MANUAL

MICROCONTROLADORES

FAMÍLIA 8031/8051

PROF. EDSON PEDRO FERLIN

CURITIBA, PR

ABRIL 2004

Sumário

CAPÍTULO 1 – INTRODUÇÃO

Neste manual detalhamos a família de microcontroladores 8031/8051 tanto no aspecto da microarquitetura quanto de programação em assembly, desta família que é amplamente utilizada no mercado e também é de fácil desenvolvimento, tendo inclusive compiladores C para ela.

Os microcontroladores são microprocessadores direcionados para aplicações de controle, que já possuem em seu interior recursos como memória de programa e dados, portas de comunicação, controladores de interrupção, timers, e em alguns casos conversores A/D. A grande vantagem é que por já possuírem estes recursos incorporados as aplicações desenvolvidas com eles são mais compactas e por conseqüência apresentam um custo menor, quando comparado com o desenvolvimento usando microprocessadores, pois estes recursos devem ser agregados no sistema.

As informações presentes neste manual foram obtidas de diversos documentos e compiladas de uma maneira a proporcionar uma melhor visão sobre estes microcontroladores, seus recursos e instruções necessários para a programação assembly e para a compreensão do funcionamento deste, versátil e prático, microcontrolador.

Além de ser utilizado para o desenvolvimento de projetos digitais compactos, direcionados para aplicações de controle e gerenciamento, ele pode ser utilizado como elemento didático para a demonstrar o funcionamento dos processadores, pois o desenvolvimento de programas usando assembly para 8051 é de fácil compreensão.

CAPÍTULO 2 – VISÃO GERAL

O Intel 8051 é um microcontrolador clássico, e é um verdadeiro microcomputador contendo E/S paralela, contadores/temporizadores, E/S serial, RAM, e EPROM ou ROM (dependendo do tipo). A família 8051 é composta por vários membros (a Intel se refere como a família MCS-51), cada um adaptado para um tipo específico de sistema.

As diferentes versões são mostradas na tabela 1. O 8051 tem dois parentes próximos, o 8751 e o 8031, e um primo, o 8052. Todas as versões contêm a mesma CPU, RAM, contadores/temporizadores, portas paralelas, e E/S serial. O 8051 contém 4Kbytes de ROM, a qual é definida/mascarada quando o *chip* é produzido. No 8751, a ROM é trocada por EPROM que se pode programar.

O 8031 é destinado para aplicações expandidas e usa memória externa. O 8031 usa três das quatro portas paralelas do *chip* para fazer o endereçamento convencional e um barramento de dados com linhas apropriadas de controle.

Desde que o 8031 ainda contenha RAM, uma porta paralela, e uma porta serial - até mesmo quando funciona como uma CPU principal de um circuito expandido - o número de *chips* eventuais necessários para expandir o E/S ou memória é ainda consideravelmente pequeno.

A Intel e outras companhias comercializam variações da família MCS-51 com mais memória interna, mais E/S, baixa potência, e assim por diante. Um 80C31 é uma versão CMOS de baixa potência do 8031, por exemplo. O 8052, é o mesmo que o 8051, exceto que tem outro contador/temporizador e RAM e ROM adicionais.

Como pode-se esperar, tanto a fabricação da máscara quanto a construção do *chip* demanda tempo e dinheiro. A programação por máscara faz sentido para uma aplicação que usa milhares de 8051 idênticos, mas isto não é prático para baixo volume de sistemas e de protótipos.

O 8751 possui todas as características do 8051, exceto que uma única EPROM substitui o programa armazenado na ROM. Fazer uma mudança no programa é tão simples como apagar a EPROM com luz ultravioleta e gravar outro programa. Muitos projetistas usam 8751 sobre o trabalho de codificação, então confiam para uma grande classe de 8051 com o programa em ROM. Se o volume do produto é suficientemente pequeno, é freqüentemente proveitoso o uso do 8751 mesmo no produto final.

