



Departamento de Engenharia Elétrica e de Computação - EESC-USP

SEL-0415 Introdução à Organização de Computadores

Aula 8 : Microcontrolador 8051 –Parte 1

Profa. Luiza Maria Romeiro Codá

MICROCONTROLADOR 8051

Introdução, características, ligação de memória externa e instruções

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

Interfaces :

O microcontrolador podem ter as mais diversas interfaces:

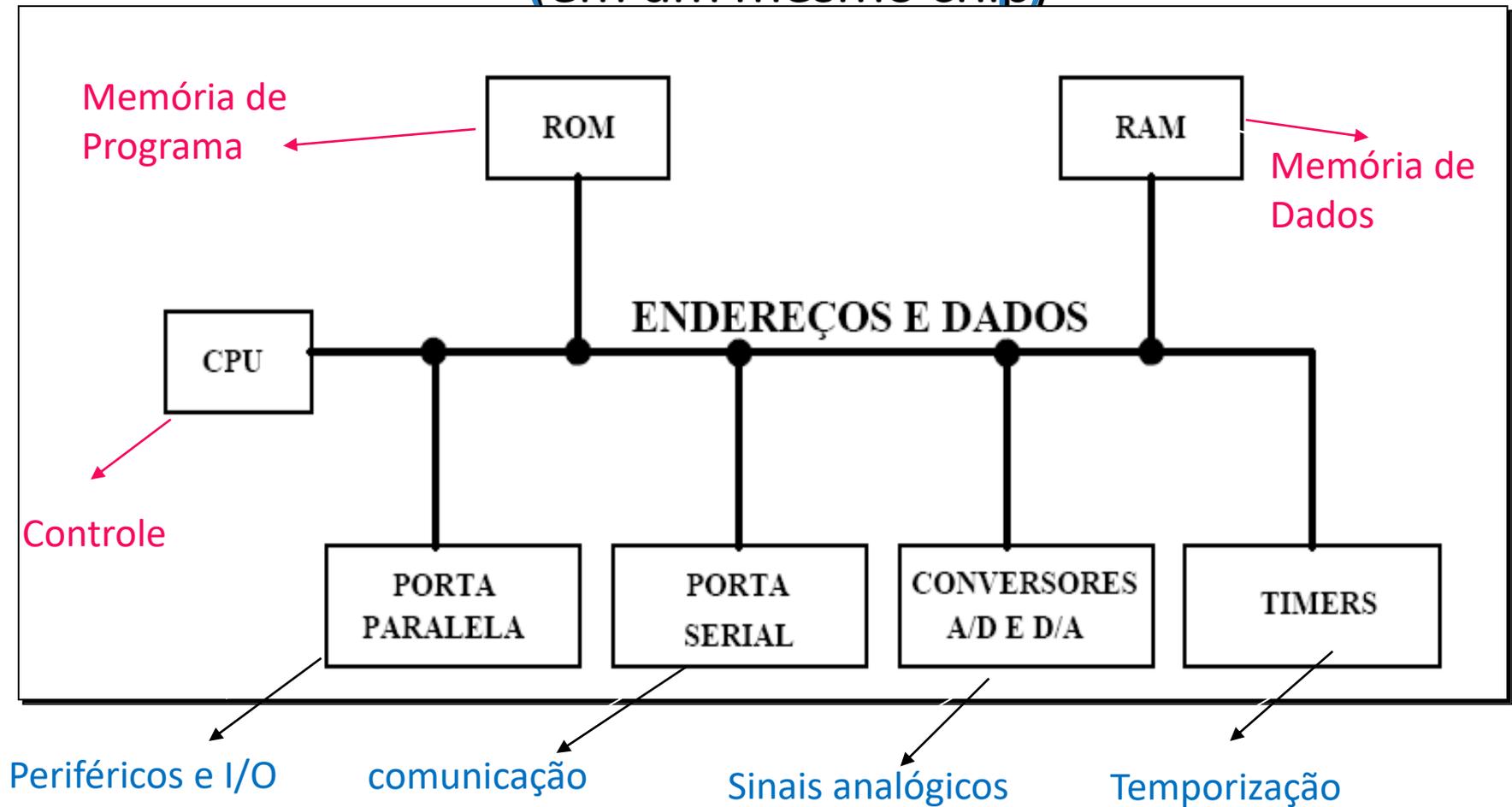
- Contador / Temporizador;
- Conversor AD / DA;
- Portas de I/O Paralelas (Entrada e Saída);
- Interface Serial.

Microcontroladores

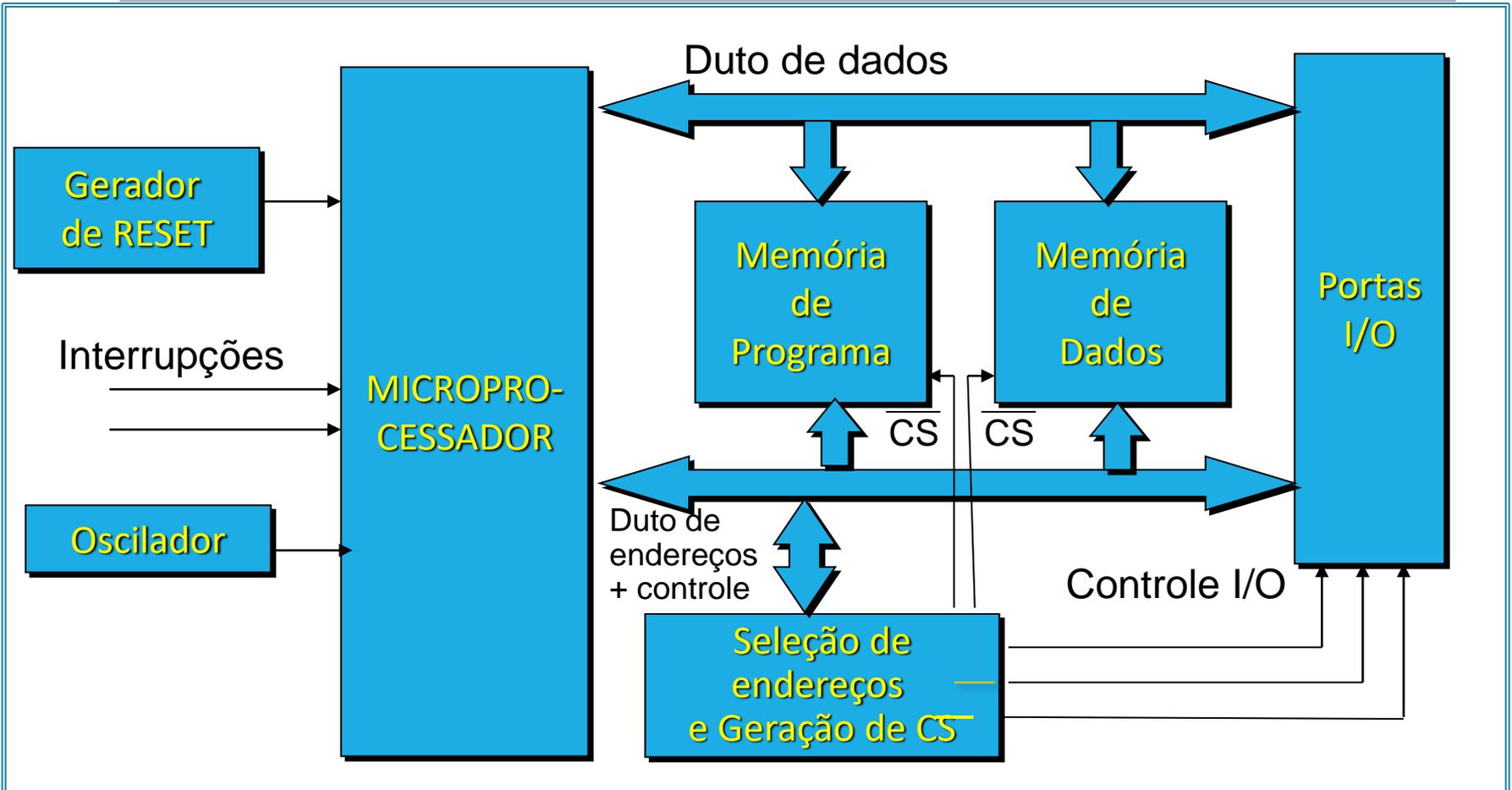
Expansão de Memória e Periféricos

O microcontrolador pode possibilitar a expansão de memórias e periféricos caso não seja suficiente a memória interna (ao CI)

Exemplo típico da arquitetura de um microcontrolador (em um mesmo chip)



Microcontroladores (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 registrador PC (que contém o endereço da instrução a ser executada) no endereço inicial (geralmente 0000H). O programa é executado a partir desse endereço;
- O microcontrolador irá buscar e executar as instruções na sequência 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) que é o registrador que contém o endereço da instrução que será executada, 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) - continuação

- 4 portas paralelas de 8 bits (32 linhas de I/O bidirecionais endereçadas individualmente)
- UART full duplex (*Universal Asynchronous Receiver 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 μ s a 400 ns).

