Departamento de Engenharia Elétrica e de Computação
EESC-USP

SEL 0415 – Introdução à Organização de Computadores

Aula 8 – Microcontrolador 8051

Profa. Luiza Maria Romeiro Codá

Autores: Prof. Dr. Marcelo A. C. Vieira
Profa. Luiza Maria Romeiro Codá
Profa. Maria Stela Veludo de Paiva

MICROCONTROLADOR 8051

Introdução, características, ligação de memória externa e instruções

SEL 0415

Microcontroladores

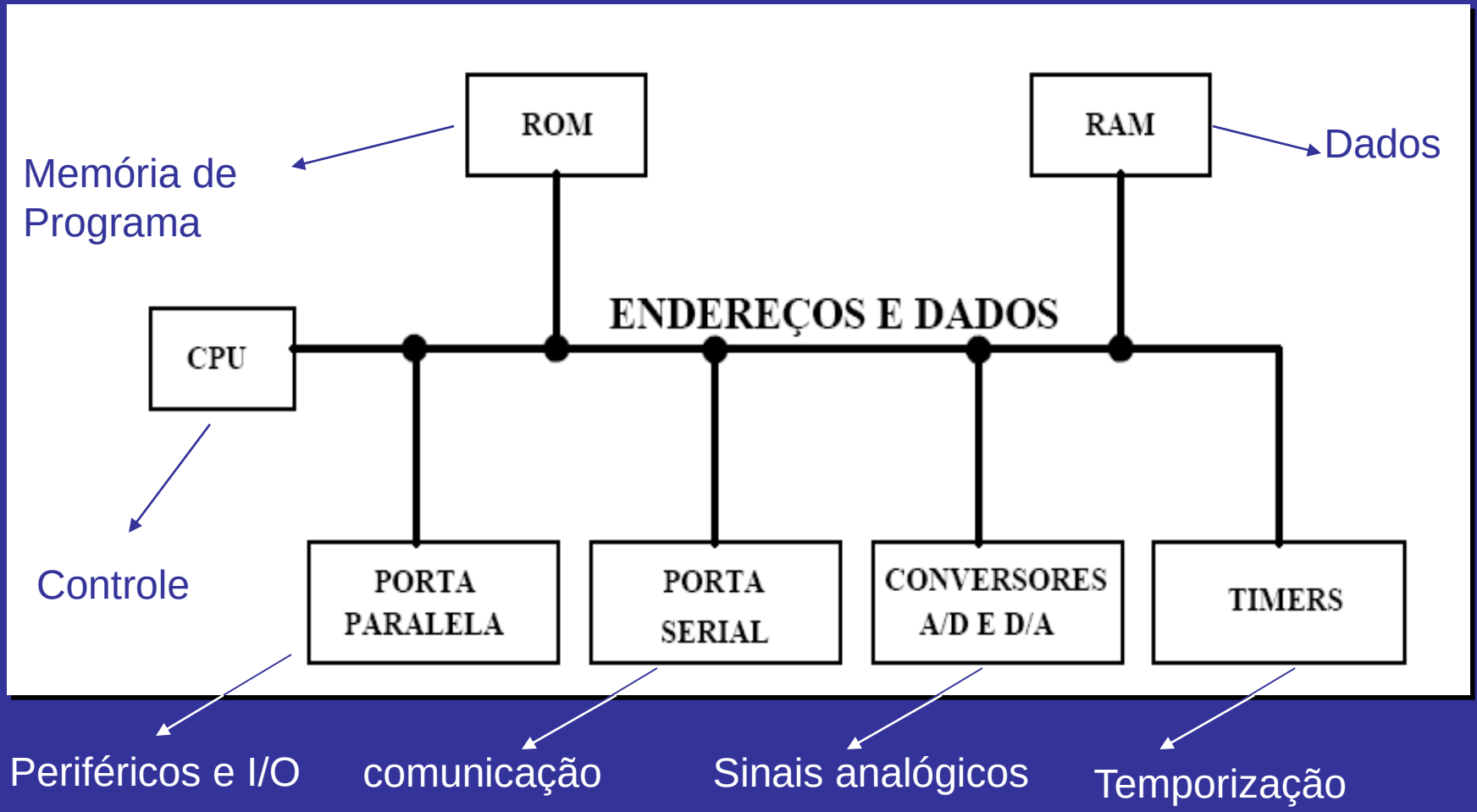
- Microcontrolador é o nome dado ao componente que incorpora em um só CI todos os elementos necessários a um microcomputador;

- Contém os seguintes módulos:
  - **Microprocessador** (ULA + Registradores + Unidade de Controle)
  - **Memórias** (Programa e Dados)
  - **Interfaces**;

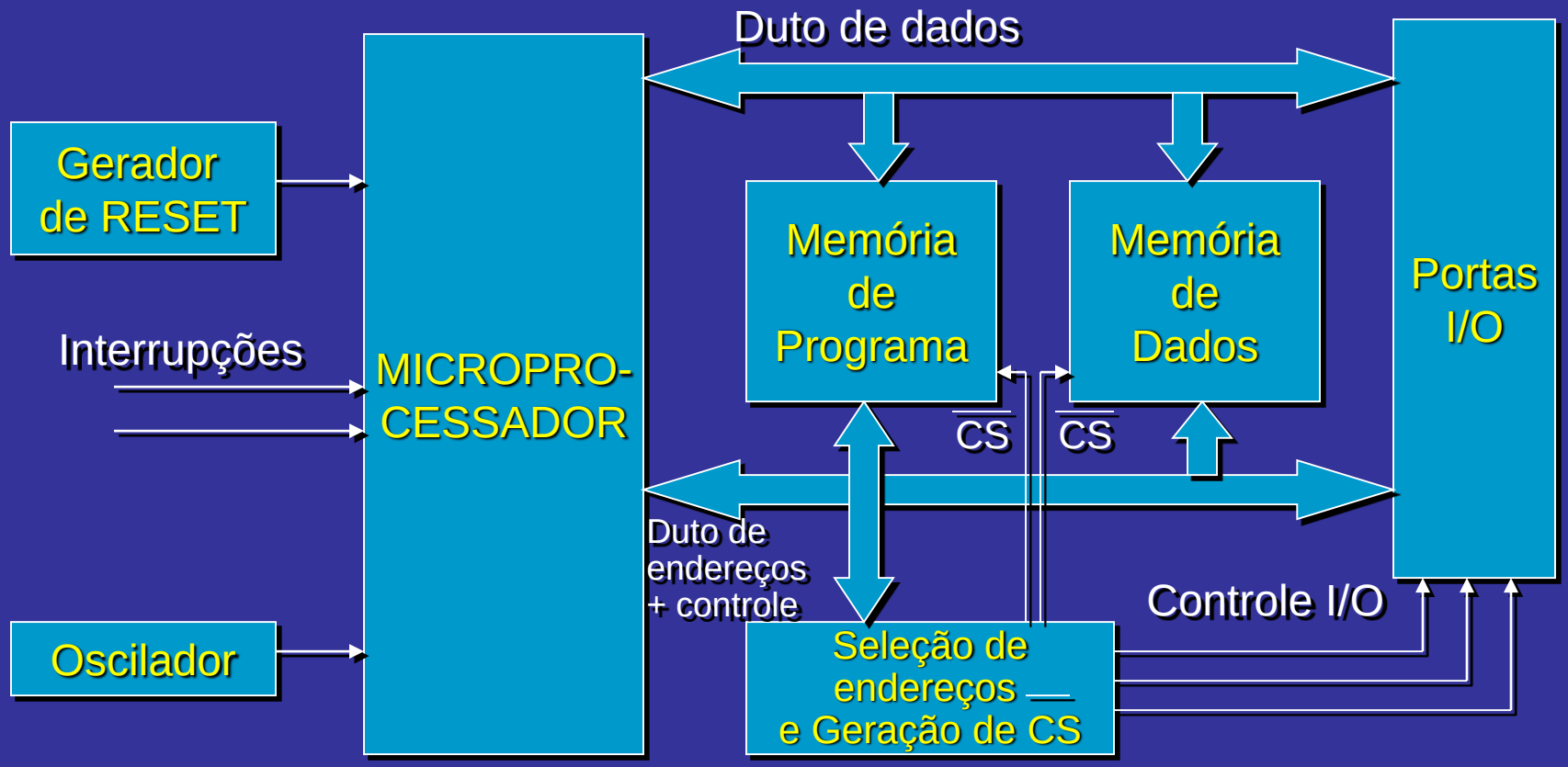
Microcontroladores

- As interfaces podem ser as mais diversas :
  - Contador / Temporizador
  - Conversor AD / DA
  - Portas de I/O Paralelas (Entrada e Saída)
  - Interface Serial
- Além disso deve permitir a expansão externa de memória e periféricos.

Exemplo típico da arquitetura de um microcontrolador



Microcontrolador de 8 Bits
(em um mesmo chip)



Operação de um Microcontrolador

- capaz de buscar e executar instruções de programas alocados na memória de programa;

- Após a energização de um microcontrolador, é gerado um sinal de *reset* que zera o *Program Counter* (PC), ou seja, posiciona o PC no endereço inicial (geralmente 0000H). O programa é executado a partir desse endereço.

- O microcontrolador irá buscar e executar as instruções na seqüência que elas estão gravadas na memória de programa, seguindo sempre o endereço de memória definida pelo PC (contador de programa);

Operação de um Microcontrolador

Ciclo de Busca: operação de leitura do opcode de uma instrução (ou parte dela) a partir da posição de memória cujo endereço é definido pelo conteúdo do PC. O opcode da instrução é armazenado em um registrador chamado de RI (Registrador de Instrução), para ser executado pela unidade de controle;

Ciclo de Execução: executa a instrução (se ela ocupar apenas uma posição) ou busca os demais bytes da instrução na memória de programa para em seguida executá-la. Nesse ciclo, o conteúdo do PC é incrementado de uma, duas ou três unidades. Isso depende do tamanho da instrução.

Operação de um Microcontrolador

- **Ciclo de Máquina**: sua definição varia de acordo com a arquitetura de cada microprocessador. Para o 8051 é: ciclo de busca do "opcode" + leitura ou gravação, em memória ou I/O (duração de 12T);

- **Ciclo de Instrução**: tempo gasto para executar uma instrução por completo. Pode necessitar de mais de um ciclo de máquina.

Operação de um Microcontrolador

• Após a energização de um microcontrolador, é gerado um sinal de *reset* que zera o *Program Counter* (PC), ou seja, posiciona o PC no endereço inicial (geralmente 0000H). O programa é executado a partir desse endereço.

Microcontrolador 80C51

- Membro da família MCS-51
- Núcleo de todos os dispositivos MCS-51 (Atmel)
- Sistema de um *chip* único, que além do microprocessador de 8 bits também contém:
  - Memória de Programa e Memória de Dados
  - Portas de I/O
  - Comunicação Serial (UART)
  - Contadores/ “Timers”
  - Lógica para Controle de Interrupção

Microcontrolador 80C51

1. Características do Núcleo (*Core*)

- CPU de 8 bits otimizada para aplicações de controle;
- Capacidade de processamento booleano (lógica de um único bit);
- Endereçamento de até 64 Kbytes de memória de programa externa;
- Endereçamento de até 64 Kbytes de memória de dados externa;
- 4 Kbytes de memória de programa (FLASH ROM) interna;
- 128 bytes (ou 256) de memória de dados (SRAM) interna para uso geral ;
- 128 bytes para mapeamento dos registradores de funções especiais (SFR).

Microcontrolador 80C51

1. Características do Núcleo (*Core*)

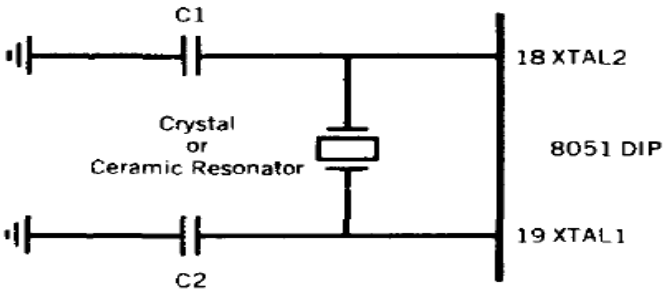
- 4 portas paralelas de 8 bits (32 linhas de I/O bidirecionais endereçadas individualmente)
- UART full duplex (*Universal Asynchronous Reciever Transmitter*)
- 2 Contadores / Temporizadores de 16 bits cada
- Estrutura de interrupção com níveis de prioridade
- Oscilador interno
- Versões disponíveis de 12 a 30 MHz (instruções de um ciclo, de 1 µs a 400 ns)

Ciclo de Máquina do 8051

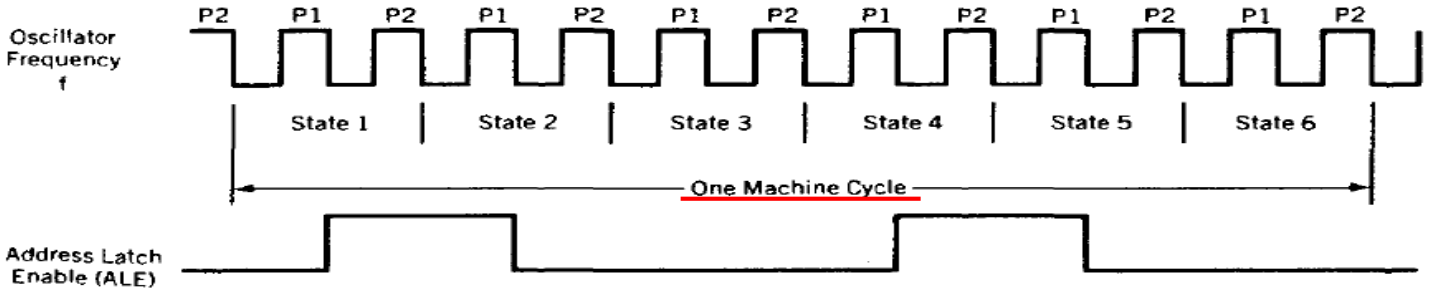
consiste em uma sequência de 6 estados, cada um formado por 2 períodos de clock

Um Ciclo de Máquina = $F_{cristal}/12$

$F_{cristal}$: 12 a 30 MHz



Crystal or Ceramic Resonator Oscillator Circuit



8051 Timing

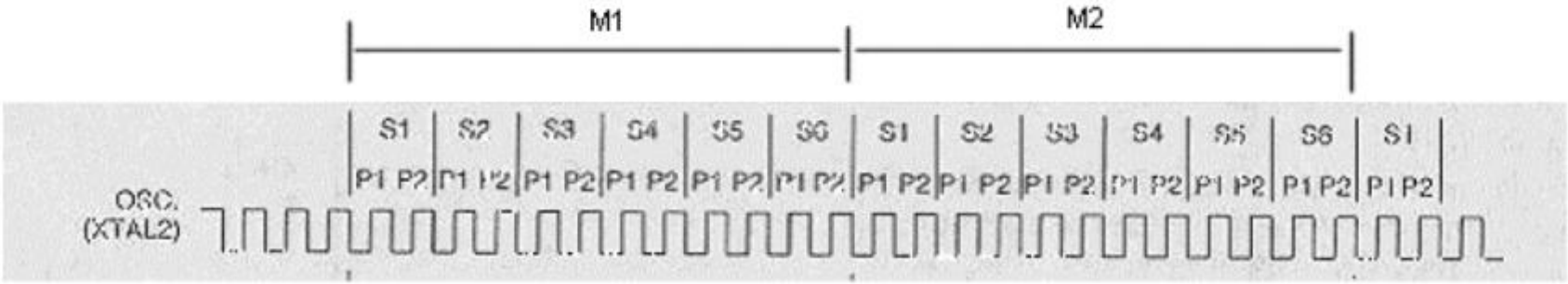
Ciclo de Máquina do 8051

Se o cristal é de 12 Mhz:

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

Ciclo de Máquina (M):

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



Rotina de atraso (*delay*) que gera temporização para o software no 8051

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

Ciclo de Máquina do 8051

- Instruções da família MCS-51 duram 12 ou 24 T_{CK} (1 ou 2 Ciclos de máquina - CM)
- Exceção ➡ ***MUL AB*** e ***DIV AB*** (usam 4 CM)

