



# *Microprocessadores*

# Motivação

---

- Uma unidade central de controle está presente em praticamente todas às áreas da tecnologia moderna;
- A indústria de MCU/MPU movimenta anualmente 50 bilhões de dólares por ano;
- Aproximadamente a metade de todas as CPU's vendidas são pequenos microcontroladores de 8 bit;
- Mais de 5 bilhões de unidades de 8 bit são vendidas anualmente.

# Objetivos

---

- Dar ao aluno o conhecimento das diversas tecnologias e arquiteturas existentes no mercado;
- Capacitar o aluno a desenvolver projetos usando microprocessadores e utilizar ferramentas de desenvolvimento modernas;
- Apresentar os conceitos básicos comuns às tecnologias comerciais existentes;
- Auxiliar o aluno à escolher o melhor dispositivo para seu projeto, levando-se em consideração desempenho, custo e confiabilidade.

# Bibliografia

---

Título	<b>Aplicações práticas do microcontrolador 8051</b>
<b>CDU</b>	<b>681.325.65</b>
<b>Cutter</b>	<b>S586a</b>
Tipo da obra	Livro
Código da obra	56672
Autor(es)	Vidal Pereira da Silva Junior
Última Edição	Editora: Érica, 13a edicao, São Paulo, 2005, 244p
Códigos	ISBN: 85-7194-194-7

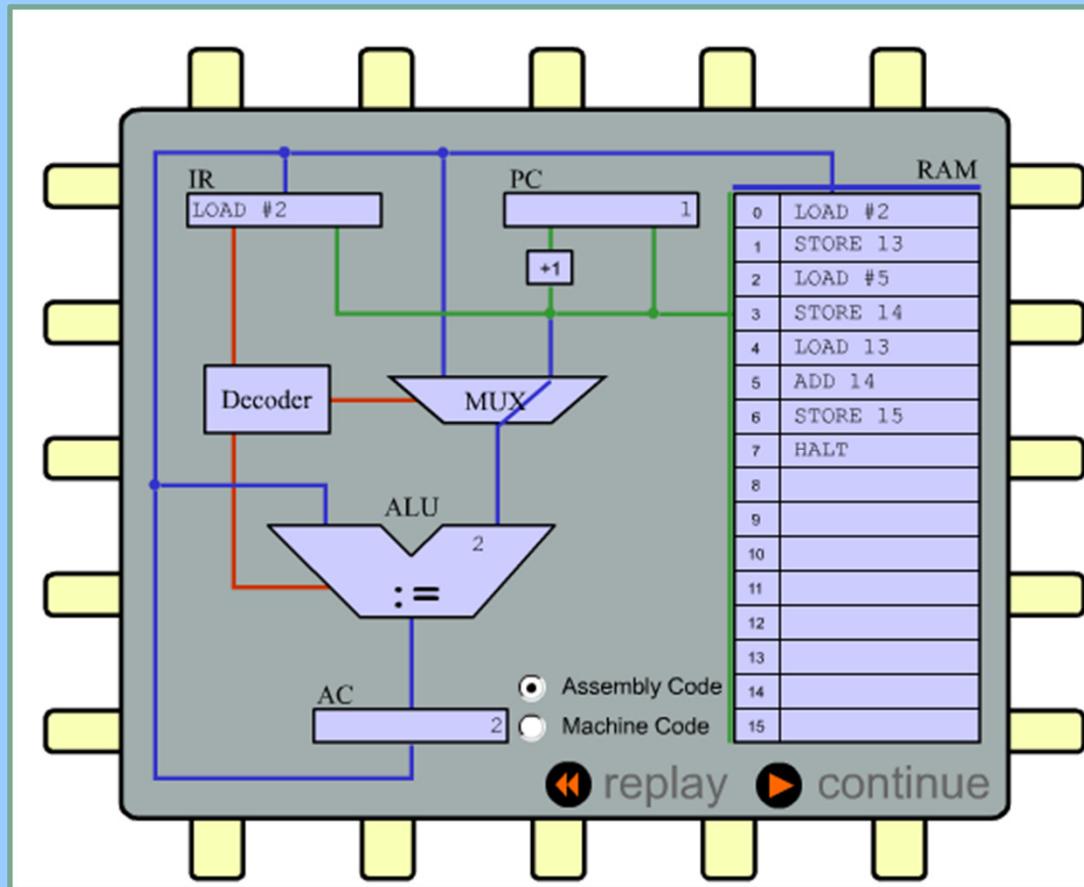
Título	<b>Microcontrolador 8051 detalhado</b>
<b>CDU</b>	<b>681.326</b>
<b>Cutter</b>	<b>N651m</b>
Tipo da obra	Livro
Código da obra	60313
Autor(es)	Denys Emilio Campion Nicolosi
Última Edição	Editora: Érica, 3a edicao, São Paulo, 2002, 221p
Códigos	ISBN: 85-7194-721-x

## 1. Definição de Microprocessador

---

- O que é um Microprocessador?
  - É um máquina de estado finita onde seus estados são definidos através de instruções binárias armazenada em uma memória e as mudanças desses estados são sincronizadas com um sinal de relógio;
  - Está normalmente associada a Unidade Central de Processamento (CPU) de um computador;

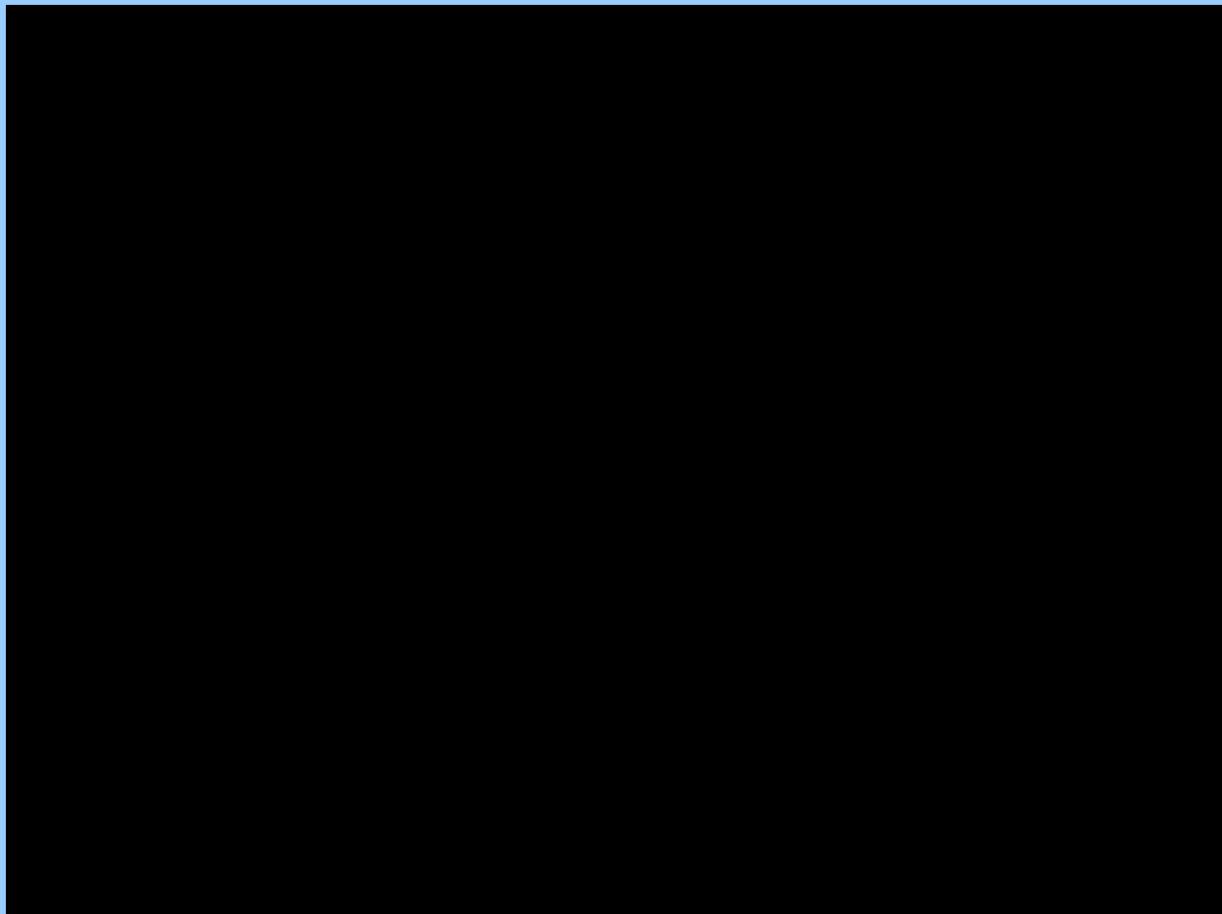
# 1. Definição de Microprocessador



Arquitetura Básica de um Microprocessador

# 1. Definição de Microprocessador

---

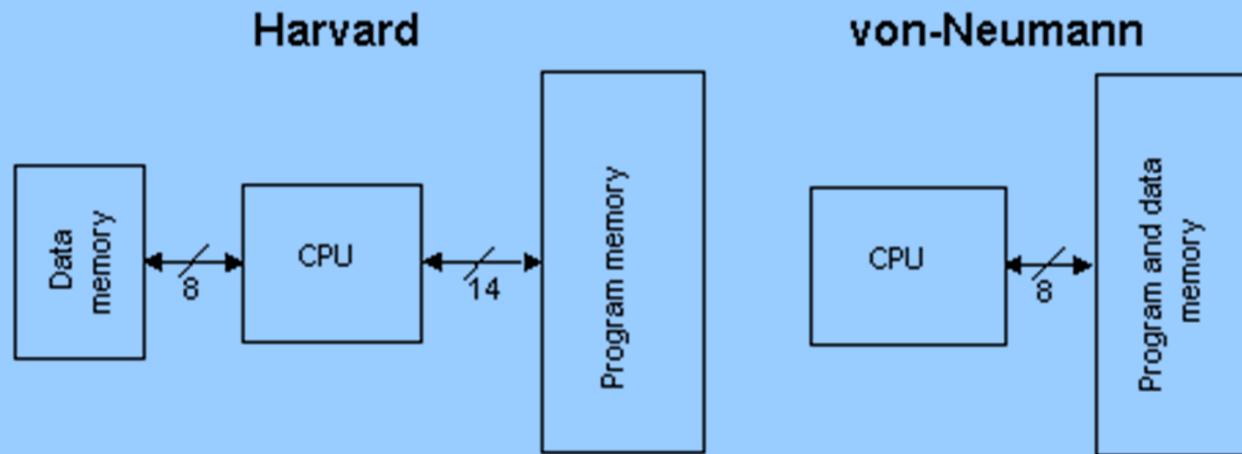


Arquitetura Básica de um Microprocessador

## 2. Conceitos Básicos de Arquitetura de Computadores

---

- Arquitetura Harvard x von Neuman



Harvard vs. von Neuman Block Architectures

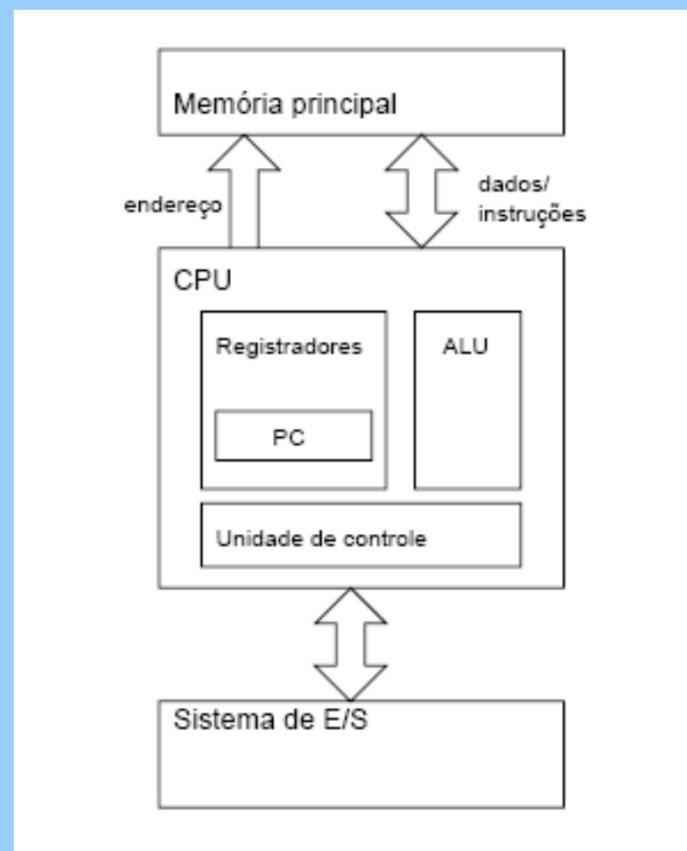
---

### Arquitetura Física

## 2. Conceitos Básicos de Arquitetura de Computadores

---

### Arquitetura von Neuman



## 2. Conceitos Básicos de Arquitetura de Computadores

---

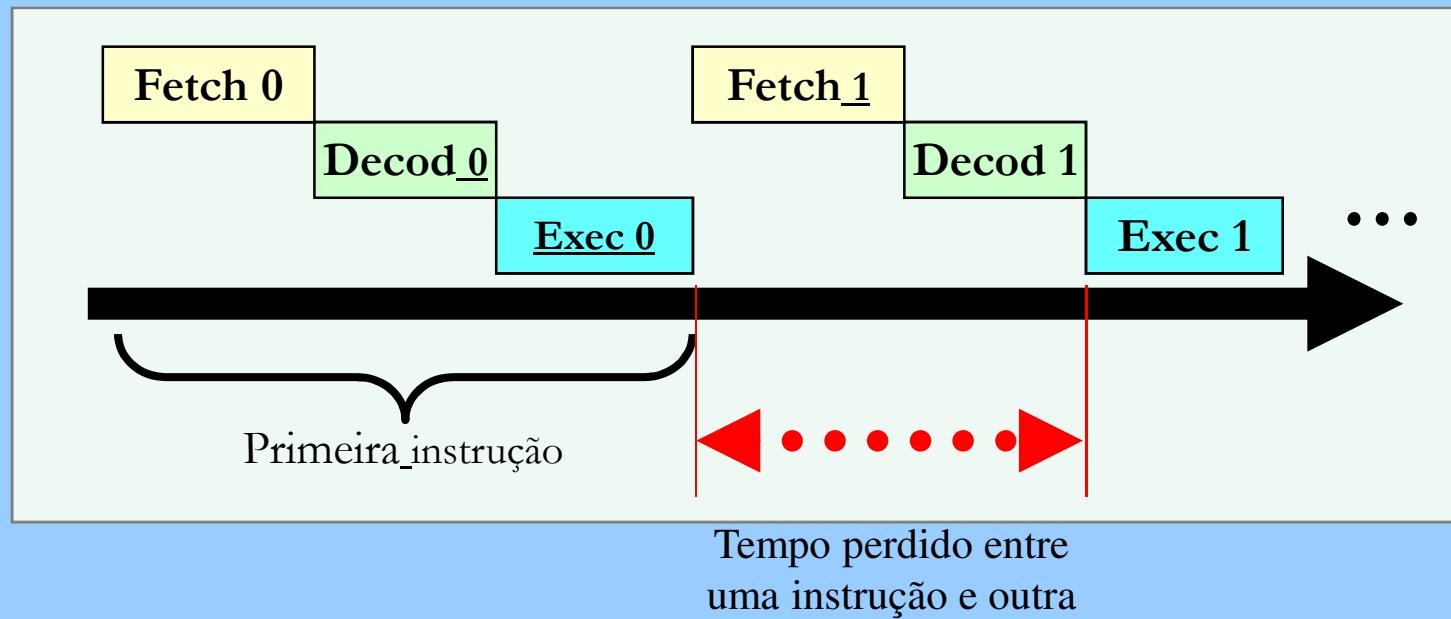
### Arquitetura von Neuman

- Barramento da memória de dados comum ao barramento da memória de programa – cria-se um gargalo;
- Conjunto de instruções maiores (CISC - Complex Instruction Set Computer) – Mais de 100 nos microcontroladores da Intel (8051) e da Motorola;
- Instruções demoram mais de um ciclo de máquina para execução;

## 2. Conceitos Básicos de Arquitetura de Computadores

---

- Ciclo de execução de uma instrução em processadores seqüenciais com arquitetura Von Neumann



## 2. Conceitos Básicos de Arquitetura de Computadores

---

### Arquitetura Harvard



## 2. Conceitos Básicos de Arquitetura de Computadores

---

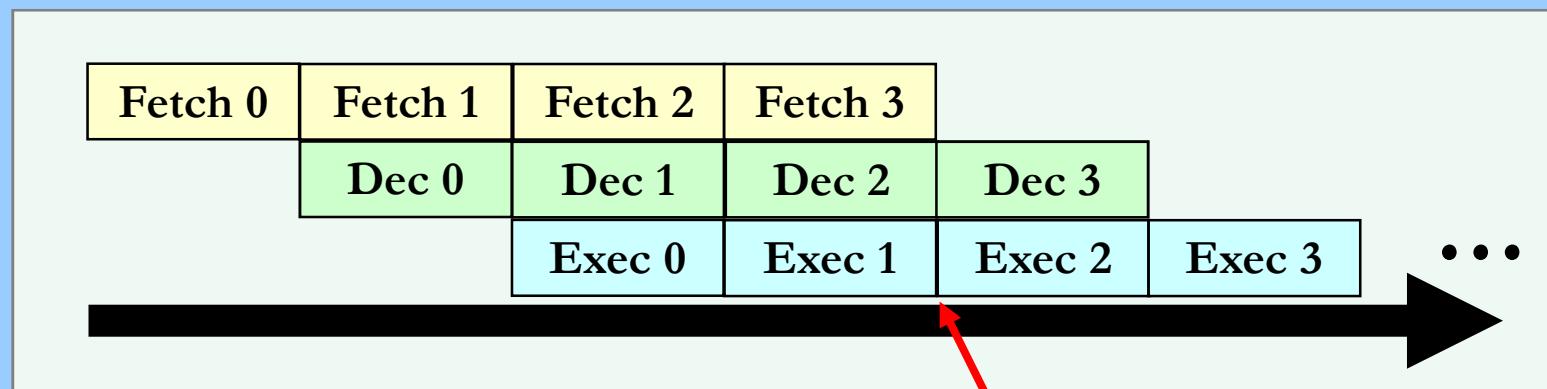
### Arquitetura Harvard

- Barramento da memória de dados separado do barramento da memória de programa;
- Instruções podem ser representadas por palavras de mais de 8 bits (14 bits no PIC16F84);
- Instruções executadas em apenas 1 ciclo de máquina;
- Aumenta o desempenho, reduz o "gargalo";
- Repertório com menos instruções (RISC - Reduced Instruction Set Computer) – 35 instruções no PIC16F84;

## 2. Conceitos Básicos de Arquitetura de Computadores

---

- *Pipeline* de Instrução
  - Ciclo de execução de uma instrução em processadores seqüenciais com Arquitetura RISC.