Microcontrolador 80C51

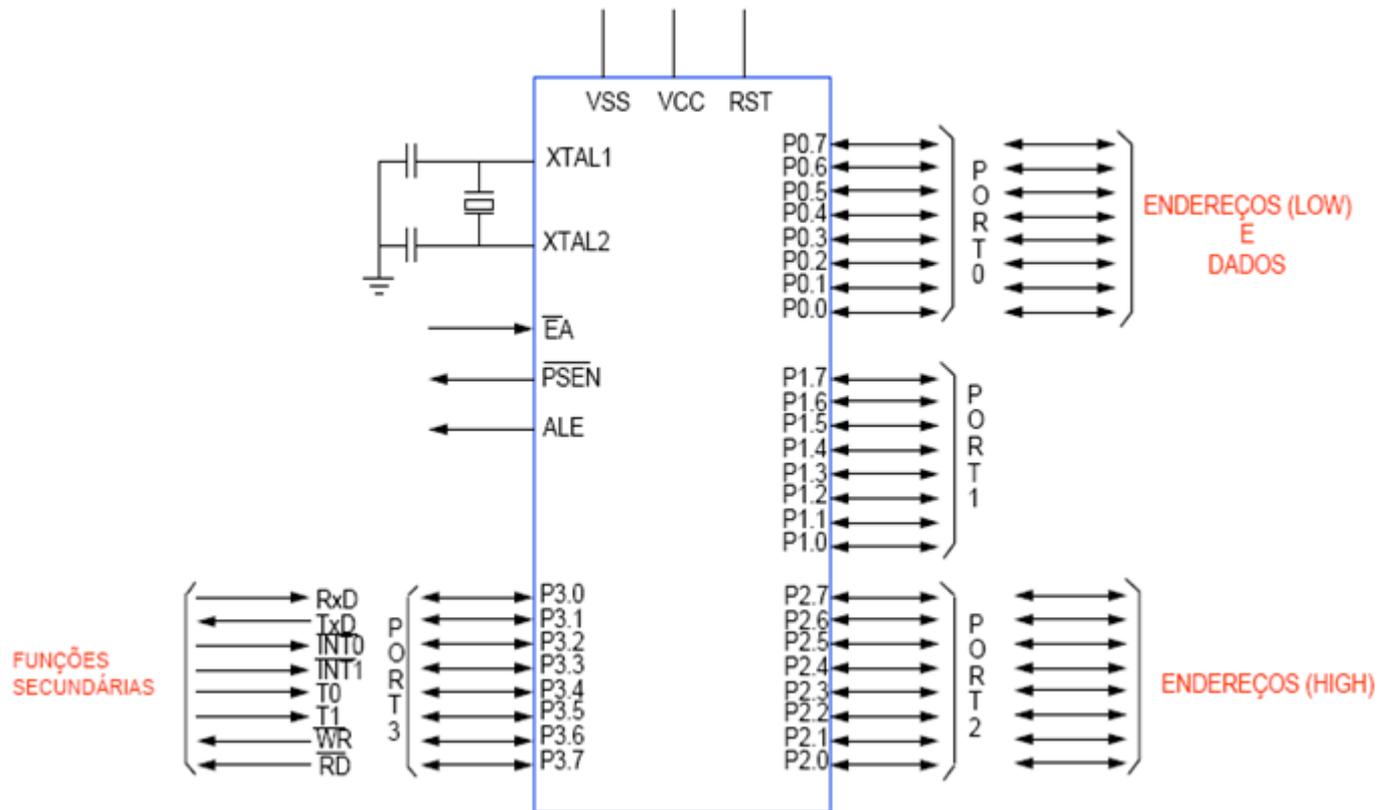
2. Arquitetura

- Arquitetura Von Neumann modificada
- 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

Configuração dos pinos

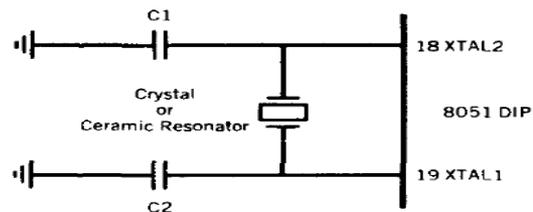


Ciclo de Máquina do 8051

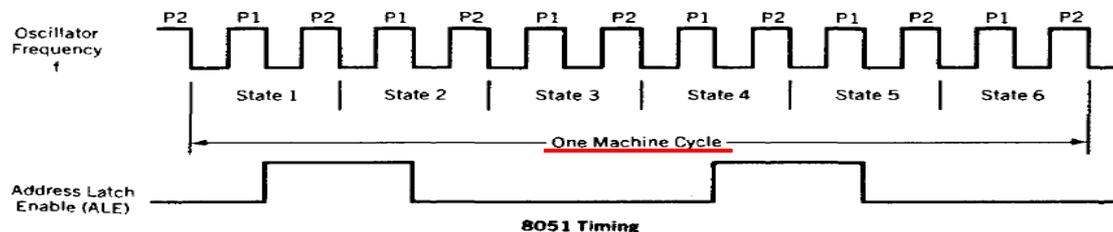
consiste em uma sequência de 6 estados, cada um formado por 2 períodos de clock

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

F_{cristal} : 12 a 30 MHz



Crystal or Ceramic Resonator Oscillator Circuit



8051 Timing

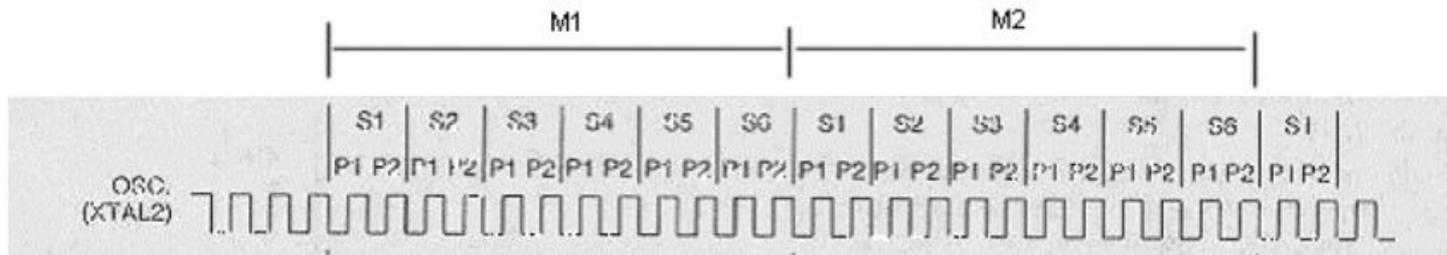
Ciclo de Máquina do 8051

Se o cristal é de 12 Mhz:

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

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:

- Armazenar em R1 o MSB
- 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 μ s	8.68 μ s
	FFh	FFh	130818	130.8 ms	141.95 ms
Maior Atraso	0	0	131843	131.8 ms	143.06 ms

DJNZ Decrementa o dado presente no registrador ou o dado na posição de memória diretamente endereçada e desvia se o resultado da operação for diferente de zero, sem afetar quaisquer flags.