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

Microcontrolador 80C51

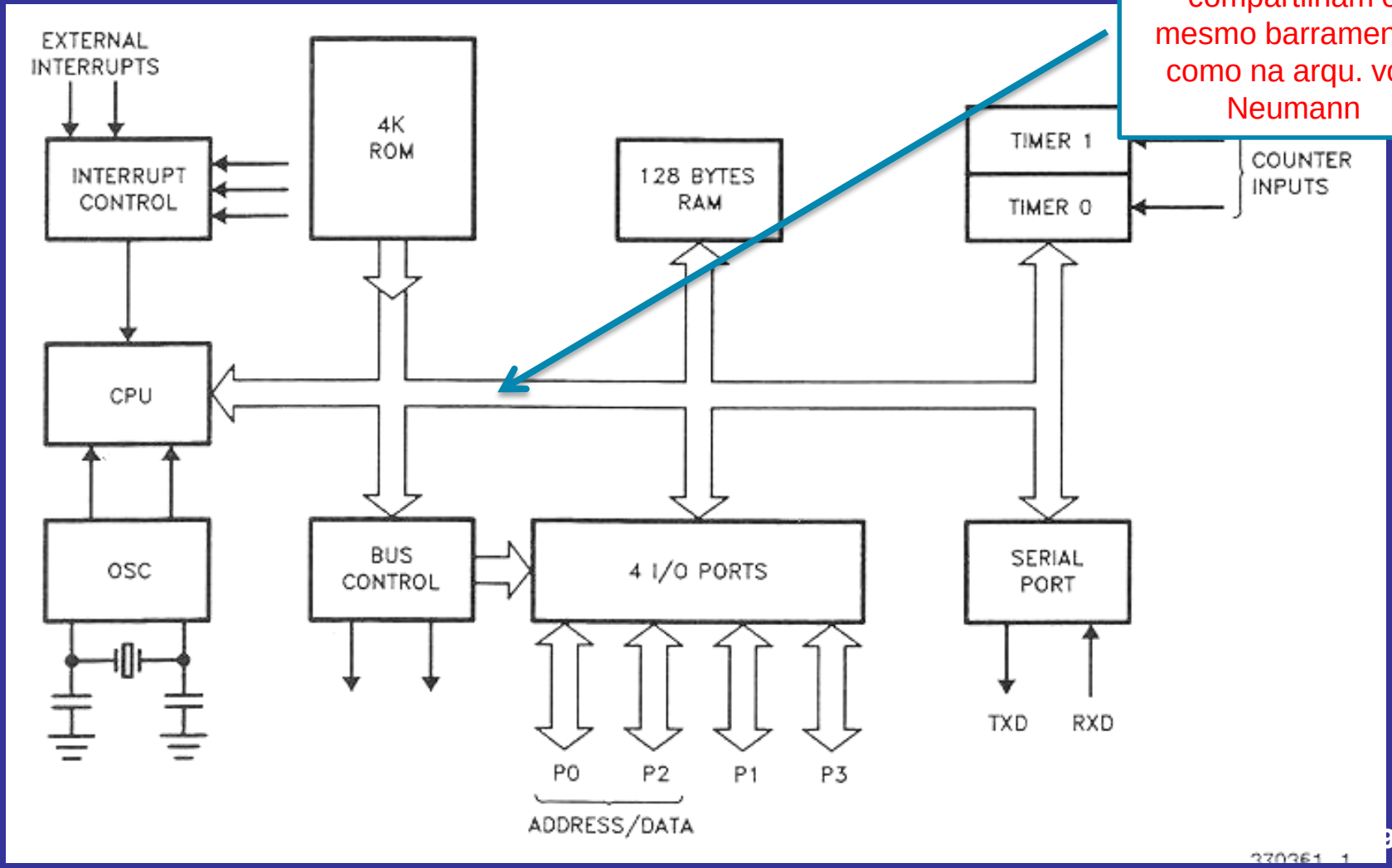
2. Arquitetura

- Arquitetura Von Neumann
- Conjunto de Instruções do tipo CISC
- 111 instruções
- O conjunto de instruções inclui:
  - ✓ Multiplicação e Divisão
  - ✓ *Bit set, reset, e test* (Instruções Booleanas).

Freq : 12 a 30 MHz

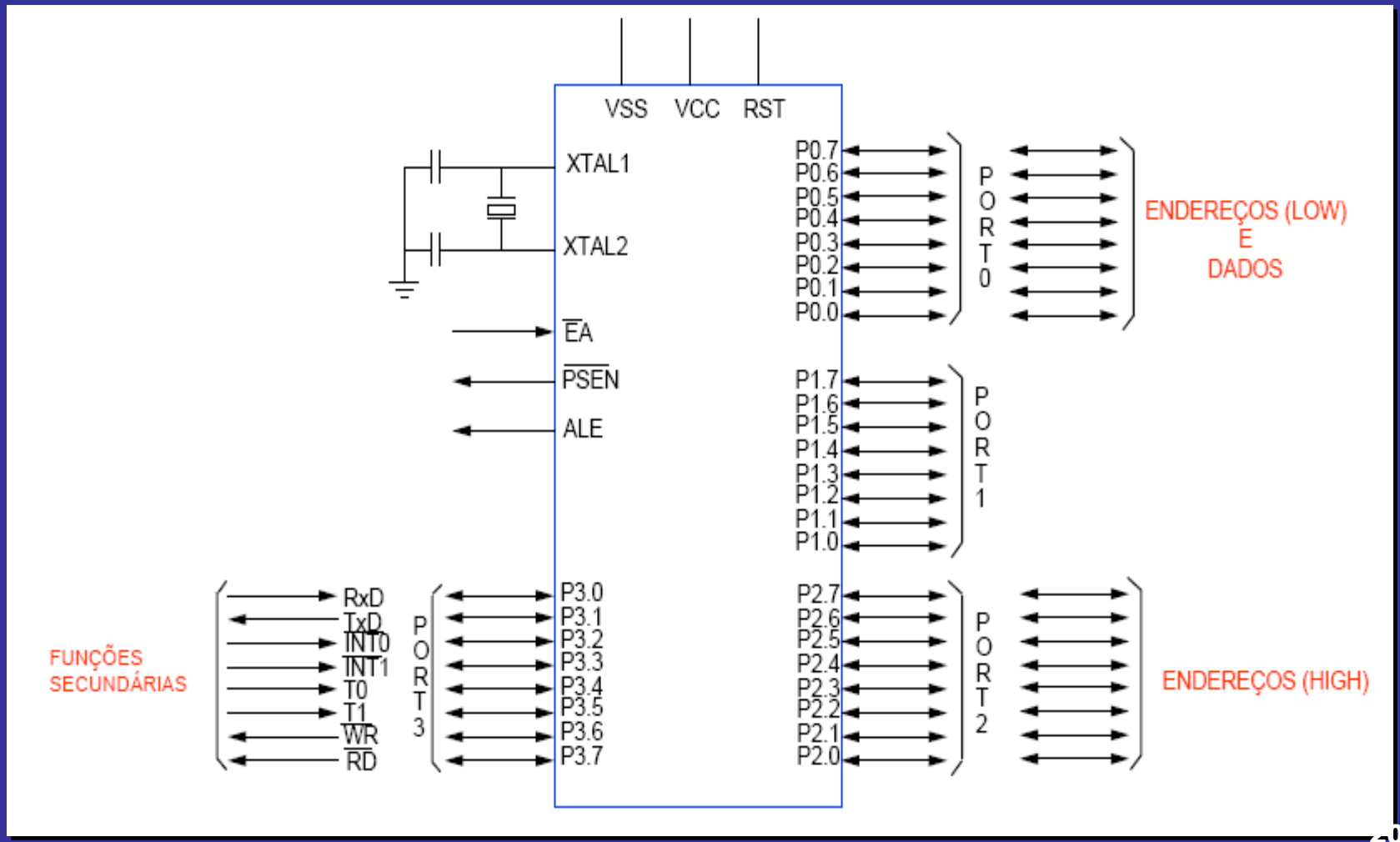
Microcontrolador 80C51

observe que as memórias compartilham o mesmo barramento, como na arqu. von Neumann



Microcontrolador 80C51

Configuração dos pinos



Ligação de Memória Externa (ROM e RAM)

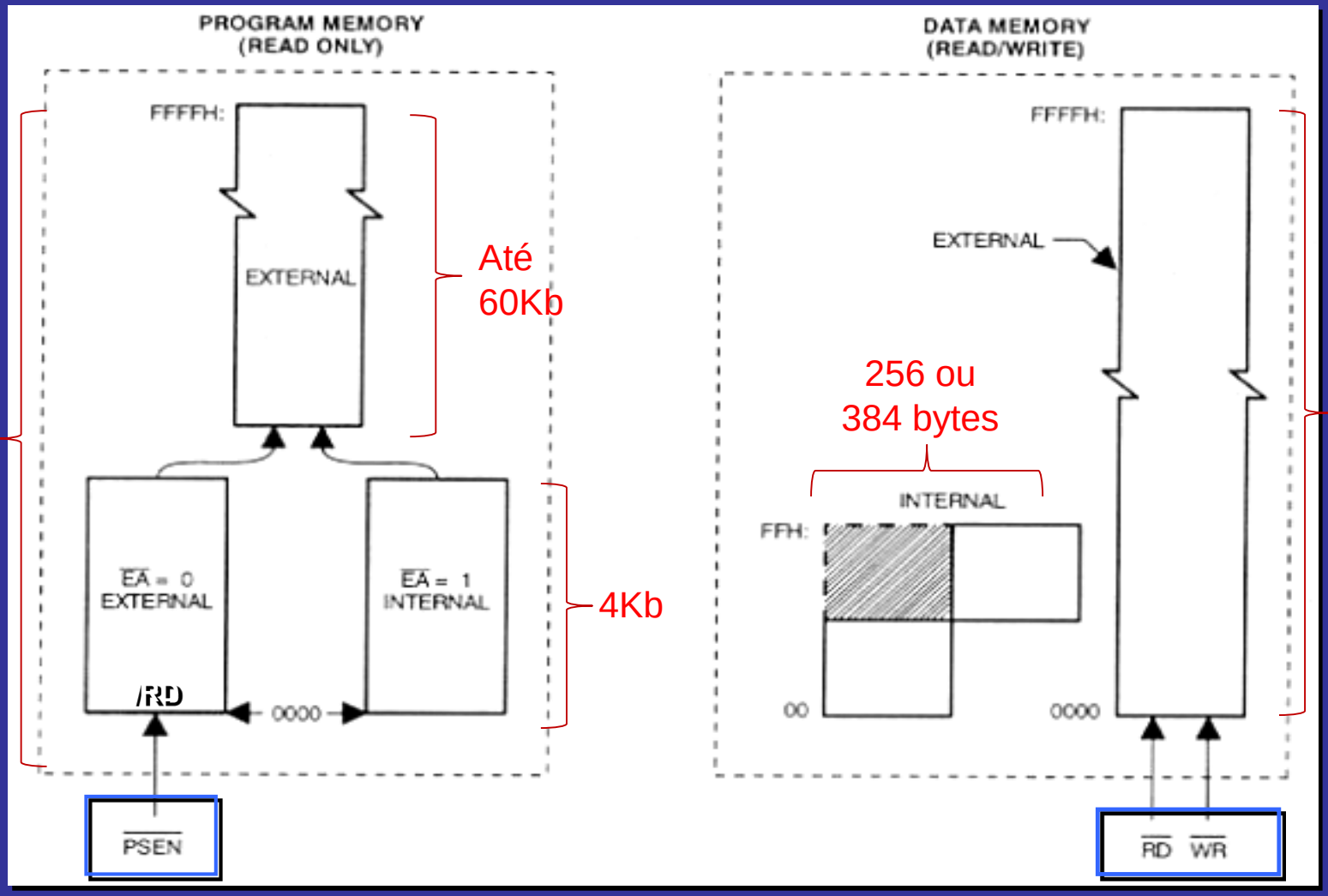
Microcontrolador 80C51

PINOS IMPORTANTES PARA INTERFACE COM MEMÓRIAS EXTERNAS

- **$\overline{\text{RD}}$** ⇨ leitura na memória de dados externa
- **$\overline{\text{WR}}$** ⇨ escrita na memória de dados externa
- **$\overline{\text{PSEN}}$** ⇨ leitura na memória de programa externa
- **P0** ⇨ multiplexado com endereços (A0-A7) e dados (D0-D7)
- **P2** ⇨ endereços A8-A15
- **ALE** ⇨ *Address Latch Enable*. Sinal para demultiplexar P0
- **$\overline{\text{EA}}$** ⇨ *External Access Enable*. Especifica o uso de memória de programa externa ou interna

Microcontrolador 80C51

Organização das Memórias na família MCS-51



Microcontrolador 80C51

Organização da Memória na família MCS-51

MEMÓRIA DE PROGRAMA

- Os 4 KB de ROM interna podem ser usados ou não, de acordo com o estado do pino $\overline{EA}$ (External Access Enable):
  - se $\overline{EA} = 0$ ⇨ até 64 KB de programa externo
  - se $\overline{EA} = 1$ ⇨ 4 KB de ROM interna e até 60 KB de programa externo

Microcontrolador 80C51

Organização da Memória na família MCS-51

MEMÓRIA DE PROGRAMA

Endereço das Memórias de Programa interna e externa

Rom Interna EA = Vcc	Endereçamento Interno	Endereçamento Externo
4 K	0000h a 0FFFh	1000h a FFFFh
8 K	0000h a 1FFFh	2000h a FFFFh
16 K	0000h a 3FFFh	4000h a FFFFh
32 K	0000h a 7FFFh	8000h a FFFFh

