P3.7	; Colocando o Status da CH3 no Carry
	MOV P1.2,C	; Transportando para o LED 3
	SJMP LOOP	; Retornar para continuar a operação
	END	

Solução_3 para o Exercício 1

	ORG 0	
	MOV P1,#00	; Apagando todos os LED's
LOOP:	MOV A,P3	; Copiando o valor das Chaves para o Acc
	SWAP A	; Trocando os nibbles de P3 (LSB – MSB)
	RR A	; Ajustando o nibble menos significativo
	MOV P1,A	; Transferindo para os LED's
	SJMP LOOP	; Retornar para continuar a operação
	END	

Exercício 2

2) Escrever um programa em Assembly do 8051 que ao ligar qualquer das chaves ocorre o seguinte:

CH1(P3.5) fechada → pisca apenas o Led L1(P1.0) na frequência de 1 Hz

CH2(P3.6) fechada → pisca apenas o Led L2(P1.1) na frequência de 1 Hz

CH3(P3.7) fechada → pisca alternadamente o Led L3(P1.2) e o Led L1(P1.0) na frequência de 1 Hz

O programa deve ficar em Loop para que a qualquer instante o operador possa alterar as opções das chaves.

Exercício 2

- Utilizar rotinas de atraso (Delay) que geram temporização por Software.

Rotina de Delay de 8 Bits

a) Armazenar em R0 o valor de contagem

C = Número de Ciclos da Rotina

$$C = (1 + (R0 \times 2) + 2)$$

$$C = ((R0 \times 2) + 3)$$

Atraso:	mov R0,#LSB	;1 ciclo
	djnz R0, \$;2 ciclos
	ret	;2 ciclos

Tempo gasto pela rotina de Delay

$$\Delta t = 12 \times \frac{1}{f} \times C$$

	R0	C = número de ciclos	Δt	
			12 MHz	11.0592
Menor Atraso	1	5	5 μs	5.42 μs
	FFh	513	513 μs	556.64 μs
Maior Atraso	0	515	515 μs	558.81 μs

$f \rightarrow$ MHz

$\Delta t \rightarrow \mu s$

$R0 = 0 \rightarrow R0 = 256$ no cálculo de C

Exercício 2

Rotina de Delay de 16 Bits

- a) Armazenar em R1 o MSB
- b) Armazenar em R0 o LSB

C = Número de Ciclos da Rotina

$$C = 1 + (1 + (R0 \times 2) + 2) \times R1 + 2$$

$$C = ((R0 \times 2) + 3) \times R1 + 3$$

Atraso:	mov R1,#MSB	;1 ciclo
Loop:	mov R0,#LSB	;1 ciclo
	djnz R0, \$;2 ciclos
	djnz R1, Loop	;2 ciclos
	ret	;2 ciclos

Tempo gasto pela
rotina de Delay

$$\Delta t = 12 \times \frac{1}{f} \times C$$

	R1 = MSB	R0 = LSB	C = número de ciclos	Δt	
				12 MHz	11.0592
Menor Atraso	1	1	8	8 μs	8.68 μs
	FFh	FFh	130818	130.8 ms	141.95 ms
Maior Atraso	0	0	131843	131.8 ms	143.06 ms

Exercício 2

Rotina de Delay de 24 Bits

- a) Armazenar em R2 o MSB
- b) Armazenar em R1 o Byte intermediário
- c) Armazenar em R0 o LSB

C = Número de Ciclos da Rotina

$$C = (((R0 \times 2) + 3) \times R1 + 3) \times R2 + 3$$

Atraso:	mov R2,#MSB	;1 ciclo
Loop1:	mov R1,#ISB	;1 ciclo
Loop:	mov R0,#LSB	;1 ciclo
	djnz R0, \$;2 ciclos
	djnz R1, Loop	;2 ciclos
	djnz R2, Loop1	;2 ciclos
	ret	;2 ciclos

Exercício 2

Rotina de Delay de 24 Bits

- a) Armazenar em R2 o MSB
- b) Armazenar em R1 o Byte intermediário
- c) Armazenar em R0 o LSB

C = Número de Ciclos da Rotina

$$C = (((R0 \times 2) + 3) \times R1 + 3) \times R2 + 3$$

Tempo gasto pela rotina de Delay

$$\Delta t = 12 \times \frac{1}{f} \times C$$

	R0	R1	R2	C = número de ciclos	Δt	
					12 MHz	11.0592
Menor Atraso	1	1	1	11	11 μs	11.94 μs
	FFh	FFh	FFh	33.358.593	33.358.593 μs	36,196 s
Maior Atraso	0	0	0	33.751.811	33.751.811 μs	36,623 s

$R0 = R1 = R2 = 0 \rightarrow R0 = R1 = R2 = 256$ no cálculo de C

Exercício 3

3) O Motor DC é ativado de acordo com a seguinte Tabela:

Horário (P2.1)	Anti-Horário (P2.7)	Sentido de Giro do Motor
0	0	Parado
0	1	Anti-Horário
1	0	Horário
1	1	Parado

Escrever um programa em Assembly do 8051 que controle uma esteira transportadora da seguinte maneira:

- Acionar o motor DC no sentido Horário.**
- Quando o produto passar pelo sensor, um sinal de Interrupção é enviado e a esteira é parada por 5 segundos para permitir a retirada do produto transportado.**
- Inverter o sentido do motor DC (Anti-horário).**
- Através de um mecanismo na esteira, uma nova interrupção é enviada pelo mesmo pino Int0 quando a esteira estiver re-posicionada para aceitar outro produto.**
- Parar a esteira por 10 segundos e re-iniciar o processo.**

Exercício para nota

Fazer os exercícios 2) e 3) e entregar em um só arquivo .ASM pelo site da disciplina até a aula da próxima semana.