Ciclo de Máquina do 8051

Instruções da família MCS-51 duram 12 ou 24 T_{clk} (1 ou 2 ciclos de máquina (CM)

Com exceção das instruções de multiplicação e divisão, MUL AB e DIV AB, respectivamente, as quais duram 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
```

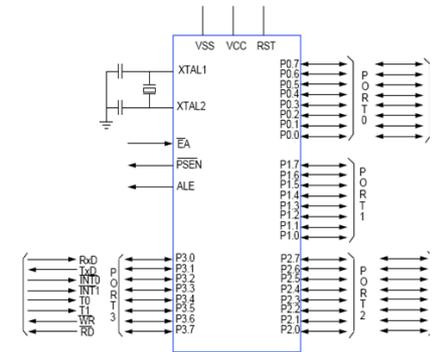
Ação de Reset Microcontrolador 80C51

O reset é ativado quando o pino de reset (pino 9) fica em nível alto por mais de dois ciclos de máquina (24 ciclos de clock); essa ação é realizada quando energiza-se o microcontrolador:

- O reset zera os registradores A, B, PSW, DPTR, PC e os registradores dos temporizadores/contadores;
- O registrador SP (stack pointer) é carregado com o valor 07h e o banco 0 de registradores é ativado;
- Os ports são inicializados com o valor FFh, para ficarem configurados como entrada;
- O registrador SCON é zerado e o registrador SBUF possui valor indeterminado;

Ligação de Memória Externa (EEPROM e RAM)

Microcontrolador 80C51



PINOS IMPORTANTES PARA INTERFACE COM MEMÓRIAS EXTERNAS

\overline{RD} : leitura na memória de dados externa

\overline{WR} : escrita na memória de dados externa

\overline{PSEN} : leitura na memória de programa externa

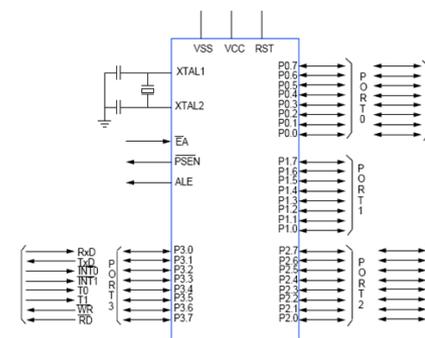
P0 : multiplexado com endereços (A0-A7) e dados (D0-D7)

P2 : endereços A8-A15

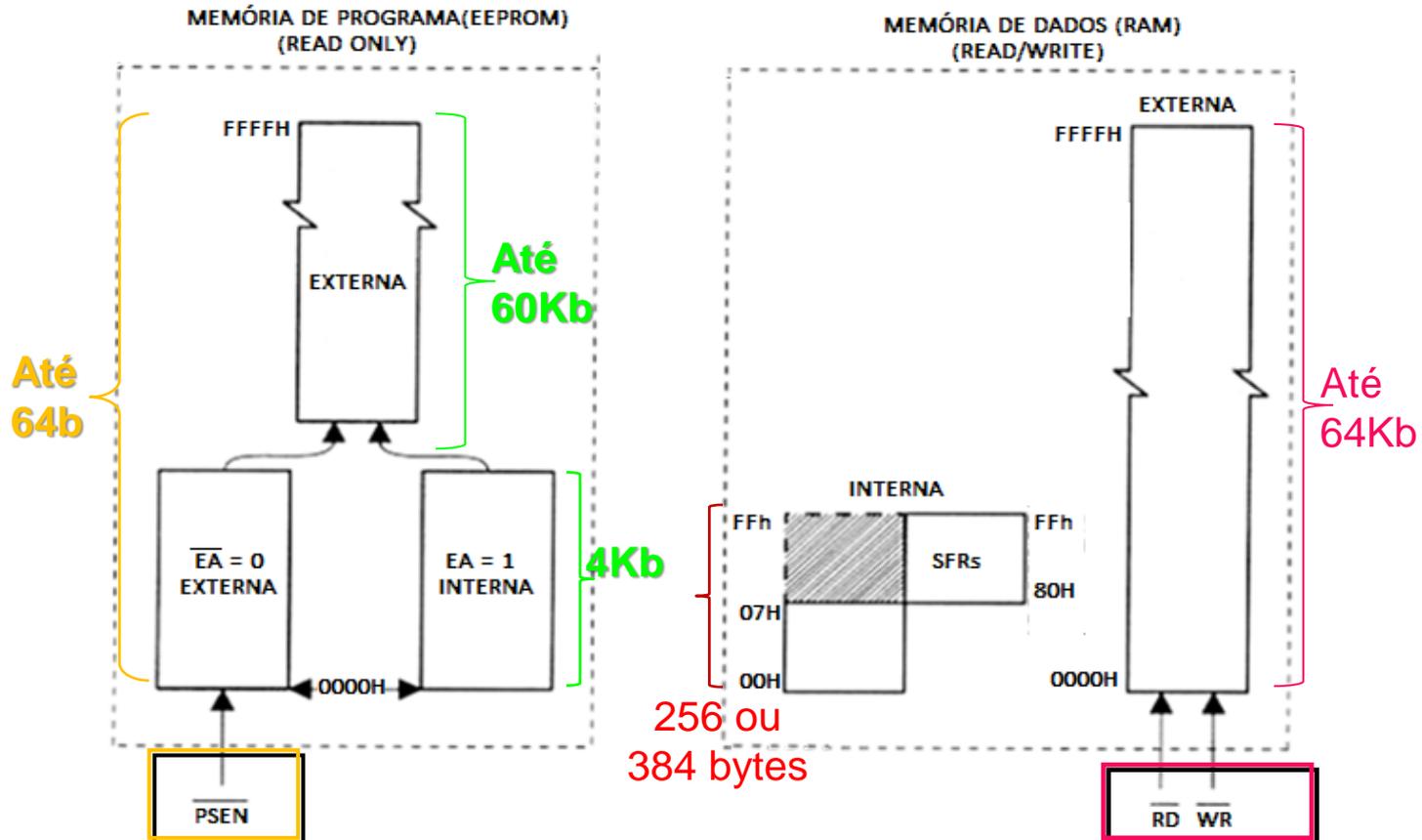
ALE: *Address Latch Enable*. Sinal para demultiplexar P0

\overline{EA} : *External Access Enable*. Especifica o uso de memória de programa externa ou interna

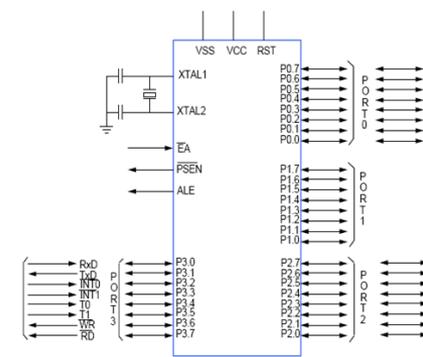
Microcontrolador 80C51



Organização das Memórias na família MCS-51



Microcontrolador 80C51



Organização das Memórias 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 $\overline{EA} = 0$ \Rightarrow até 64 KB de programa externo
- se $\overline{EA} = 1$ \Rightarrow 4 KB de ROM interna e até 60 KB de memória (EEPROM) de programa externa

Microcontrolador 80C51

Organização das Memórias 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 $\overline{EA} = 0 \rightarrow$ toda a memória de programa é externa : 0000H a FFFFH

Microcontrolador 80C51

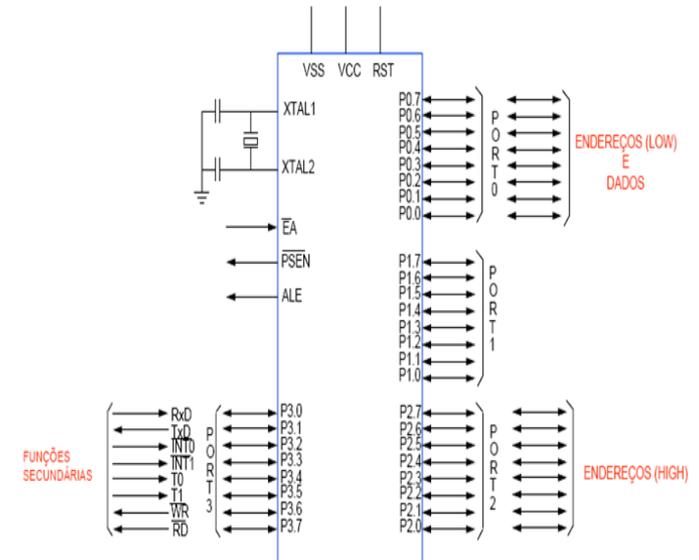