O 8031 não tem o programa armazenado no próprio *chip*. O sistema incluirá uma EPROM externa e um "*latch*" de endereço, como visto na figura 1. Considerando a queda de preço das EPROMs e o pesado custo do uso de ambos os 8051s ou 8751s em pouca quantidade, o 8031 é uma alternativa viável apesar dos *chips* adicionais. Para muitos sistemas pequenos, a combinação 8031/EPROM é muito melhor em custo/benefício do que um 8051.

Todos os membros da família MCS-51 tem o mesmo núcleo do hardware e, portanto, usam o mesmo núcleo do conjunto de instruções.

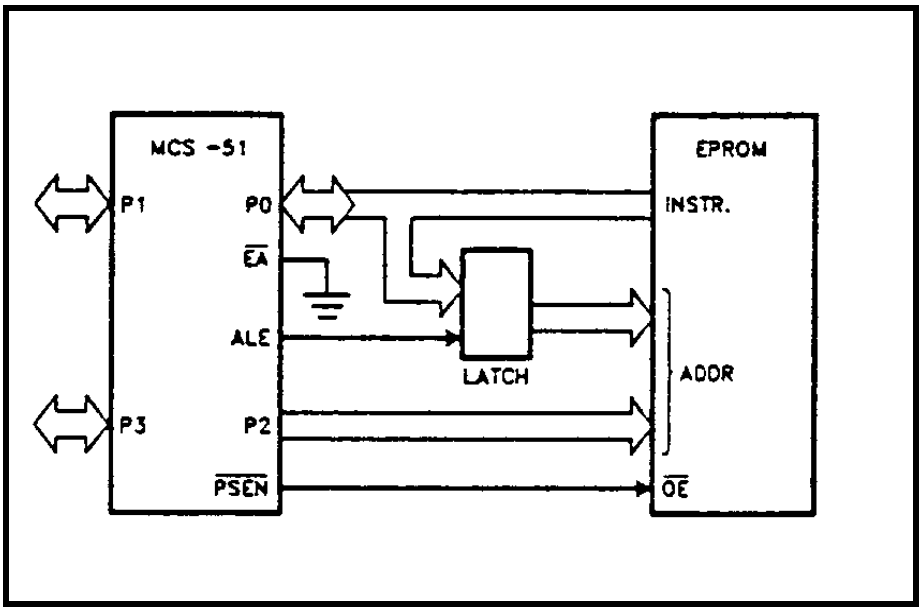


Figura 1. Sistema Mínimo com ROM externa

A família de microcontroladores MCS - 51 foi desenvolvida para aplicações de controle e oferece características de hardware sofisticadas que permitem seu uso no Controle Industrial, Periféricos Inteligentes e em uma variada gama de produtos.

O 8031 é um membro desta família que não possui memória interna de programa, embora toda a arquitetura interna seja a mesma. As características de hardware e software da família 8051 permitem a manipulação de bits com extrema facilidade e possibilitam o desenvolvimento de sistema microprocessado de um único *chip*.

Para os membros da família que contem ROM interna podem utilizam RAM adicional, como pode ser visto na Figura 2.