Se EA = 0 → toda a memória de programa é externa : 0000H a FFFFH

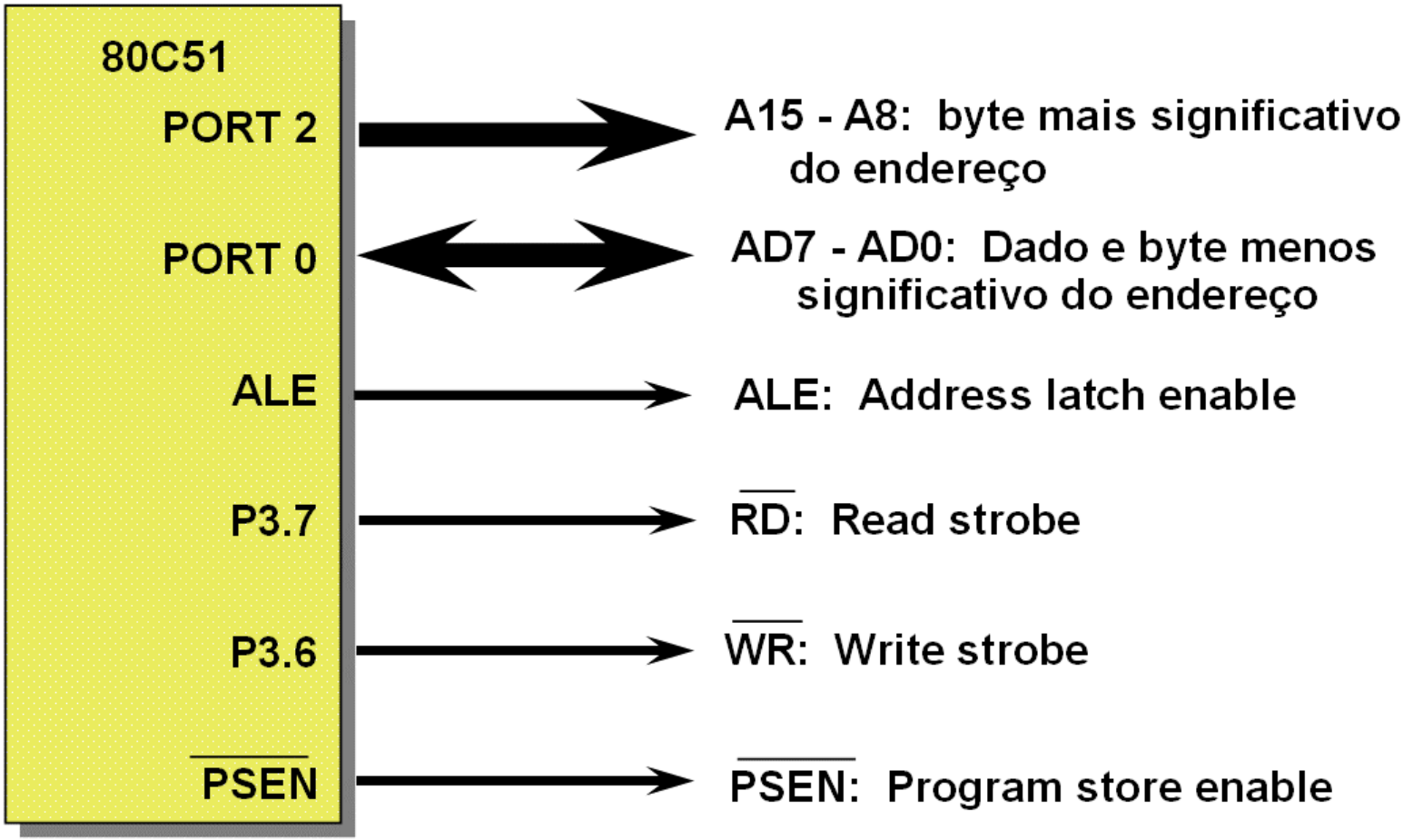
Microcontrolador 80C51

Organização da Memória na família MCS-51

MEMÓRIA DE DADOS

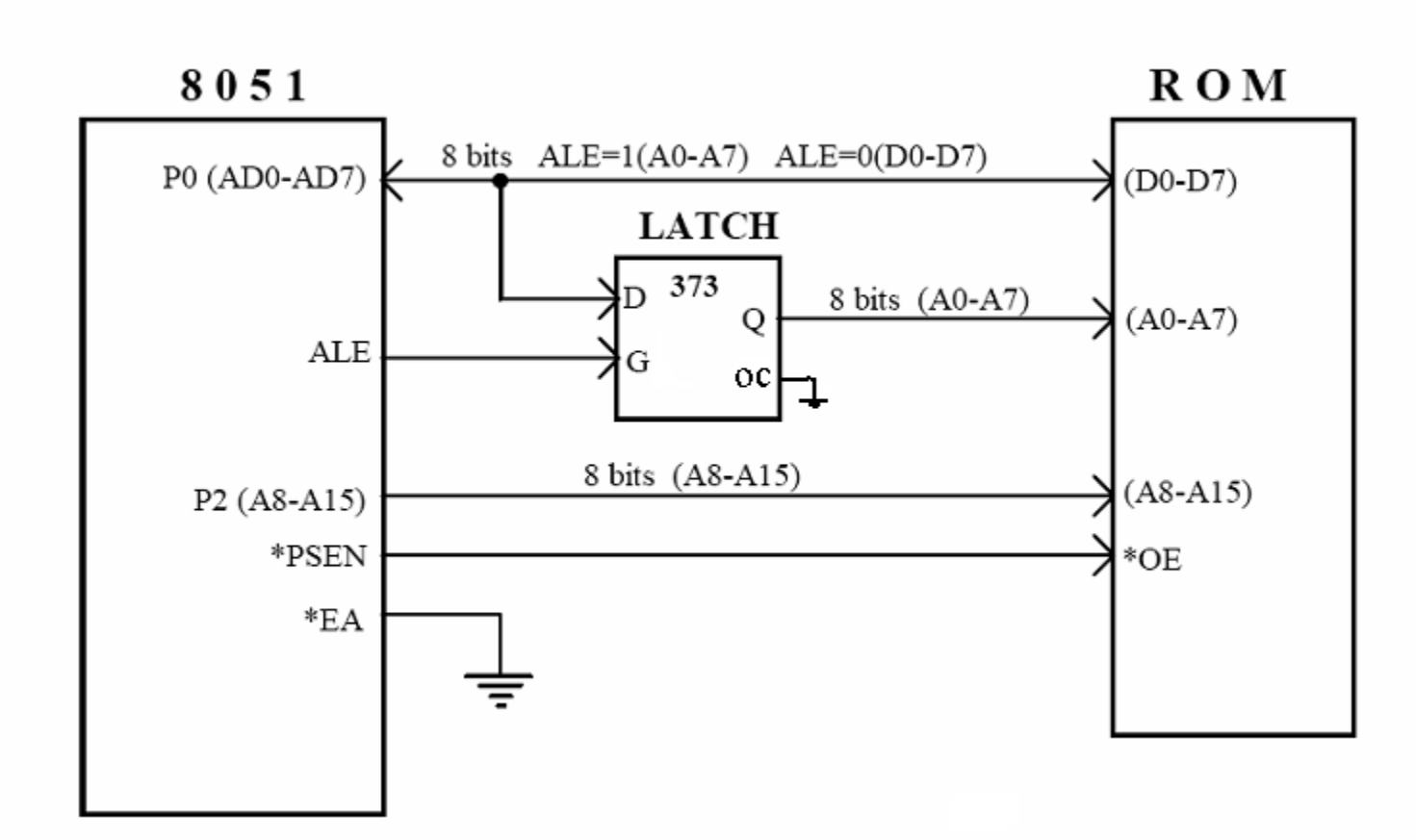
- **Externa** (DADOS) ➡ expansão com até 64 K RAM
- Pode-se usar as memórias RAM interna e externa simultâneamente
- As **instruções** de transferência de dados são **diferentes** para a memória externa e interna
- **Interna:** endereçamento direto (endereço do dado)
- **Externa**: endereçamento indireto (por ponteiro)

Expansão do Duto Externo



Mapeamento da Memória Externa de Programa

Mapeamento completo (64 Kbytes)



Memória de programa só pode ser lida. São sempre emitidos endereços de 16 bits ⇨ as portas P0 e P2 são sacrificadas quando se usa memória de programa externa.

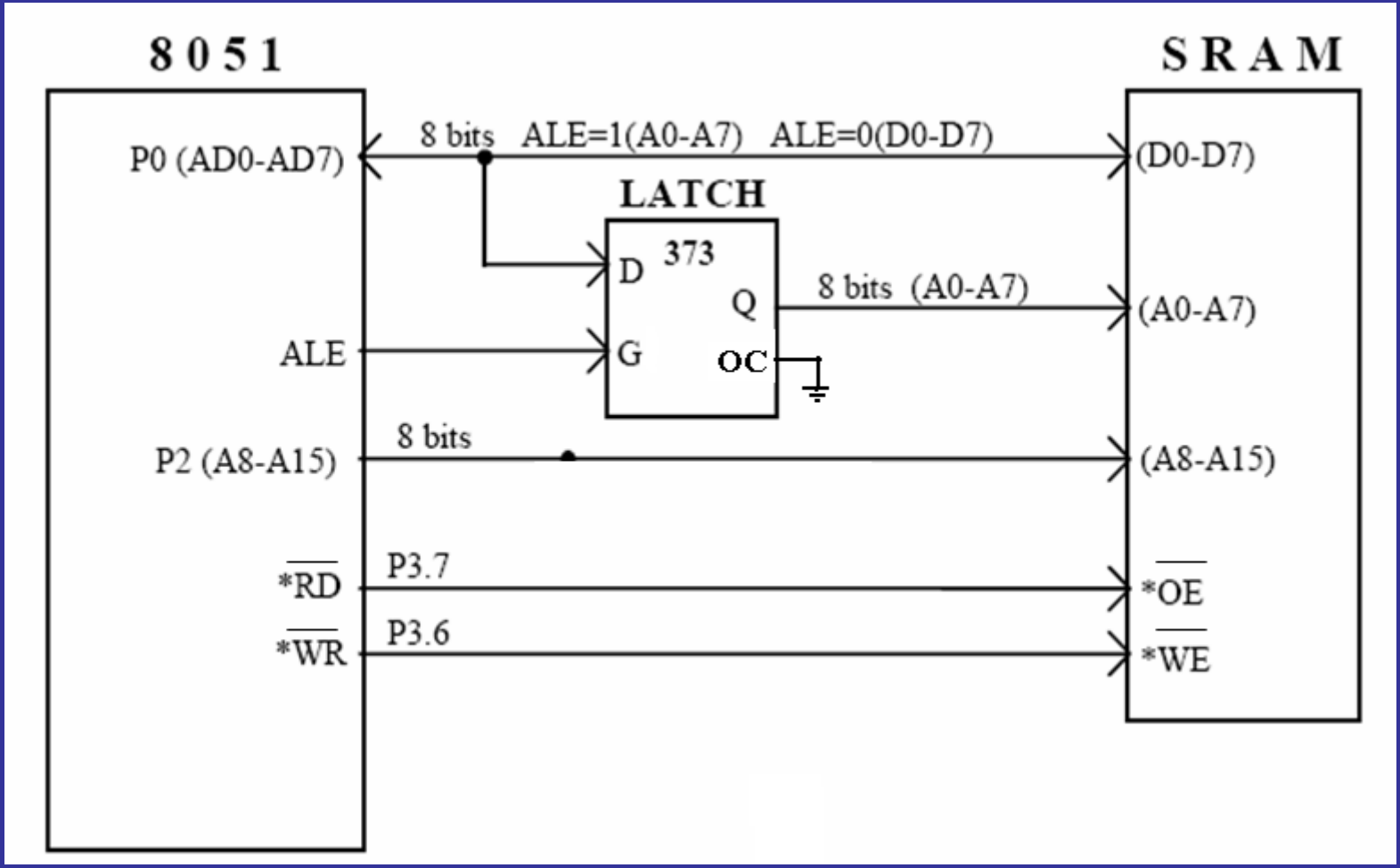
Memória de programa externa (EPROM)

- Porta P2 ➡ 8 bits mais significativos do endereço
- Porta P0 ➡ 8 bits menos significativos do endereço multiplexado com os dados dados

1. pino ALE (address latch enable) ➡ quando em nível lógico 1, indica a presença de endereço no duto multiplexado (P0)
2. latch externo (ex.74373) ➡ comandado pelo ALE ➡ se ALE = 1 bits A0-A7 são armazenados na saída do 74373 (OC em “0”)
3. pino $\overline{\text{PSEN}}$ (ativo em nível lógico 0)➡ ligado no Output Enable da memória externa
4. Opcode ou operandos lidos da EPROM pelo microprocessador, são colocados nos pinos da porta P0

Mapeamento da Memória Externa de Dados (RAM)

Mapeamento completo (64 Kbytes)



Memória de Dados externa

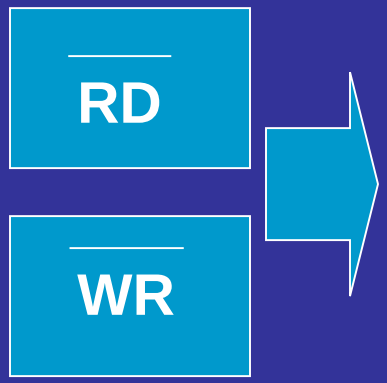
- Porta 2 ➡ 8 bits mais significativos do endereço
- Porta 0 ➡ 8 bits menos significativos do endereço + dados

1. pino ALE (address latch enable) ➡ quando em nível lógico 1, indica a presença de endereço no duto multiplexado (P0)

- latch externo (ex.74373) ➡ comandado pelo ALE ➡ se ALE = 1 bits A0-A7 são armazenados na saída do 74373 (OC em “0”)
- pinos /RD e /WR ➡ quando em nível lógico “0” corresponde respectivamente a operação de leitura e escrita, na memória RAM
- o dado lido da memória ou a ser gravado na memória é colocado no duto AD0-AD7-(porta P0)

Memória de dados externa

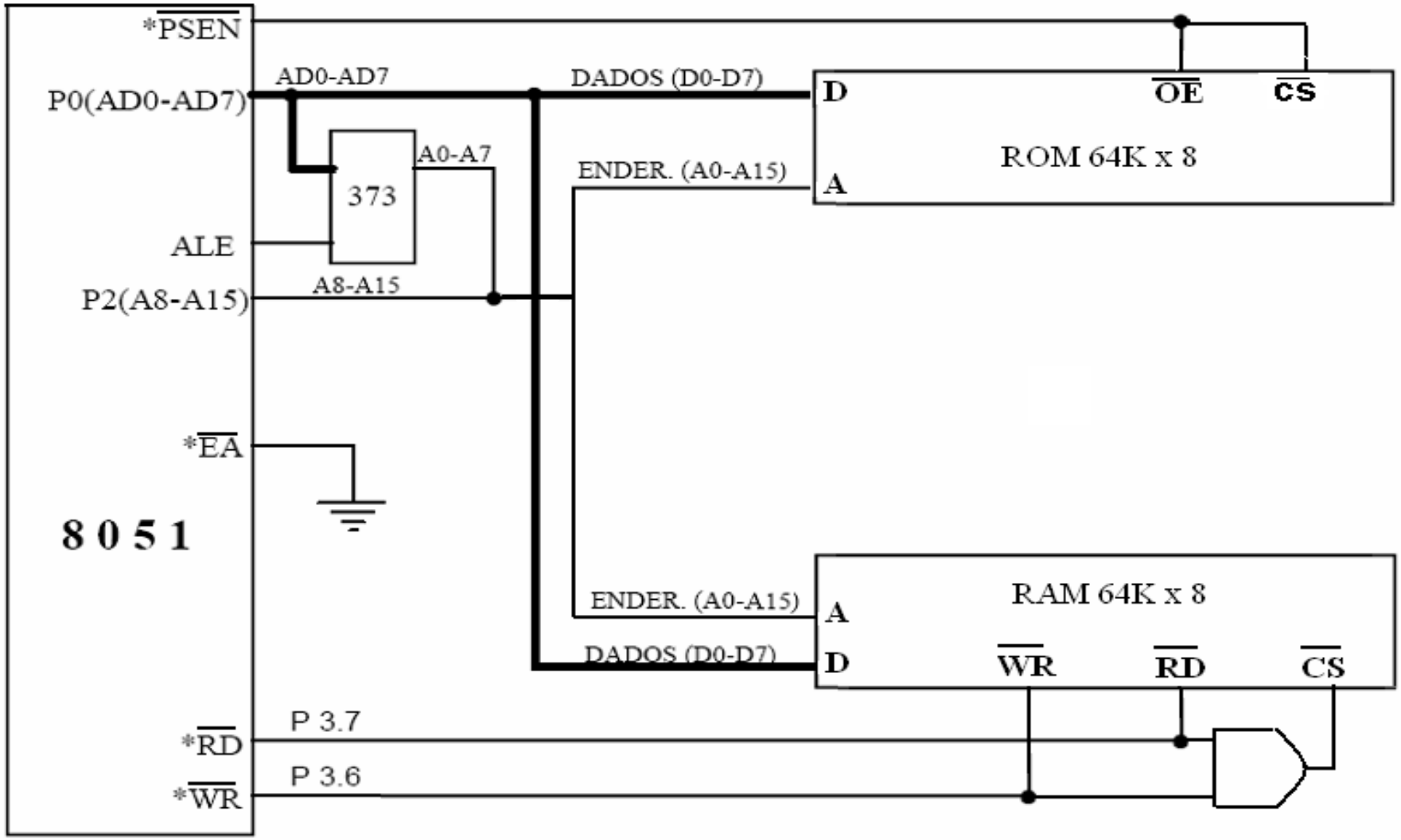
Portas I/O para expansão da memória



Gerados pelos pinos da Porta 3

- Ativos só se a CPU está executando instrução de leitura e escrita na memória de dados (RAM) externa
- Leitura da RAM ➡ $\overline{RD} = 0$ $\overline{WR} = 1$
- Escrita na RAM ➡ $\overline{RD} = 1$ $\overline{WR} = 0$
- **ROM externa ➡ $\overline{PSEN} = 0$ ➡** $\overline{RD} = 1$ $\overline{WR} = 1$

Mapeamento da Memória Externa de Dados e Programa



Microcontrolador 80C51

Registradores de Uso do Programador

- **A** : acumulador de 8 bits
- **B** : registrador de 8 bits
- **8** registradores nomeados de **R0 a R7**
  - ❖ há 4 bancos de registradores RO – R7, mapeados em RAM interna
  - ❖ apenas um banco pode ser selecionado por vez
- Há vários registradores para a programação dos timers, do controlador da serial e de interrupção; esses registradores são mapeados na RAM dedicada aos SFR