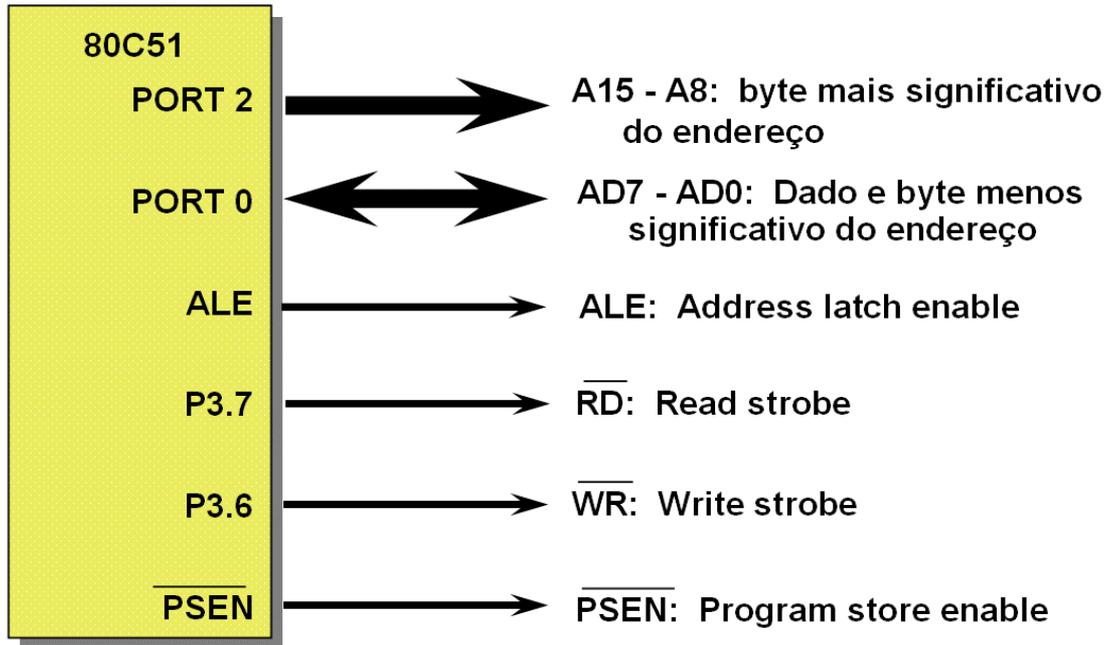
Organização das Memórias na família MCS-51

MEMÓRIA DE DADOS

- **Externa (DADOS)** → expansão com até 64 K RAM
- Pode-se usar as memórias RAM interna e externa simultaneamente
- As **instruções** de transferência de dados são **diferentes** para a memória externa e interna

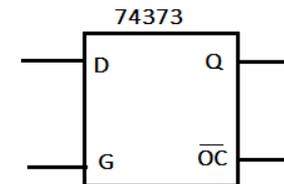
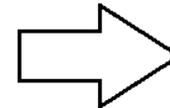
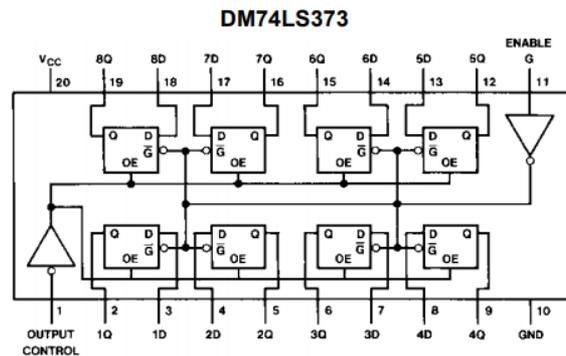
Microcontrolador 80C51

Mapeamento de Memória Externa



Microcontrolador 80C51

Informações sobre o 74373



Function Tables

DM74LS373

Output Control	Enable G	D	Output
L	H	H	H
L	H	L	L
L	L	X	Q ₀
H	X	X	Z

H = HIGH Level (Steady State) L = LOW Level (Steady State) X = Don't Care

↑ = Transition from LOW-to-HIGH level Z = High Impedance State

Q₀ = The level of the output before steady-state input conditions were established.

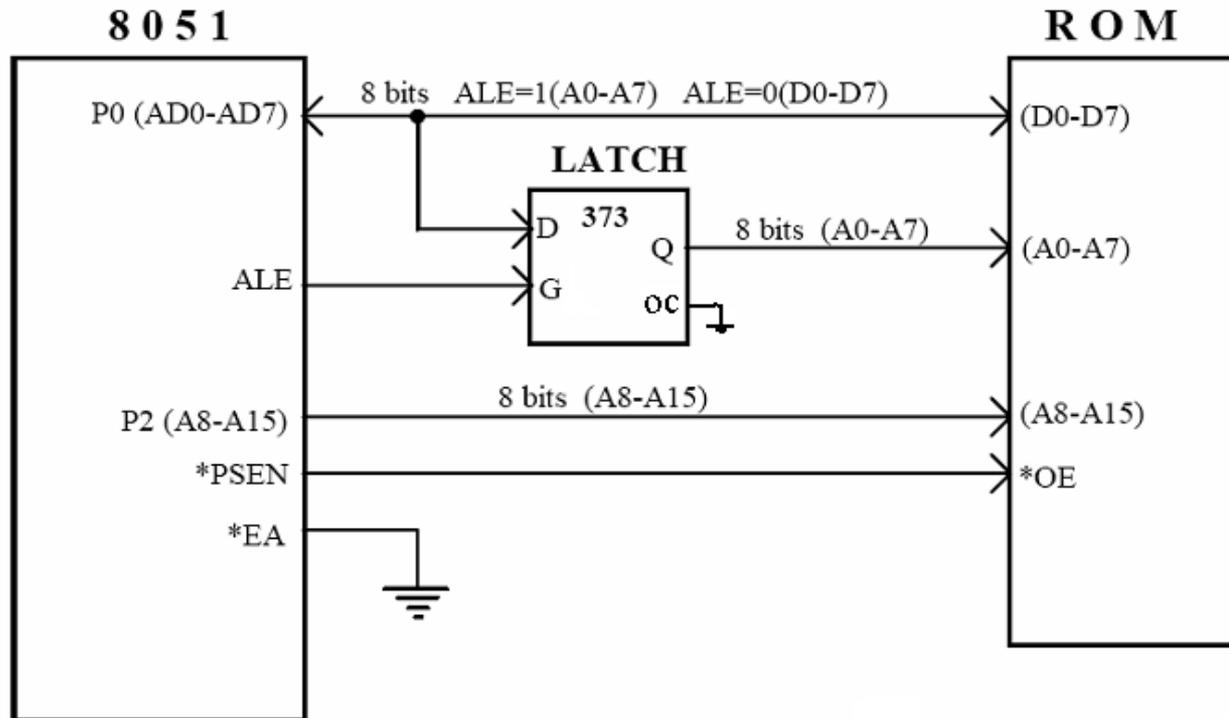
Para leitura:

$\overline{OE} = '0'$

E = '1'

Mapeamento da Memória Externa de Programa

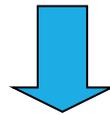
Mapeamento completo (64 Kbytes)



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 de programa (EEPROM)

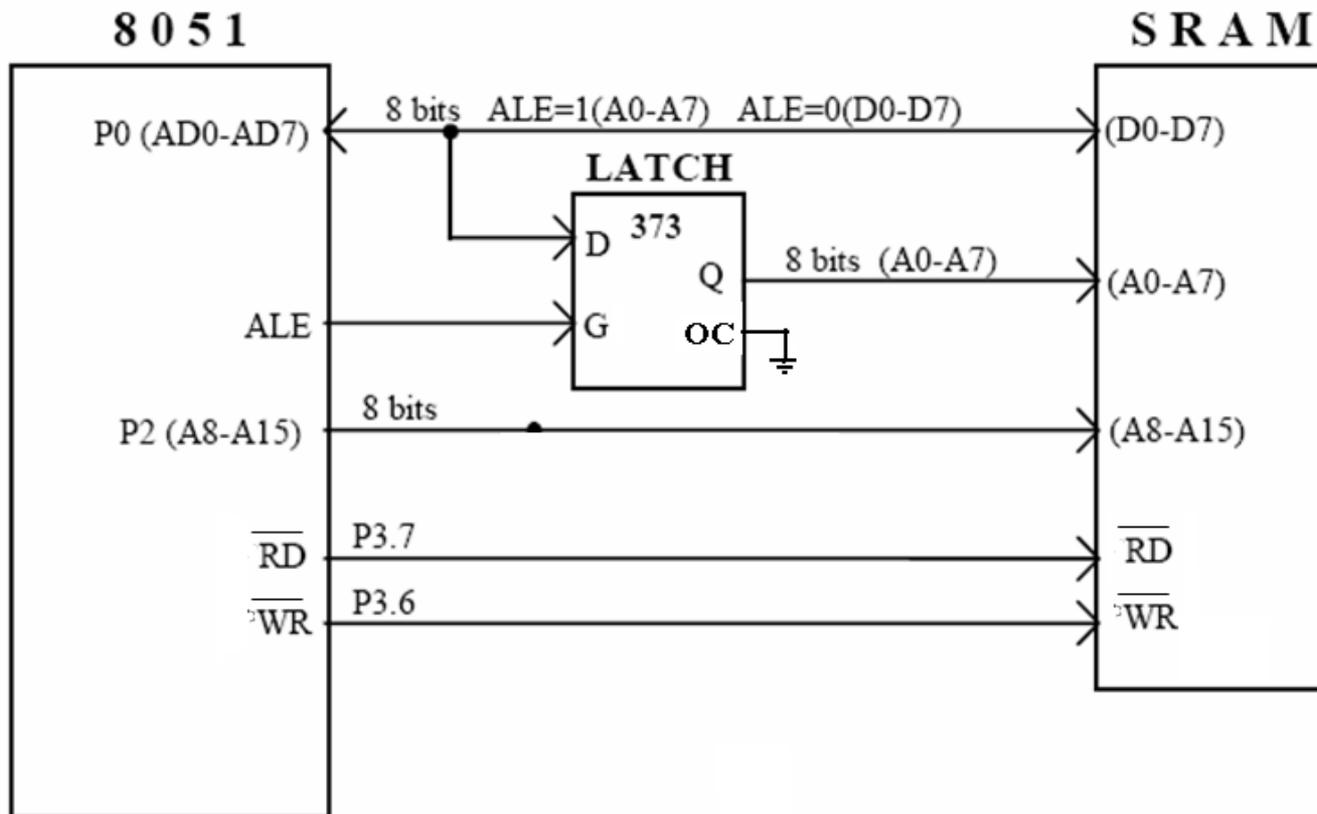
- **Porta P2** → 8 bits mais significativos do endereço
- **Porta P0** → 8 bits menos significativos do endereço multiplexado com os dados



1. pino ALE (address latch enable) quando em nível lógico 1, indica a presença de endereço no duto multiplexado (porta P0);
2. latch externo (ex. 74373) é comandado pelo ALE :
se ALE = 1 bits A0-A7 são armazenados na saída do 74373 (OC em "0")
3. pino 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, o qual é o duto D7-D0.

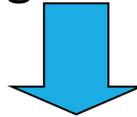
Mapeamento da Memória Externa de Dados (RAM)

Mapeamento completo (64 Kbytes)



Memória de Dados externa (RAM)

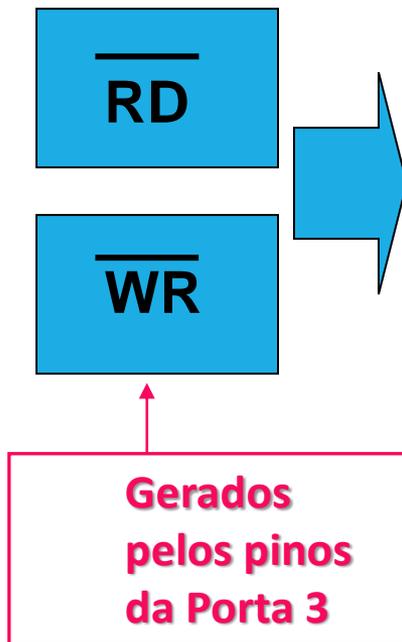
- Porta 2 → 8 bits mais significativos do endereço
- Porta 0 → 8 bits menos significativos do endereço + dados



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

Memória de dados externa

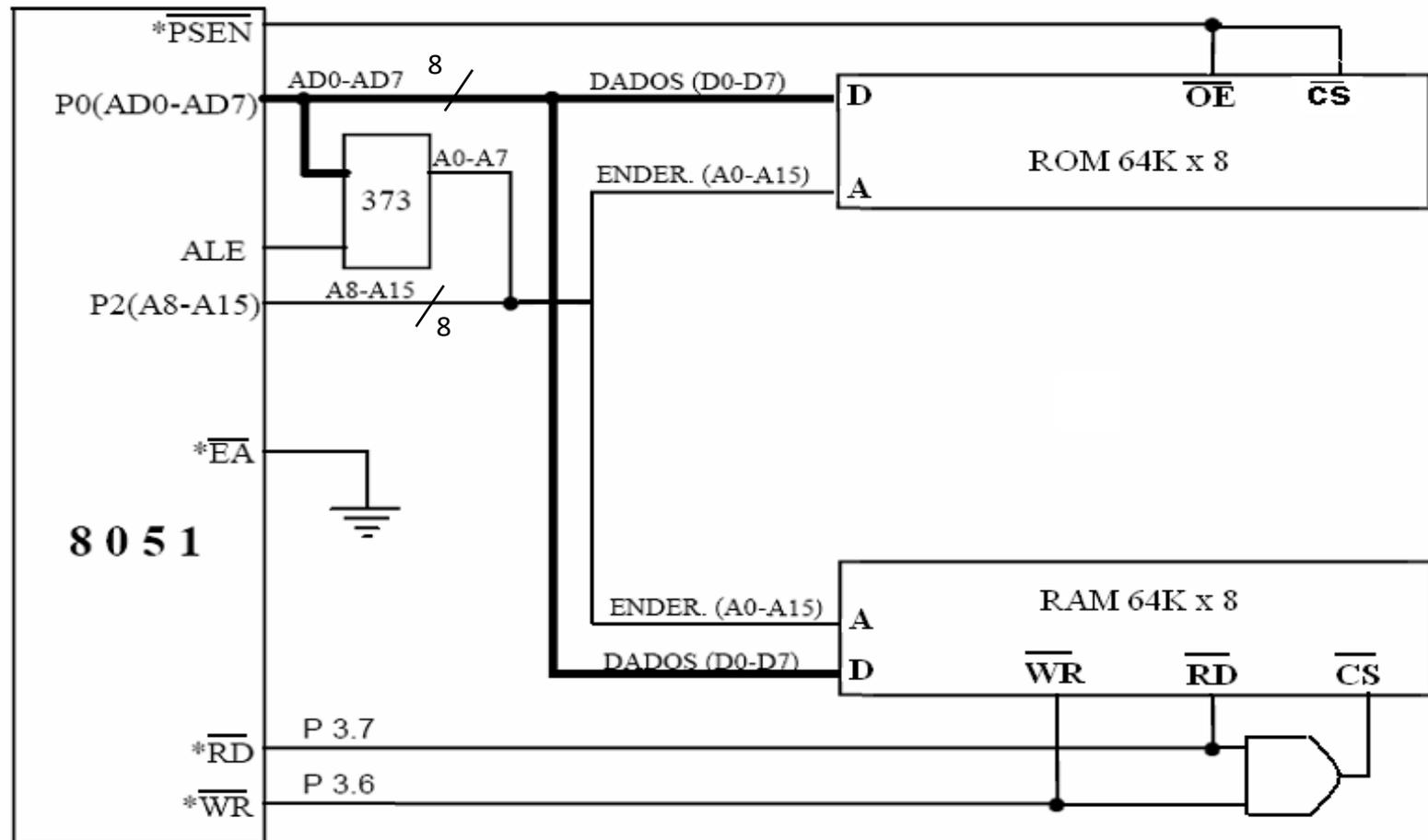
Portas I/O para expansão da memória



Ativos só se a CPU está executando instrução de leitura e escrita na memória de dados (RAM) externa

