

# Aula 1

## SEL-614 MICROPROCESSADORES E APLICAÇÕES



**80C51**



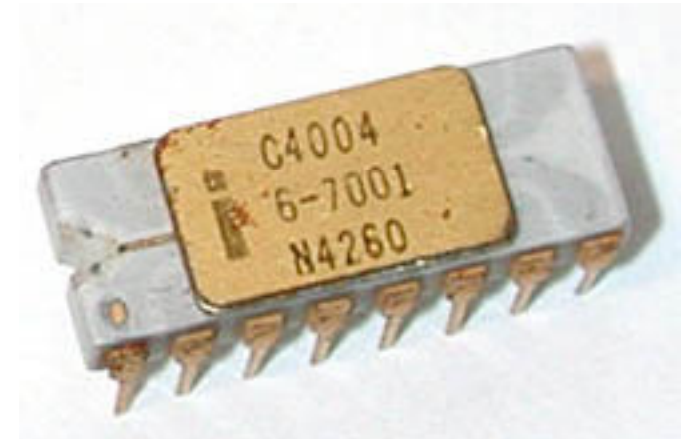
Prof: Adilson Gonzaga

# HISTÓRICO

## *Microprocessador*



- **Circuito integrado (“chip”) capaz de executar instruções.**



- **1971**

**Intel Corporation lançou no mercado o microprocessador 4004  
(3.000 transistores )**

# HISTÓRICO



## *Microprocessador*

**O microprocessador é um dispositivo lógico programável em um único chip de silício, concebido sob a tecnologia VLSI (circuito integrado em alta escala).**

**Age sob o controle de um programa armazenado em memória, executando operações aritméticas, lógica booleana, tomadas de decisão, além de entrada e saída de dados, permitindo a comunicação com outros dispositivos periféricos.**

## Microprocessador:

- parte principal de um microcomputador
- executa instruções

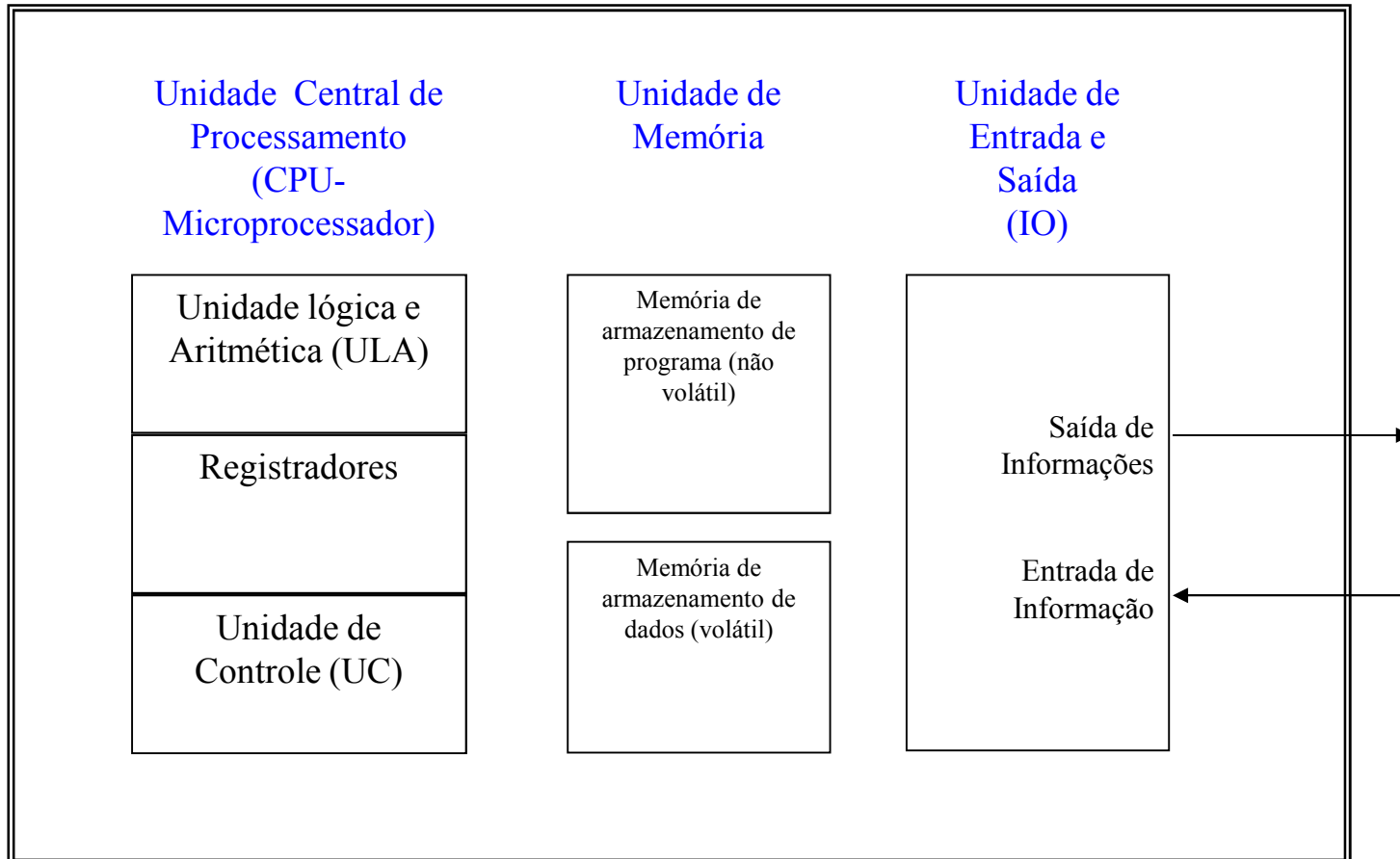
### Unidades básicas de um Microprocessador:

**Unidade Lógica Aritmética (ULA)** - responsável pela realização das operações lógicas e aritméticas.

• **Unidade de Controle (UC)** - responsável pela decodificação e execução das instruções, fornecendo os sinais de temporização adequados para as diversas partes do processador e do próprio computador.

• **Registradores** - armazenamento da Informação Binária (dados, endereços e instruções).

# ARQUITETURA DE MICROCOMPUTADORES



# MICROPROCESSADORES E MICROCONTROLADORES

**Microcomputador:** computador digital com velocidade e recursos limitados, e tipicamente é constituído por:

- unidade central de processamento – CPU;
- memória;
- circuitos de entrada e saída.

Suas aplicações são também limitadas quando comparadas às de um computador de maior porte.

# MICROPROCESSADORES E MICROCONTROLADORES

**Microprocessador:** geralmente implementado em um único componente.

- unidade central de processamento – CPU;
- Máquina seqüencial de uso geral, cujo comportamento no tempo é determinado por um **programa** externo colocado em memória.

**Associado a CI's periféricos, ele pode gerar:**

- microcomputadores e controles lógicos de uso específicos
- microcomputadores de uso geral

# MICROPROCESSADORES E MICROCONTROLADORES

1976 - 1º microcontrolador – Intel 8048

1980 – Intel 8051

**Microcontroladores:** possuem em um único componente:

- a unidade central de processamento
- memória (ROM e RAM);
- periféricos dedicados (serial, paralela, timer, etc...).