Pipeline de Instruções em processadores RISC

Na maioria das vezes, uma instrução é executada imediatamente após a outra.

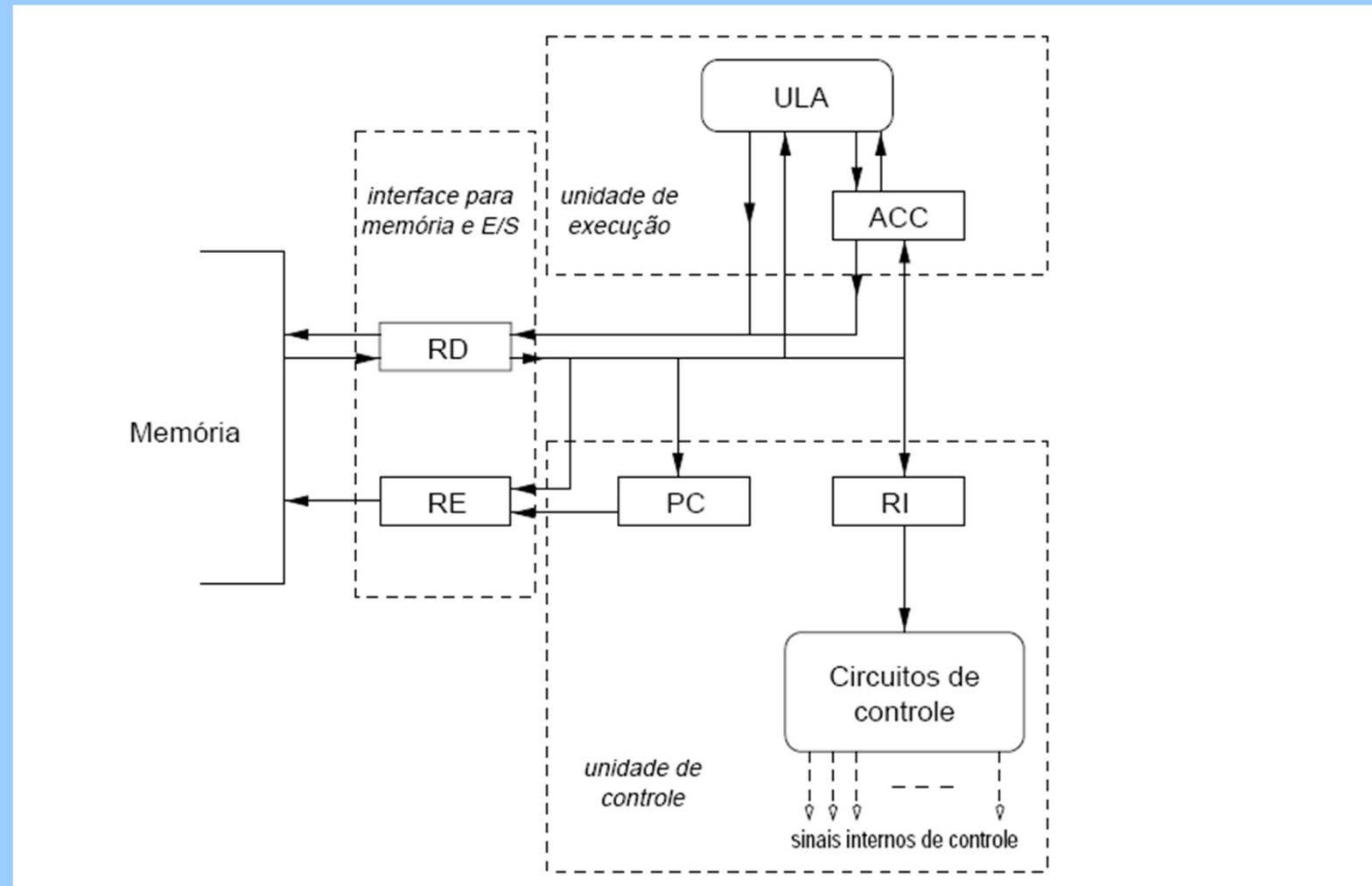
## 2. Conceitos Básicos de Arquitetura de Computadores

---

- Arquiteturas de CPU

Dispositivo	Arquitetura Física	Arquitetura Lógica
Z80	Von Neumann	CISC
8051	Von Neumann	CISC
PIC	Harvard	RISC
ARM	Harvard	RISC
PSoC	Harvard	CISC

### 3. Arquitetura Básica de um Microprocessador



### 3. Arquitetura Básica de um Microprocessador

---

#### Unidades básicas de um Microprocessador

- **Unidade Lógico Aritmética (ULA)**, responsável pela realização das operações lógicas e aritméticas;
- **Unidade de Controle**, responsável pela decodificação e execução das instruções, fornecendo os sinais de temporização adequados para as diversas partes do processador e do próprio computador;
- **ACC, RD, RE, RI, PC, Registradores para armazenamento da Informação Binária (dados, endereços e instruções).**

### 3. Arquitetura Básica de um Microprocessador

---

#### Memórias

➤ **Quanto ao acesso:**

- **Acesso seqüencial** - precisa-se passar por endereços intermediários (Fita Magnética);
- **Acesso aleatório (RAM)** - acessa qualquer dado em qualquer endereço sem a necessidade de ter que passar por outros endereços intermediários.

➤ **Volatilidade:**

- **Memórias voláteis:** são aquelas que perdem as informações quando é cortada sua alimentação. São memórias que geralmente usam como elemento de memória o flip-flop;
- **Memórias não voláteis:** são memórias que mesmo desligando-se sua alimentação, não perdem as informações armazenadas. EX: Memórias magnéticas e as eletrônicas ROM, PROM, EPROM, EEPROM, FLASH;

### 3. Arquitetura Básica de um Microprocessador

---

#### Memórias

##### ➤ Memórias de escrita/leitura ou somente leitura

- **Escrita/leitura:** são memórias que podem ser acessadas pela CPU tanto para leitura quanto para escrita. Elas são usadas para armazenar dados que serão utilizados durante a execução do programa (memórias RAM's, EEPROM's);

- **Somente leitura:** são as memórias que armazenam o programa, ou seja, são as memórias que só serão lidas pela CPU e que já vêm gravadas para o sistema (memórias ROM's ,PROM's, EPROM's, FLASH, etc).

### 3. Arquitetura Básica de um Microprocessador

---

#### Memórias

➤ **Tipo de armazenamento:**

- **Estáticas (SRAM – Static RAM):** memórias estáticas são aquelas nas quais as informações permanecem armazenadas enquanto houver alimentação;
- **Dinâmicas (DRAM – Dynamic RAM):** são memórias que necessitam de um sinal de *refresh* para manter a informação armazenada mesmo com a presença de alimentação. Isso acontece porque cada célula tem um transistor MOSFET e um capacitor que armazena um dado (bit 1).

### 3. Arquitetura Básica de um Microprocessador

---

#### Tipos de Memórias

- **Memórias RAM (Random Access Memory):**
  - São memórias de acesso aleatório. Elas podem ser estáticas ou dinâmicas e consideradas voláteis;
- **Memórias ROM (Ready Only Memory):**
  - Essas memórias são utilizadas no sistema somente para a leitura normalmente usada para armazenamento do programa residente da CPU (Firmware);

### 3. Arquitetura Básica de um Microprocessador

---

#### Tipos de Memórias

- **Memórias PROM (Programmable Read Only Memory):**
  - Essas memórias são utilizadas no sistema somente para a leitura. Geralmente usadas como memórias de programa só podem ser gravadas com gravadores específicos e só uma vez. Não voláteis.
- **EPROM (Erasable Programmable Read Only Memory):**
  - Essas memórias são utilizadas no sistema somente para a leitura, também empregadas como memórias de programa e só podem ser gravadas com gravadores específicos. Podem ser apagadas por raios ultravioleta e regravadas por muitas vezes. Não voláteis.

### 3. Arquitetura Básica de um Microprocessador

---

#### Tipos de Memórias

- **EEPROM (Electrically-Erasable Programmable Read Only Memory):**
  - Apagáveis eletricamente. Possui número limitado de apagamento e regravação variando de 100.000 a 1 milhão de vezes;
- **Memórias FLASH**
  - É uma memória do tipo EEPROM – Toshiba 1980;
  - Permitem gravação e leitura por blocos;
  - Mais lentas que as DRAM's porém são não Voláteis;
  - Número finito de ciclos de escrita/apagamento;

### 3. Arquitetura Básica de um Microprocessador

---

#### Memórias Flash

- Baixo Custo;
- Ocupação mínima de espaço;
- Baixo consumo de energia;
- Alta resistência mecânica e alta durabilidade;
- Alta segurança contando com recursos ECC (Error Correcting Code);

### 3. Arquitetura Básica de um Microprocessador

---

#### Memórias Flash

- Tipos de Memórias FLASH – (FLASH NOR e FLASH NAND):

##### FLASH NOR

- A memória flash **NOR** permite acessar os dados da memória de maneira **aleatória** e com alta velocidade;
- Possui custo mais elevado e **alto tempo de gravação e apagamento** (operação de gravação mais lentas) em relação ao tipo NAND;
- Usadas nas BIOS das placas-mãe e também em firmwares de vários dispositivos, que antes eram armazenados em memória ROM ou EPROM;
- Largamente utilizada até hoje em celulares, palmtops e firmware.

### 3. Arquitetura Básica de um Microprocessador

---

#### Memórias Flash

##### FLASH NAND

- A memória **FLASH NAND** possui tempos de escrita (gravação) e apagamento mais rápido;
- **Acesso seqüencial** às células de memória e trata-as em conjunto, isto é, em blocos de células (leitura por blocos);
- Cada bloco consiste em páginas tipicamente de 512, 2048 ou 4096 bytes;
- Maior densidade de armazenamento e menor custo por bit;
- Maior vida útil (até 10 vezes mais em relação a NOR).

### 3. Arquitetura Básica de um Microprocessador

---

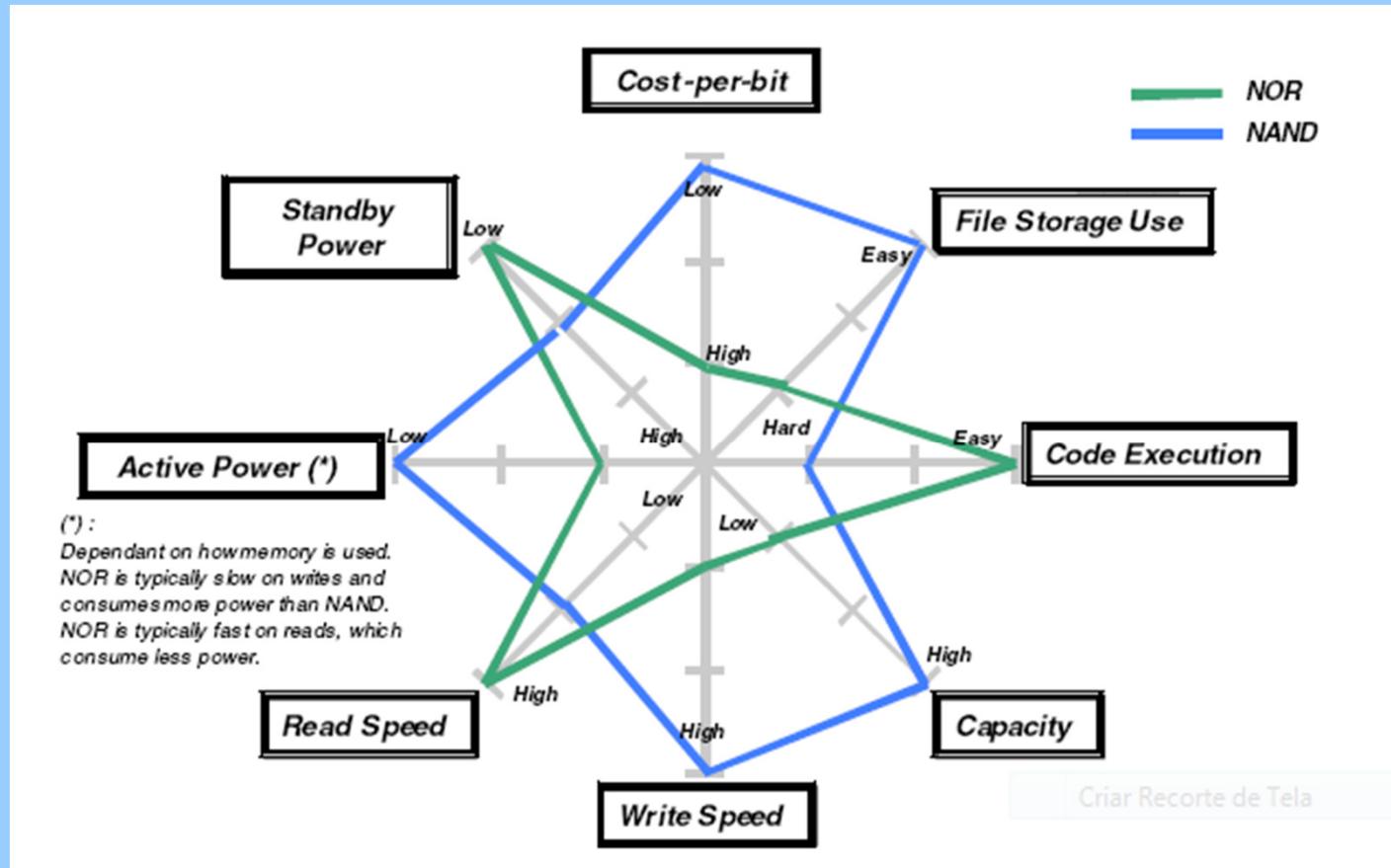
#### Memórias Flash

##### ➤ Principais diferenças entre NOR e NAND

- As conexões das células individuais de memória são diferentes;
- A densidade de armazenamento no chip é atualmente mais elevado em memórias **NAND (Pen Drivers)**;
- O custo da NOR é muito mais elevado.
- A NOR permite acessos aleatórios, enquanto a NAND permite apenas acesso seqüencial à memória;
- A leitura é muito mais rápida na **NOR (BIOS e Firmware)**.

### 3. Arquitetura Básica de um Microprocessador

#### Memórias Flash



Comparativo entre FLASH NOR e NAND

## 4. Arquitetura Básica de um Microcontrolador

---

Microcontrolador x Microprocessador

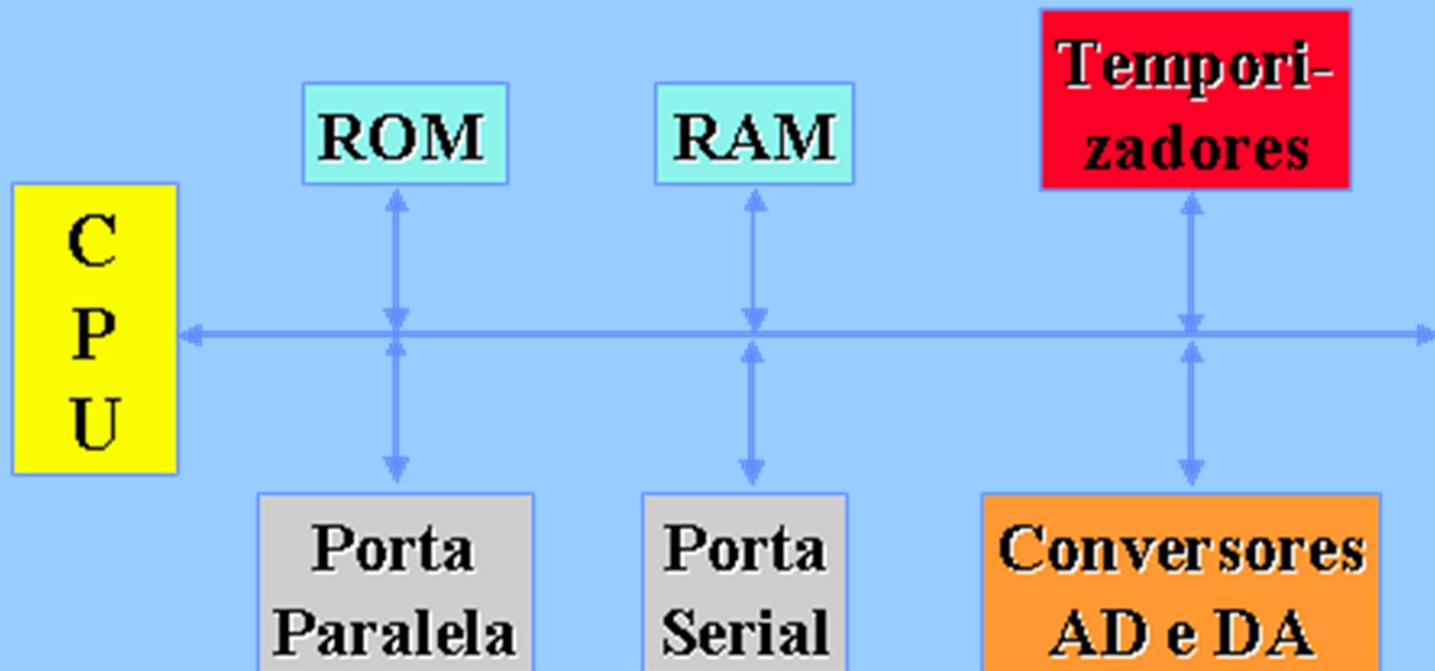
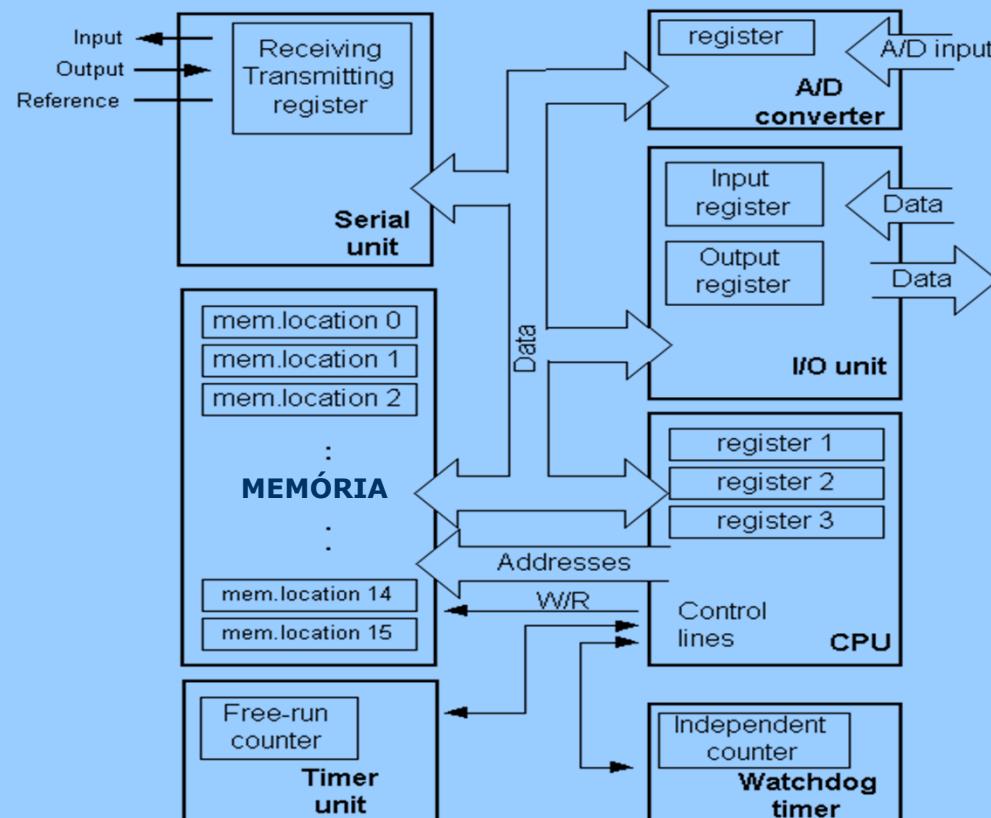