Microcontrolador 80C51

Ponteiros (contem endereço)

- **PC** (*Program Counter*): ponteiro de 16 bits para área de programa (EPROM)
- **DPTR** (*Data Pointer*) : ponteiro de 16 bits para área de dados em memória EPROM ou RAM Externa
- **SP** (*Stack Pointer*) : ponteiro de pilha (8 bits), localizado na RAM Interna (inicia com endereço 07H, e incrementa o seu valor de uma unidade antes de guardar um dado na pilha.
- **R0** : ponteiro de 8 bits para RAM interna ou RAM externa
- **R1** : ponteiro de 8 bits para RAM interna ou RAM externa

Modos de Endereçamento do 8051

1. Endereçamento Imediato

   • Opera sobre o dado localizado na própria instrução

• Identificado através do sinal #

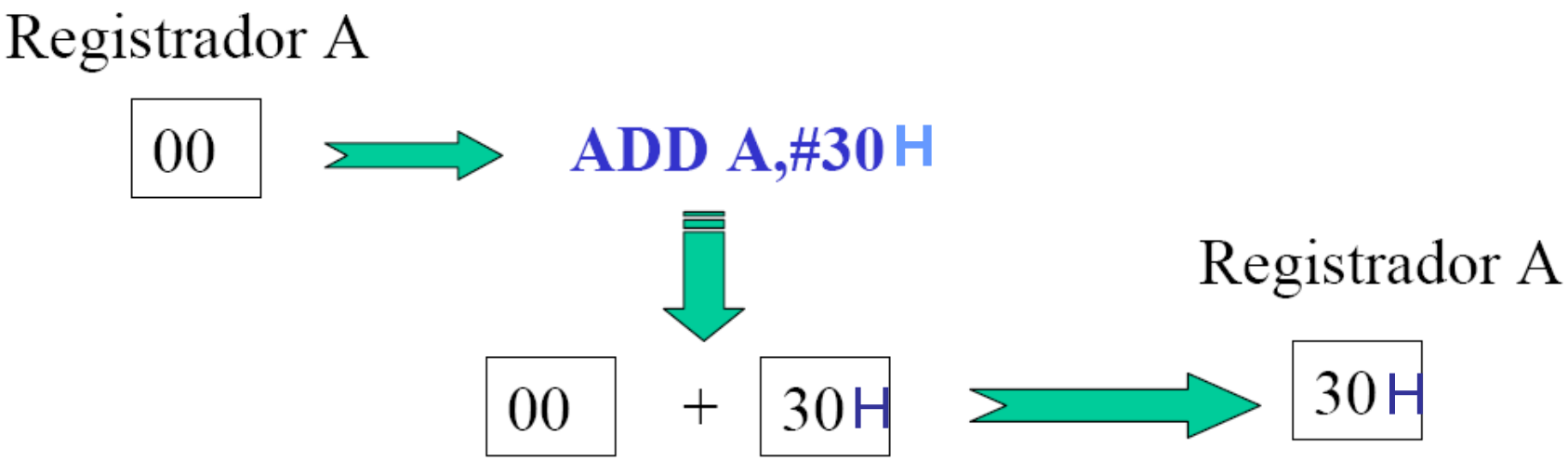
• Exemplo: **ADD A,#30H**

O dado 30 é somado ao Registrador A

Modos de Endereçamento do 8051

1. Endereçamento Imediato

ADD A,#30H



Modos de Endereçamento do 8051

2. Endereçamento Direto

- Opera sobre o dado cujo endereço está na instrução

- Exemplo:

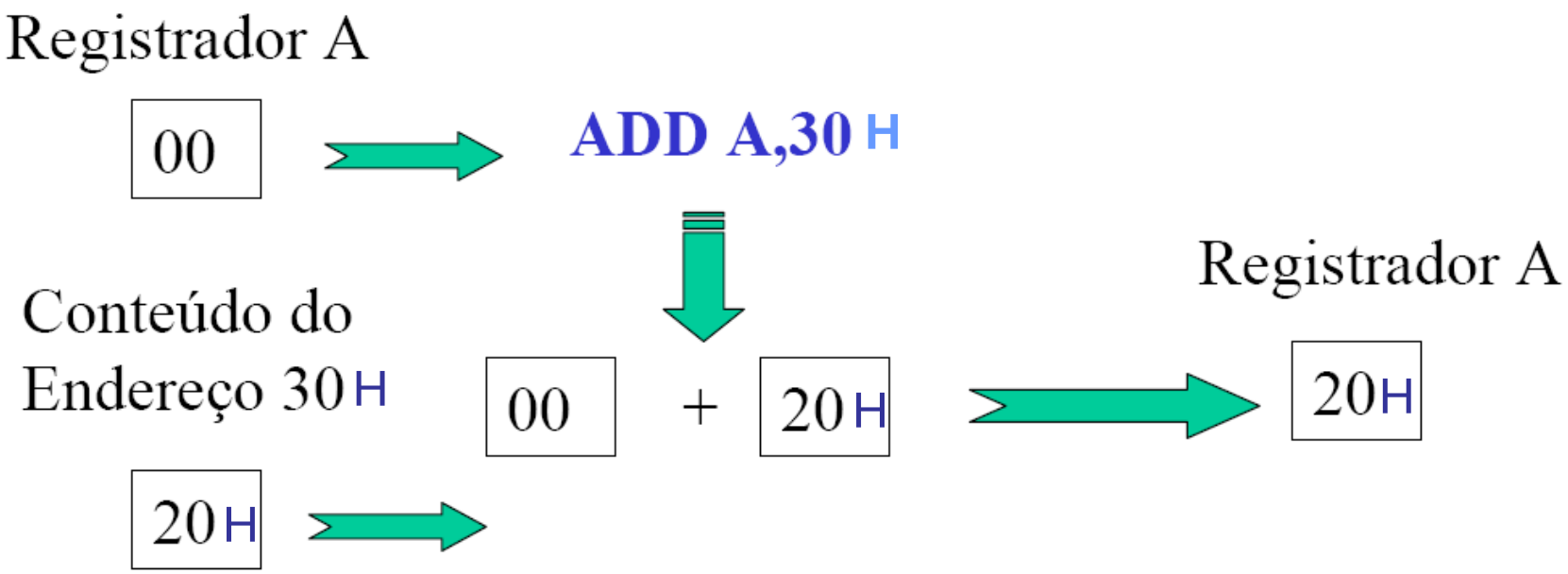
ADD A,30H

O dado armazenado no endereço 30 é somado ao Registrador A

Modos de Endereçamento do 8051

2. Endereçamento Direto

ADD A,30H



Modos de Endereçamento do 8051

3. Endereçamento Indireto

- Opera sobre o dado cujo endereço está armazenado em um Registrador apontado na instrução

- Identificado através do sinal @

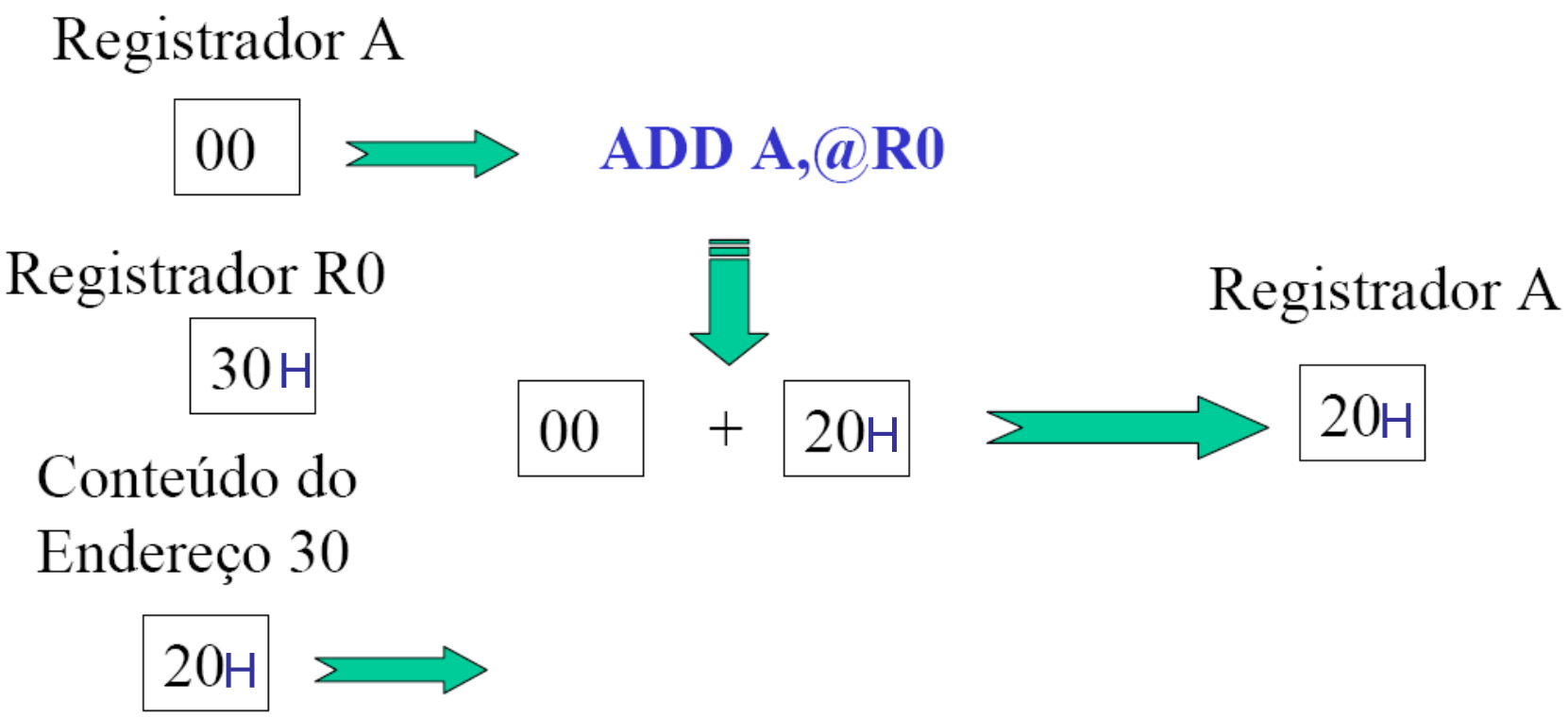
- Exemplo: **ADD A,@R0**

O dado armazenado no endereço apontado pelo Registrador R0 é somado ao Registrador A

Modos de Endereçamento do 8051

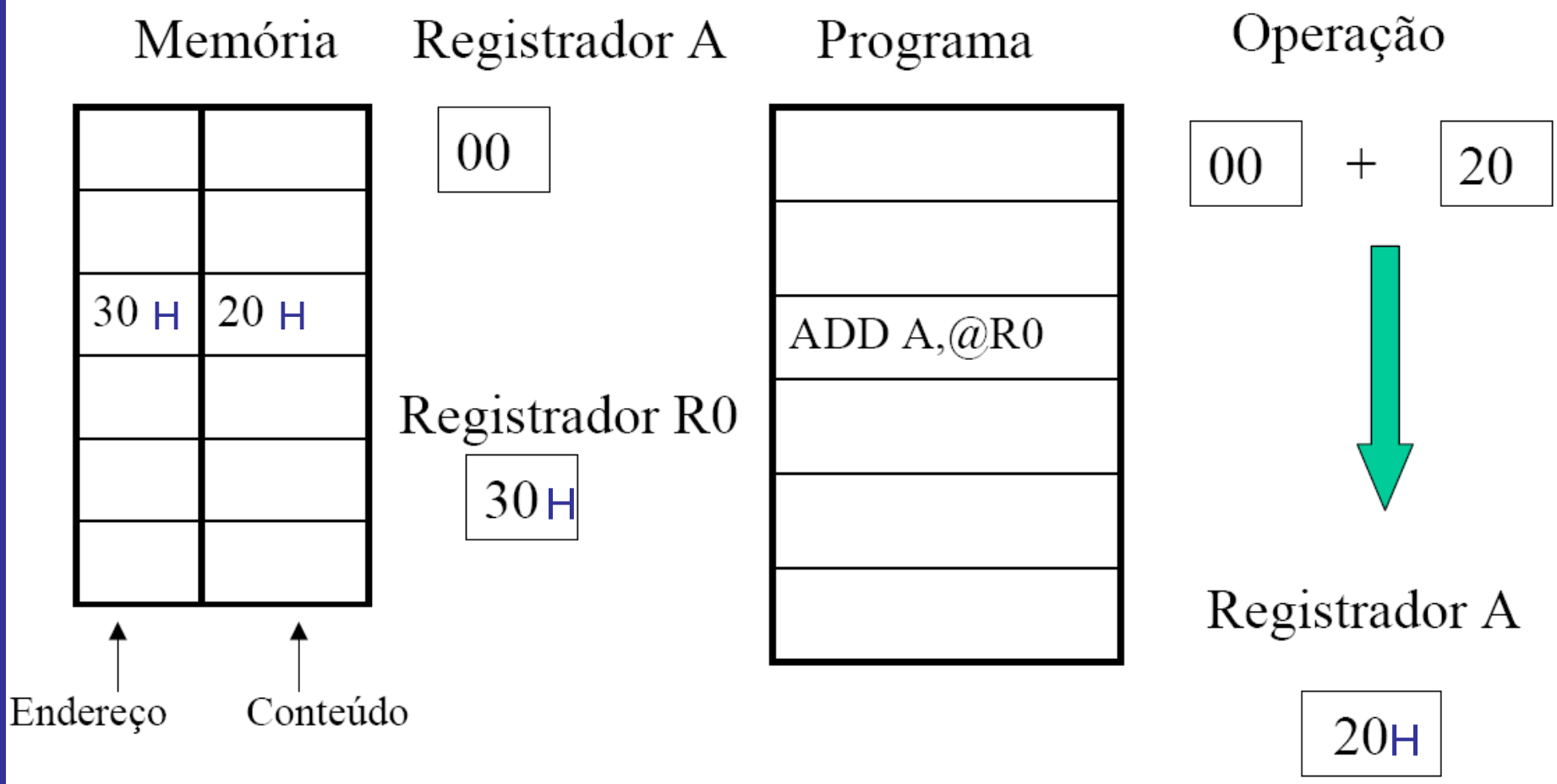
3. Endereçamento Indireto

ADD A,@R0



Modos de Endereçamento do 8051

ADD A,@R0



Instrução para Memória de Programa

- Acesso a área de dados em EPROM
- É denominado modo de endereçamento Indexado, pois facilita o acesso a tabelas
- **é endereçável pelo ponteiro de dados DPTR.**

Instrução :

MOVC A,@A+DPTR

Exemplo:

```
MOV   DPTR, #0F0BH
CLR   A
MOVC A,@A+DPTR
```

Instruções para Memória de Dados Externa

- espaço de endereço de 64K bytes
- espaço todo é indiretamente endereçável pelo ponteiro de dados DPTR.