- Leitura da RAM $\Rightarrow \overline{RD} = 0 \quad \overline{WR} = 1$
- Escrita na RAM $\Rightarrow \overline{RD} = 1 \quad \overline{WR} = 0$
- Quando Leitura em **EEPROM externa**:
PSEN = 0 $\Rightarrow \overline{RD} = 1 \quad \overline{WR} = 1$

Mapeamento da Memória Externa de Dados e Programa



Microcontrolador 80C51

Ponteiros : registradores que contém endereço

- **PC** (*Program Counter*) : ponteiro de **16 bits** para área de programa (EEPROM);
- **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 (ponteiro incrementável);
- **R0** : ponteiro de **8 bits** para RAM interna;
- **R1** : ponteiro de **8 bits** para RAM interna.

Microcontrolador 80C51

Registadores 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 R0 – R7, mapeados em RAM interna
 - ❖ apenas um banco pode ser selecionado por vez

Existem outros registradores para a programação dos timers, do controlador da serial e de interrupção. Esses registradores são mapeados na RAM dedicada aos SFR.

Microcontrolador 80C51

Tipos de Instruções

As instruções do 8051 podem ser classificadas em 5 tipos diferentes:

- **Transferência de dados;**
Ex: MOV A, Rn
- **Aritméticas;**
Ex: SUBB A, #dato8
- **Lógicas;**
Ex: ANL A, Rn
- **Booleanas;**
Ex: CPL C
- **Desvio;**
Ex: LJMP addr

Microcontrolador 80C51

Modo de acesso às Memórias

Modo de acesso às memórias ou de endereçamento das memórias, é a forma como a instrução, através de registradores, lê ou escreve na memória ou dispositivos de I/O. Para acessar um dado, as instruções do microcontrolador 8051 contam com cinco tipos de endereçamentos:

- **Imediato**
- **Direto**
- **Indireto**
- **por Registradores ou por Registrador Específico**
- **Indexado.**

Modo de acesso às memórias

Microcontrolador 80C51

- **Endereçamento Imediato:** quando uma constante é apresentada na instrução após #
- **Endereçamento Direto:** o endereço do operando é especificado por um campo de 8bits na instrução. Somente RAM de dados interna e a região SFR pode ser diretamente endereçadas.
- **Endereçamento Indireto:** a instrução especifica um registrador que contém o endereço do operando.