Diagrama em blocos de um sistema controlador

# 4. Arquitetura Básica de um Microcontrolador

- Microcontrolador



Arquitetura típica de um microcontrolador integrada em um único chip

## 5. Breve Histórico e Fabricantes de Microcontroladores

---

- **1978 - O primeiro microcontrolador 8048 da Intel ;**
- **1983 - Nasce a família 8051 da Intel;**
- **Motorola com o 68HC11;**
- **Zilog com sua família Z8, ;**
- **National com o COP8;**
- **Microchip com os PICs;**
- **Atmel com os AVRs.**

## 6. Fabricantes de Microcontroladores Família 8051

---

Fabricante	Microcontrolador	Relógio	RAM	ROM	Serial	Timer	AD
Intel	87C51-24	24 MHz	256	4K	1	2	não
Philips	P80C31	12 MHz	128 B	não	1	3	não
Philips	P87C51	12 MHz	128	4 KB	1	3	não
Philips	P87C51MB2	24 MHz	2 KB	4 KB	1	4	não
Philips	P80C552	24 MHz	256	não	2	3	8/10 bits
Atmel	AT87F51	24 MHz	128	4 KB	1	2	não
Atmel	AT87F51RC	24 MHz	512	32 KB	1	3	não
Atmel	AT89C5115	40 MHz	512	18 KB	1	3	8/10 bits
Dallas	DS87C550	33 MHz	1 KB	8 KB	2	3	8/10 bits

## 7. Microcontroladores PIC (Peripheral Integrated Controller)

---

- Microcontroladores fabricados pela Microchip Technology;
- Arquitetura Harvard e conjunto de instruções RISC (sets de 35 instruções e de 76 instruções);
- Processam dados de 8 bits (foi lançada uma família de 16 bits com prefixo 24F);
- Recursos de programação por memória FLASH, EEPROM e OTP;
- Famílias de 12 bits, 14 bits e 16 bits de núcleo de processamento;

## 7. Microcontroladores PIC (Peripheral Integrated Controller)

---

- Velocidade de 0kHz (ou DC) a 48MHz usando ciclo de instrução mínimo de 4 períodos de clock o que permite uma velocidade chegue ao máximo de 10 MIPS;
- Possuem interrupções tanto externas como de periféricos internos;
- Alimentação de 2 a 6V;
- Encapsulamento de 6 a 100 pinos em diversos padrões (SOT23, DIP, SOIC, TQFP, etc);

## 7. Microcontroladores PIC (Peripheral Integrated Controller)

---

- Principais periféricos internos (a disponibilidade varia conforme o modelo):
  - Conversores Analógico-Digitais de 8 a 12 bits;
  - Contadores e timers de 8 e 16 bits;
  - Comparadores Analógicos;
  - USARTs;
  - Controladores de comunicação I2C, USB;
  - Controladores PWM;

## 7. Microcontroladores PIC (Peripheral Integrated Controller)

---

- Controladores de LCD;
- Controladores de motores;
- Periféricos para LAN, CAN;
- Controladores Ethernet;
- Periféricos IRDA;
- Codificadores para criptografia Keeloq;
- Watchdog timer;
- Detetores de falha na alimentação;

## 7. Microcontroladores PIC (Peripheral Integrated Controller)

---

- Portas digitais com capacidade de 25mA (fornecer ou drenar) para acionar circuitos externos;
- Osciladores internos.

## 8. Família do Microcontrolador 8051

---

Versões do 8051:

- 8031 – sem ROM (ROMLESS), 128 bytes de RAM e 2 T/C;
- 8051 – com 4KB de ROM, 128 bytes de RAM e 2 T/C;
- 8751 – com 4KB de EPROM, 128 bytes de RAM e 2 T/C;
- 8032 – sem ROM (ROMLESS), 256 bytes de RAM e 3 T/C;
- 8052 – com 8KB de ROM, 256 bytes de RAM e 3 T/C;
- 8752 - com 8KB de EPROM, 256 bytes de RAM e 3 T/C;

# 8. Família do Microcontrolador 8051

---

## Variações:

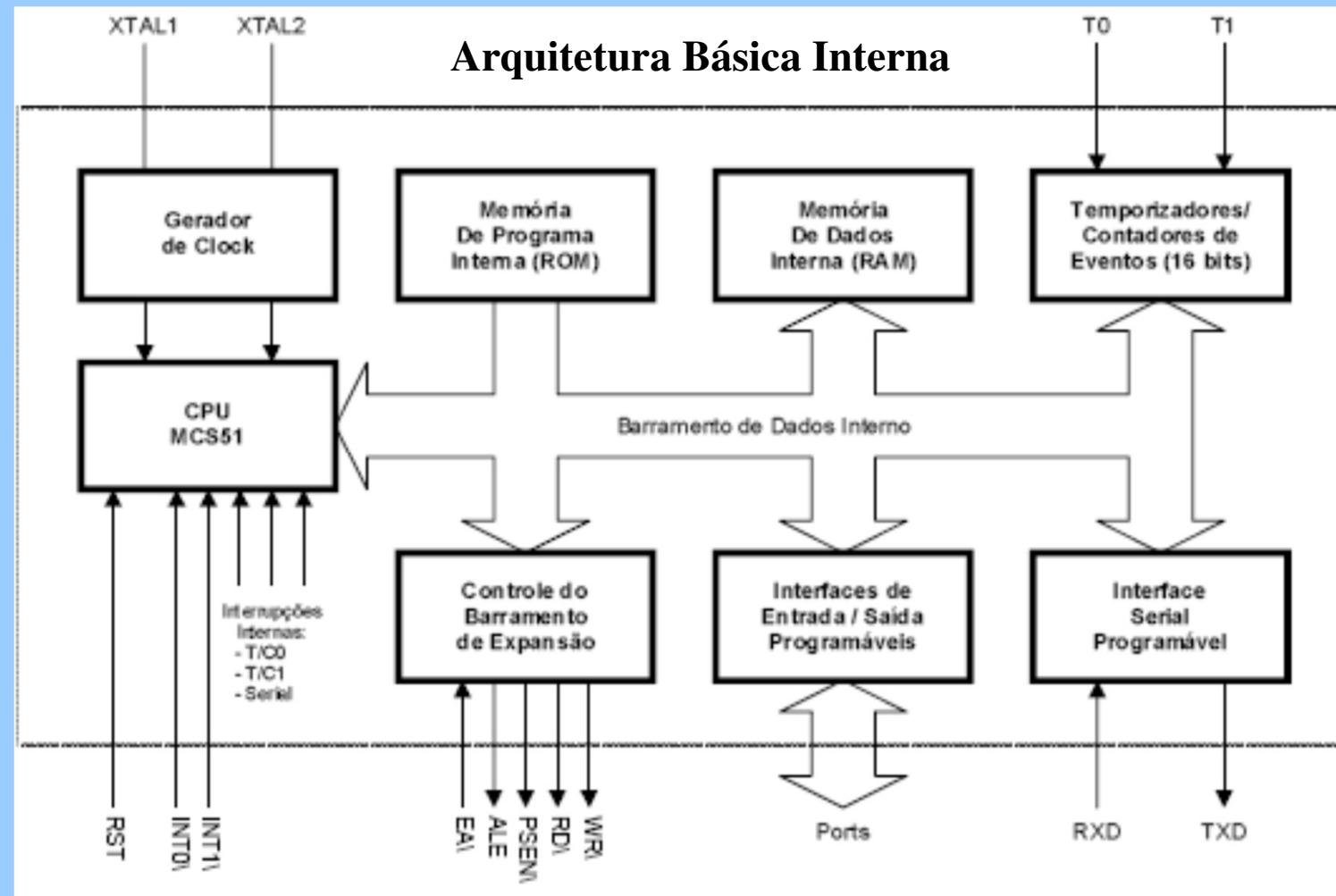
- 80C31, 80C32, 80C51 e 80C52 – versões CMOS, incluindo modos de baixo consumo de energia;
- 80LV31, 80LV32, 80LV51 e 80LV52 – versões low-voltage (ISSI);
- 89C51 e 89C52 – versões com memória Flash reprogramável (Atmel, ISSI, Philips);
- 89C1051, 89C2051 e 89C4051 – versões com memória Flash reprogramável, comparadores analógicos e invólucro reduzido (Atmel)

## 8. Família do Microcontrolador 8051

---

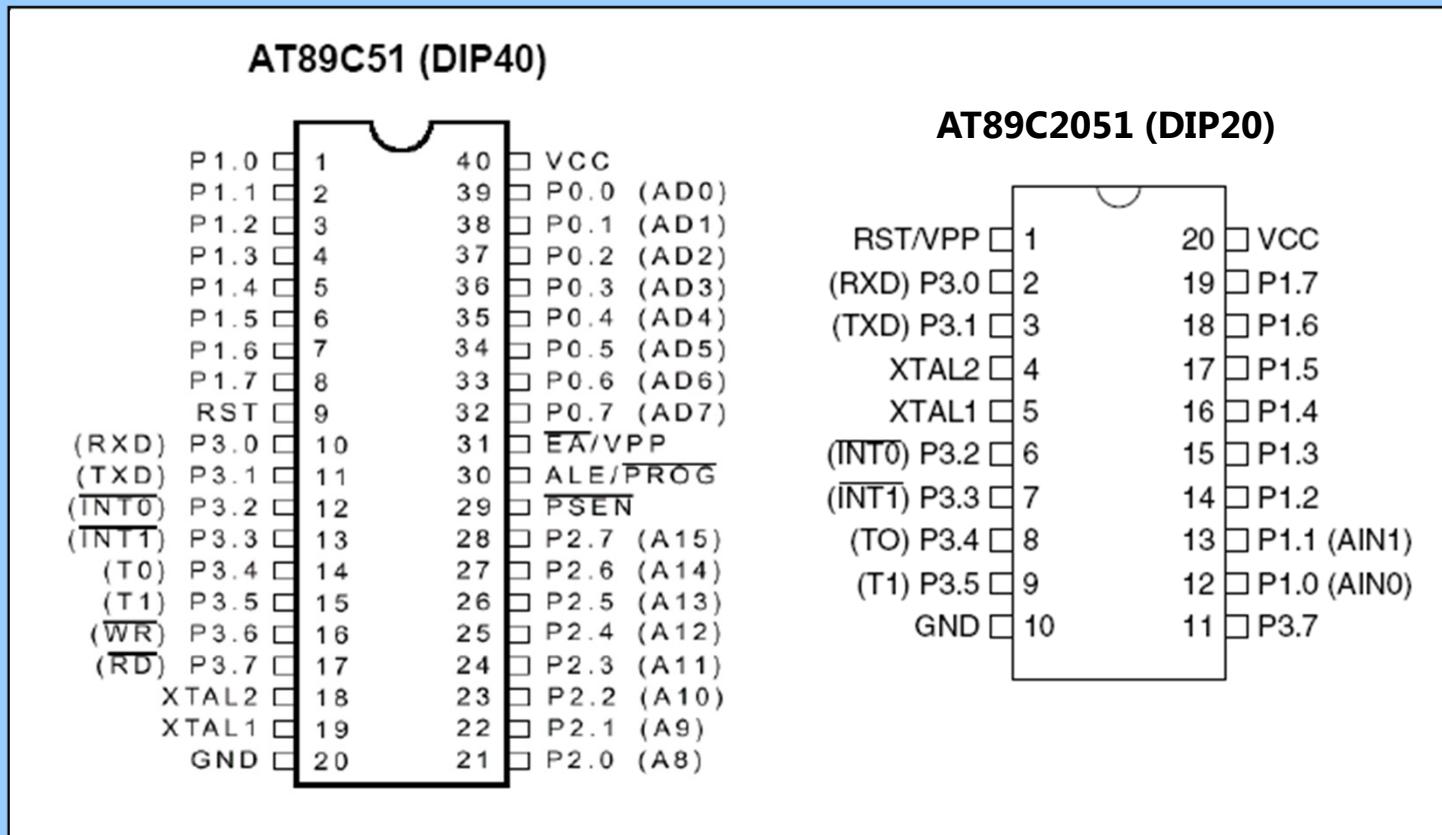
- 89C420 – versão com memória flash e clock otimizado (três vezes mais veloz), capaz de operar em até 33MHz (Dallas/Maxim Semiconductor);
- C505L – versão com 32KB de ROM, 512 bytes de RAM, conversor A/D de 10 bits e interface para LCD (Infineon Technologies);
- P51XA-G3, P51XA-H3 e P51XAS3 – versões com arquitetura de 16 bits (Philips);
- Família C8051 da Silicon Labs – Processadores de alto desempenho com estruturas de pipeline implementada.

# 9. Arquitetura do Microcontrolador 8051



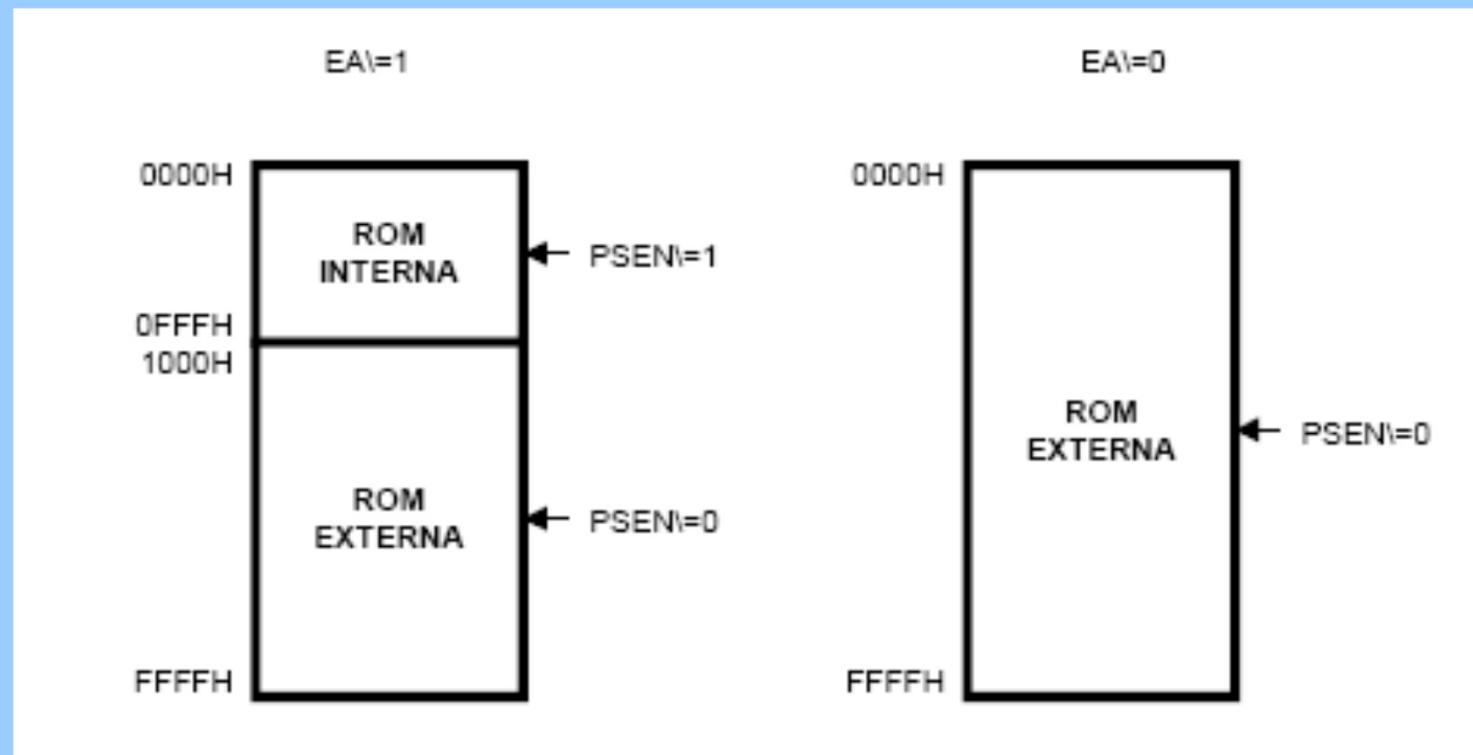
# 9. Arquitetura do Microcontrolador 8051