Instruções :

movx a,@DPTR

movx @DPTR,a

Instruções para Memória de Dados INTERNA (RAM)

- Todas as instruções MOV
- Exemplo:

  - **Endereçamento imediato:** MOV **Ri**,#0F8H
  - **Endereçamento direto:** MOV **Ri**,6FH

    MOV 34H, 7FH
  - **Endereçamento indireto :** MOV A, @R1

    MOV A, @R0
  - **Endereçamento por registrador** : MOV A,**Ri**

  OBS: Ri pode ser escolhido de R0 a R7

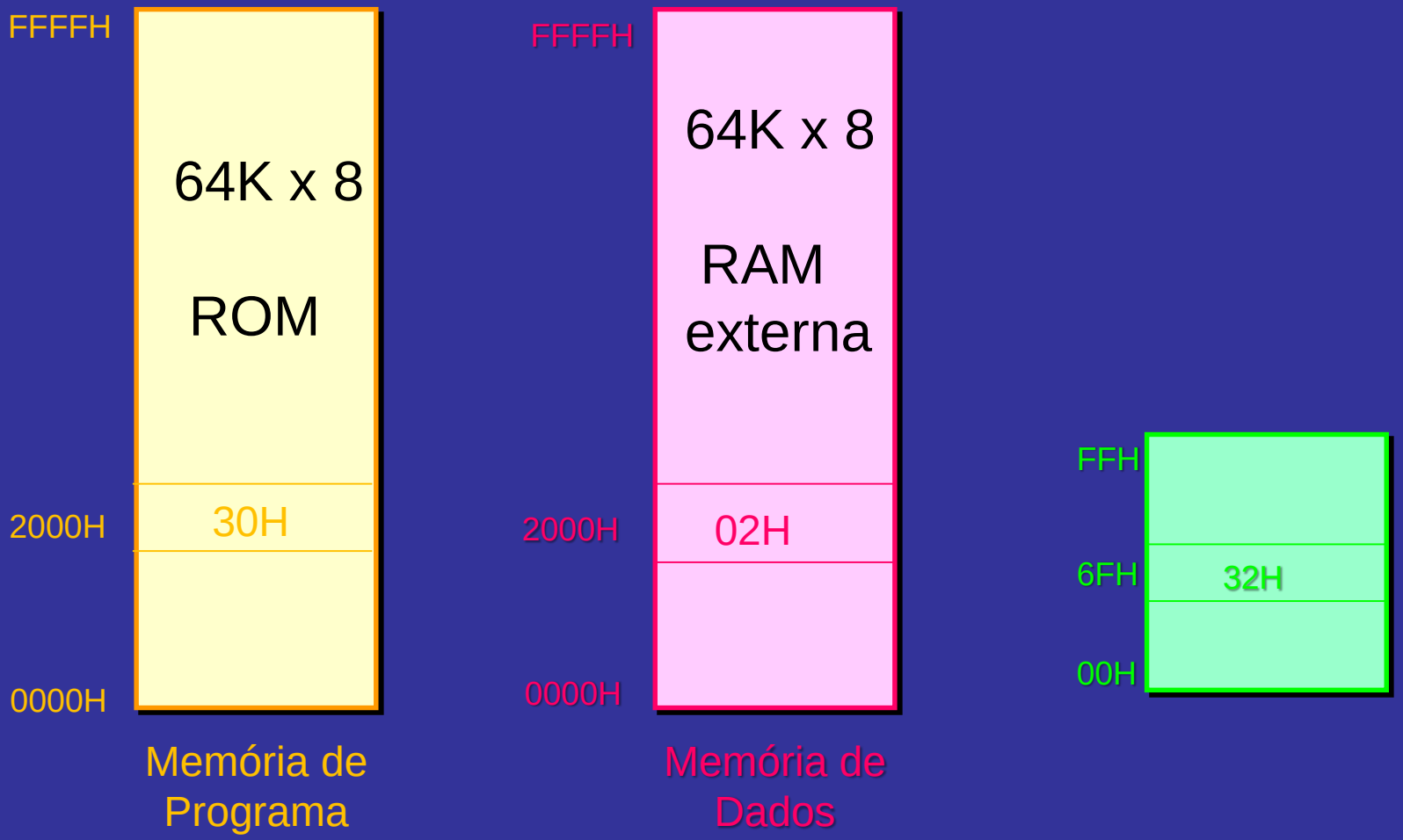
Exemplo de Programa em Assembly pra o 8051

- Programa que lê um dado na posição 2000h da memória de programa e soma com um dado armazenado na memória externa na mesma posição e armazena na posição 6Fh da RAM interna.

```
            ORG 0
MOV         DPTR, #2000H ; ponteiro DPTR contém 2000H
CLR         A            ; limpa acumulador (A=0)
MOVC A,@A+DPTR   ;  valor do acumulador ( A=0) é somado com  o valor do  DPTR (2000H)
                 ; e o conteúdo da posição de memória de programa EEPROM
                 ; (2000H +  0 = 2000H) é carregado no acumulador (por ex:
                 ;considerando que o conteúdo do endereço 2000H da EEPROM seja
                 ;30H, A=30H)
MOV R1,A         ; o valor do acumulador é copiado em R1 (R1= 30H)
MOVX A,@DPTR     ;  o conteúdo da posição apontada pelo DPTR (2000H) na RAM externa
                 ;é copiado para o acumulador (por ex: considerando que o conteúdo da
                 ;posição 2000H da RAM externa seja 02H, então A=02H)
ADD A,R1         ; o conteúdo de R1(30H) é somado com o conteúdo do acumulador e o
                  ;resultado é armazenado no acumulador, então A= 02H + 30H = 32H
MOV 6Fh,A        ; o valor do acumulador é copiado para a posição 6Fh da RAM interna,
                 ;então o conteúdo de 6FH na RAM interna  torna-se 32H
SJMP $           ; fim do processamento
END              ; fim físico do programa
```

Exemplo de Programa em Assembly pra o 8051

- Programa que lê um dado na posição 2000h da memória de programa e soma com um dado armazenado na memória externa na mesma posição e armazena na posição 6Fh da RAM interna.

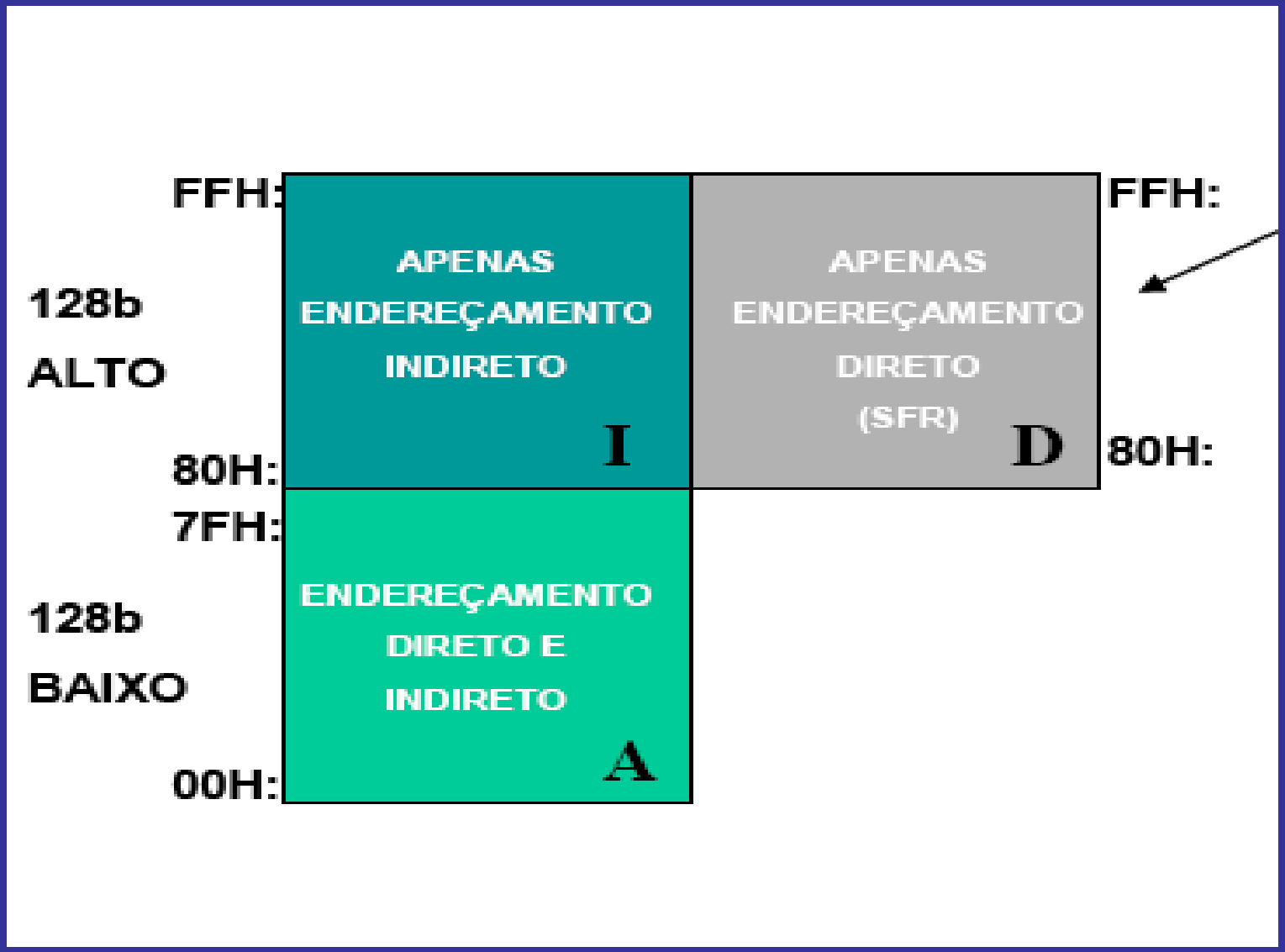


Mapeamento de memória interna, pilha e Interrupção

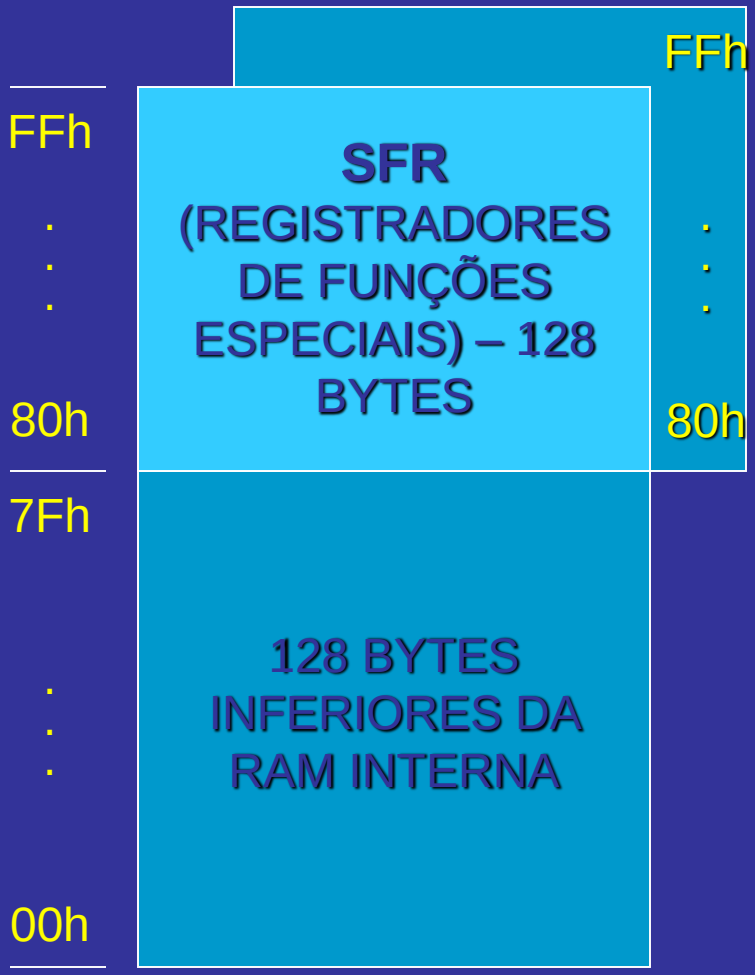
Memória de dados interna (RAM)

- 8051 oferece uma memória de dados interna, com um mínimo de 128 bytes
  - Vantagem 1: rápido acesso aos dados e, em muitas aplicações, pode eliminar a necessidade da RAM externa ⇨ custo menor;
  - Vantagem 2: **áreas de RAM interna acessíveis bit** a bit ⇨ útil para operações booleanas.
- Espaço de endereçamento para acessar a RAM interna = 8 bits ➡ máximo de 256 bytes (8052 ➡ mais 128 bytes = 384 bytes)
- RAM interna ➡ 4 bancos de 8 registradores **(R0 – R7)** que podem ser utilizados pelo usuário

Memória de dados interna (RAM)



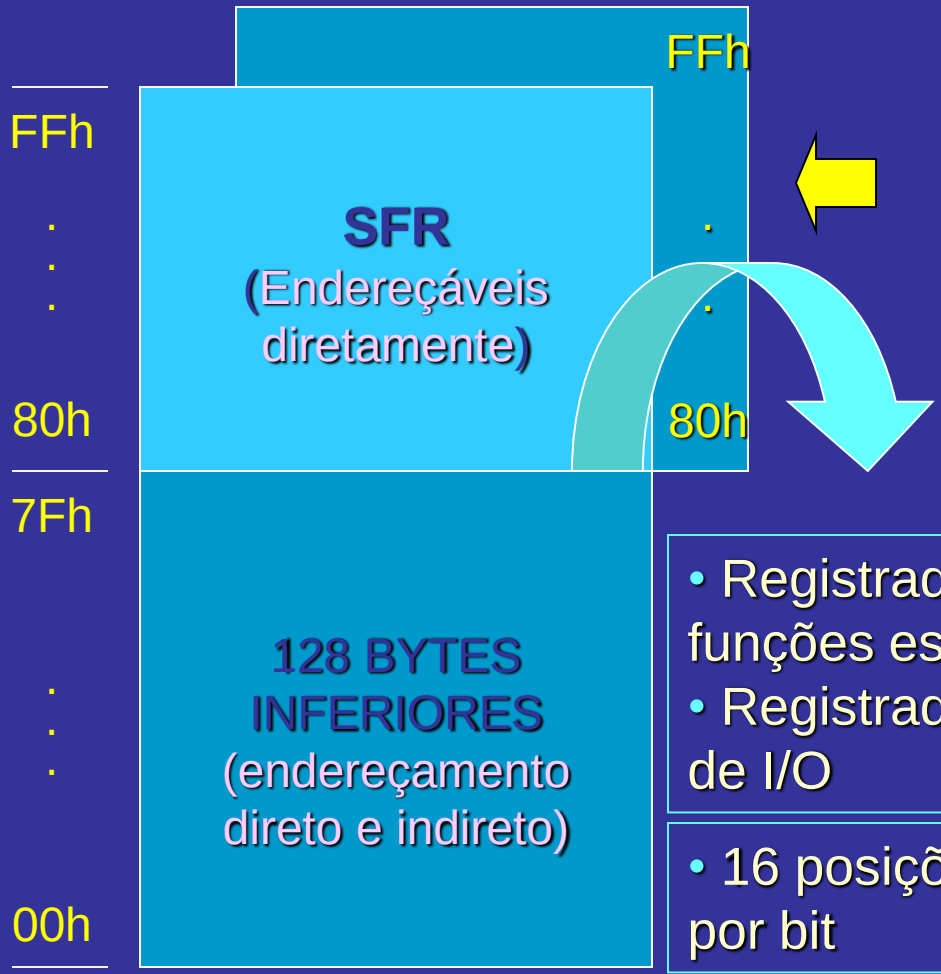
Memória de dados interna (RAM)



128 BYTES SUPERIORES**

(**) Só para 8052 – endereçáveis indiretamente

Registradores de Funções Especiais (SFR)