Os **microcontroladores** apresentam:

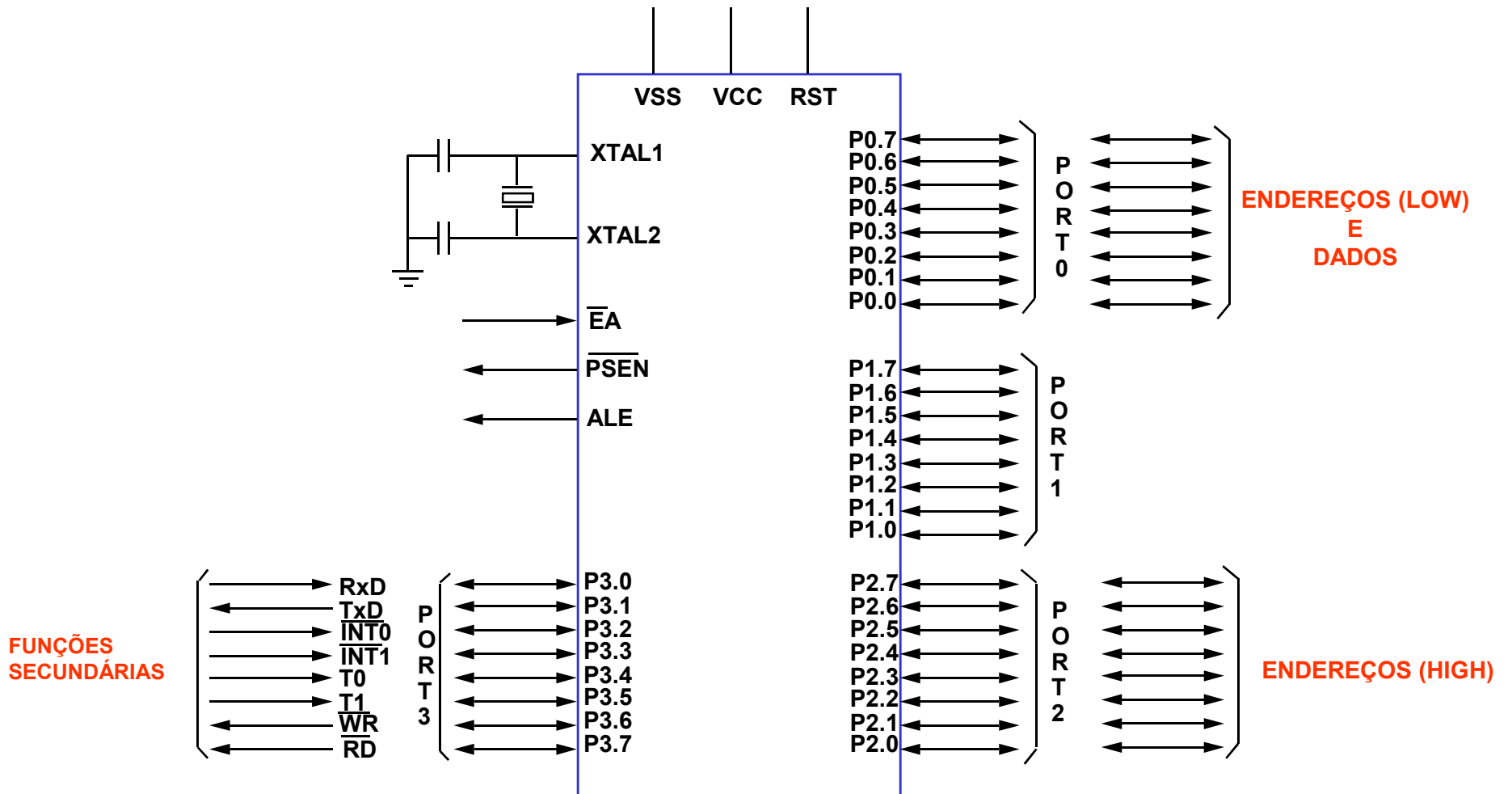
- menor desempenho que os microprocessadores,
- custo muito baixo (alguns dólares tipicamente),
- destinados a aplicações onde as dimensões, custo, tamanho e consumo do produto são muito importantes.



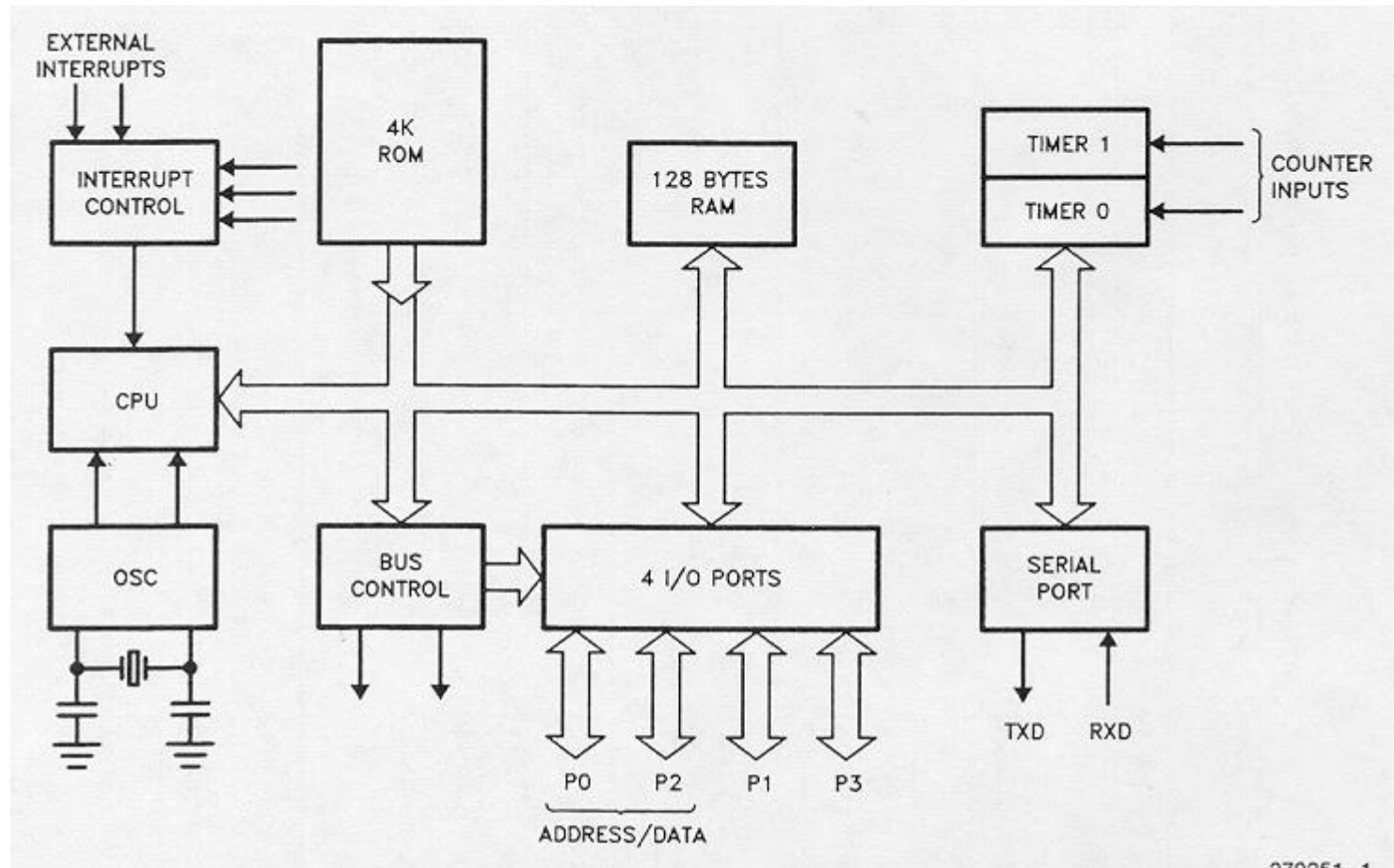
# Microcontrolador 8051

- ✚ O 8051 é membro da família MCS-51, e constitui o núcleo de todos os dispositivos MCS-51
  