## Configuração dos Pinos:



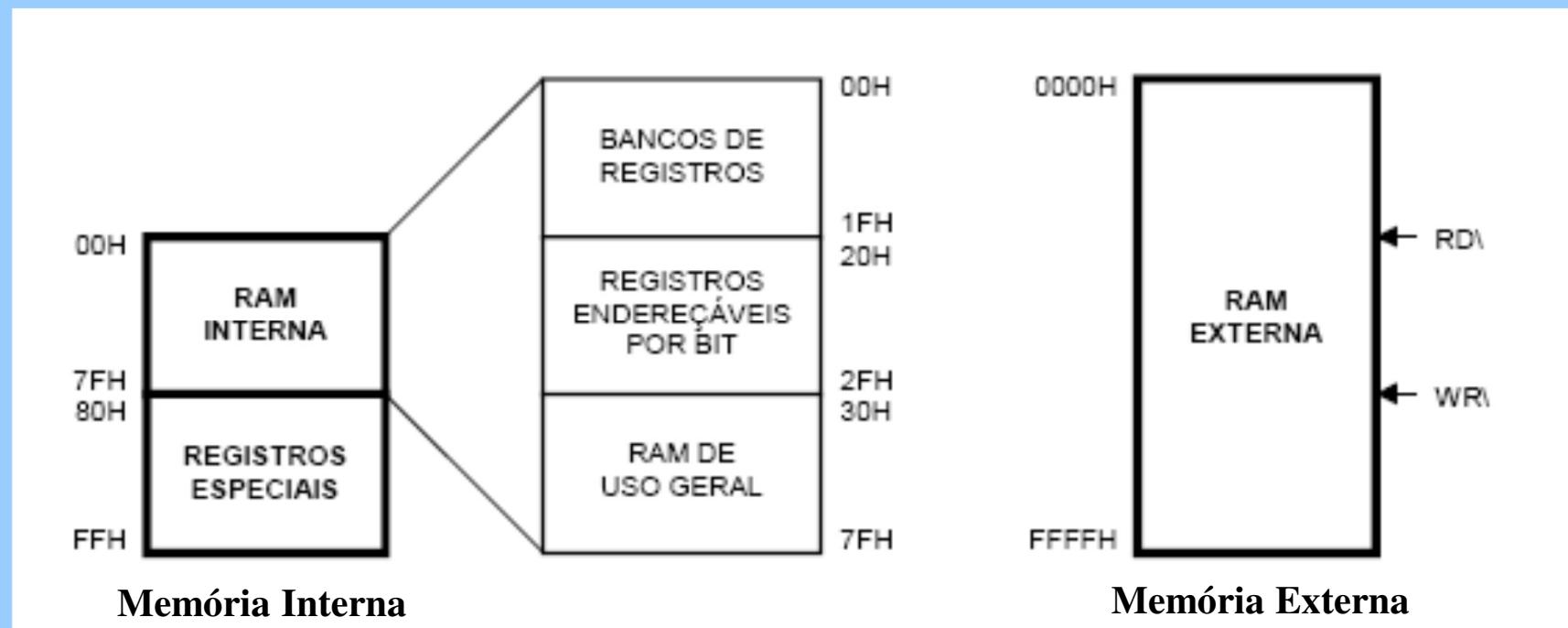
# 10. Organização de Memória no 8051

## Memória de Programa:



# 10. Organização de Memória no 8051

## Memória de Dados:



# 11. Registros Internos no 8051

## Bancos de Registradores:

Banco	PSW:RS1	PSW:RS0	Endereço	Registro
0	0	0	00H	R0
			01H	R1
			02H	R2
			03H	R3
			04H	R4
			05H	R5
			06H	R6
			07H	R7
1	0	1	08H	R0'
			09H	R1'
			0AH	R2'
			0BH	R3'
			0CH	R4'
			0DH	R5'
			0EH	R6'
			0FH	R7'
2	1	0	10H	R0"
			11H	R1"
			12H	R2"
			13H	R3"
			14H	R4"
			15H	R5"
			16H	R6"
			17H	R7"
3	1	1	18H	R0\
			19H	R1\
			1AH	R2\
			1BH	R3\
			1CH	R4\
			1DH	R5\
			1EH	R6\
			1FH	R7\

# 12. Registros Internos no 8051

## Registros de Bits Endereçáveis:

Registro	Endereços Individuais dos Bits								
	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
20H	07H	06H	05H	04H	03H	02H	01H	00H	
21H	0FH	0EH	0DH	0CH	0BH	0AH	09H	08H	
22H	17H	16H	15H	14H	13H	12H	11H	10H	
23H	1FH	1EH	1DH	1CH	1BH	1AH	19H	18H	
24H	27H	26H	25H	24H	23H	22H	21H	20H	
25H	2FH	2EH	2DH	2CH	2BH	2AH	29H	28H	
26H	37H	36H	35H	34H	33H	32H	31H	30H	
27H	3FH	3EH	3DH	3CH	3BH	3AH	39H	38H	
28H	47H	46H	45H	44H	43H	42H	41H	40H	
29H	4FH	4EH	4DH	4CH	4BH	4AH	49H	48H	
2AH	57H	56H	55H	54H	53H	52H	51H	50H	
2BH	5FH	5EH	5DH	4CH	5BH	5AH	59H	58H	
2CH	67H	66H	65H	64H	63H	62H	61H	60H	
2DH	6FH	6EH	6DH	6CH	6BH	6AH	69H	68H	
2EH	77H	76H	75H	74H	73H	72H	71H	70H	
2FH	7FH	7EH	7DH	7CH	7BH	7AH	79H	78H	

# 12. Registros Internos no 8051

Registros de  
Funções  
Especiais:

High 128 bytes of Internal RAM (SFR)							
F8h							
F0h	B						
E8h							
E0h	ACC						
D8h							
D0h	PSW						
C8h							
C0h							
B8h	IP						
B0h	P3						
A8h	JF						
A0h	P2						
98h	SCON						
90h	P1	SBUF					
88h	TCON	TMOD	TL0	TL1	TH0	TH1	
80h	P0	SP	DPL	DPH			PCON
	0	1	2	3	4	5	6
							7

# 12. Registros Internos no 8051

## Registros de Funções Especiais Bits Endereçáveis:

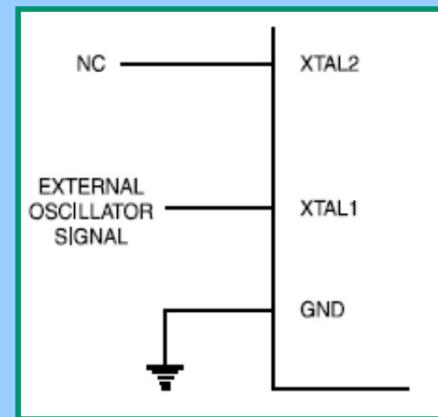
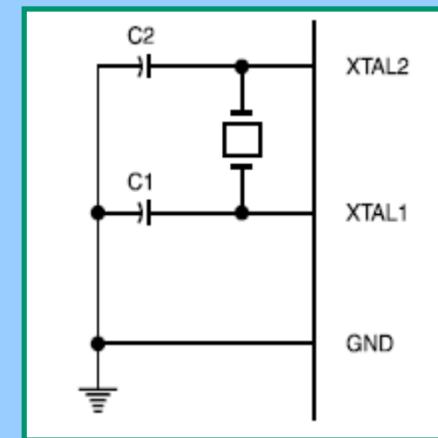


Registro	Endereços Individuais dos Bits							
	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
P0	87H	86H	85H	84H	83H	82H	81H	80H
P1	97H	96H	95H	94H	93H	92H	91H	90H
P2	A7H	A6H	A5H	A4H	A3H	A2H	A1H	A0H
P3	B7H	B6H	B5H	B4H	B3H	B2H	B1H	B0H
A	E7H	E6H	E5H	E4H	E3H	E2H	E1H	E0H
B	F7H	F6H	F5H	F4H	F3H	F2H	F1H	F0H
TCON	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
	8FH	8EH	8DH	8CH	8BH	8AH	89H	88H
SCON	SM1	SM2	SM3	REN	TB8	RB8	T1	R1
	9FH	9EH	9DH	9CH	9BH	9AH	99H	98H
IE	EA			ES	ET1	EX1	ET0	EX0
	AFH			ACH	ABH	AAH	A9H	A8H
IP				PS	PT1	PX1	PT0	PX0
				BCH	BBH	BAH	B9H	B8H
PSW	CY	AC	F0	RS1	RS0	OV		P
	D7H	D6H	D5H	D4H	D3H	D2H		D0H

# 13. Clock no 8051

---

- A freqüência de operação do 8051 pode chegar a um máximo de 8, 10, 12 MHz na versão básica ou 16, 24, 33MHz em versões avançadas;
- Ciclos de máquina do 8051 têm a duração de 12 ciclos de clock;
- Versões otimizadas da Dallas/Maxim (89C420 e similares) os ciclos de máquina levam apenas 4 ciclos de clock (3 x 8051 padrão).



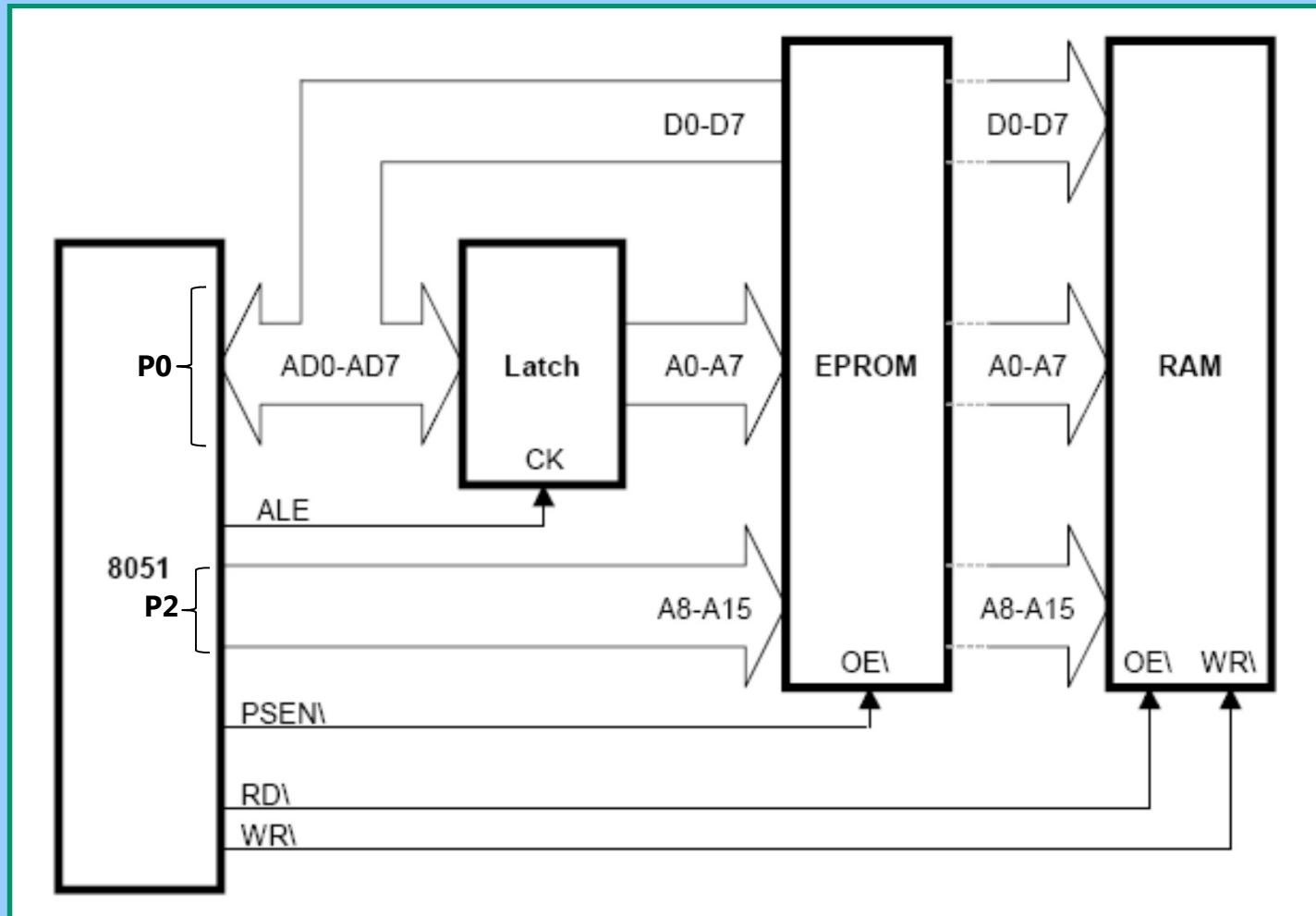
# 14. Reset no 8051

---

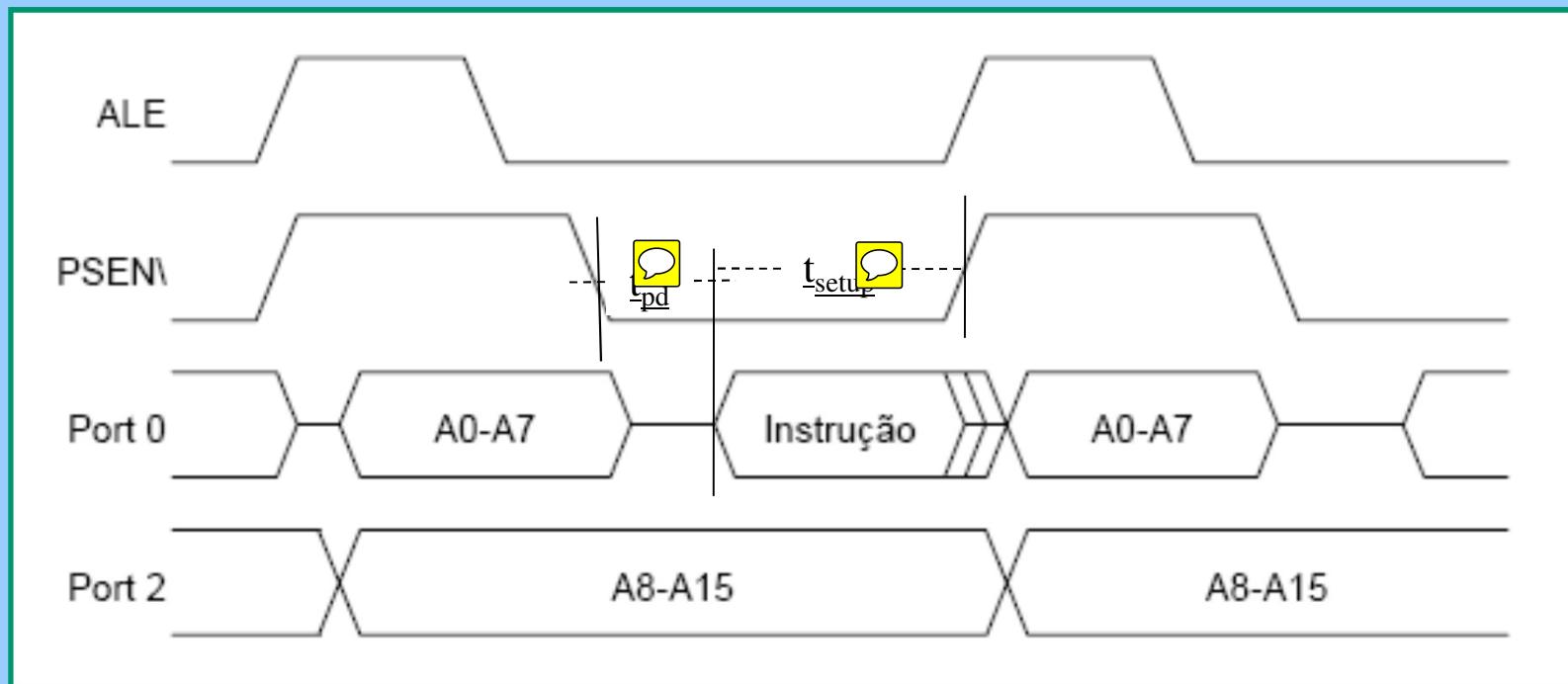
Condição após o Reset:

- Os registros A, B, PSW, DPTR, PC e todos os registros dos temporizadores/contadores de eventos são zerados;
- O registro SP é carregado com o valor 07H;
- Os Ports P0, P1, P2 e P3 são carregados com FFH;
- O registro SCON é zerado e o registro SBUF fica com valor indeterminado;
- O registro PCON tem apenas o seu bit mais significativo zerado;
- Os registros IE e IP são carregados com o valor XXX00000B, (X=indeterminado);
- A RAM interna não tem o seu conteúdo afetado pelo reset.