Ambas externa e interna RAM's podem ser indiretamente acessadas.
Se o endereço possui 8bits, seu valor é armazenado em R0 ou R1.
Se o endereço for de 16bits, seu valor é armazenado no registrador DPTR.
- **Endereçamento por Registradores:** instruções que acessam registradores de R0 a R7 dos bancos de registradores ou algumas instruções específicas a certos registradores. Tais instruções são eficientes por eliminarem um byte de endereço. O próprio opcode é capaz de realizar tal tarefa.
- **Endereçamento Indexado:** somente a Memória de Programas, cuja única operação é leitura, pode ser acessada por esse modo. É usado para fazer leituras em tabelas na Memória de Programas. Um registrador de 16bits (DPTR) aponta para o início da tabela enquanto o Acumulador é ajustado para a n-ésima posição da mesma. O endereço de uma entrada para a tabela é formado pela soma entre o Acumulador

Modo de endereçamento (acesso) da memória RAM interna 80C51

Endereçamento Imediato: quando uma constante é apresentada na instrução após #
Ex: MOV A,#0FH

1. Endereçamento Imediato

- Opera sobre o dado localizado na própria instrução

• Identificado através do sinal #

• Exemplo:

ADD A,#30H

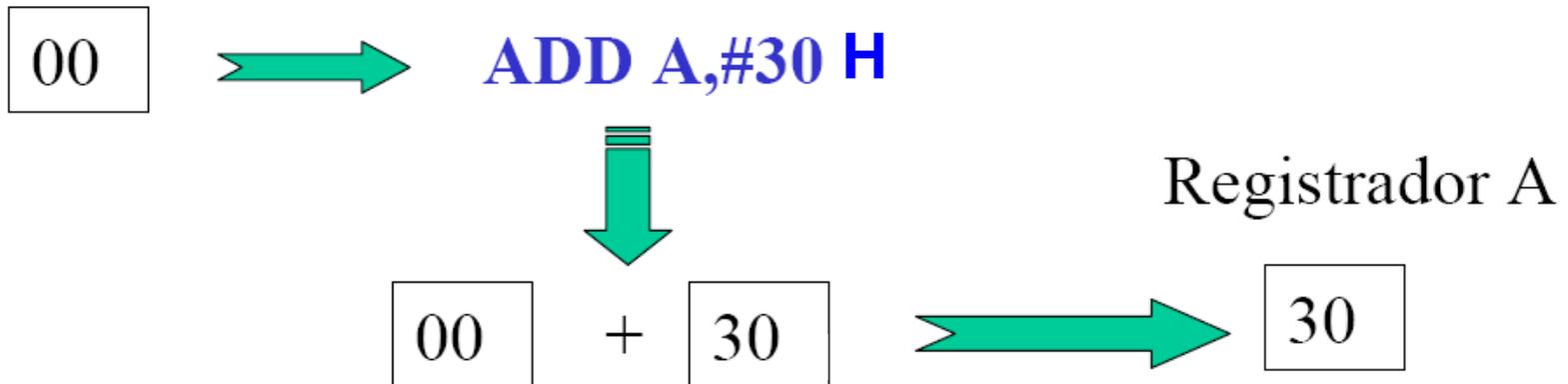
O dado 30 é somado ao Registrador A

Modo de endereçamento (acesso) da memória RAM interna 80C51

1. Endereçamento Imediato

ADD A,#30 H

Registrador A



Modo de endereçamento (acesso) da memória RAM interna 80C51

• **Endereçamento Direto:** o endereço do operando é especificado por um campo de 8bits na instrução. Somente RAM de dados interna e a região SFR pode ser diretamente endereçadas. Ex : MOV A, 0FH

2. Endereçamento Direto

- Opera sobre o dado cujo endereço está na instrução

• Exemplo:

ADD A,30H

O dado armazenado no endereço 30H é somado ao Registrador A

Modo de endereçamento (acesso) da memória RAM interna 80C51

2. Endereçamento Direto

ADD A,30H

Registrador A

00



ADD A,30 H



Conteúdo do
Endereço 30

00

+

20



Registrador A

20

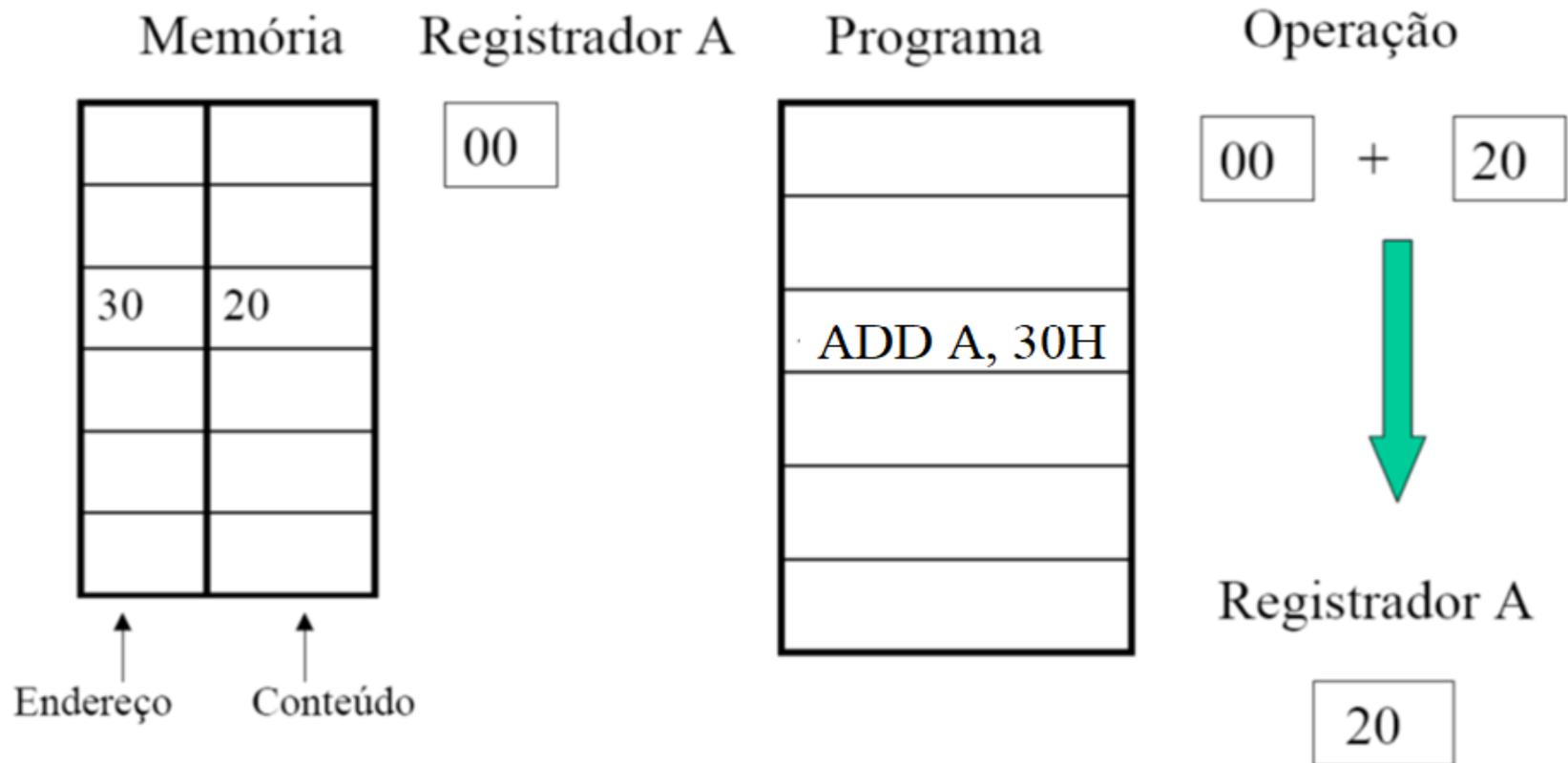
20



Modo de endereçamento (acesso) da memória RAM interna 80C51

2. Endereçamento Direto

`ADD A,30H`



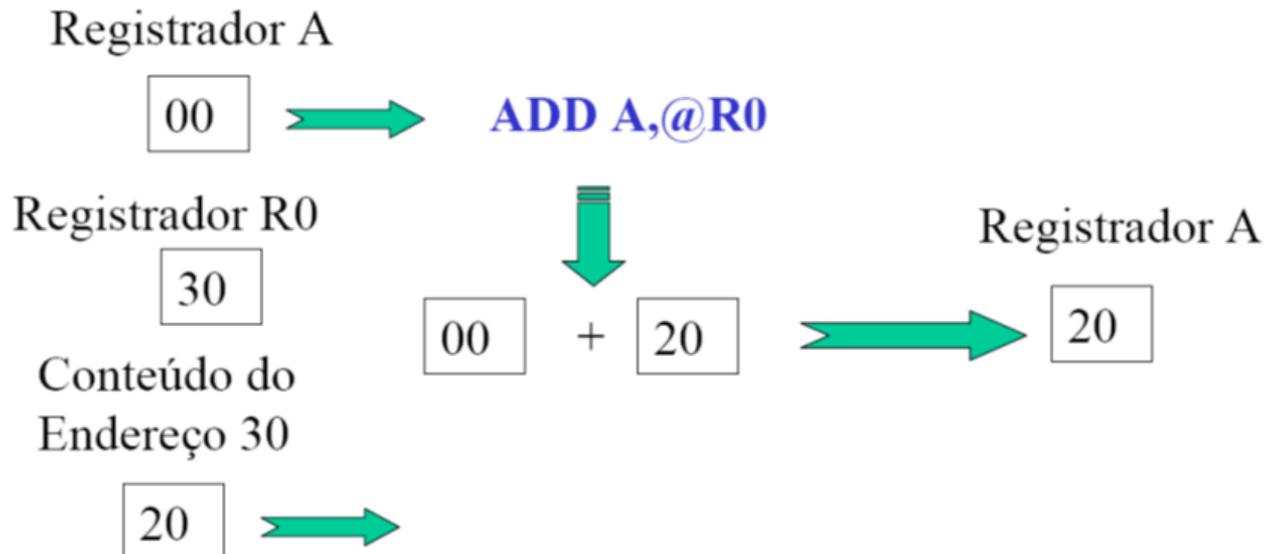