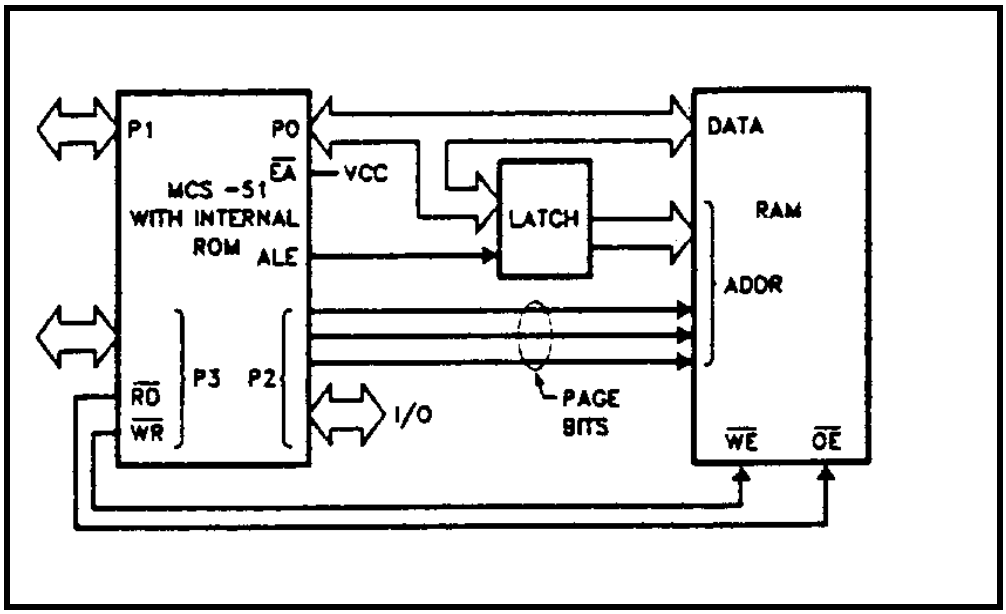


Figura 2. (Sistema com ROM interna e RAM adicional externa)

- **Versões com ROM:**

  8051, 80C51, 80CL51, 8052, 83C053, 83CL410, 83C451, 83C528, 83C550, 83C552, 83C562, 83C575, 83C592, 83C652, 83C654, 83C751, 83C752, 83C851, 83C852

- **Versões sem ROM:**

  8031, 80C31, 8032, 80C32, 80CL410, 80C451, 80C528, 80C550, 80C552, 80C562, 80C575, 80C592, 80C652, 80C851

- **Versões com EPROM:**

  87C51, 87C52, 87C054, 87C451, 87C528, 87C550, 87C552, 87C575, 87C592, 87C652, 87C654, 87C751, 87C752

1.1) Características de Hardware

As características funcionais apresentadas abaixo referem-se ao microcontrolador 8051.

- CPU de 8 bits otimizada para aplicações de controle;
- Memória Interna de Programa (4Kbytes);
- Memória Interna de Dados (256 bytes);
- 2 Contadores/Temporizadores Programáveis de 16 bits;
- 1 Porta Serial *Full - Duplex;*
- 32 linhas de E/S;
- Estrutura de Interrupção com dois níveis de prioridade e 5 fontes (2 externas, 3 internas);
- RAM interna endereçável bit-a-bit;
- 64Kbytes para Memória Externa de Programas;
- 64Kbytes para Memória Externa de Dados.

Cada dispositivo da família MCS-51 contêm todas as características principais apresentadas e mais algumas adicionais. Uma comparação das características de todos os dispositivos da família MCS-51 é mostrado na tabela abaixo.

Device	ROMless Version	EPROM Version	ROM Bytes	RAM Bytes	8-Bit I/O Ports	16-Bit Timer/ Counters	Programmable Counter Array (PCA)	UART	Serial Expansion Port (SEP)	Global Serial Channel (GSC)	DMA Channels	A/D Channels	Interrupt Sources/ Vectors	Power Down and Idle Modes
8051	8031	—	4K	128	4	2		✓					6/5	
8051AH	8031AH	8751H 8751BH	4K	128	4	2		✓					6/5	
8052AH	8032AH	8752BH	8K	256	4	3		✓					8/6	
80C51BH	80C31BH	87C51	4K	128	4	2		✓					6/5	✓
80C52	80C32	—	8K	256	4	3		✓					8/6	✓
83C51FA	80C51FA	87C51FA	8K	256	4	3	✓	✓					14/7	✓
83C51FB	80C51FA	87C51FB	16K	256	4	3	✓	✓					14/7	✓
83C152JA	80C152JA	—	8K	256	5	2		✓		✓	2		19/11	✓
—	80C152JB	—	—	256	7	2		✓		✓	2		19/11	✓
83C152JC	80C152JC	—	8K	256	5	2		✓		✓	2		19/11	✓
—	80C152JD	—	—	256	7	2		✓		✓	2		19/11	✓
83C452	80C452	87C452P	8K	256	5	2		✓					9/8	✓