- ✚ É um sistema de um *chip* único, que além do microprocessador de 8 bits pode conter:
  - . Memória de Programa e Memória de Dados
  - . Portas de I/O
  - . Comunicação Serial
  - . Contadores/ “Timers”
  - . Lógica para Controle de Interrupção
  - . etc ...

# Configuração dos pinos do 8051



# Diagrama em Blocos do 8051

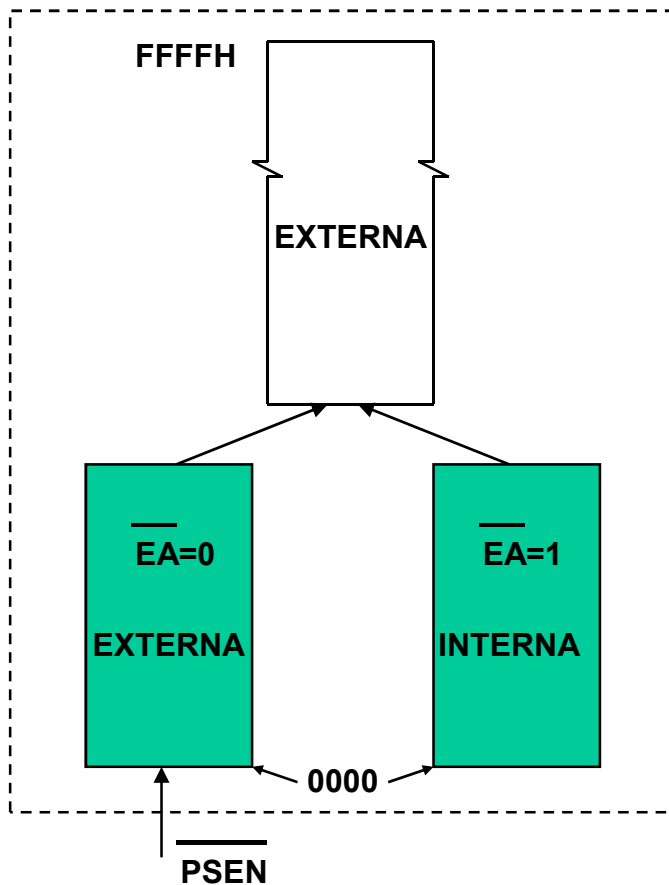


## Organização da memória da família MCS-51

- Memórias de dados e de programas separadas.

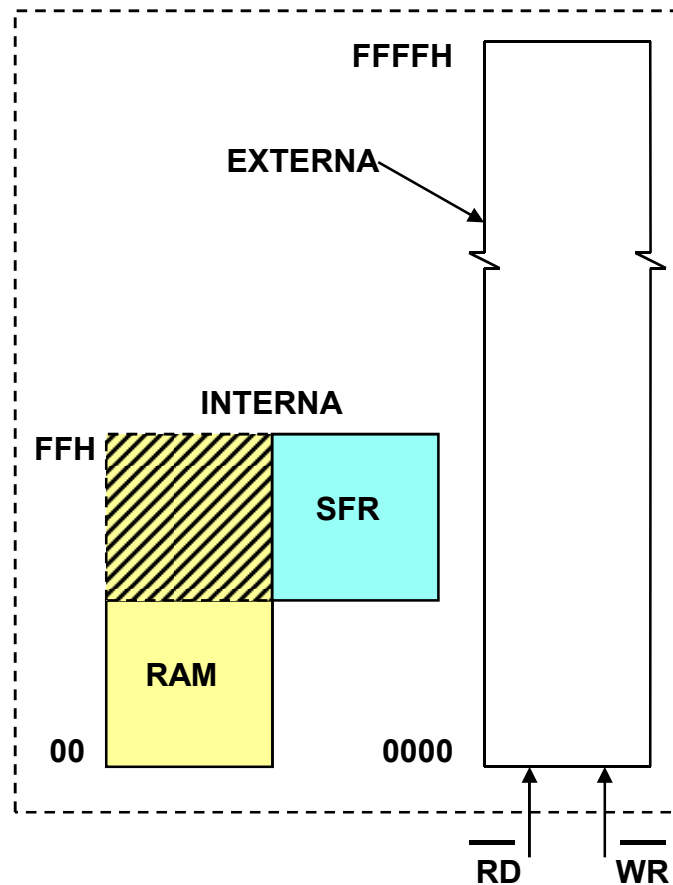
### MEMÓRIA DE PROGRAMA

(Read only)



### MEMÓRIA DE DADOS

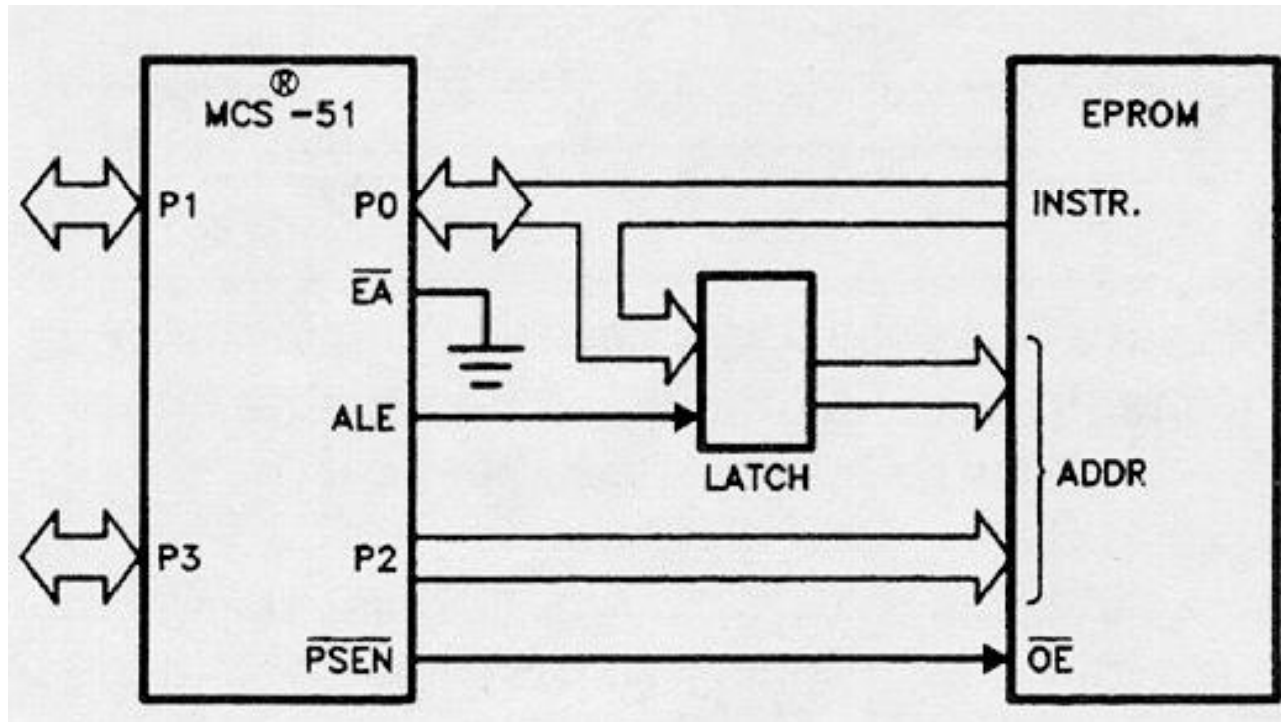
(Read/Write)