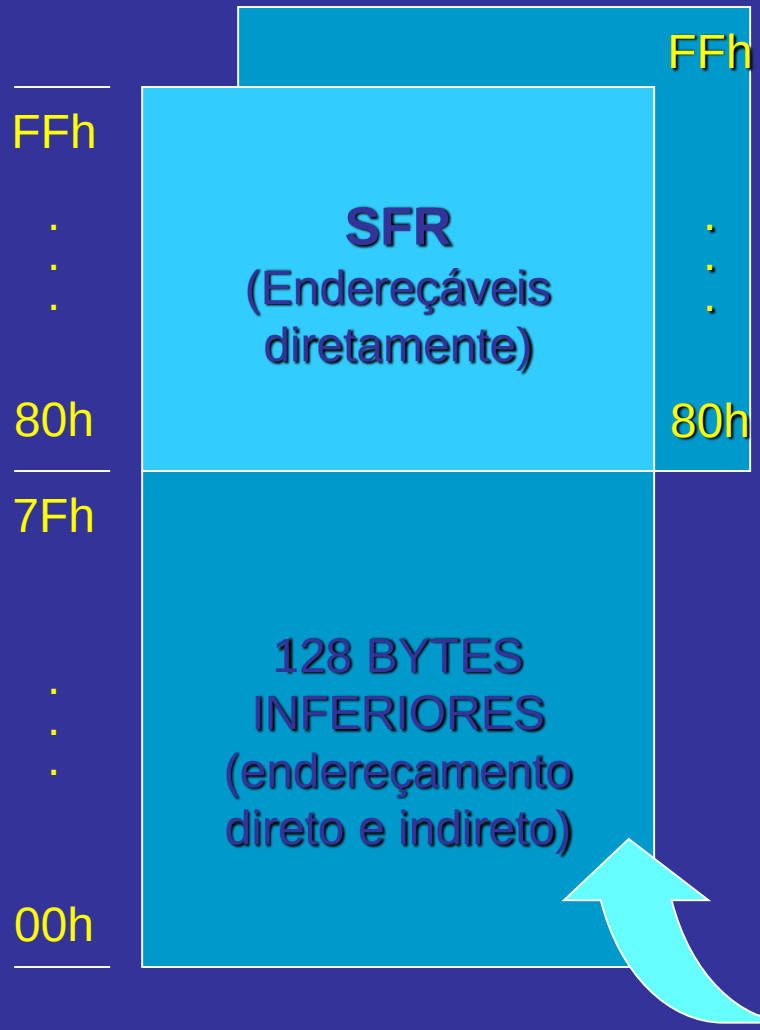


128 BYTES SUPERIORES (apenas endereçamento indireto)

- Registradores da CPU para funções especiais
- Registradores de controle de I/O

- 16 posições endereçáveis por bit

Registradores de Propósito Geral (GPR)



128 BYTES SUPERIORES (apenas endereçamento indireto)

- Registradores para uso do programador
- 4 bancos de registradores Ri para facilitar a programação
- 16 posições endereçáveis por bit

Memória de dados interna (RAM)

(**) Endereçamento direto e indireto

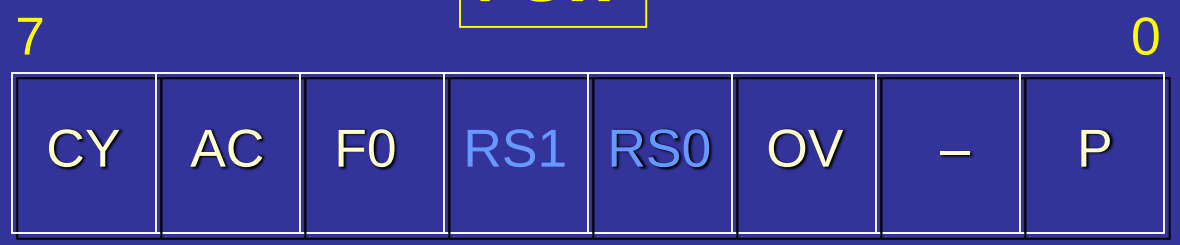
(*) Endereçamento feito com 8 bits

Memória de dados interna (RAM)

BANCO DE REGISTRADORES

- Formados pelos registradores R0 a R7
- Seleção entre os Bancos, feita pelos bits 3 e 4 do byte PSW (*program status word*)

PSW



Flag de CARRY

Flag de CARRY Auxiliar*

Flag 0

Bits de controle do banco de registradores

Flag de OVERFLOW

Flag de Paridade

RS1	RS0	Banco	Endereço
0	0	0	00 – 07h
0	1	1	08 – 0Fh
1	0	2	10 – 17h
1	1	3	18 – 1Fh

(*) *Para operações BCD*

Microcontrolador 80C51

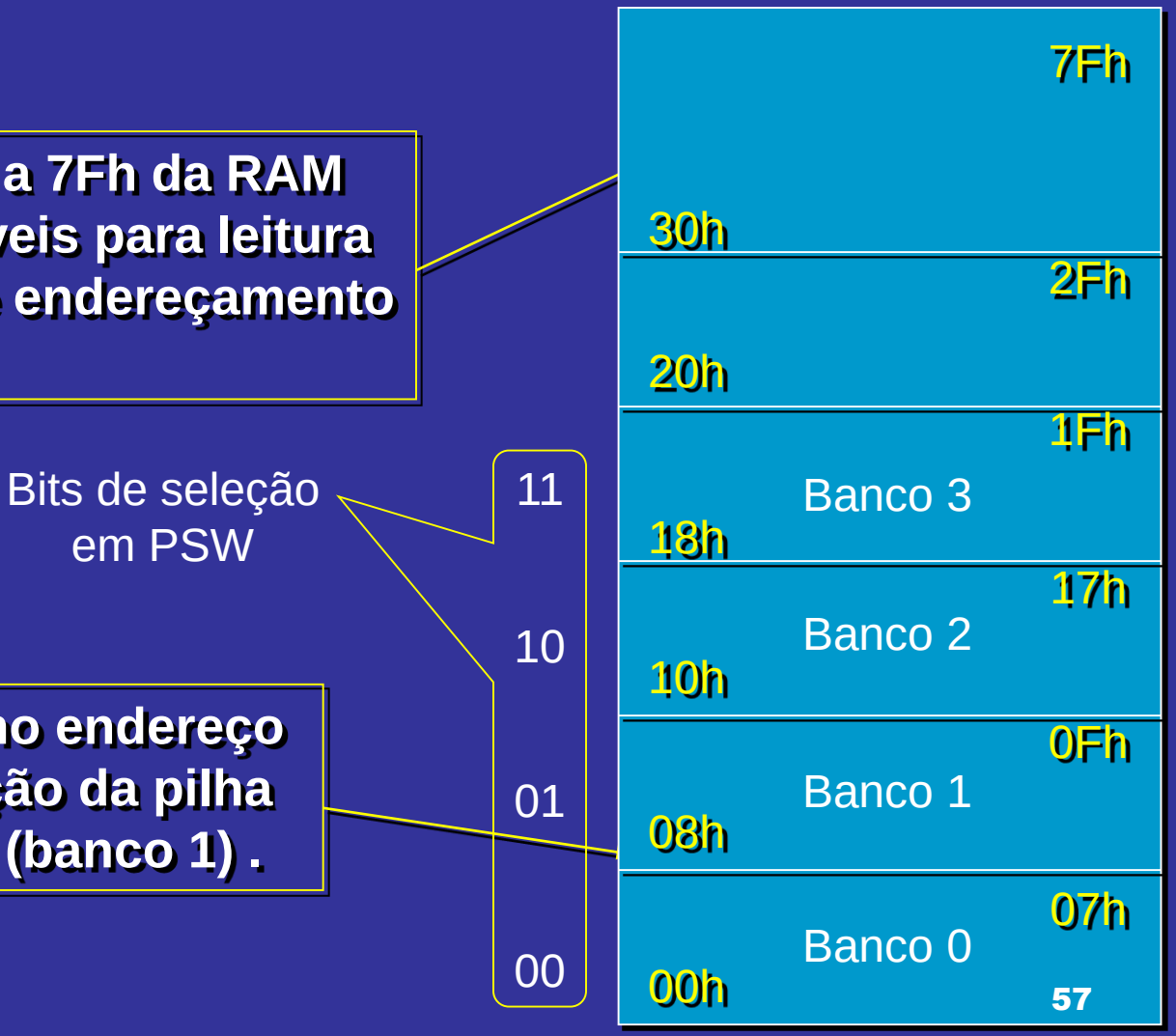
- *Flags* ➡ bits indicadores de estado:
  - contidos no registrador PSW (palavra de status do programa – *"program status word"*)
  - são colocados em "1" ou "0" dependendo do resultado das operações da CPU
  - algumas instruções testam *flags* para definir se a execução do programa prossegue na instrução seguinte, ou se salta para outra parte do programa
  - *flags* típicas: *SIGN, CARRY, ZERO, OVERFLOW*
  - bit de *flag* usualmente se refere ao estado do A
  - bit de sinal = MSB do A após a operação da ULA

Memória de dados interna (RAM)

Área de dados

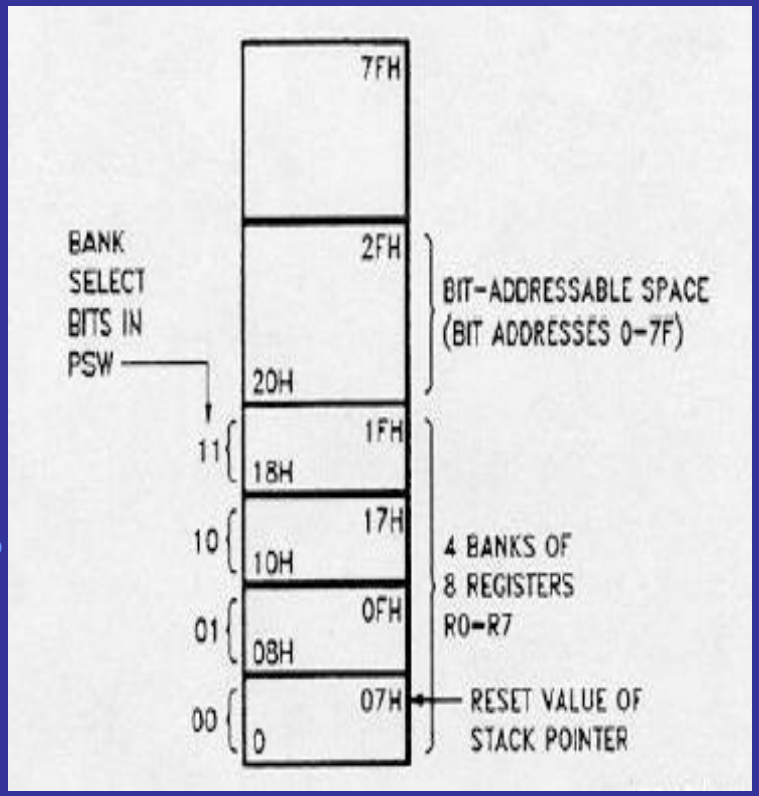
As posições de 30h a 7Fh da RAM interna são disponíveis para leitura e escrita, através de endereçamento direto e indireto.

O SP é inicializado no endereço 07H, mas a 1ª. Posição da pilha a ser gravada é 08H (banco 1) .



Memória de dados interna (RAM)

Bytes endereçáveis



Cada uma dessas posições de memória pode também ser acessada direta ou indiretamente por byte .

Ex.:

a) Endereçamento Direto

mov 20h,#0AAh

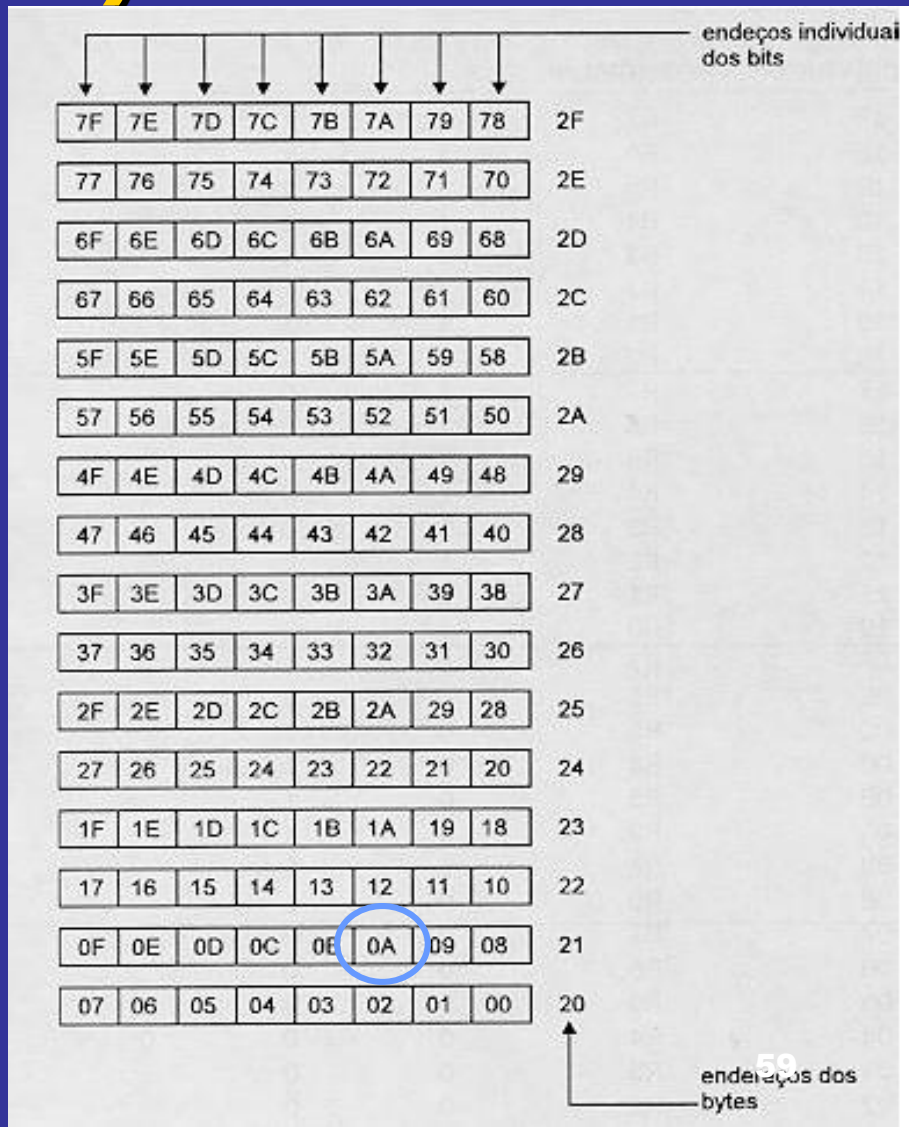
b) Endereçamento Indireto

mov R0,#20h

mov @R0,#0AAh

Memória de dados interna (RAM)

Faixa de RAM interna endereçável por Bit (de 20H a 2FH).



Ex. : Setar o bit 2 da posição 21h

setb 0Ah

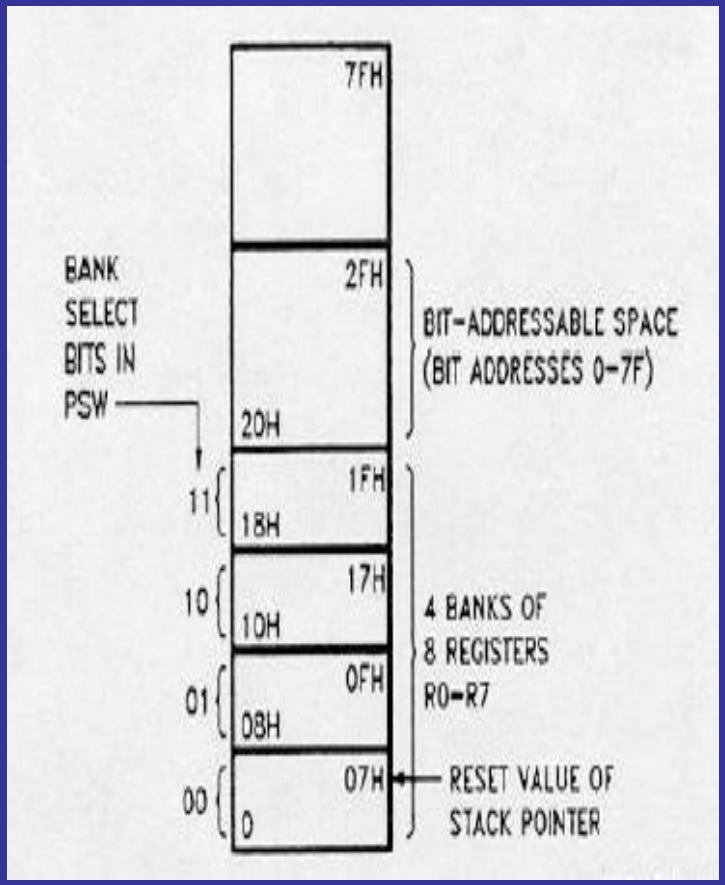
ou

setb 21h.2

Existe um conjunto de instruções específicos para endereçamento a bit.