# 15. Memória Externa de Programa e de Dados



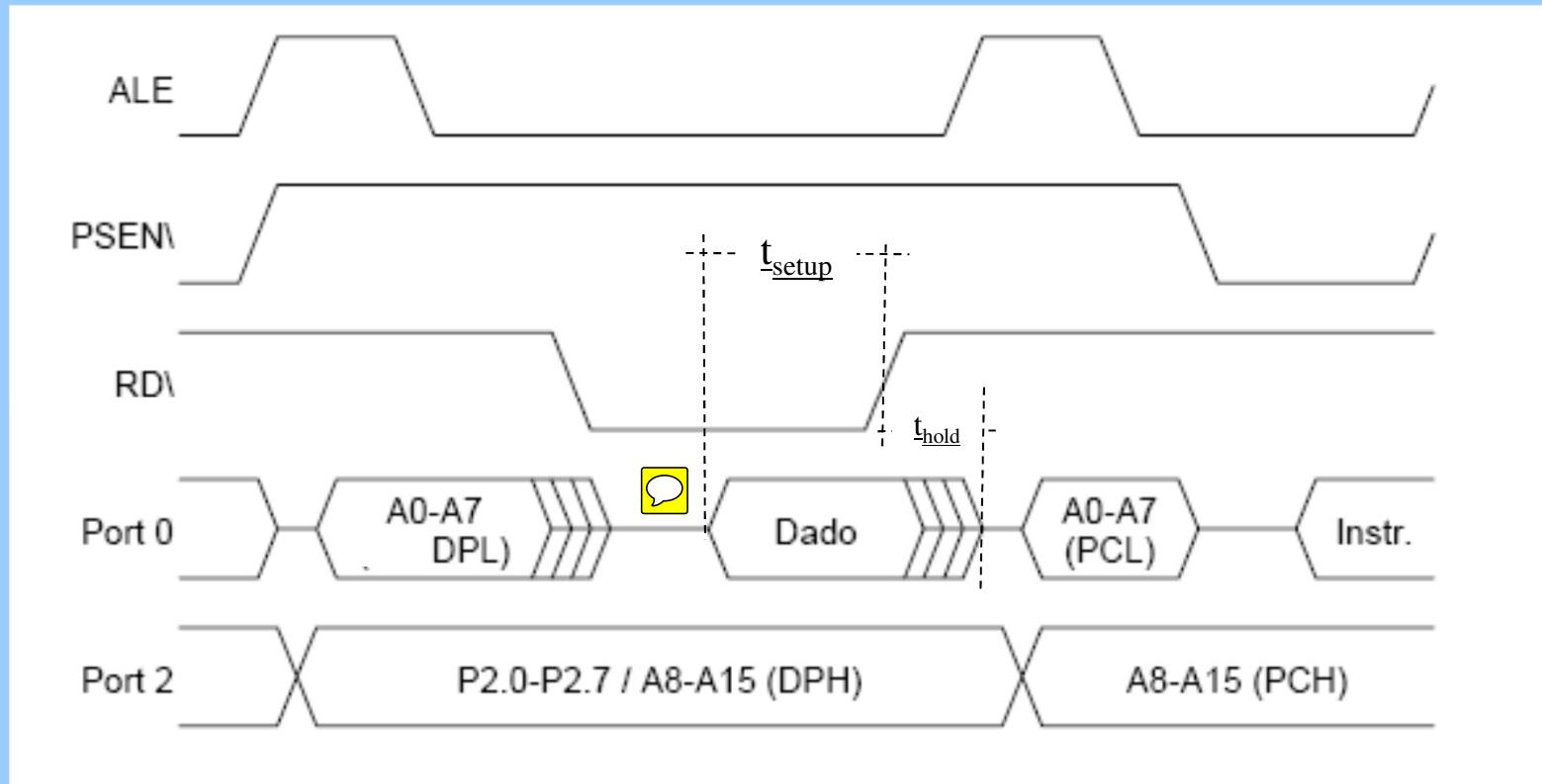
# 15. Memória Externa de Programa e de Dados



**Ciclo de Leitura na Memória de Programa Externa**

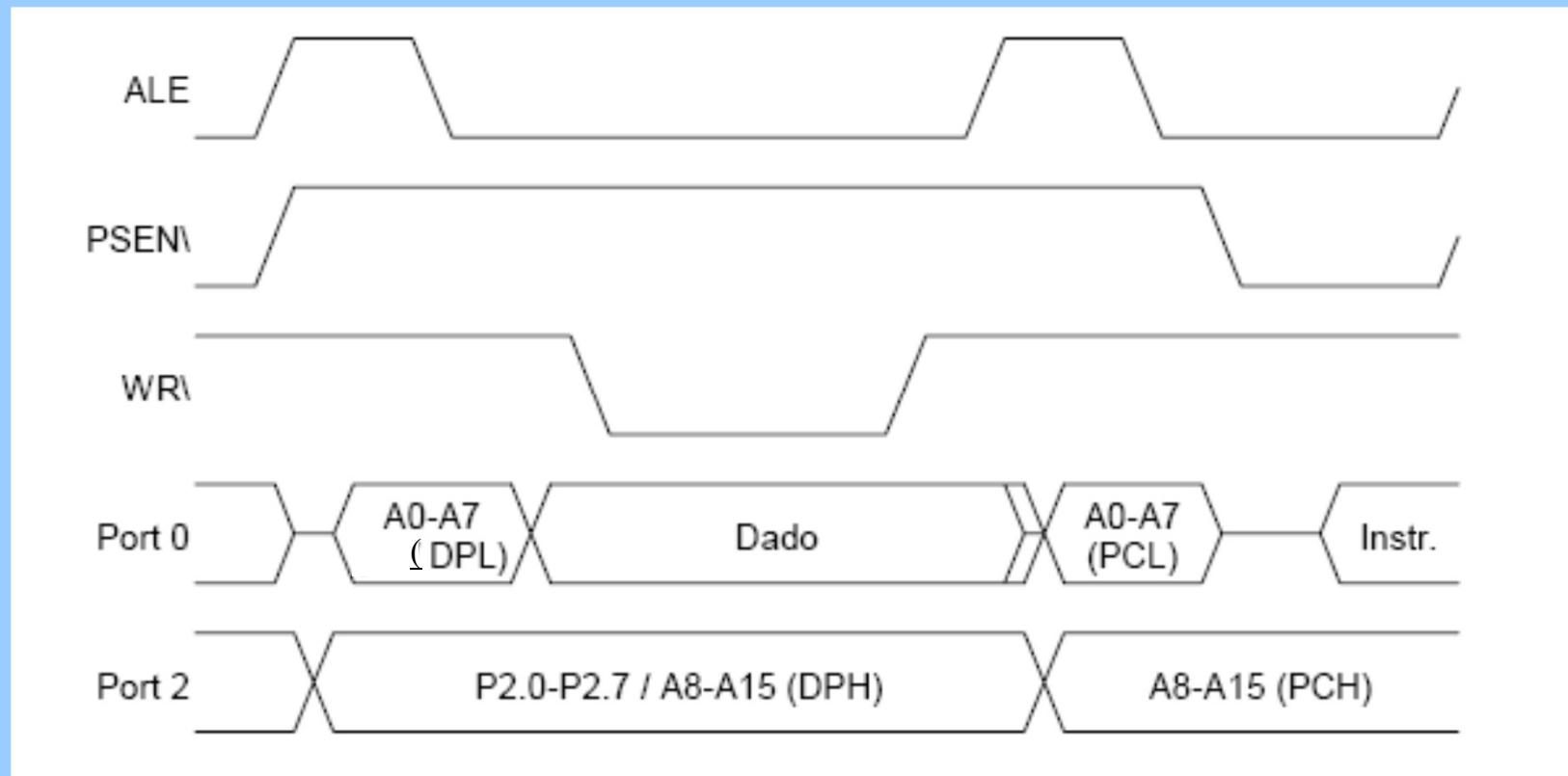


## 15. Memória Externa de Programa e de Dados



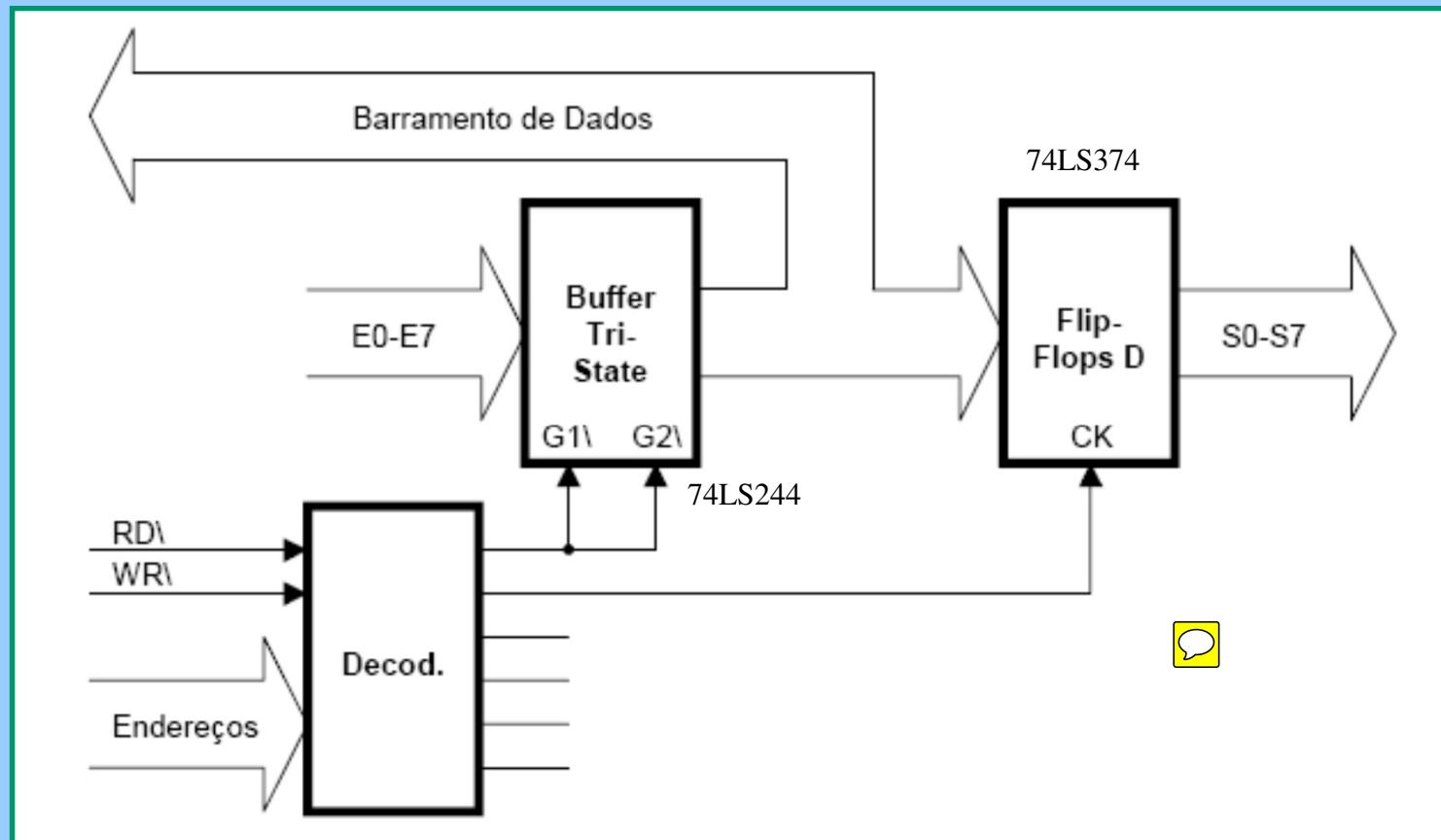
**Ciclo de Leitura na Memória de Dados Externa**

# 15. Memória Externa de Programa e de Dados



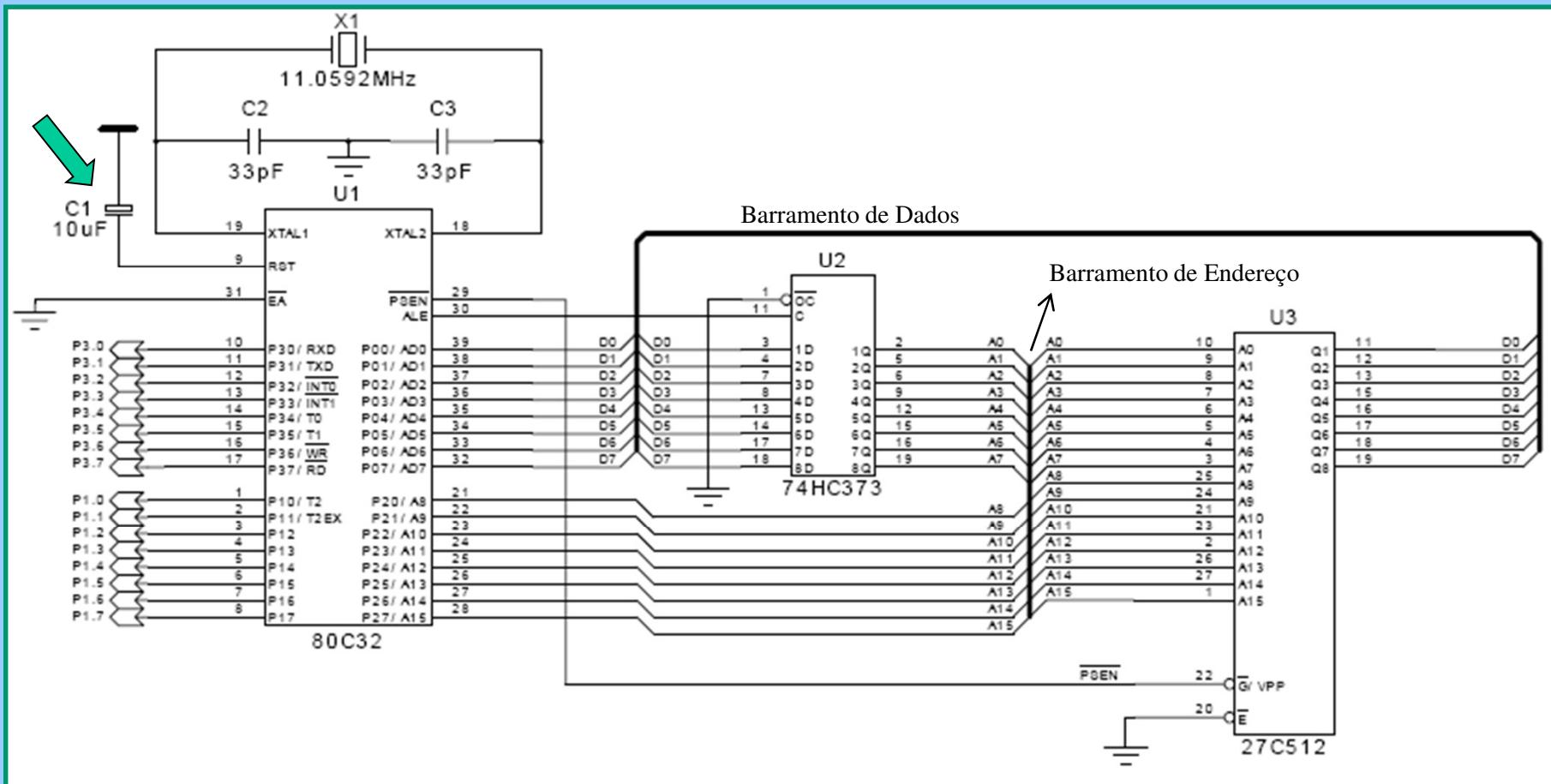
**Ciclo de Escrita na Memória de Dados Externa**

# 16. Expansão de Entradas e Saídas I/O



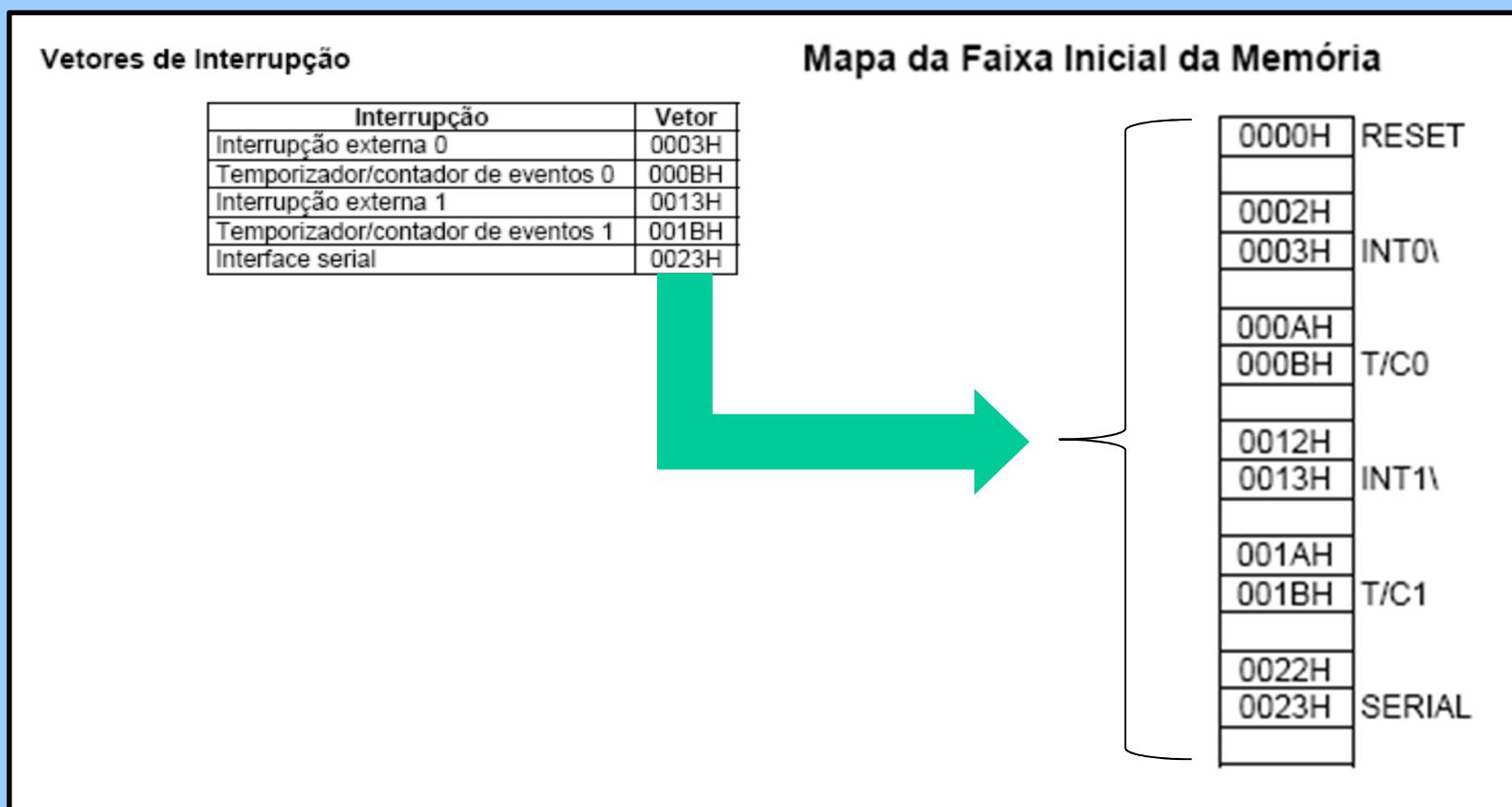
Expansão de I/O

# 17. Sistema mínimo baseado em 8051



Sistema Mínimo com ROM Externa

# 18. Interrupções no 8051



# 19. Programação no 8051

---

- Código de instruções de 8 bits;
- As instruções podem ter tamanho variável de 1 a 3 bytes;

Ex. Mov A, Rn (1 byte) (12 clocks) (1110 1rrr)

Mov A, #dado (2 bytes) (12 clocks) (0111 0100 byte)

Mov DPTR, #dado 16 bits (3 bytes) (24 clocks) (1001 0000 byte  
high, byte low);

- Instruções de apenas 1 byte não requerem operandos ou então já os possuem embutidos nos 8 bits do código da instrução;
- Instruções de 2 ou 3 bytes possuem um ou dois bytes adicionais, correspondentes ao(s) operando(s);
- Para cada instrução o tempo de execução pode ser de um a quatro ciclos de máquina (cada ciclo de máquina corresponde a 12 períodos de clock), conforme a complexidade da operação a ser executada e o número de bytes extras a serem buscados na memória de programa;

# 19. Programação no 8051

---

- Na programação em linguagem Assembly (baixo nível) são utilizados símbolos mnemônicos (Mov, sjmp, setb, clr, cjne, movx, movc, swap, etc);
- A codificação do programa propriamente dita é realizada por um aplicativo montador (assembler);
- Linguagem de programação de alto nível como a linguagem C utiliza-se um compilador, que é o responsável pela tradução para o Assembly e posterior montagem do código.