## Memória de Programa

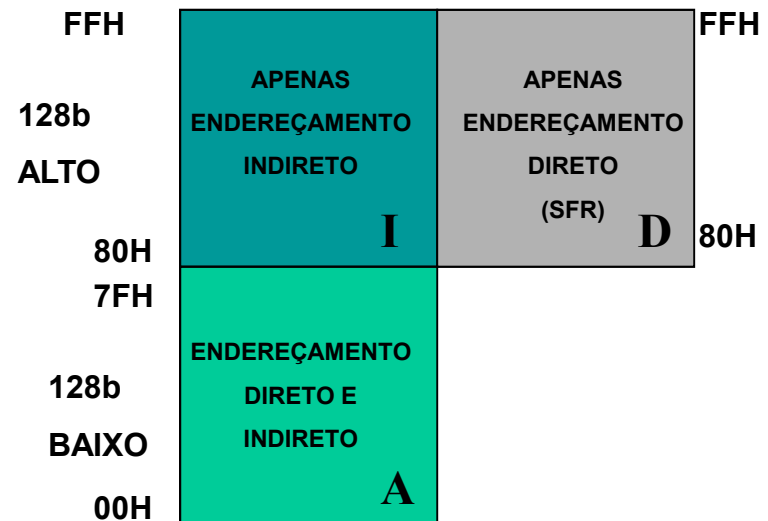
- Mapeamento de memória de programa externa

- Mapeamento completo ( 64 Kb externo )



## Memória de Dados Interna (RAM Interna)

- O endereçamento é feito com 8 bits
- Chips com 128 bytes de RAM não possuem a área I (Apenas Endereçamento Indireto)



**Área A:** 128 bytes inferiores (00h a 7Fh) , acessíveis por endereçamento direto e indireto (existe em toda a família MCS-51)

**Área D:** SFR (special function register) acessível por endereçamento direto (80h a FFh) também existe em todos os membros da família MCS-51

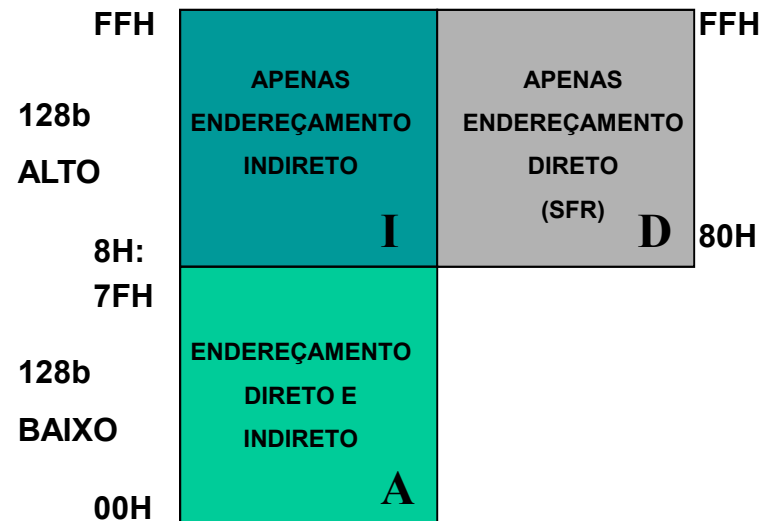
**Área I:** 128 bytes superiores (80h a FFh acessível somente por endereçamento indireto, só existe nos chips de 256 bytes de RAM interna (8032,8052,...)).

## Memória de Dados Interna (RAM Interna)

### Exemplos:

a. Escrever 0AAh no endereço 80h da RAM Interna (área D)

*MOV 80h,#0AAh*



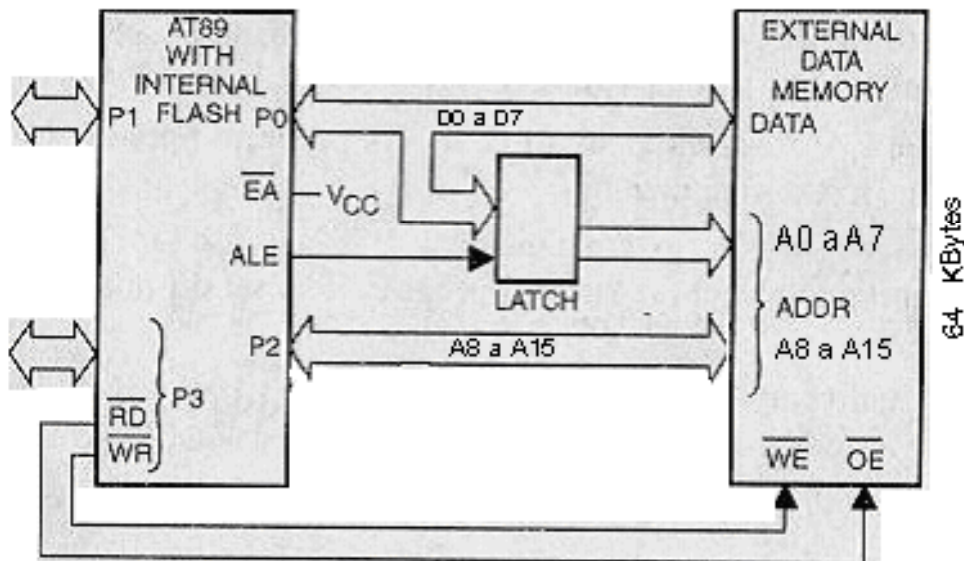
b. Escrever 0AAh no endereço 80h da RAM (área I de um microcontrolador com 256 bytes de RAM interna)

*MOV R0,#80h*

*MOV @R0,#0AAh*

## Memória de Dados Externa

**Acesso através de endereço de 16 bits**



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

**Instruções :**

```
MOVX A,@DPTR  
MOVX @DPTR,A
```



## Memória de Dados Externa

**Acesso através de endereço de 16 bits**

**Exemplo :**

**a. Armazenar 3Fh na posição 34CBh da memória externa :**

```
MOV DPTR,#34CBh  
MOV A,#3Fh  
MOVX @DPTR,a
```

**b. Ler o conteúdo da posição 13F4h da memória externa :**

```
MOV DPTR,#13F4h  
MOVX A,@DPTR
```

# Programação de Microprocessadores



- **Microprocessadores** são **‘Máquinas de Estado Seqüenciais Síncronas’** que operam mediante a execução de uma seqüência de códigos binários armazenados em memória.