Memória de dados interna (RAM)

Bits endereçáveis



INSTRUÇÕES DE BIT:

CLR bit → **zera o bit diretamente**

SETB bit → **seta o bit diretamente**

CPL bit → **complementa o bit diretamente**

ANL C,bit → **AND entre o bit e o *carry***

ANL C,/bit → **AND entre o complemento do bit e o *carry***

ORL C,bit → **OR entre o bit e o *carry***

ORL C,/bit → **OR entre o complemento do bit e o *carry***

MOV C,bit → **move o bit para o *carry***

MOV bit,C → **move o *carry* para o bit**

JB bit,rel → **jmp para rel se bit = 1**

JNB bit,rel → **jmp para rel se bit = 0**

JBC bit,rel → **jmp para rel se bit = 1 e zera o bit**

Registradores de Funções Especiais (SFR) – Mapa

Registradores de Funções Especiais (SFR)

Qualquer dos SFR pode ser endereçado a byte **diretamente ou imediatamente** através do endereço de cada um ou do nome

Exemplo:

mov P0,#3Fh	ou	***mov 80h,#3fh***
mov DPL,DPH	ou	***mov 82h,83h***

Registradores de Funções Especiais (SFR)

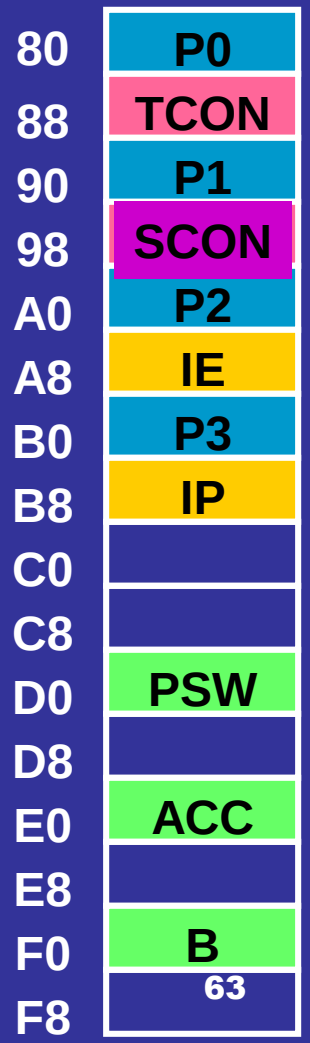
SFR endereçáveis a bit

- Os SFR's cujos endereços terminam em 0 ou 8h podem também ser endereçados a bit
- Modos de acesso ao bit:

(I) por endereço do Bit dentro do Byte:

1. ***setb 80h.1***; seta o bit 1 do endereço 80h (Porta 0)
2. ***clr 80h.2*** ; zera o bit 2 do endereço 80h (Porta 0)

P0	P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0
80h	80.7	80.6	80.5	80.4	80.3	80.2	80.1	80.0



Registradores de Funções Especiais (SFR)

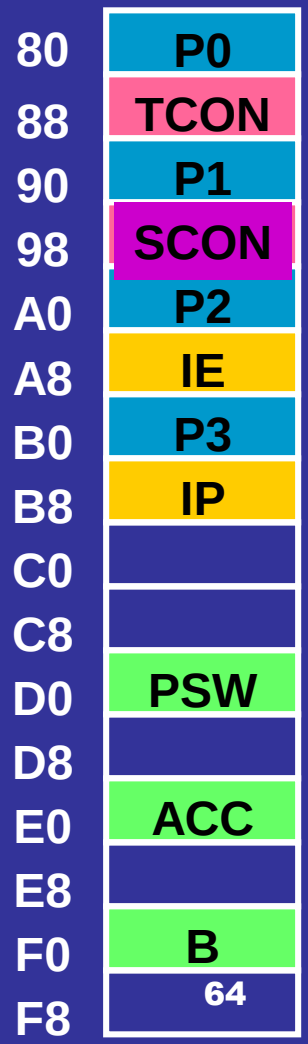
SFR endereçáveis a bit

- Os SFR's cujos endereços terminam em 0 ou 8h podem também ser endereçados a bit
- Modos de acesso ao bit:

(II) por nome :

1. ***setb P0.1*** ; set no bit 1, do endereço 80h (Porta 0)
2. ***clr P0.2*** ; zera o bit 2, do endereço 80h (Porta 0)

P0	P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0
80h	80.7	80.6	80.5	80.4	80.3	80.2	80.1	80.0



Registradores de Funções Especiais (SFR)

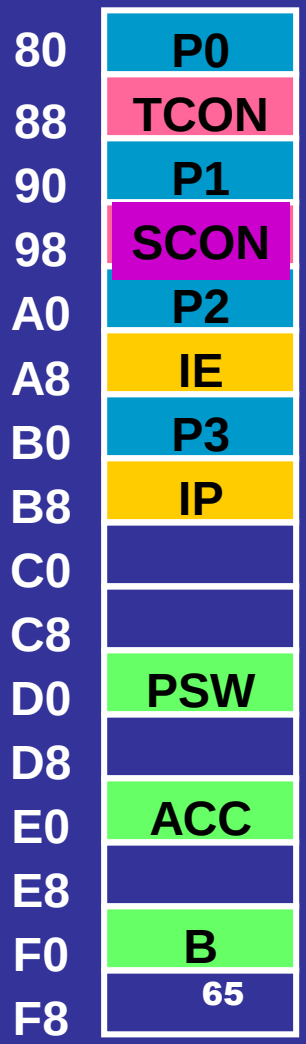
SFR endereçáveis a bit

- Os SFR's cujos endereços terminam em 0 ou 8h podem também ser endereçados a bit
- Modos de acesso ao bit:

(III) pelo endereço absoluto do bit :

1. ***setb 81h*** ; seta o bit 1 do endereço 80h (Porta 0)
2. ***clr 82h*** ; zera o bit 2 do endereço 80h (Porta 0)

P0	P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0
80h	80.7	80.6	80.5	80.4	80.3	80.2	80.1	80.0
	87	86	85	84	83	82	81	80



Registradores de Funções Especiais (SFR)

SFR endereçáveis a bit

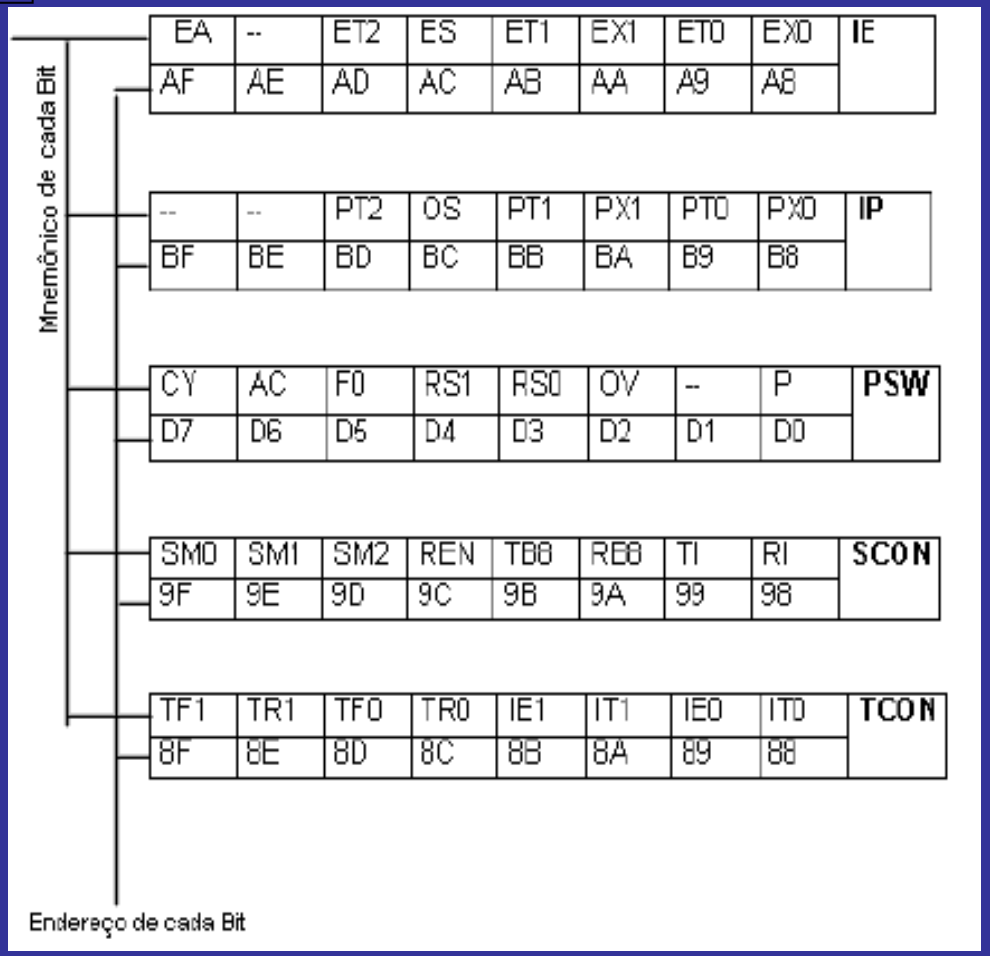
(IV) pelo nome do bit :

Os SFRs endereçáveis a bit que determinam funções, podem ser endereçados através do nome de cada bit:

Ex.:

setb EA ; faz o bit 7 de IE=1

setb 0AFh ; idem



Microprocessador de 8 Bits

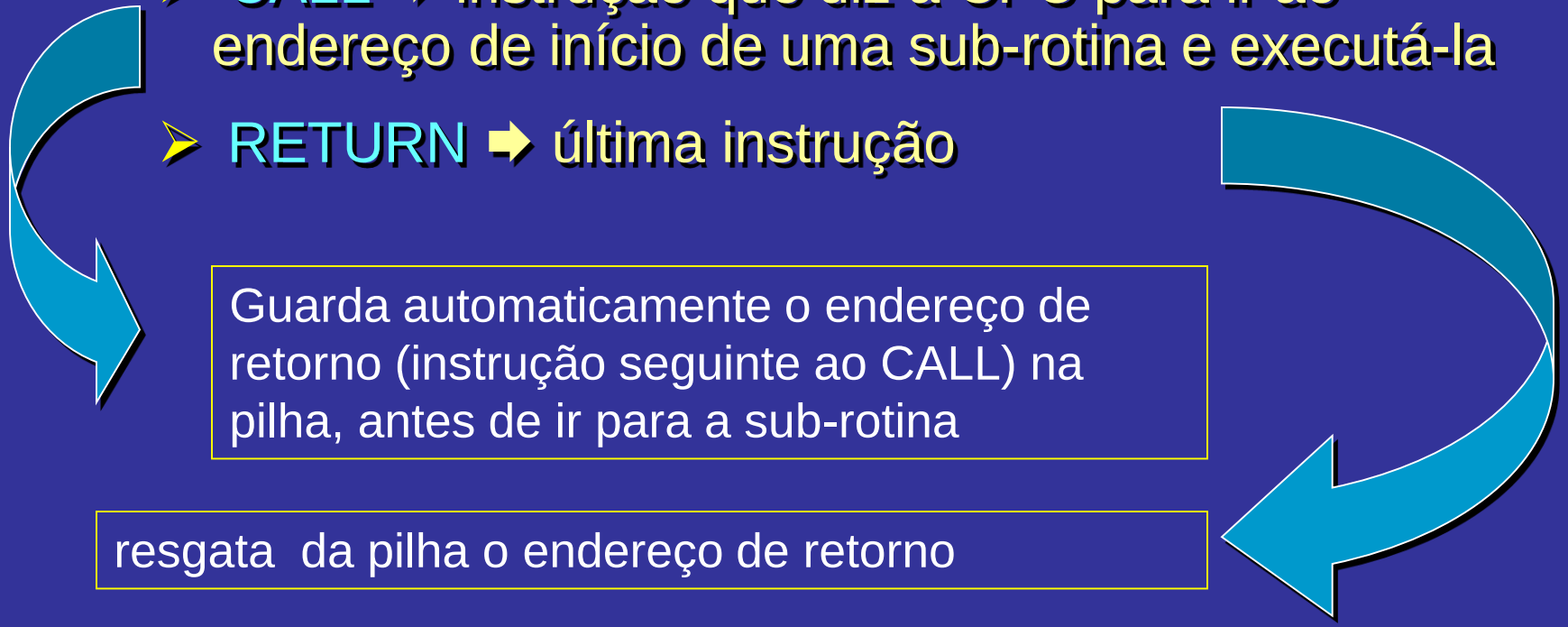
PILHA

- Pilha ➡ parte da RAM interna dedicada para armazenar dados sob o controle do ponteiro SP.
  - Solicitação de Interrupção e chamada de subrotina usam a pilha.
  - Existem instruções para armazenar e retirar dados da pilha.
  - PONTEIRO DE PILHA (*"Stack pointer"*) ➡ aponta para o topo da pilha onde o último dado foi armazenado.
  - No armazenamento os dados são empilhados sequencialmente.

Microprocessador de 8 Bits

ALGUMAS DEFINIÇÕES ÚTEIS

- uso mais importante ➡ chamada de sub-rotina:
- CALL ➡ instrução que diz à CPU para ir ao endereço de início de uma sub-rotina e executá-la
- RETURN ➡ última instrução

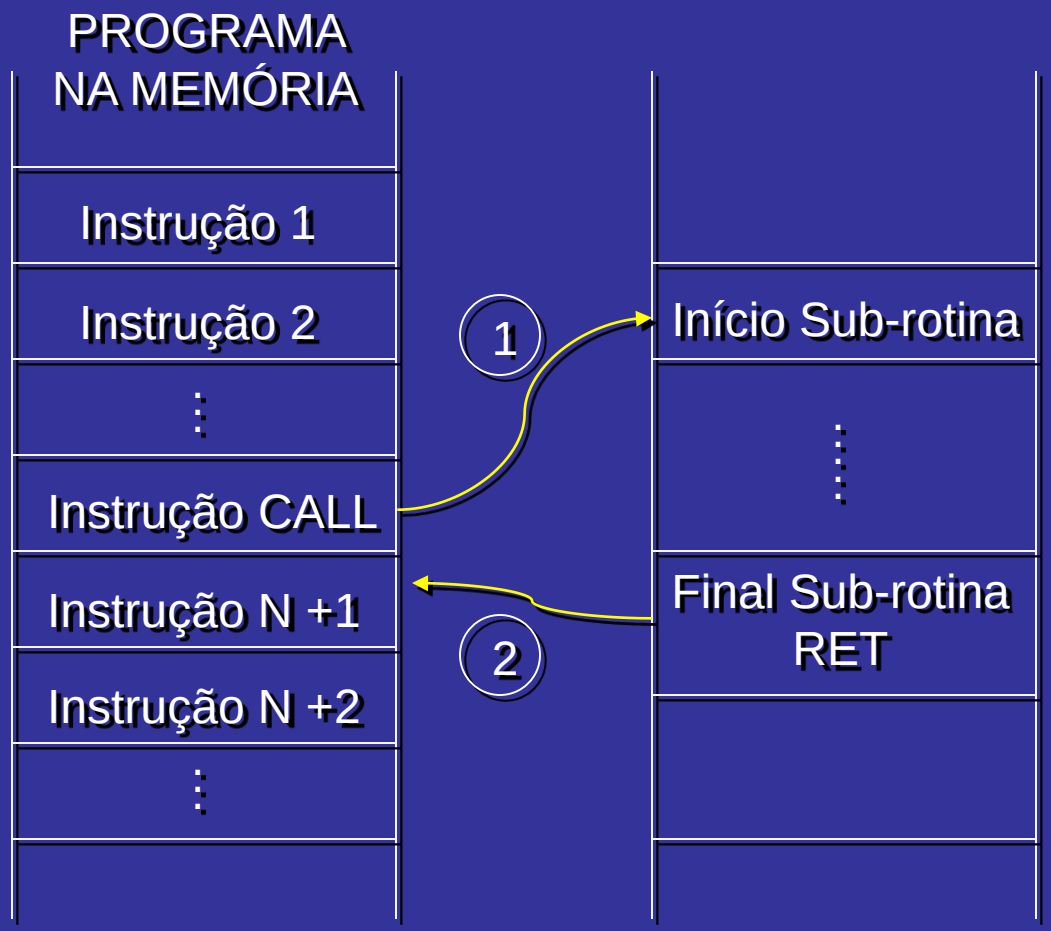


Guarda automaticamente o endereço de retorno (instrução seguinte ao CALL) na pilha, antes de ir para a sub-rotina

resgata da pilha o endereço de retorno

Microprocessador de 8 Bits

ALGUMAS DEFINIÇÕES ÚTEIS

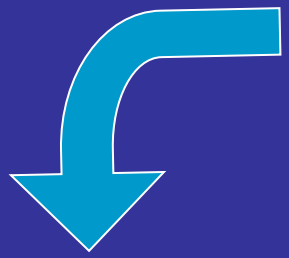


1 ➡ Armazena end. retorno na pilha e vai à sub-rotina

2 ➡ End. retorno da pilha e volta ao programa

Memória de dados interna (RAM)