Tabela 1. Dispositivos da família MCS-51

1.1) Pinagem do 8051

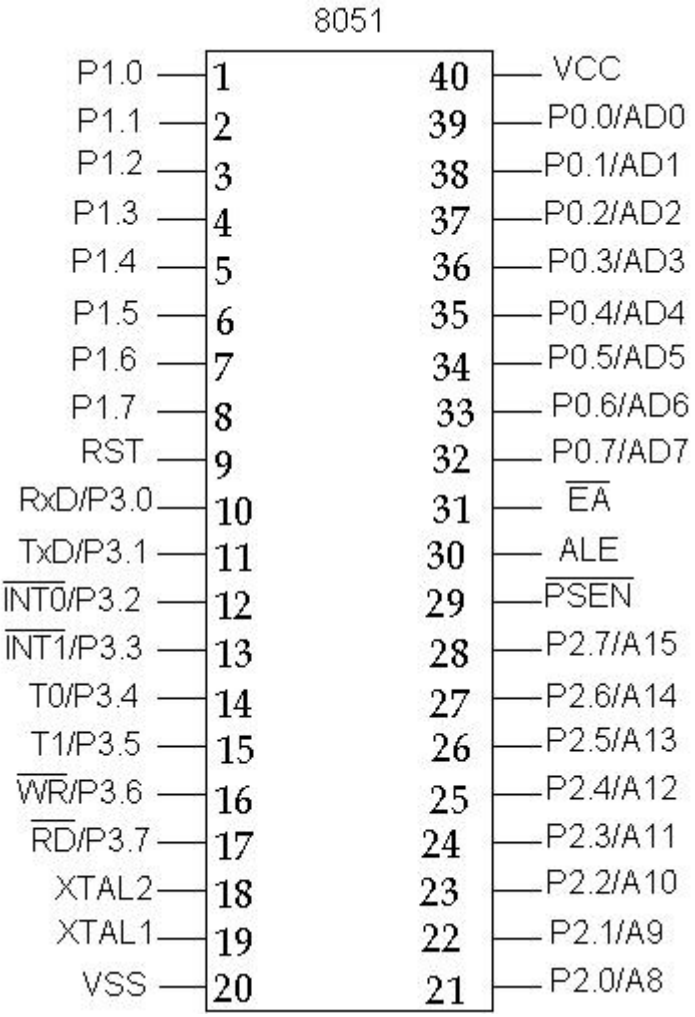


Figura 1. Sistema Mínimo com ROM externa

1.2) Organização da Memória

Todos os membros da família MCS-51 possuem espaços de endereçamento separados para Dados e Programas (arquitetura Harvard).

- **Memória de Programa**

O espaço de endereçamento para a memória de programa é de 64Kbytes. No 8051, os 4Kbytes mais baixos estão na própria CPU (memória interna de programa). Após o *reset*, a CPU inicia a execução no endereço 0000H, onde deve residir uma instrução de salto para o endereço de inicio do programa. As posições 0003H a 0023H (002BH) estão reservadas para as rotinas de atendimento a interrupções.