Modo de endereçamento (acesso) da memória RAM interna 80C51

Endereçamento Indireto: a instrução especifica um registrador que contém o endereço do operando. Ambas externa e interna RAM's podem ser indiretamente acessadas.

Ex : MOV @R1,A

3. Endereçamento Indireto

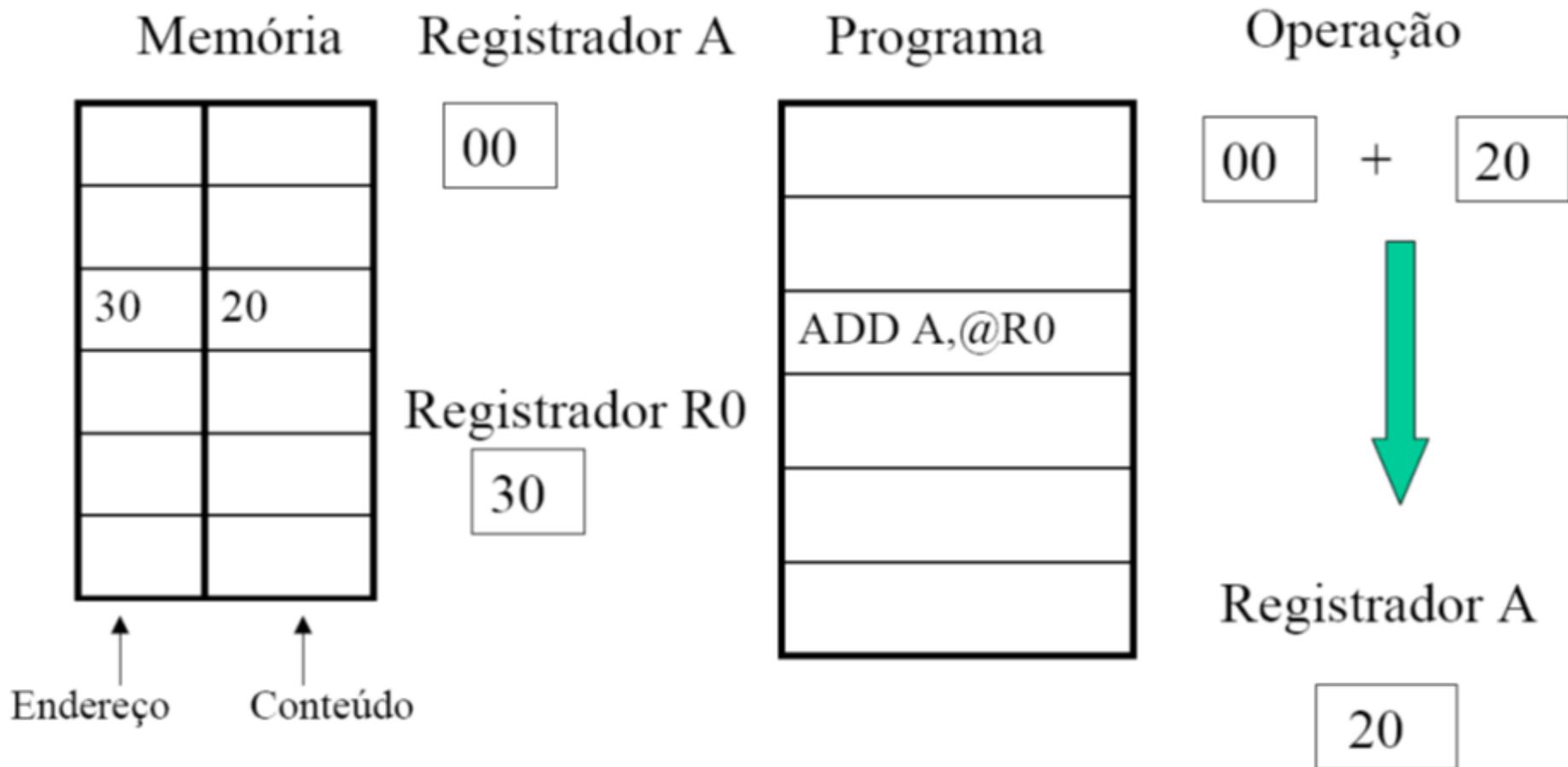
ADD A,@R0



Modo de endereçamento (acesso) da memória RAM interna 80C51

3. Endereçamento Indireto

`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 interna)

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

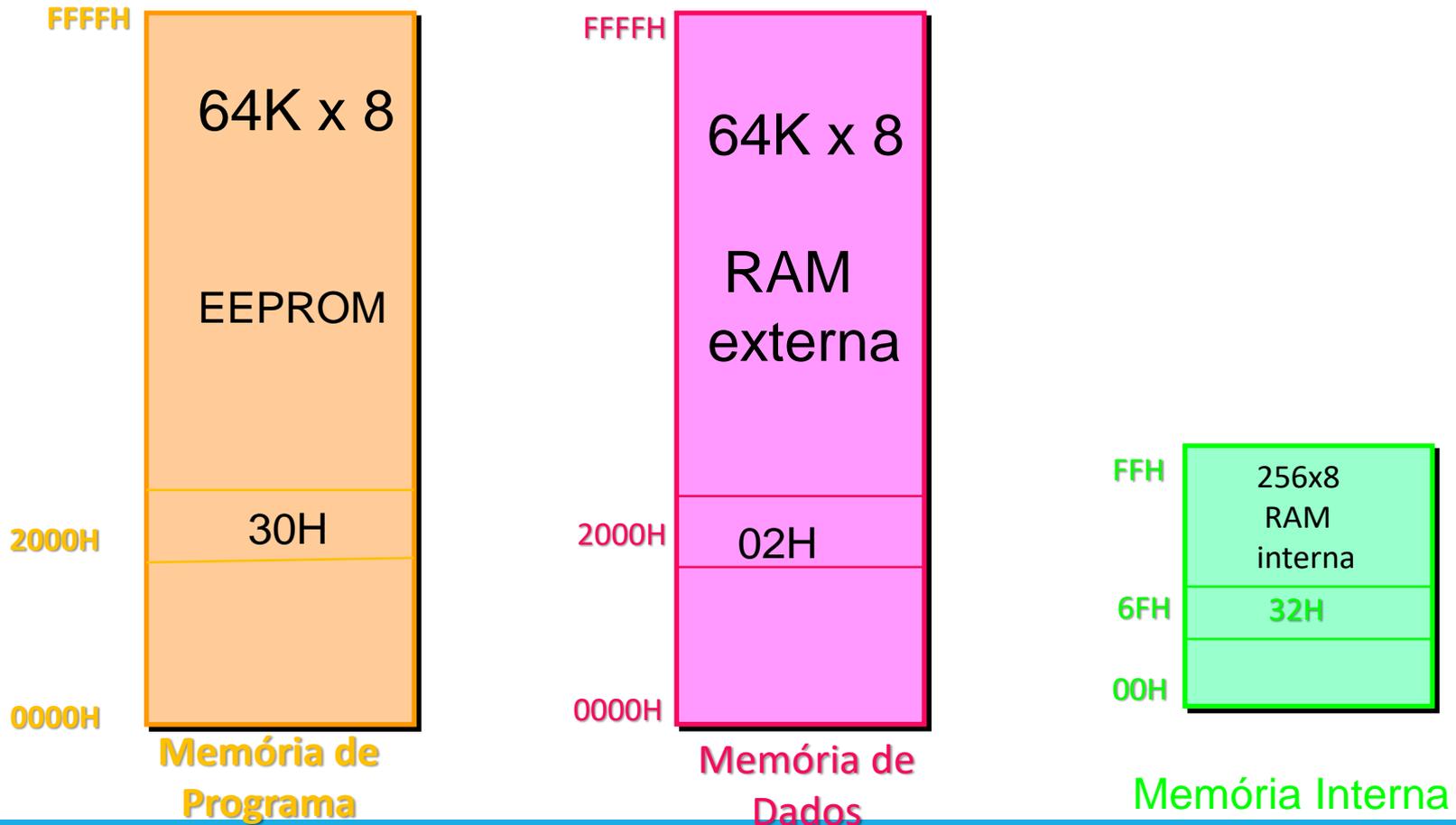
ORG 2000H

DB 30H ; diretiva que define dado, no caso como 30H

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.



Exemplo de Programa em Assembly pra o 8051

memória EEPROM

Endereço em hexadecimal	Conteúdo em hexadecimal	Instrução
0000	90	MOV DPTR, #2000H
0001	20	
0002	00	
0003	E4	CLR A
0004	93	MOVC A,@A+DPTR
0005	F9	MOV R1,A
0006	29	ADD A,R1
0007	F5	MOV 6Fh,A
0008	6F	
0009	80	SJMP \$
000A	FE	
• • •	• • •	• • •
2000	30	

Conjunto de instruções e OPCODES do 8051

(C) 2005-7 www.efton.sk		LOWER NIBBLE															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
UPPER NIBBLE	0	NOP	AJMP page+00XX	LJMP 0000-FFFF	RR A	INC A	INC direct	INC @R0	INC @R1	INC R0	INC R1	INC R2	INC R3	INC R4	INC R5	INC R6	INC R7
	1	JBC bit, -128+127	ACALL page+00XX	LCALL 0000-FFFF	RRC A	DEC A	DEC direct	DEC @R0	DEC @R1	DEC R0	DEC R1	DEC R2	DEC R3	DEC R4	DEC R5	DEC R6	DEC R7
	2	JB bit, -128+127	AJMP page+01XX	RET	RL A	ADD A, #data	ADD A, direct	ADD A, @R0	ADD A, @R1	ADD A, R0	ADD A, R1	ADD A, R2	ADD A, R3	ADD A, R4	ADD A, R5	ADD A, R6	ADD A, R7
	3	JNB bit, -128+127	ACALL page+01XX	RETI	RLC A	ADDC A, #data	ADDC A, direct	ADDC A, @R0	ADDC A, @R1	ADDC A, R0	ADDC A, R1	ADDC A, R2	ADDC A, R3	ADDC A, R4	ADDC A, R5	ADDC A, R6	ADDC A, R7
	4	JC -128+127	AJMP page+02XX	ORL direct, A	ORL direct, #data	ORL A, #data	ORL A, direct	ORL A, @R0	ORL A, @R1	ORL A, R0	ORL A, R1	ORL A, R2	ORL A, R3	ORL A, R4	ORL A, R5	ORL A, R6	ORL A, R7
	5	JNC -128+127	ACALL page+02XX	ANL direct, A	ANL direct, #data	ANL A, #data	ANL A, direct	ANL A, @R0	ANL A, @R1	ANL A, R0	ANL A, R1	ANL A, R2	ANL A, R3	ANL A, R4	ANL A, R5	ANL A, R6	ANL A, R7