- As ordens ou comandos compreendidos por um determinado Microprocessador, são **INSTRUÇÕES** seqüencialmente armazenadas na Memória.
- Ao conjunto de Instruções compreendidos por um determinado Microprocessador dá-se o nome de “**INSTRUCTION SET**”.
- Cada Microprocessador tem seu próprio Instruction Set que é em geral, diferente do Instruction Set de outro Microprocessador de fabricantes diferentes.

- Uma seqüência de Instruções do Instruction Set, armazenadas na memória e que realiza alguma operação, recebe o nome de **PROGRAMA**.

- Cada Instrução do Microprocessador é um código binário formada em geral por um ou mais Bytes.

- A cada código binário equivalente a uma Instrução está associado um **Mnemônico** para facilitar a compreensão da função que a Instrução executa.

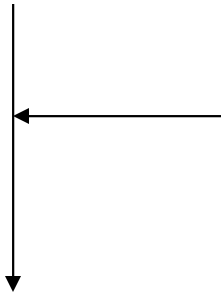
- Ao conjunto de Instruções e seus Mnemônicos equivalentes dá-se o nome de **LINGUAGEM ASSEMBLY**.

# Fluxograma

- Para a documentação lógica de um Programa em Assembly utiliza-se um **Fluxograma ou Diagrama de Blocos**.
- Cada bloco do Fluxograma equivale a um sub-conjunto do Instruction Set do Microprocessador.
- O Fluxograma é uma forma de se implementar logicamente um programa, antes que o mesmo seja codificado na Linguagem Assembly do Microprocessador.

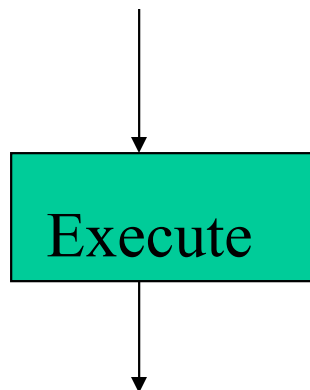
# Fluxograma

- Linhas de Fluxo do Programa



- Mostram a seqüência de execução das Instruções.
- Cada Bloco do Fluxograma possui apenas uma linha de Fluxo de Entrada e uma ou duas de saída

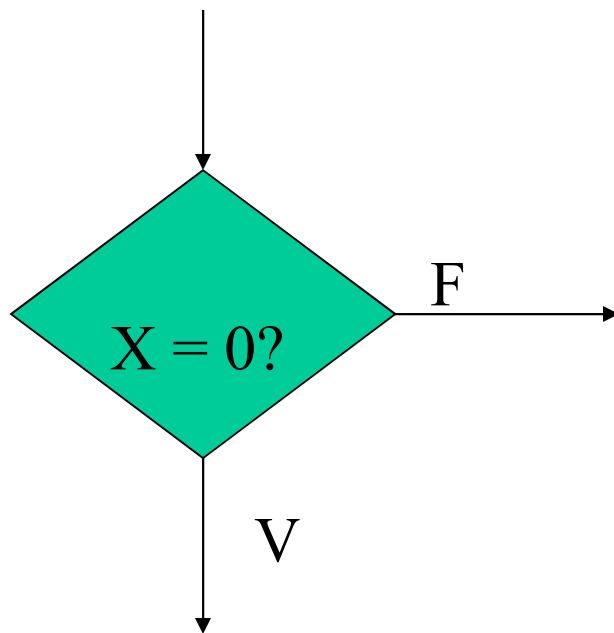
- Bloco de Processo



- Equivalem às Instruções que realizam alguma operação do tipo:
  - Movimento de Dados
  - Operação Aritmética
  - Operação Lógica

# Fluxograma

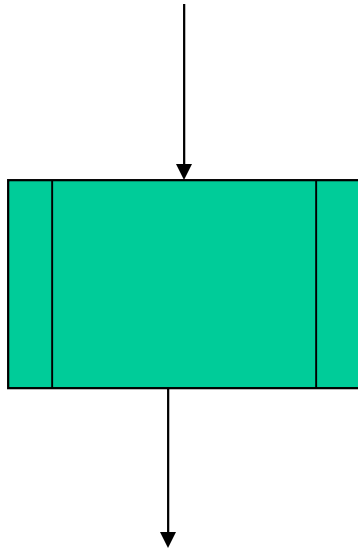
- Bloco de Decisão



- Equivale às Instruções que decidem sobre o Fluxo do Programa.
- Se a função dentro do bloco for Verdadeira(V) o programa continua abaixo, se for Falsa(F) o programa muda o fluxo.

# Fluxograma

- Processo Pré-definido

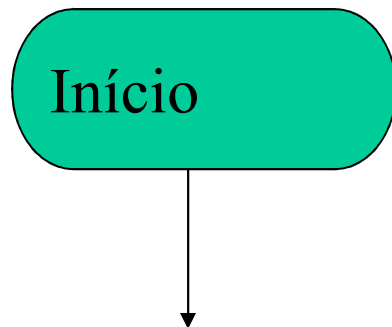


- Equivale às Instruções que mandam executar uma Sub-rotina armazenada em outro lugar da Memória.
- Observe que quando a sub-rotina termina, o fluxo do programa continua normalmente.



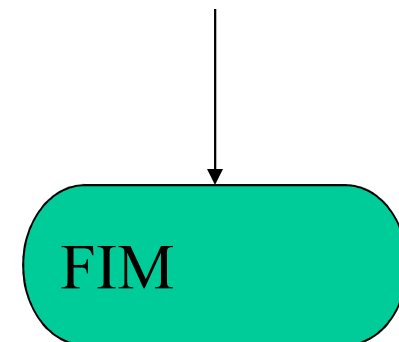
# Fluxograma

- Bloco de Início de Programa



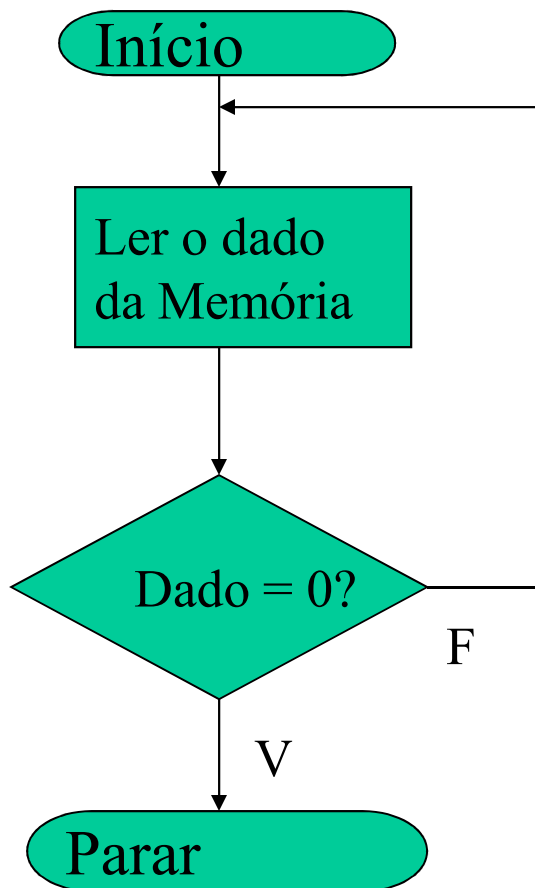
- O Bloco de Início de Programa não equivale a uma Instrução específica do Instruction SET.