# 16. Programação no 8051

---

## MODOS DE ENDEREÇAMENTO:

- **Modo Imediato:**

Permite carregar constantes em posições da memória de dados interna ou registradores. Nas instruções da família 8051, as constantes são precedidas pelo símbolo #. Ex. **Mov A, #05h; Mov 10h,#33h**

- **Modo Direto**

Essa forma de endereçamento utiliza diretamente o endereço da posição da memória de dados interna na qual será efetuada a operação. Ex. **Mov A, 30h ; ANL A, 30h;**

- **Modo Indireto**

Os registros R0, R1 e DPTR atuam como ponteiros nessas instruções, sendo R0 e R1 ponteiros para endereços de 8 bits e DPTR um ponteiro para endereços de 16 bits. Nas instruções da família 8051, os ponteiros são precedidos pelo símbolo @.

Ex. **Mov A,@Ri ; Mov @Ri,A ; Movx A,@dptr ; Movx @dptr,A**

- **Modo Registro**

Permite o acesso aos registros R0 a R7 do banco de registros em uso (working registers), definido pelos bits de controle RS1 e RS0. Ex. **Mov R0, #03h.**

- **Modo Indexado**

Quando o endereço efetivo é a soma do acumulador e um registro de 16 bits ( DPTR ou PC ).Ex. **Movc @A+DPTR**

# 19. Programação no 8051

---

## CONJUNTO DE INSTRUÇÕES

- Transferência de Dados:

MOV, MOVC, MOVX, PUSH, POP, XCH e XCHD;

**MOV <destino>, <fonte>**

Ex:      MOV A,B  
              MOVC A, @A+DPTR  
              MOVX A,@DPTR  
              PUSH A  
              POP A  
              XCH A,R1  
              XCH A,@R1  
              XCHD A,@R2

# 19. Programação no 8051

---

- Operações Lógicas;

Ex: **clr A, cpl A, swap A**

**ANL <operando1>, <operando2>**

**ORL <operando1>, <operando2>**

**XRL <operando1>, <operando2>**

**rr A, rrc A**

**rl A, rlc A**

- Operações Aritméticas:

**ADD A, <operando>**

**ADDC A, <operando>**

**SUBB A, <operando>**

**INC <operando>**

**DEC <operando>**

**MUL AB**

**DIV AB**

**DA A**

# 19. Programação no 8051

---

- Manipulação de Variáveis Booleanas (Bits):

**CLR <operando>**  
**SETB <operando>**  
**CPL <operando>**  
**ANL C, <operando>**  
**ORL C, <operando>**  
**MOV C, <fonte>**  
**MOV <destino>, C**

- Controle de Programa:

**LJMP <end16>**  
**AJMP <end11>**  
**SJMP <endereço>**  
**JMP @A+DPTR**  
**LCALL end16**  
**ACALL end11**  
**RET**  
**RETI**

# 19. Programação no 8051

---

**JNZ <endereço>**

**JZ <endereço>**

**JNC <endereço>**

**JC <endereço>**

**JNB <operando>, <endereço>**

**JB <operando>, <endereço>**

**JBC <operando>, <endereço>**

**CJNE <operando1>, <operando2>, <endereço>**

**DJNZ <operando>, <endereço>**

**NOP**

# 19. Programação no 8051

## Conjunto de Instruções Completo

	0_	1_	2_	3_	4_	5_	6_	7_
_0	NOP	JBC bit,rel	JB bit,rel	JNB bit,rel	JC rel	JNC rel	JZ rel	JNZ rel
_1	AJMP end11	ACALL end11	AJMP end11	ACALL end11	AJMP end11	ACALL end11	AJMP end11	ACALL end11
_2	LJMP adr16	LCALL adr16	RET	RETI	ORL end8,A	ANL end8,A	XRL end8,A	ORL C.bit
_3	RR A	RRC A	RL A	RLC A	ORL end8,#dt8	ANL end8,#dt8	XRL end8,#dt8	JMP @A+DPTR
_4	INC A	DEC A	ADD A,#dt8	ADDC A,#dt8	ORL A,#dt8	ANL A,#dt8	XRL A,#dt8	MOV A,#dt8
_5	INC end8	DEC end8	ADD A,end8	ADDC A,end8	ORL A,end8	ANL A,end8	XRL A,end8	MOV end8,#dt8
_6	INC @R0	DEC @R0	ADD A,@R0	ADDC A,@R0	ORL A,@R0	ANL A,@R0	XRL A,@R0	MOV @R0,#dt8
_7	INC @R1	DEC @R1	ADD A,@R1	ADDC A,@R1	ORL A,@R1	ANL A,@R1	XRL A,@R1	MOV @R1,#dt8
_8	INC R0	DEC R0	ADD A,R0	ADDC A,R0	ORL A,R0	ANL A,R0	XRL A,R0	MOV R0,#dt8
_9	INC R1	DEC R1	ADD A,R1	ADDC A,R1	ORL A,R1	ANL A,R1	XRL A,R1	MOV R1,#dt8
_A	INC R2	DEC R2	ADD A,R2	ADDC A,R2	ORL A,R2	ANL A,R2	XRL A,R2	MOV R2,#dt8
_B	INC R3	DEC R3	ADD A,R3	ADDC A,R3	ORL A,R3	ANL A,R3	XRL A,R3	MOV R3,#dt8
_C	INC R4	DEC R4	ADD A,R4	ADDC A,R4	ORL A,R4	ANL A,R4	XRL A,R4	MOV R4,#dt8
_D	INC R5	DEC R5	ADD A,R5	ADDC A,R5	ORL A,R5	ANL A,R5	XRL A,R5	MOV R5,#dt8
_E	INC R6	DEC R6	ADD A,R6	ADDC A,R6	ORL A,R6	ANL A,R6	XRL A,R6	MOV R6,#dt8
_F	INC R7	DEC R7	ADD A,R7	ADDC A,R7	ORL A,R7	ANL A,R7	XRL A,R7	MOV R7,#dt8

# 19. Programação no 8051

## Conjunto de Instruções Completo

	<b>8_</b>	<b>9_</b>	<b>A_</b>	<b>B_</b>	<b>C_</b>	<b>D_</b>	<b>E_</b>	<b>F_</b>
<b>_0</b>	SJMP rel	MOV DPTR,#dt16	ORL C,/bit	ANL C,/bit	PUSH end8	POP end8	MOVX A,@DPTR	MOVX @DPTR,A
<b>_1</b>	AJMP end11	ACALL end11	AJMP end11	ACALL end11	AJMP end11	ACALL end11	AJMP end11	ACALL end11
<b>_2</b>	ANL C,bit	MOV bit,C	MOV C,bit	CPL bit	CLR bit	SETB bit	MOVX A,@R0	MOVX @R0,A
<b>_3</b>	MOVC A,@A+PC	MOVC A,@A+DPTR	INC DPTR	CPL C	CLR C	SETB C	MOVX A,@R1	MOVX @R1,A
<b>_4</b>	DIV AB	SUBB A,#dt8	MUL AB	CJNE A,#dt8,rel	SWAP A	DA A	CLR A	CPL A
<b>_5</b>	MOV end8,end8	SUBB A,end8	-	CJNE A,end8,rel	XCH A,end8	DJNZ end8,rel	MOV A,end8	MOV end8,A
<b>_6</b>	MOV end8,@R0	SUBB A,@R0	MOV @R0,end8	CJNE @R0,#dt8,rel	XCH A,@R0	XCHD A,@R0	MOV A,@R0	MOV @R0,A
<b>_7</b>	MOV end8,@R1	SUBB A,@R1	MOV @R1,end8	CJNE @R1,#dt8,rel	XCH A,@R1	XCHD A,@R1	MOV A,@R1	MOV @R1,A
<b>_8</b>	MOV end8,R0	SUBB A,R0	MOV R0,end8	CJNE R0,#dt8,rel	XCH A,R0	DJNZ R0,rel	MOV A,R0	MOV R0,A
<b>_9</b>	MOV end8,R1	SUBB A,R1	MOV R1,end8	CJNE R1,#dt8,rel	XCH A,R1	DJNZ R1,rel	MOV A,R1	MOV R1,A
<b>_A</b>	MOV end8,R2	SUBB A,R2	MOV R2,end8	CJNE R2,#dt8,rel	XCH A,R2	DJNZ R2,rel	MOV A,R2	MOV R2,A
<b>_B</b>	MOV end8,R3	SUBB A,R3	MOV R3,end8	CJNE R3,#dt8,rel	XCH A,R3	DJNZ R3,rel	MOV A,R3	MOV R3,A
<b>_C</b>	MOV end8,R4	SUBB A,R4	MOV R4,end8	CJNE R4,#dt8,rel	XCH A,R4	DJNZ R4,rel	MOV A,R4	MOV R4,A
<b>_D</b>	MOV end8,R5	SUBB A,R5	MOV R5,end8	CJNE R5,#dt8,rel	XCH A,R5	DJNZ R5,rel	MOV A,R5	MOV R5,A
<b>_E</b>	MOV end8,R6	SUBB A,R6	MOV R6,end8	CJNE R6,#dt8,rel	XCH A,R6	DJNZ R6,rel	MOV A,R6	MOV R6,A
<b>_F</b>	MOV end8,R7	SUBB A,R7	MOV R7,end8	CJNE R7,#dt8,rel	XCH A,R7	DJNZ R7,rel	MOV A,R7	MOV R7,A

# 19. Programação no 8051

---

## LINGUAGEM ASSEMBLY

- **Pseudo-instruções:**

**ORG endereço**

Define a origem (endereço) do programa, subrotina, tabela, etc.

**DB dado, [...]**

Define constantes (mensagens, tabelas, etc.) a serem gravadas na memória de programa.

Ex. ORG 03h  
sjmp trata\_int0  
:

- **Definições:**

São declarações que atribuem nomes a constantes, permitindo referências ao longo do programa.

# 19. Programação no 8051

---

## rótulo EQU valor

Ex: Hora

EQU 30h

Tempo\_de\_Espera

EQU 15

Atribui o nome do rótulo à constante especificada (valor).

## Rótulos ou Label:

São nomes atribuídos aos endereços de rotinas de tratamento de interrupções, sub-rotinas, tabelas ou simplesmente locais de desvio do fluxo do programa utilizados em estruturas de programação (decisões e repetições), facilitando referências ao longo do programa.

## Comentários:

Qualquer texto digitado após ponto-e-vírgula (;) é desprezado pelo assembler durante a montagem do programa. Dessa forma, pode-se fazer uso de comentários úteis para a documentação e melhor entendimento do programa.

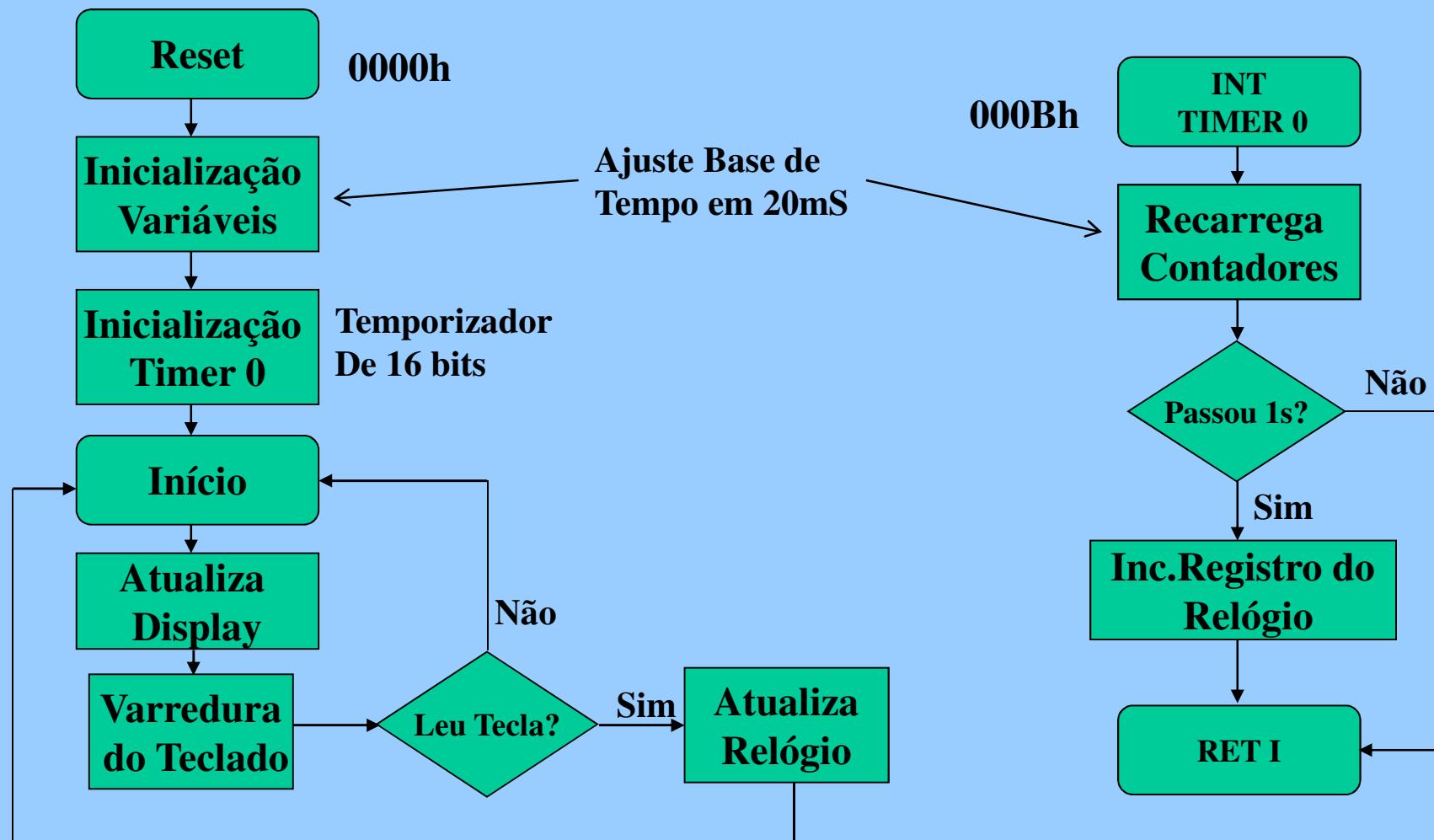
# 19. Programação no 8051

---

## Estrutura de um Programa em Assembly

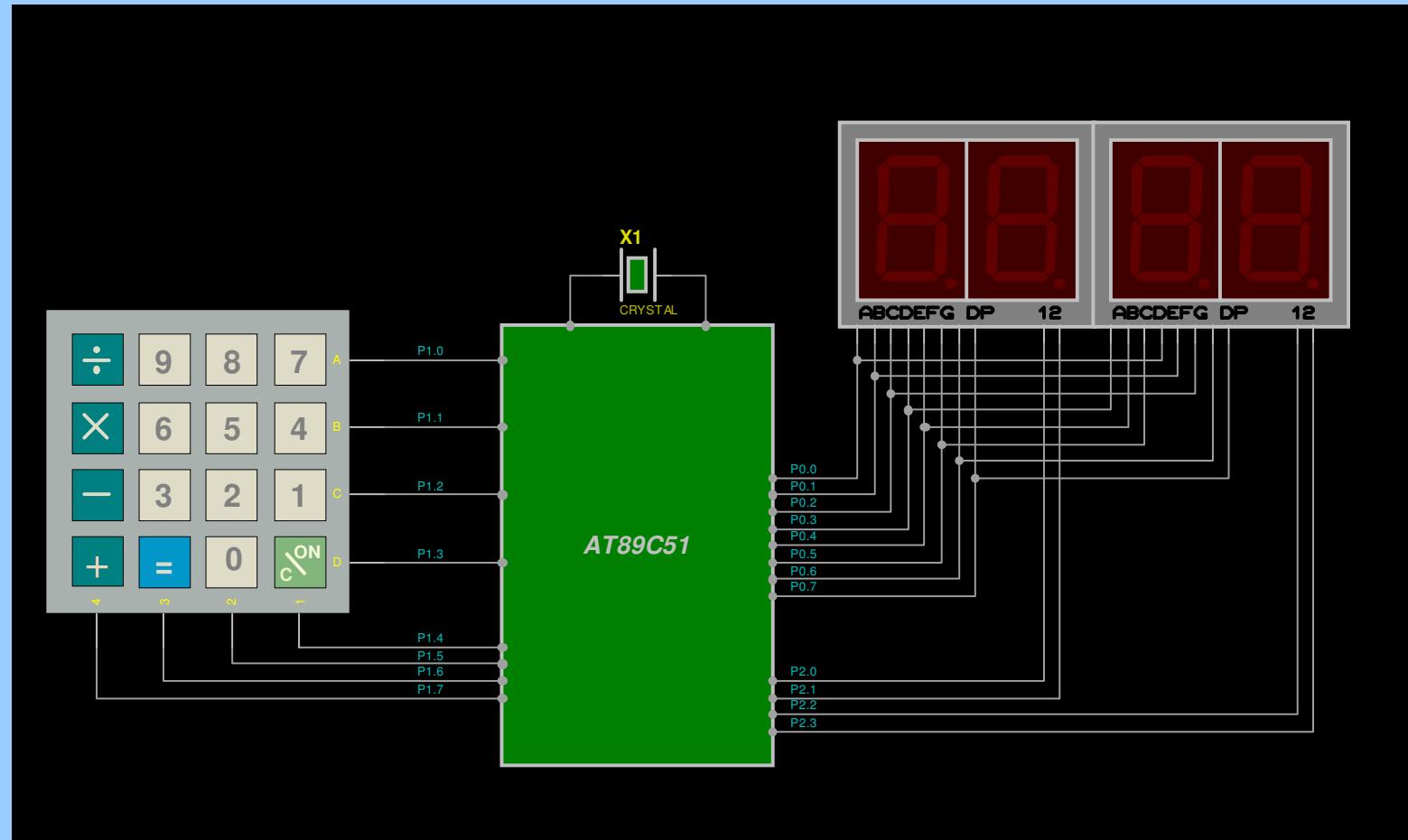
- Exemplo de estruturação de programas assembly (ver arquivos exemplos);
- Editor de texto (Context);
- Montador (x8051 e linkador XLINK);
- Simulação, DEBUG, e Emulação (PROTEUS).