PILHA

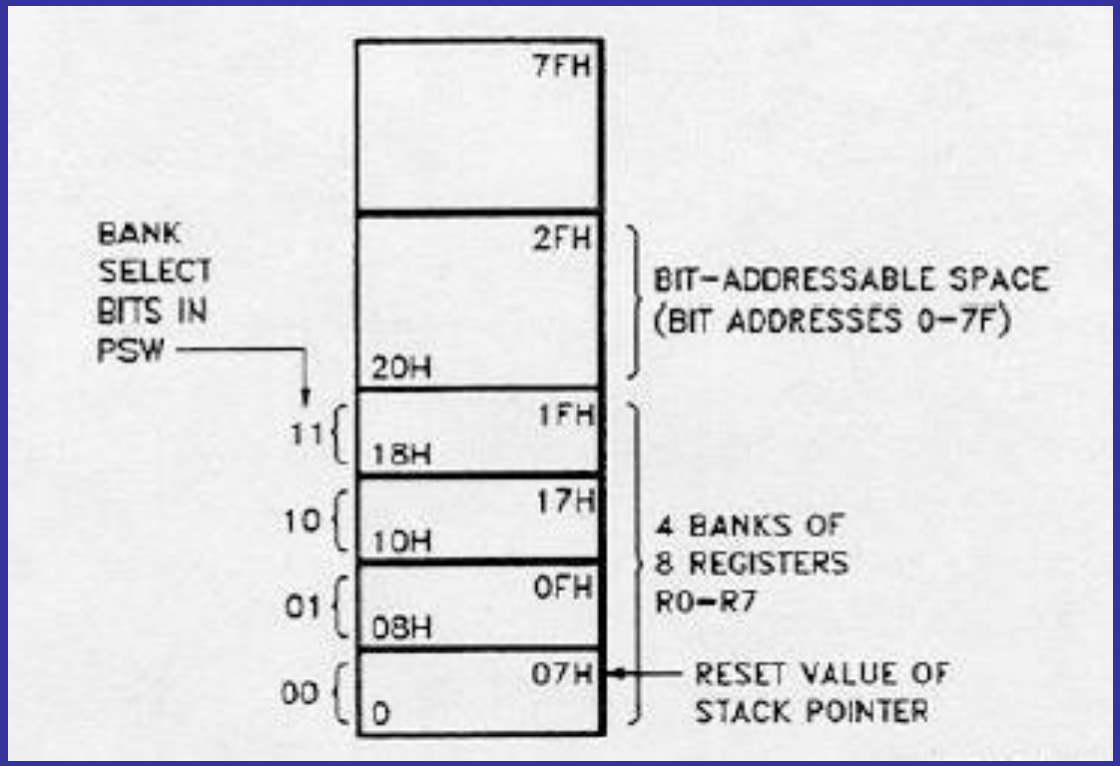


pilha começa em 08H (banco 1 de registradores)

- **O reset inicializa o PONTEIRO DA PILHA (*Stack Pointer* – SP) na posição 07h, e é incrementando a cada vez que é usado.**
- **como o SP tem largura de 8 bits, são necessárias 2 posições para armazenamento de cada endereço**
- **a pilha cresce através da memória (SP é incrementado antes dos dados serem armazenados)**
- **o ponteiro da pilha pode ser inicializado ficar em qualquer endereço na RAM interna(a escolha do programador)**

Memória de dados interna (RAM)

PILHA



- Ao se aplicar reset na CPU, RS1 e RS0 = 0
- o banco de registradores 'default' é o **Banco 0.**

*Para usar mais de um Banco de Registradores, **SP** deve ser inicializado em outra posição da RAM (por ex, 30h)*

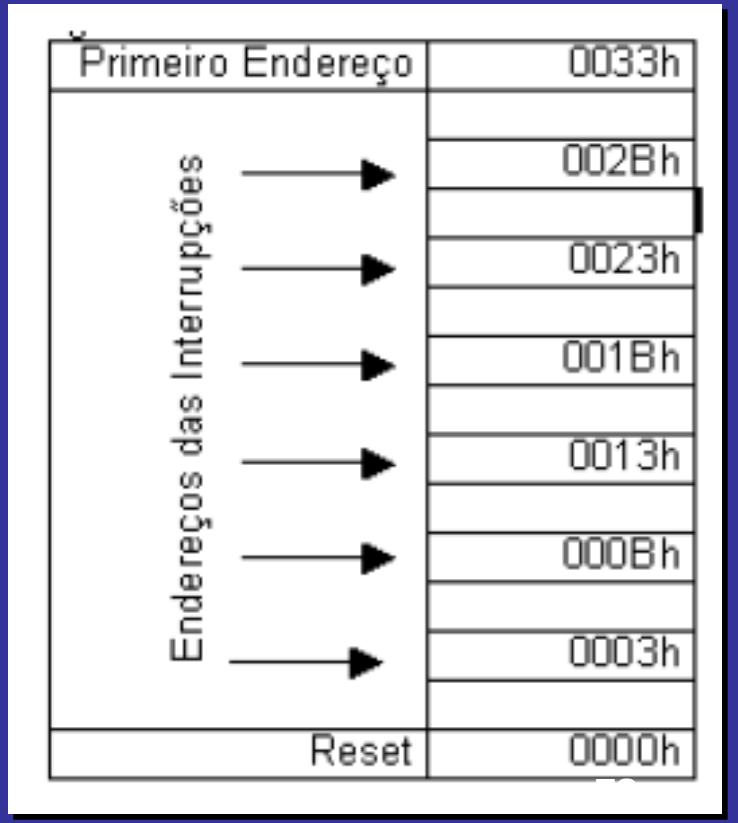
Microcontrolador 80C51

Interrupção

MEMÓRIA DE PROGRAMA

Endereço das interrupções:

- Cada interrupção causa um salto para um endereço fixo na memória de programa (ROM , EPROM ,...) a partir do endereço 0003 H
- 5 interrupções: 2 externas, 2 timers/contadores e 1 porta serial;



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**.

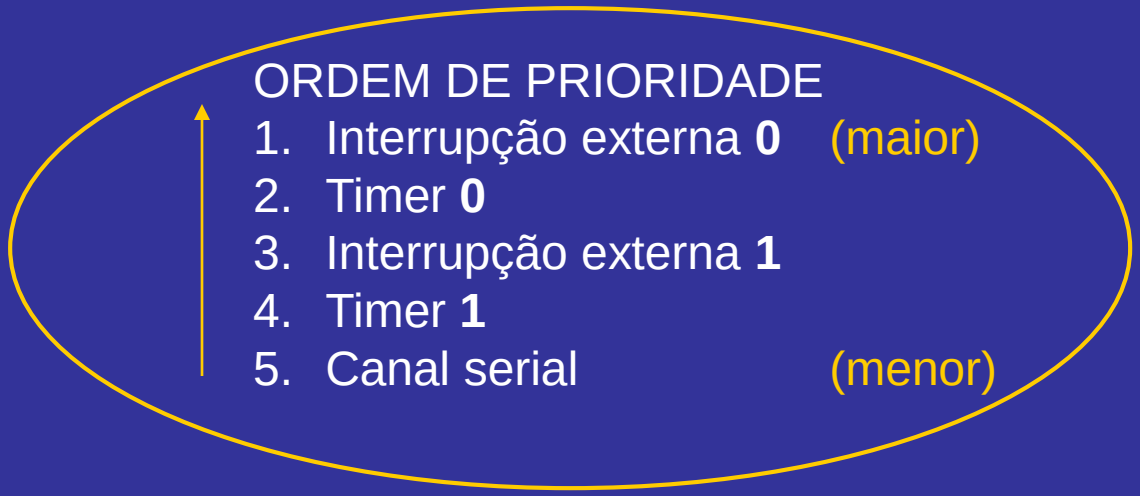
Ao terminar a execução desta sub-rotina o controle volta para o programa inicial no endereço imediatamente abaixo do ponto onde foi interrompido.

Estrutura de Interrupção

- Procedimento:
  - Atendendo a uma interrupção, o μP pára a execução do programa e vai executar uma sub-rotina (*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.

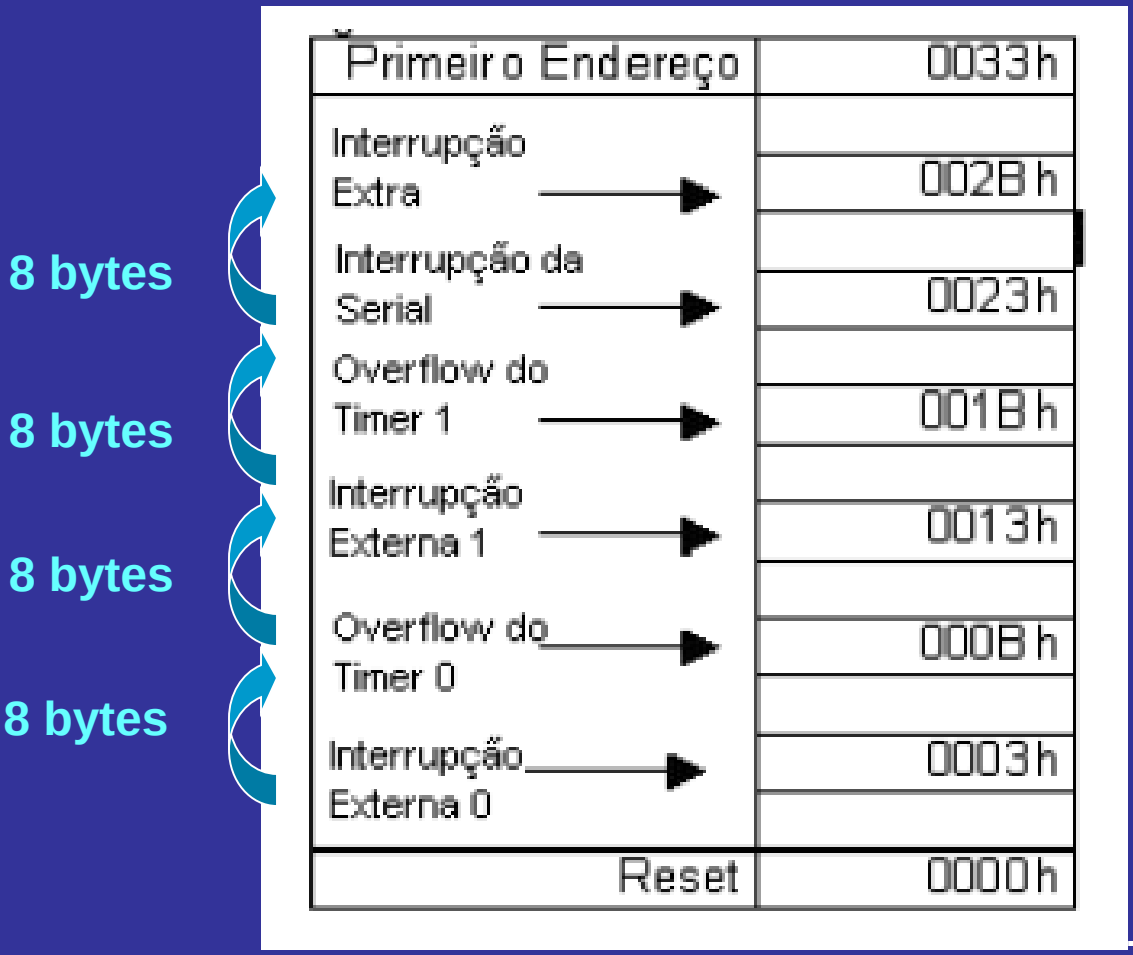
Interrupção

- Cada interrupção pode ser habilitada individualmente ou não e todas podem ser desabilitadas de uma só vez
- Cada interrupção pode ter níveis de prioridade:



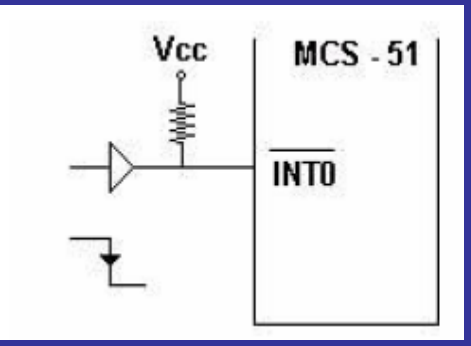
Interrupção

- Endereço das interrupções:



Estrutura da Interrupção

Ex. Programação da Interrupção Externa 0 sensível à borda de descida



IE0 → 0003h

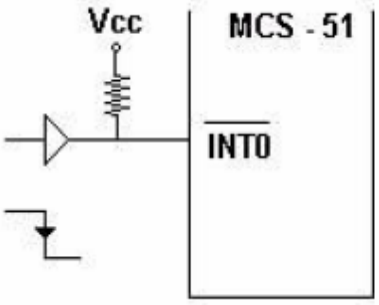
Programa principal:

```
ORG 100H        ; Origem do Programa fora da área de Interrupções

SETB EA         ; Habilita o uso de Interrupções
SETB IE.0       ; Habilita a Interrupção Externa 0
SETB IT0        ; Estabelece que deve ser sensível a descida de borda
...
...             ; Comandos do Programa Principal
...
END
```

Estrutura da Interrupção

Ex. Programação da Interrupção Externa 0 sensível à borda de descida



IE0 → 0003h

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
```

Software Real

```
        ORG 0
        MOV A,#00000001B ;faz acumulador = 00000001
        MOV P1, A        ;move acumulador para a Porta 1
LEITURA:
        JNB P3.5, LEFT   ;pula para LEFT se P3.5 = 0, senão próx. linha
RIGHT:
        RR A             ;roda byte do Acumulador para direita
        MOV P1, A        ;move Acumulador para a Porta 1
        ACALL TEMPO      ;gasta tempo
        SJMP LEITURA     ;lê bit P3.5 novamente
LEFT:
        RL A             ;roda byte do Acumulador para esquerda
        MOV P1, A        ;move Acumulador para a Porta 1
        ACALL TEMPO      ;gasta tempo
        SJMP LEITURA     ;lê bit P3.5 novamente
TEMPO:
        xxx              ;sub-rotina para gastar tempo
        RET              ;retorna da sub-rotina
        END              ;fim do programa (compilador)
```

Como fica no 8051?

Code Window (Disassembly)

Addr	Opcodes	ASC	Label	Disassembly	Comments
0000	74 01	t☺		MOV A,#01h	faz acumulador = 00000001
0002	F5 90	õ☺		MOV P1,A	move acumulador para a Porta 1
0004	30 B5 07	0µ•	LEITURA	JNB P3.5,LEFT	pula para LEFT se P3.5 = 0, senão próx.
0007	03	♥	RIGHT	RR A	roda byte do Acumulador para direita
0008	F5 90	õ☺		MOV P1,A	move Acumulador para a Porta 1
000A	11 15	‼§		ACALL TEMPO	gasta tempo
000C	80 F6	€ö		SJMP LEITURA	lê bit P3.5 novamente
000E	23	#	LEFT	RL A	roda byte do Acumulador para esquerda
000F	F5 90	õ☺		MOV P1,A	move Acumulador para a Porta 1
0011	11 15	‼§		ACALL TEMPO	gasta tempo
0013	80 EF	€ï	(EXT1 INT)	SJMP LEITURA	lê bit P3.5 novamente
0015	22	"	TEMPO	RET	
0016	00	♥		NOP	
0017	00	♥		NOP	

FIM