- Bloco de Fim de Programa



- O Bloco de FIM equivale a uma instrução que termina o Programa. É chamado de **FIM LÓGICO** do Programa.

# Exemplo de Fluxograma de um Programa de Microprocessador



- O programa ao lado deve Ler um Dado da memória, verificar se é igual a zero. Se não for zero, continua em LOOP. Se for zero para o programa.

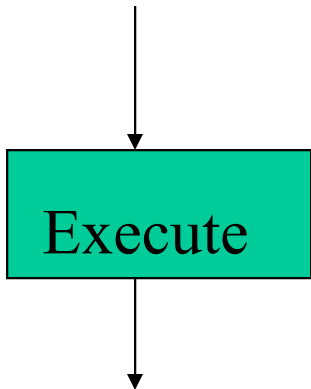
# Codificação Assembly

- Para Codificar um Programa escrito através de um Fluxograma, deve-se escolher o Microprocessador, ou seja, conhecer seu Conjunto de Instruções.

• Os Microcontroladores da família MCS-51 serão os dispositivos a serem aplicados nesta disciplina.

# Codificação Assembly do 8051

- Instruções equivalentes ao Bloco de Processo



- Instruções Aritméticas

SUBB	A, direct
------	-----------

ADD	A, Rn
-----	-------

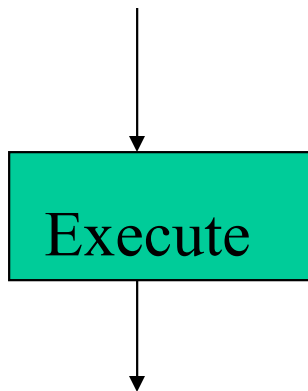
INC	A
-----	---

DEC	A
-----	---

DA	A
----	---

# Codificação Assembly do 8051

- Instruções equivalentes ao Bloco de Processo



- Instruções Lógicas

**ANL**    **A, Rn**

**ORL**    **A, direct**

**XRL**    **A, #data**

**CLR**    **A**

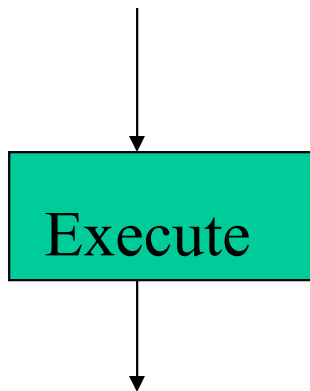
**CPL**    **A**

**RL**    **A**

**SWAP**    **A**

# Codificação Assembly do 8051

- Instruções equivalentes ao Bloco de Processo



- Instruções de Transferência de Dados

**MOV**    **A, Rn**

**MOVC**    **A, @A+DPTR**

**MOVX**    **A,@DPTR**

**PUSH**    **direct**

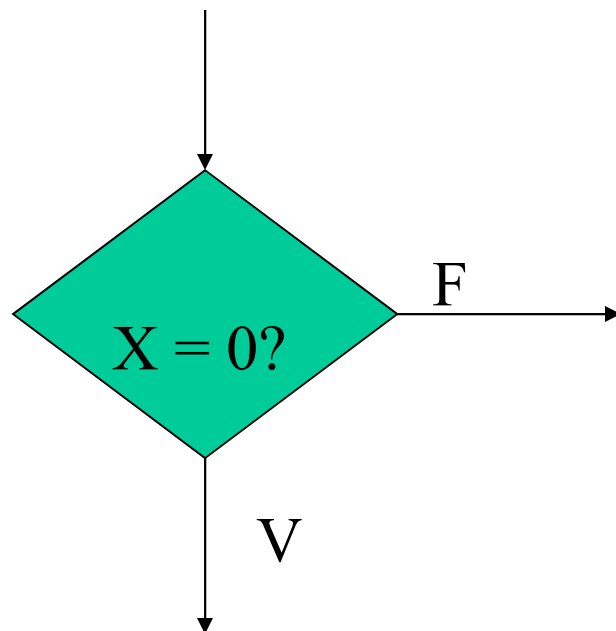
**POP**      **direct**

**XCH**     **A, Rn**

# Codificação Assembly do 8051

- Instruções equivalentes ao Bloco de Decisão

- Instruções de Desvio



JZ rel

JNZ rel

CJNE A, direct, rel

JC rel

JNC rel

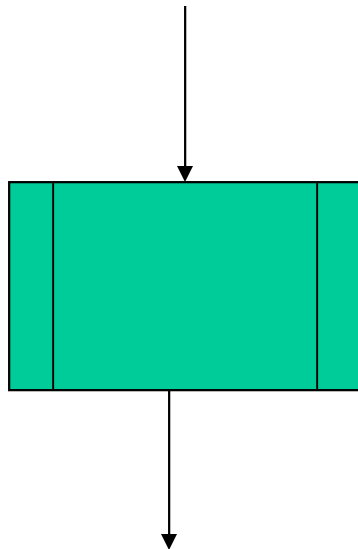
JB bit, rel

JNB bit, rel

DJNZ Rn, rel

# Codificação Assembly do 8051

- Instruções equivalentes ao Bloco de Processo Pré-definido



- Instruções de Sub-Rotina

LCALL addr16

ACALL addr11

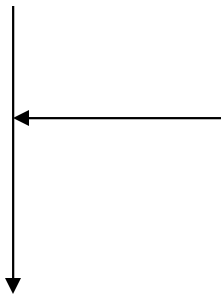
RET

RETI



# Codificação Assembly do 8051

- Instruções equivalentes a Mudança de Fluxo



- Instruções de Saltos

LJMP	addr16	AJMP	addr11
SJMP	rel		
JMP	@A+DPTR		

# Modos de Endereçamento do 8051

## 1. Endereçamento Imediato

- Opera sobre o dado localizado na própria instrução

• Identificado através do sinal #

• Exemplo:

**ADD A,#30h**

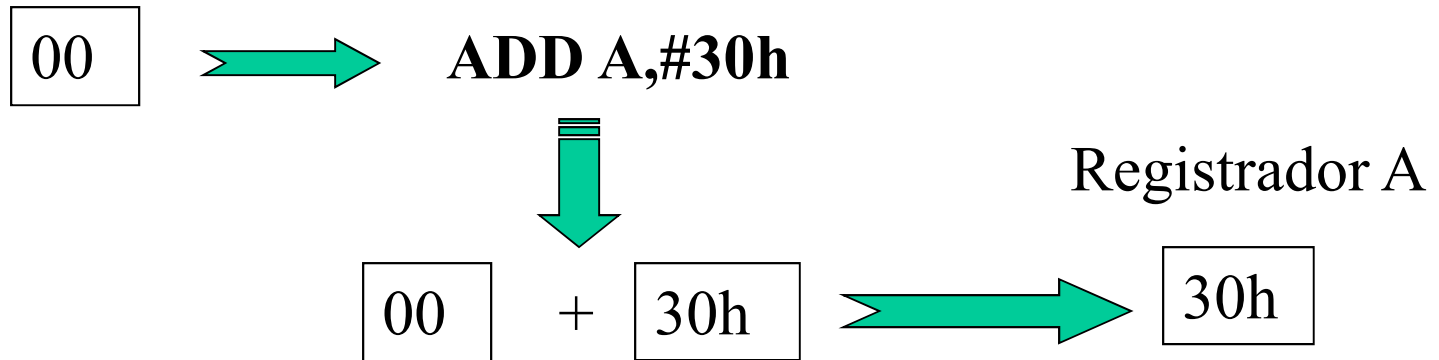
O dado 30h é somado ao Registrador A

# Modos de Endereçamento do 8051

## 1. Endereçamento Imediato

**ADD A,#30h**

Registrador A



# Modos de Endereçamento do 8051

**ADD A,#30h**

Memória

Registrador A

Programa

Operação

30	20

00

ADD A,#30h

00 + 30



Registrador A

30

↑  
Endereço  
(hexa)