# 20. Relógio Digital

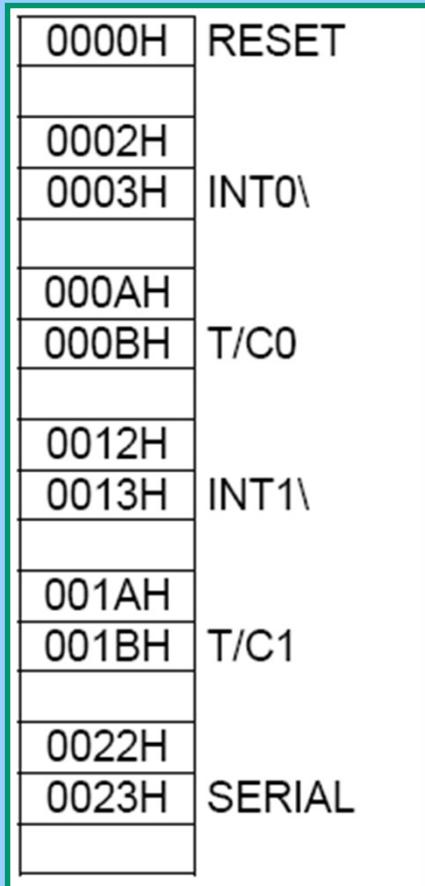


# 21. Diagrama de Blocos do Circuito

0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111



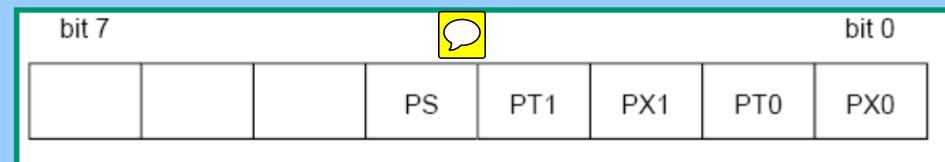
# 22. Estudo das Interrupções



Vetores de Interrupção



Registro IE



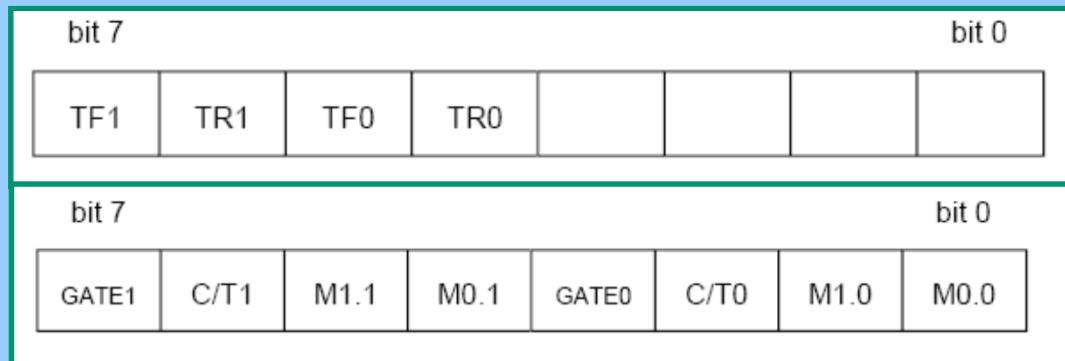
Registro IP



Registro TCON



## 23. Registros Especiais de Controle dos T/C



- **GATE 0/1:**

**Nível 0 – a contagem do T/C será habilitada se apenas o bit de controle correspondente (TR1 ou TR0) no registro TCON estiver em nível lógico 1.**

**Nível 1 – a contagem do T/C será habilitada se o bit de controle correspondente (TR1 ou TR0) no registro TCON estiver em nível lógico 1 e o pino (INT1\ ou INT0\ ) também estiver em nível lógico 1.**

## 24. Registros Especiais de Controle dos T/C

- C/T1 – Counter / Timer 1; C/T0 – Counter / Timer 0

Nível 0 – o T/C correspondente funciona como temporizador (sinal de contagem interno – freqüência de clock dividida por 12);

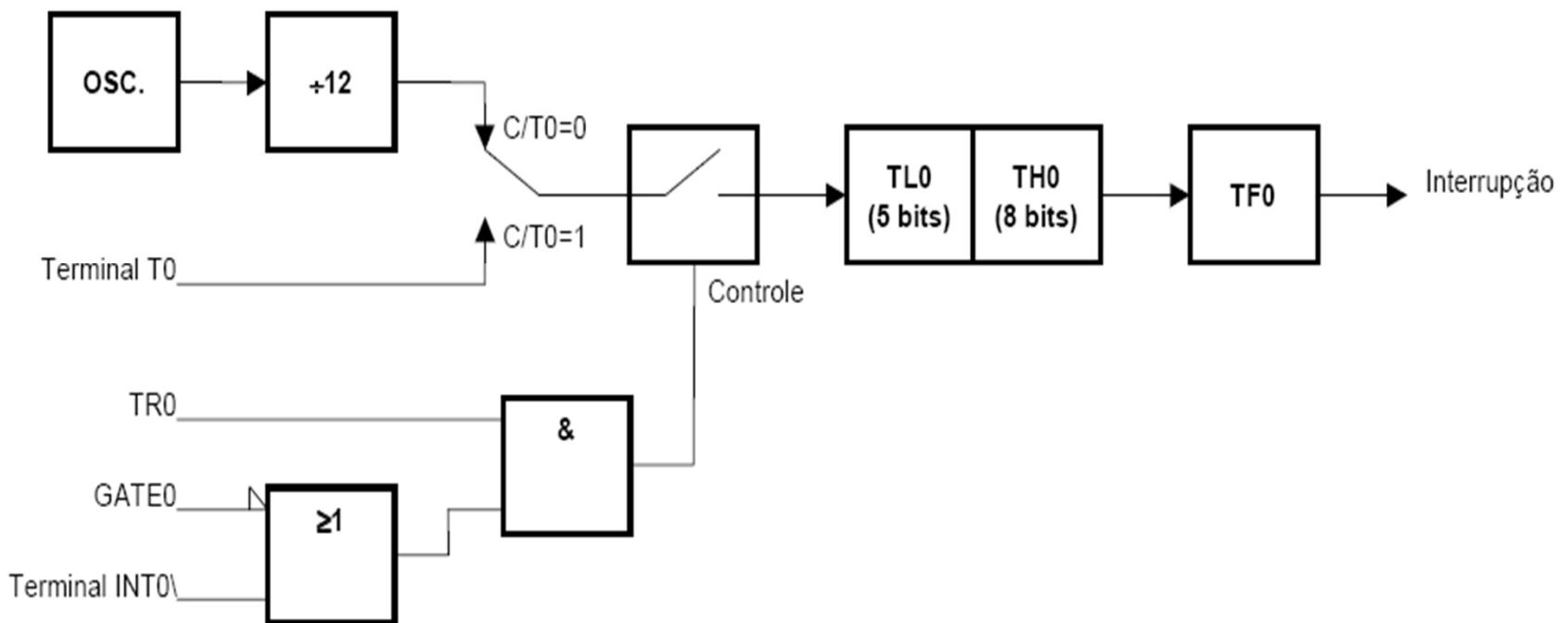
Nível 1 – o T/C correspondente funciona como contador (sinal de contagem externo – terminal T1 ou T0 do Port3);

- M1.1/M0.1 – Mode T/C1; M0.1/M0.0 – Modos de operação do T/C0

<b>M1.<math>\times=0</math> e M0.<math>\times=0</math></b>	T/C de 8 bits com divisor de freqüência (prescaler) de 5 bits. 5 BMS TLx = prescaler 2 a 32.
<b>M1.<math>\times=0</math> e M0.<math>\times=1</math></b>	T/C de 16 bits
<b>M1.<math>\times=1</math> e M0.<math>\times=0</math></b>	T/C de 8 bits com recarga automática (THx = reload)
<b>M1.<math>\times=1</math> e M0.<math>\times=1</math></b>	Duplo T/C de 8 bits ( TH0 {TR1 e TF1} e TL0 {TR0 e TF0} )

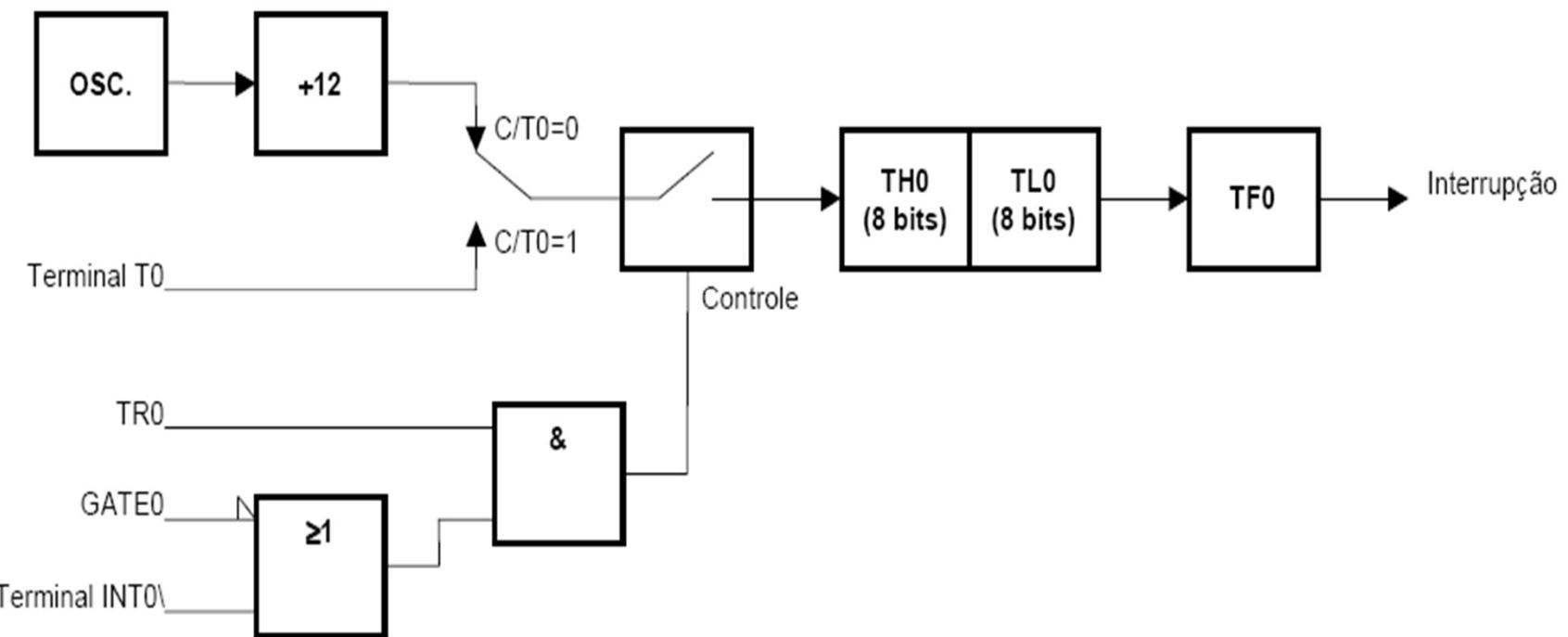
## 24. Registros Especiais de Controle dos T/C

### Operação no Modo 0



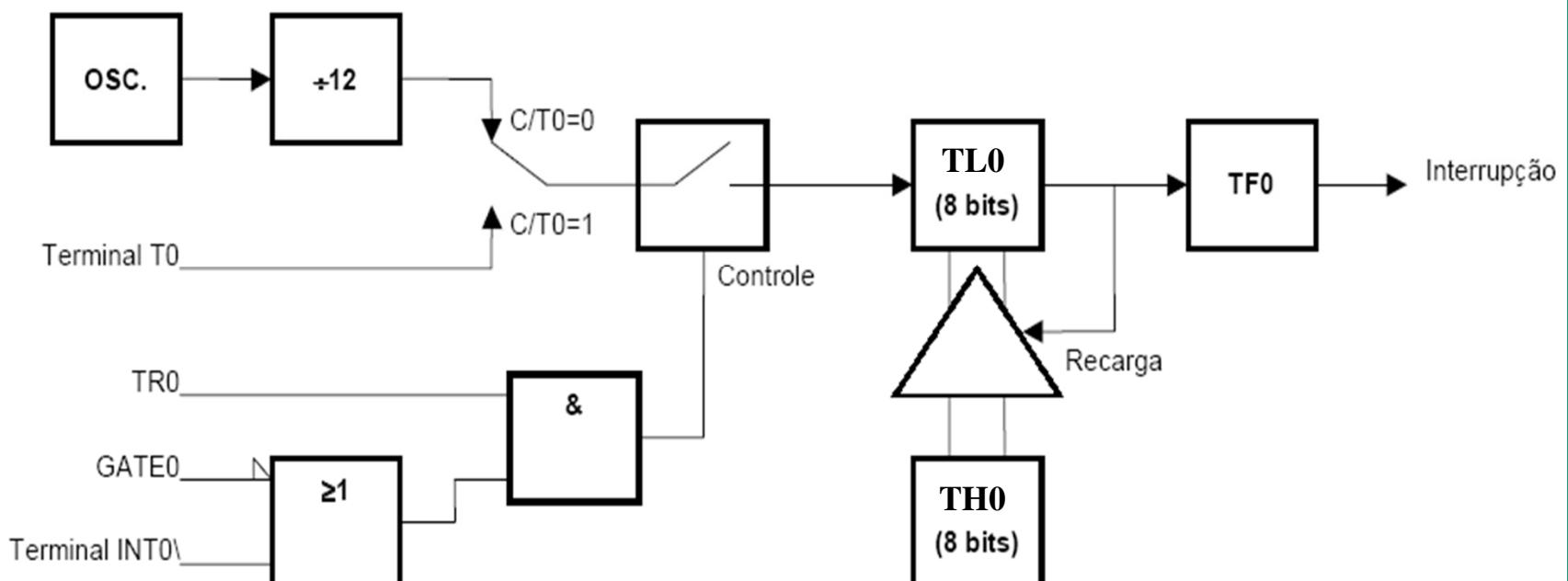
# 24. Registros Especiais de Controle dos T/C

## Operação no Modo 1



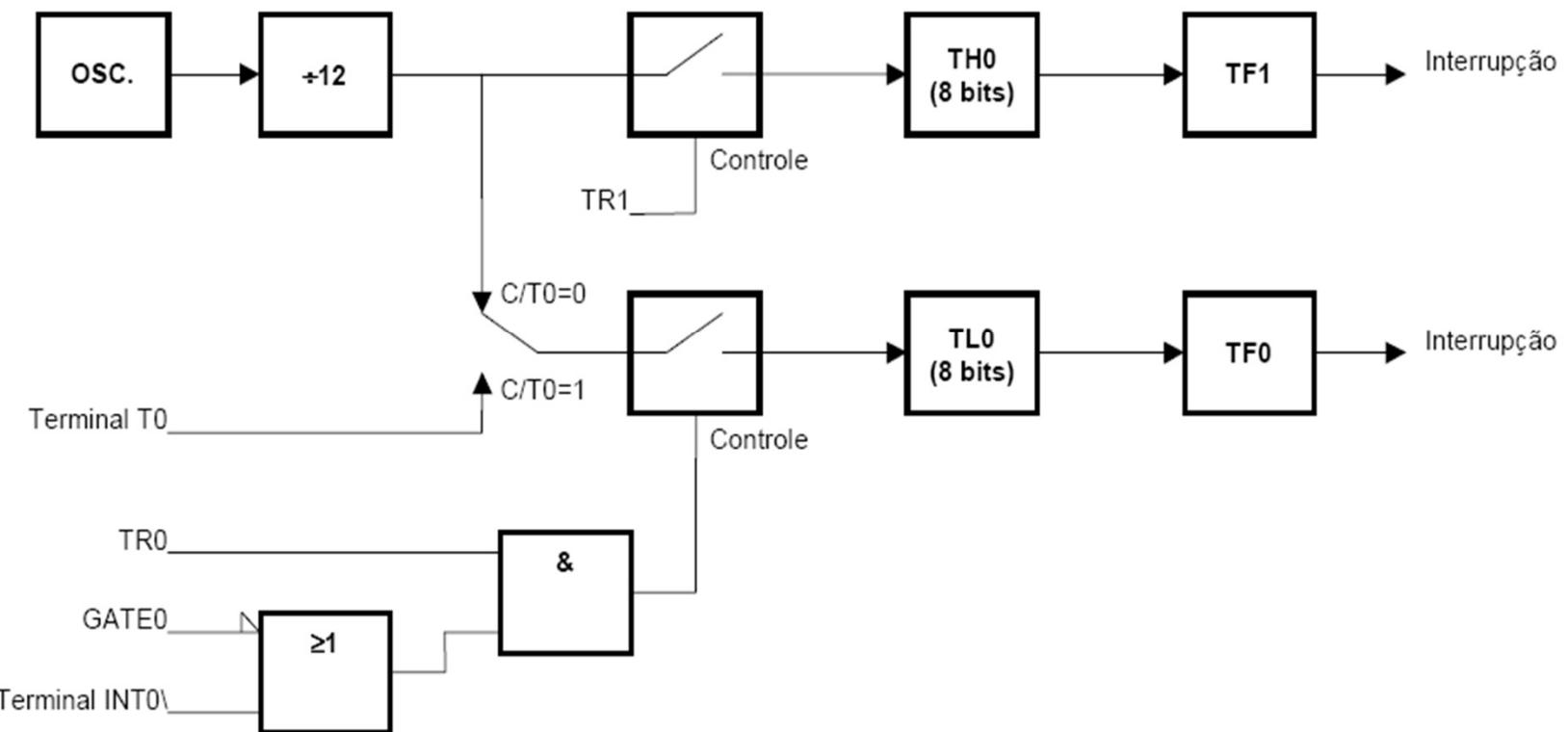
# 24. Registros Especiais de Controle dos T/C

## Operação no Modo 2



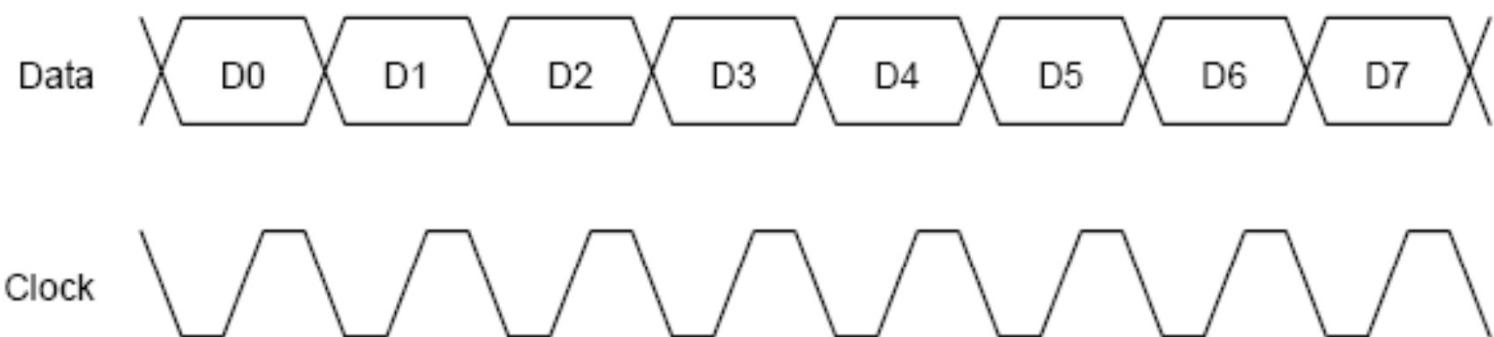
# 24. Registros Especiais de Controle dos T/C

## Operação no Modo 3



# 25. Estudo do Canal Serial do 8051

## Comunicação Serial Síncrona



## Comunicação Serial Assíncrona



# 25. Estudo do Canal Serial do 8051

- **Registro SCON – Serial Control:** Esse registro tem a função de controlar e programar o funcionamento da interface serial do 8051.