	6	JZ -128+127	AJMP page+03XX	XRL direct, A	XRL direct, #data	XRL A, #data	XRL A, direct	XRL A, @R0	XRL A, @R1	XRL A, R0	XRL A, R1	XRL A, R2	XRL A, R3	XRL A, R4	XRL A, R5	XRL A, R6	XRL A, R7
	7	JNZ -128+127	ACALL page+03XX	ORL C, bit	JMP @A+DPTR	MOV A, #data	MOV direct, #data	MOV @R0, #data	MOV @R1, #data	MOV R0, #data	MOV R1, #data	MOV R2, #data	MOV R3, #data	MOV R4, #data	MOV R5, #data	MOV R6, #data	MOV R7, #data
	8	SJMP -128+127	AJMP page+04XX	ANL C, bit	MOVC A, @A+PC	DIV AB	MOV direct1, direct2	MOV direct, @R0	MOV direct, @R1	MOV direct, R0	MOV direct, R1	MOV direct, R2	MOV direct, R3	MOV direct, R4	MOV direct, R5	MOV direct, R6	MOV direct, R7
	9	MOV DPTR, #data16	ACALL page+04XX	MOV bit, C	MOVC A, @A+DPTR	SUBB A, #data	SUBB A, direct	SUBB @R0, @R0	SUBB @R1, @R1	SUBB A, R0	SUBB A, R1	SUBB A, R2	SUBB A, R3	SUBB A, R4	SUBB A, R5	SUBB A, R6	SUBB A, R7
	A	ORL C, /bit	AJMP page+05XX	MOV C, bit	INC DPTR	MUL AB	???	MOV @R0, direct	MOV @R1, direct	MOV R0, direct	MOV R1, direct	MOV R2, direct	MOV R3, direct	MOV R4, direct	MOV R5, direct	MOV R6, direct	MOV R7, direct
	B	ANL C, /bit	ACALL page+05XX	CPL bit	CPL C	CJNE A, #data, -128+127	CJNE A, direct, -128+127	CJNE @R0, #data, -128+127	CJNE @R1, #data, -128+127	CJNE R0, #data, -128+127	CJNE R1, #data, -128+127	CJNE R2, #data, -128+127	CJNE R3, #data, -128+127	CJNE R4, #data, -128+127	CJNE R5, #data, -128+127	CJNE R6, #data, -128+127	CJNE R7, #data, -128+127
	C	PUSH direct	AJMP page+06XX	CLR bit	CLR C	SWAP A	XCH A, direct	XCH A, @R0	XCH A, @R1	XCH A, R0	XCH A, R1	XCH A, R2	XCH A, R3	XCH A, R4	XCH A, R5	XCH A, R6	XCH A, R7
	D	POP direct	ACALL page+06XX	SETB bit	SETB C	DA A	DJNZ direct, -128+127	XCHD A, @R0	XCHD A, @R1	DJNZ R0, -128+127	DJNZ R1, -128+127	DJNZ R2, -128+127	DJNZ R3, -128+127	DJNZ R4, -128+127	DJNZ R5, -128+127	DJNZ R6, -128+127	DJNZ R7, -128+127
	E	MOVX A, @DPTR	AJMP page+07XX	MOVX A, @R0	MOVX A, @R1	CLR A	MOV A, direct	MOV A, @R0	MOV A, @R1	MOV A, R0	MOV A, R1	MOV A, R2	MOV A, R3	MOV A, R4	MOV A, R5	MOV A, R6	MOV A, R7
	F	MOVX @DPTR, A	ACALL page+07XX	MOVX @R0, A	MOVX @R1, A	CPL A	MOV direct, A	MOV @R0, A	MOV @R1, A	MOV R0, A	MOV R1, A	MOV R2, A	MOV R3, A	MOV R4, A	MOV R5, A	MOV R6, A	MOV R7, A

FIM