↑  
Conteúdo  
(hexa)

# Modos de Endereçamento do 8051

## 2. Endereçamento Direto

- Opera sobre o dado cujo endereço está na instrução

• Exemplo:

**ADD A,30h**

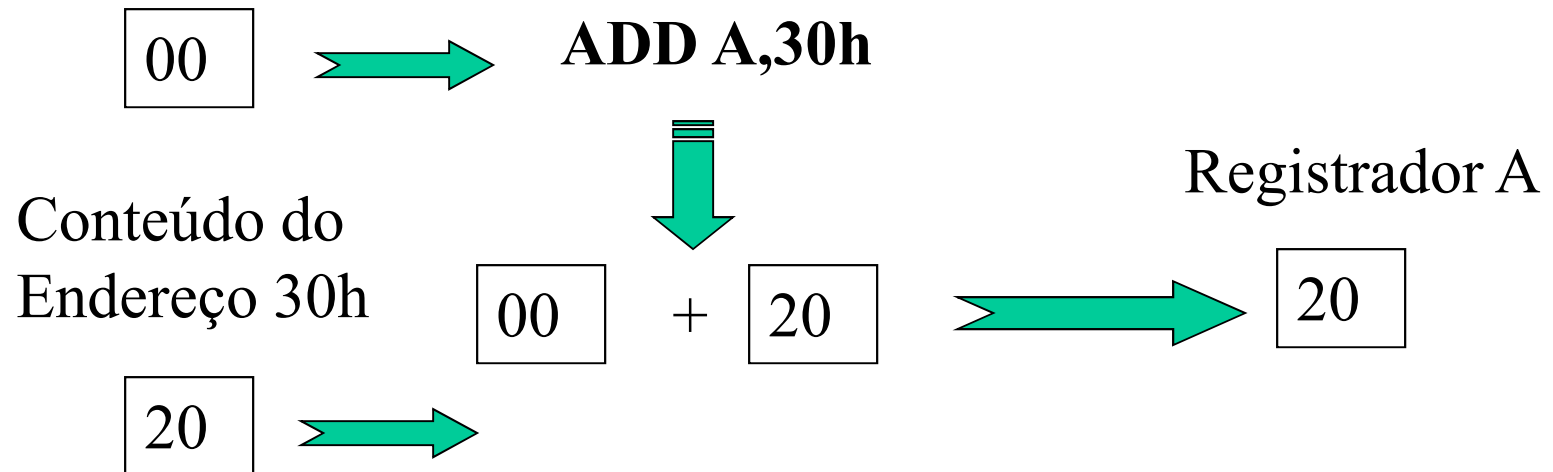
O dado armazenado no endereço 30h é somado ao Registrador A