## SM0/SM1 – Serial Mode

Esses bits de controle permitem a obtenção de quatro modos de funcionamento distintos para a interface serial.

SM0	SM1	Modo	Taxa de Transmissão e Recepção
0	0	0	Freqüência de clock dividida por 12
0	1	1	Programável
1	0	2	Freqüência de clock dividida por 32 ou 64
1	1	3	Programável

# 25. Estudo do Canal Serial do 8051

---

## **SM2 – Serial Mode**

Possui diferentes funções em cada modo de funcionamento:

- Modo 0: não altera em nada o funcionamento, devendo permanecer no nível lógico 0.
- Modo 1: quando em nível lógico 1 desabilita o pedido de interrupção se for recebido um stop bit inválido.
- Modos 2 e 3: quando em nível lógico 1 habilita a comunicação serial entre vários microcontroladores e desabilita o pedido de interrupção se for recebido um nono bit de dado igual a 0.

# 25. Estudo do Canal Serial do 8051

## Modos de Operação da Interface Serial

- **Modo 0 (SM0=0, SM1=0)**

É o único modo de operação onde ocorre comunicação de dados síncrona half-duplex. A taxa de transmissão e recepção é fixa (freqüência de clock do 8051 dividida por 12).

São transmitidos ou recebidos sempre 8 bits de dados, sendo o menos significativo transmitido primeiramente. Os dados (data) são transmitidos ou recebidos pelo terminal RXD e o sinal de sincronismo (clock) pelo terminal TXD.

- **Modo 1 (SM0=0, SM1=1)**

Nesse modo de operação e nos dois modos seguintes ocorre comunicação de dados assíncrona full-duplex com taxa de transmissão e recepção programável. A recepção dos dados é feita pelo terminal RXD e a transmissão pelo terminal TXD.

No Modo 1 cada pacote de dados transmitido ou recebido possui 10 bits, sendo um start bit seguido de 8 bits de dados e um stop bit. Na recepção o stop bit vai para o bit RB8 do registro SCON.

# 25. Estudo do Canal Serial do 8051

---

- **Modo 2 (SM0=1, SM1=0)**

No Modo 2 cada pacote de dados transmitido ou recebido possui 11 bits, sendo um start bit seguido de 8 bits de dados, um nono bit e um stop bit.

Na transmissão o nono bit a ser transmitido deverá estar no bit TB8 do registro SCON. Na recepção o nono bit vai para o bit RB8.

A taxa de transmissão e recepção pode ser programada para 1/32 ou 1/64 da freqüência de clock do microcontrolador.

- **Modo 3 (SM0=1, SM1=1)**

O Modo 3 possui funcionamento idêntico ao do Modo 2, exceto pela taxa de transmissão que é programável.

# 25. Estudo do Canal Serial do 8051

## Taxa de Transmissão e Recepção Serial

- Modo 0: a taxa é igual a 1/12 da freqüência de clock (fixa).
- Modo 2: a taxa depende apenas do valor do bit SMOD no registro PCON. Para SMOD=0 a taxa é igual a 1/32 da freqüência de clock e para SMOD=1 a taxa é igual a 1/64 da freqüência de clock.
- Modos 1 e 3: a taxa é fornecida pelo T/C1, de maneira que os dados são transmitidos a cada estouro de contagem. Nesse caso o modo de funcionamento mais comum do T/C1 é o Modo 2 (recarga automática). A interrupção do T/C1 deve ser desabilitada.

$$Taxa = \frac{2^{SMOD}}{32} \cdot \frac{f_{clock}}{12.(256 - TH1)}$$

- Registro PCON – Power Control

bit 7					GF1	GF0	PD	IDL	bit 0
SMOD									

# 25. Estudo do Canal Serial do 8051

Crystal Frequency (Mhz)	Timer Reload	Loader Baud Rate	Error %
32.0000	F6	16667	-1.3
	F3	12821	-0.2
	EC	8333	-1.3
29.4912	FC	38400	0.0
	F8	19200	0.0
	F4	12800	0.0
24.5760	F6	12800	0.0
	F5	11636	-1.0
	F4	10667	-1.8
24.0000	F3	9615	-0.2
	F0	7812	-1.7
	E6	4808	-0.2
22.1184	FF	115200	0.0
	FE	57600	0.0
	FD	38400	0.0
20.0000	F8	13021	-1.7
	F0	6510	-1.7
	E8	4340	-1.7
16.0000	FB	16667	-1.3
	F6	8333	-1.3
	F4	6944	-2.4
11.0592	FF	57600	0.0
	FE	28800	0.0
	FD	19200	0.0
1.84320	FF	9600	0.0
	FE	4800	0.0
	FD	3200	0.0

# 25. Estudo do Canal Serial do 8051

---

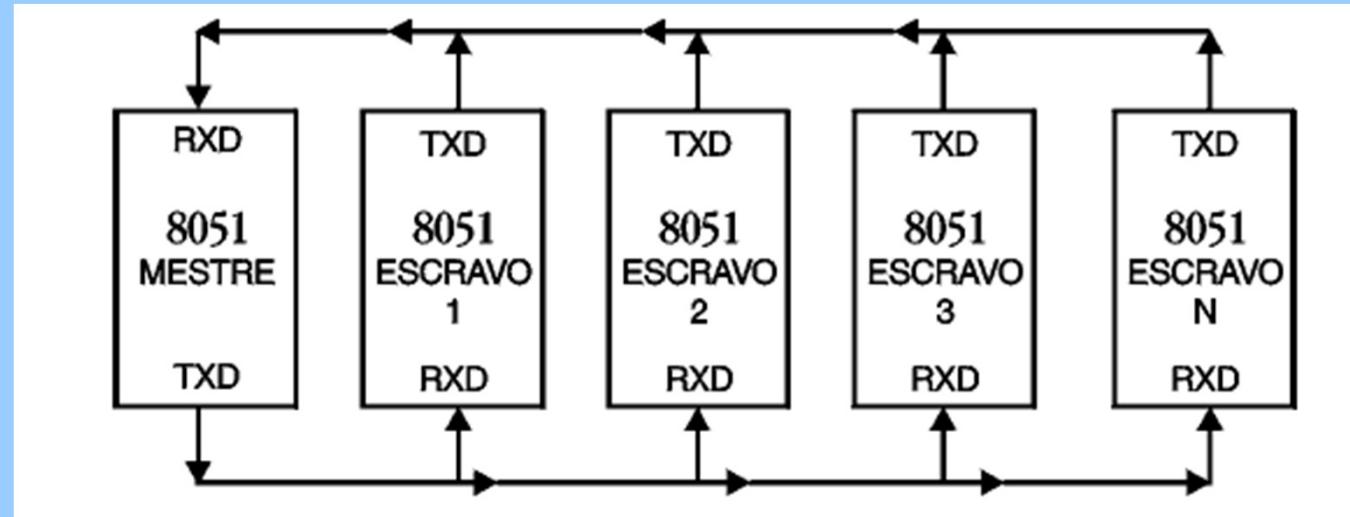
Mestre-Escravo:

Na recepção, se  $SM2 = 1$  e  $RB8 = 1$ , a interrupção do canal serial será atendida. Assim, podemos criar condições para que um 8051 mestre possa enviar dados serialmente para vários 8051 ‘escravos’, da seguinte forma:

- 1 – No início do trabalho, todos os ‘escravos’ estarão com  $SM2 = 1$  (prontos para receber dados, cujo nono bit seja 1);
- 2 – Quando o 8051 mestre desejar enviar dados para alguns dos ‘escravos’;
- 3 – Escreverá um em seu bit TB8;
- 4 – Enviará serialmente o endereço do escravo desejado (8 bits);
- 5 – Como teremos todos os bits dos ‘escravos’ com  $RB8$  em 1, então, todos serão interrompidos para poder verificar se é seu endereço o enviado;
- 6 – O escravo que for selecionado zerará seu bit  $SM2$  e estará preparado para receber os dados, os quais deverão ter agora o nono bit ( $RB8$ ) em 0. Os demais ‘escravos’ permanecerão com  $SM2$  em 1 e, desta forma, não serão mais interrompidos.

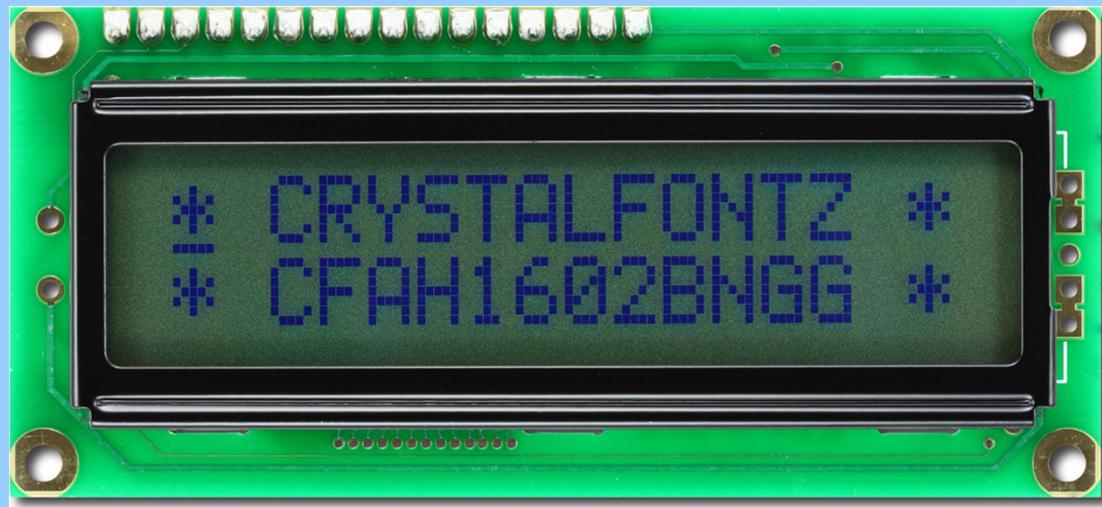
# 25. Estudo do Canal Serial do 8051

---



# 26. Display de Cristal Líquido -LCD

---



- LCD podem ser gráficos ou de caracteres;
- LCD gráficos são encontrados com resoluções de 122x32, 128x64, 240x64 e 240x128 dots pixel ou maior, e geralmente estão disponíveis com 20 pinos para conexão;
- São especificados em número de linhas por colunas;
- Podem ser encontrados com ou sem *LED backlight*;
- Permite ajuste de contraste mediante ajuste de tensão em um pino específico;

# 26. Display de Cristal Líquido -LCD

Pino	Função	Descrição
1	Alimentação	Terra ou GND
2	Alimentação	VCC ou +5V
3	V0	<b>Tensão para ajuste de contraste (ver Figura 1)</b>
4	RS Seleção:	1 - Dado, 0 - Instrução
5	R/W Seleção:	1 - Leitura, 0 - Escrita
6	E Chip select	1 ou (1 → 0) - Habilita, 0 - Desabilitado
7	B0 LSB	Barramento de Dados
8	B1	
9	B2	
10	B3	
11	B4	
12	B5	
13	B6	
14	B7 MSB	Anodo p/ <i>LED backlight</i>
15	A (qdo existir)	
16	K (qdo existir)	Catodo p/ <i>LED backlight</i>

Pinagem dos módulos LCD

# 26. Display de Cristal Líquido -LCD

---

R/W	RS	DESCRIÇÃO	Endereço
0	0	Instrução - Escrita no modulo	0000h
0	1	Dados - Escrita no modulo	0100h
1	0	Instrução - Leitura no modulo	0200h
1	1	Dados - Leitura no modulo	0300h

Endereçamento do módulo LCD para o aplicativo de Teste

# 26. Display de Cristal Líquido -LCD

## PROGRAMAÇÃO / INSTRUÇÕES

DESCRIÇÃO	MODO	RS	R/W	Código h
Display	Liga (sem cursor)	0	0	0C
	Desliga	0	0	0A / 08
Limpa Display com Home cursor		0	0	01
Controle do Cursor	Liga	0	0	0E
	Desliga	0	0	0C
	Desloca para Esquerda	0	0	10
	Desloca para Direita	0	0	14
	Cursor Home	0	0	02
	Cursor Piscante	0	0	0D
	Cursor com Alternância	0	0	0F
Sentido de deslocamento do cursor ao entrar com caracter	Para a esquerda	0	0	04
	Para a direita	0	0	06
Deslocamento da mensagem ao entrar com caracter	Para a esquerda	0	0	07
	Para a direita	0	0	05
Deslocamento da mensagem sem entrada de caracter	Para a esquerda	0	0	18
	Para a direita	0	0	1C
End. da primeira posição	primeira linha	0	0	80
	segunda linha	0	0	C0

# 26. Display de Cristal Líquido -LCD

INSTRUÇÃO	R S	R/ W	B7	B6	B5	B4	B3	B2	B1	B0	DESCRIÇÃO e tempo de execução (uS)	t	
Limpa Display	0	0	0	0	0	0	0	0	0	1	-Limpa todo o display e retorna o cursor para a primeira posição da primeira linha	1.6 mS	
Home p/ Cursor	0	0	0	0	0	0	0	0	1	*	-Retorna o cursor para a 1. coluna da 1. Linha -Retorna a mensagem previamente deslocada a sua posição original	1.6 mS	
Fixa o modo de funcionamento	0	0	0	0	0	0	0	1	X	S	-Estabelece o sentido de deslocamento do cursor (X=0 p/ esquerda, X=1 p/ direita) -Estabelece se a mensagem deve ou não ser deslocada com a entrada de um novo caractere (S=1 SIM, X=1 p/ direita) -Esta instrução tem efeito somente durante a leitura e escrita de dados.	40 uS	
Controle do Display	0	0	0	0	0	0	1	D	C	B	-Liga (D=1) ou desliga display (D=0) -Liga(C=1) ou desliga cursor (C=0) -Cursor Piscante(B=1) se C=1	40 uS	
Desloca cursor ou mensagem	0	0	0	0	0	1	C	R	*	*	-Desloca o cursor (C=0) ou a mensagem (C=1) para a Direita se (R=1) ou esquerda se (R=0) - Desloca sem alterar o conteúdo da DDRAM	40 uS	
Fixa o modo de utilização do módulo LCD	0	0	0	0	1	Y	N	F	*	*	-Comunicação do módulo com 8 bits(Y=1) ou 4 bits(Y=0) -Número de linhas: 1 (N=0) e 2 ou mais (N=1) -Matriz do caractere: 5x7(F=0) ou 5x10(F=1) - Esta instrução deve ser ativada durante a inicialização	40 uS	
Posiciona no endereço da CGRAM	0	0	0	1	Endereço da CGRAM				-Fixa o endereço na CGRAM para posteriormente enviar ou ler o dado (byte)				40 uS
Posiciona no endereço da DDRAM	0	0	1	Endereço da DDRAM				-Fixa o endereço na DDRAM para posteriormente enviar ou ler o dado (byte)				40 uS	
Leitura do Flag Busy	0	1	B F	AC				-Lê o conteúdo do contador de endereços (AC) e o BF. O BF (bit 7) indica se a última operação foi concluída (BF=0 concluída) ou está em execução (BF=1).				0	
Escreve dado na CGRAM / DDRAM	0	1	Dado a ser gravado no LCD				- Grava o byte presente nos pinos de dados no local apontado pelo contador de endereços ( <i>posição do cursor</i> )				40 uS		
Lê Dado na CGRAM / DDRAM	1	1	Dado lido do módulo				- Lê o byte no local apontado pelo contador de endereços ( <i>posição do cursor</i> )				40 uS		

# **26. Display de Cristal Líquido -LCD**

---

## **TABELAS DE ENDEREÇOS DOS CARCTERES NA DDRAM**

<b>LCD 16x2</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>	<b>11</b>	<b>12</b>	<b>13</b>	<b>14</b>	<b>15</b>	<b>16</b>
linha 1	80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
linha 2	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF

<b>LCD 20x4</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>	<b>11</b>	<b>12</b>	<b>13</b>	<b>14</b>	<b>15</b>	<b>16</b>	<b>17</b>	<b>18</b>	<b>19</b>	<b>20</b>
linha 1	80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F	90	91	92	93
linha 2	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF	D0	D1	D2	D3
linha 3	80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F	90	91	92	93
linha 4	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF	D0	D1	D2	D3

# 26. Display de Cristal Líquido -LCD

Char. code	0 0 0 0 0 0 0 1 1 1 1 1 1	0 0 0 1 1 1 1 0 0 1 1 1 1	0 1 1 0 0 1 1 1 1 0 0 1 1	0 0 1 0 1 0 1 0 1 0 1 0 1
xxxxx0000	0 0 P ^ P - ♪ E & p			
xxxxx0001	! 1 A Q a q 。 A T 4 ä q			
xxxxx0010	" 2 B R b r 「イツ×βθ			
xxxxx0011	# 3 C S c s ピテモ。			
xxxxx0100	\$ 4 D T d t エトヤムΩ			
xxxxx0101	% 5 E U e u • オナユシÜ			
xxxxx0110	& 6 F U f v ヲカニヨロΣ			
xxxxx0111	' 7 G W 9 w アキラロπ			
xxxxx1000	( 8 H X h x イクネリ♪ 口			
xxxxx1001	) 9 I Y i y ヲケルレ" ピ			
xxxxx1010	* ; J Z j z エコハレ i キ			
xxxxx1011	+ ; K [ k ( オサヒロ× 穴			
xxxxx1100	, < L ¥ 1   ャシフワΦ 円			
xxxxx1101	- = M ] m ) ュスヘンモ ×			
xxxxx1110	. > N ^ n + ボセホ ^ ハ			
xxxxx1111	/ ? 0 _ o ← ツリマ " ö ■			

MAPA DE CARCTERES NA CGRAM