# Modos de Endereçamento do 8051

## 2. Endereçamento Direto

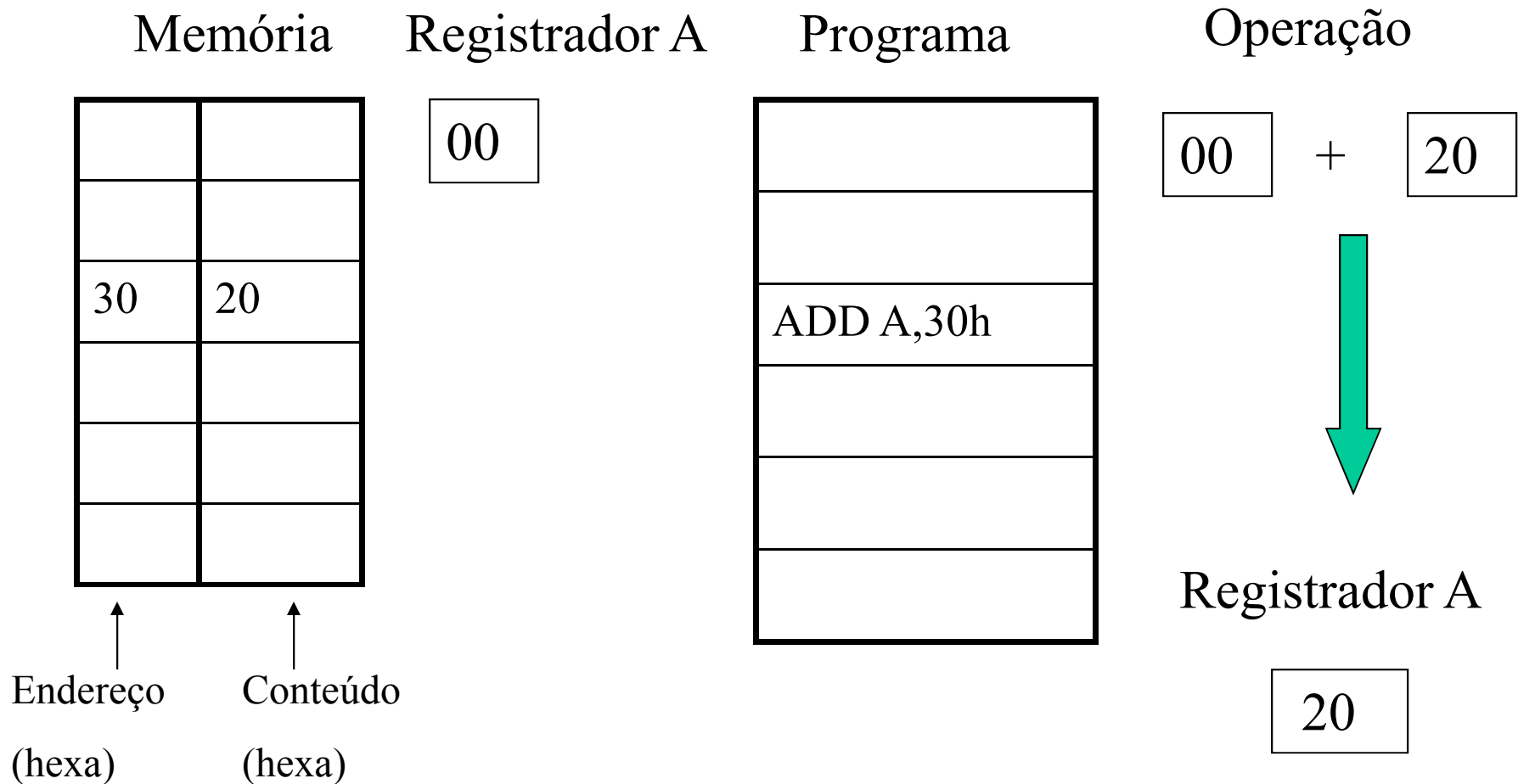
**ADD A,30h**

Registrador A



# Modos de Endereçamento do 8051

**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**

Registrador A

00 → **ADD A,@R0**

Registrador R0

30h

Conteúdo do  
Endereço 30h

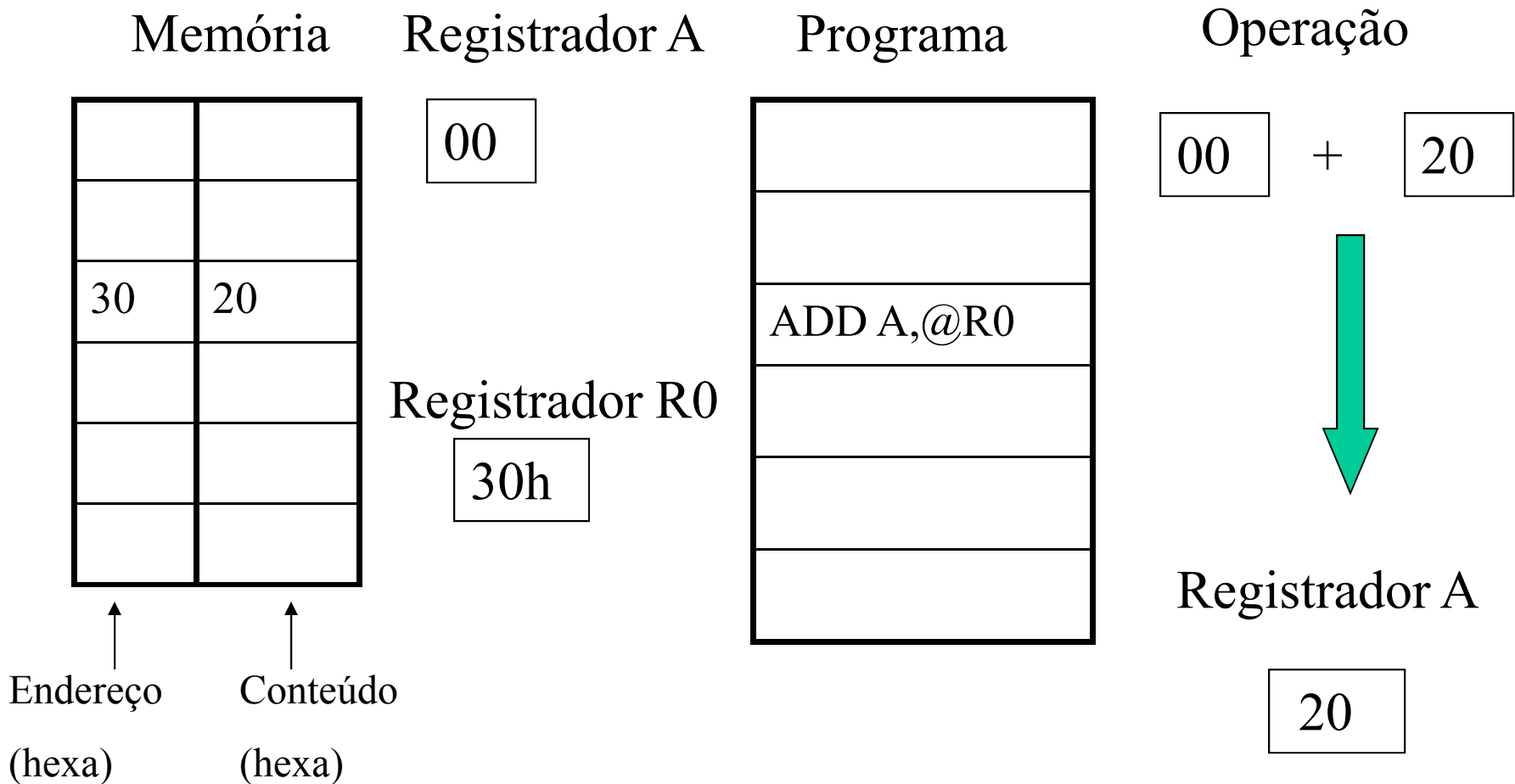
20 →

00 + 20 → Registrador A

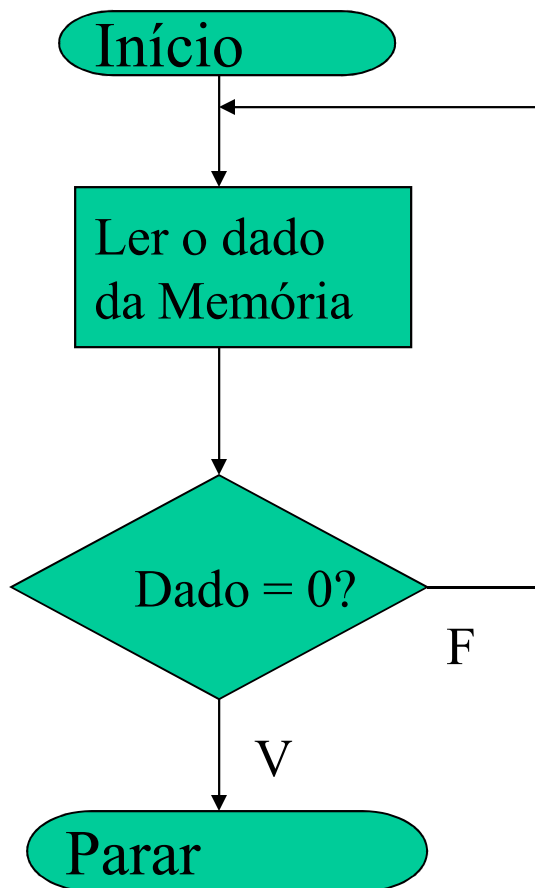
20

# Modos de Endereçamento do 8051

**ADD A,@R0**



# Exemplo de um Programa Assembly do 8051



```
ORG 0
LOOP:
    MOV A,30H
    CJNE A,#00,LOOP
AQUI:
    SJMP AQUI
```

# Exemplo de um Programa Assembly do 8051

Mnemônicos (Programa Assembly)

```
                ORG 0
LOOP:           MOV A,30H
                CJNE A,#00,LOOP
AQUI:           SJMP AQUI
```

COMPILADOR

Código Compilado (Opcode)

Addr	Opcodes	ASC	Label	Disassembly
0000	E5 30	â0	LOOP	MOV A,30h
0002	B4 00 FB	î0		CJNE A,#00h,LOOP
0005	80 FE	€p	AQUI	SJMP AQUI

# Exemplo de um Programa Assembly do 8051

Memória de  
Programa

00	E5
01	30
02	B4
03	00
04	FB
05	80
06	FE

↑                      ↑  
Endereço            Conteúdo

Addr	Opcodes	ASC	Label	Disassembly
0000	E5 30	â0	LOOP	MOV A,30h
0002	B4 00 FB	î0		CJNE A,#00h,LOOP
0005	80 FE	€p	AQUI	SJMP AQUI