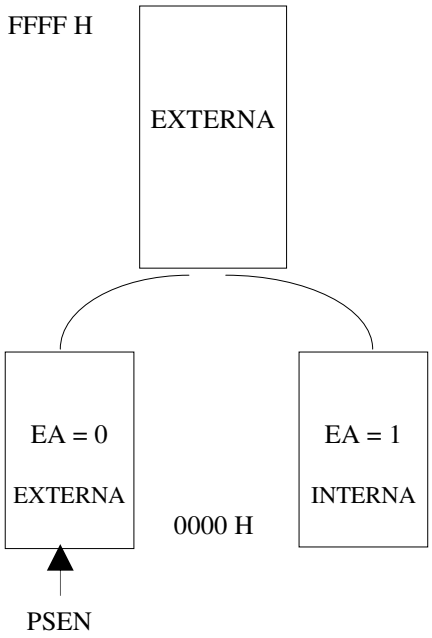


Figura 3. Memória de Programa: Interna e Externa

Os 4Kbytes mais baixos da memória de programa podem residir tanto internamente no *chip* como na memória externa. A seleção é feita pelo pino EA (*External Access*). Quando EA=1, a CPU busca as instruções de endereços 0000H a 0FFFH na ROM interna e as instruções de endereços 1000H a FFFFH na memória externa. Se EA=0, então todas as buscas serão feitas na memória externa. No 8031, como não existe ROM interna, o pino EA deve estar sempre em 0.

No 8052 (8032) que tem 8Kbytes de memória ROM, quando EA=1, a CPU busca as instruções de endereços 0000H a 1FFFH na ROM interna, e as instruções de endereço 2000H a FFFFH na memória externa.

- **Memória de Dados**

O 8051 pode acessar dados tanto na Memória Interna como na Memória Externa de dados. A Memória Externa de Dados pode ter até 64Kbytes e é acessada através da instrução MOVX. A Memória Interna de Dados do 8051 é composta por 2 blocos de 128 bytes. O bloco inferior (00H a 7FH) é usado como RAM e pode ser endereçado direto ou indiretamente. O bloco superior (80H a FFH) é um espaço reservado para mapear os registradores internos da CPU (SFRs - *Special Function Registers*). Este bloco só pode ser acessado diretamente.

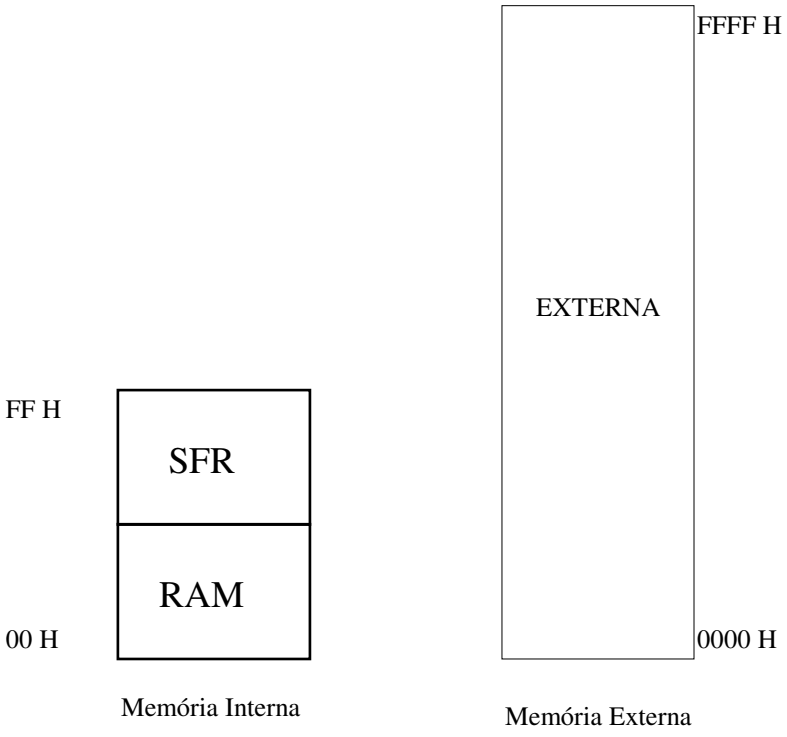


Figura 4. Memória de Dados: Interna e Externa

O bloco inferior da Memória Interna de Dados (00H a 7FH) é dividido em três partes:

- Banco de Registradores (00H a 1FH): 32 bytes
- Área endereçável bit-a-bit (20H a 2FH): 16 bytes
- Área de rascunho (*scratch-pad*) (30H a 7FH): 80 bytes

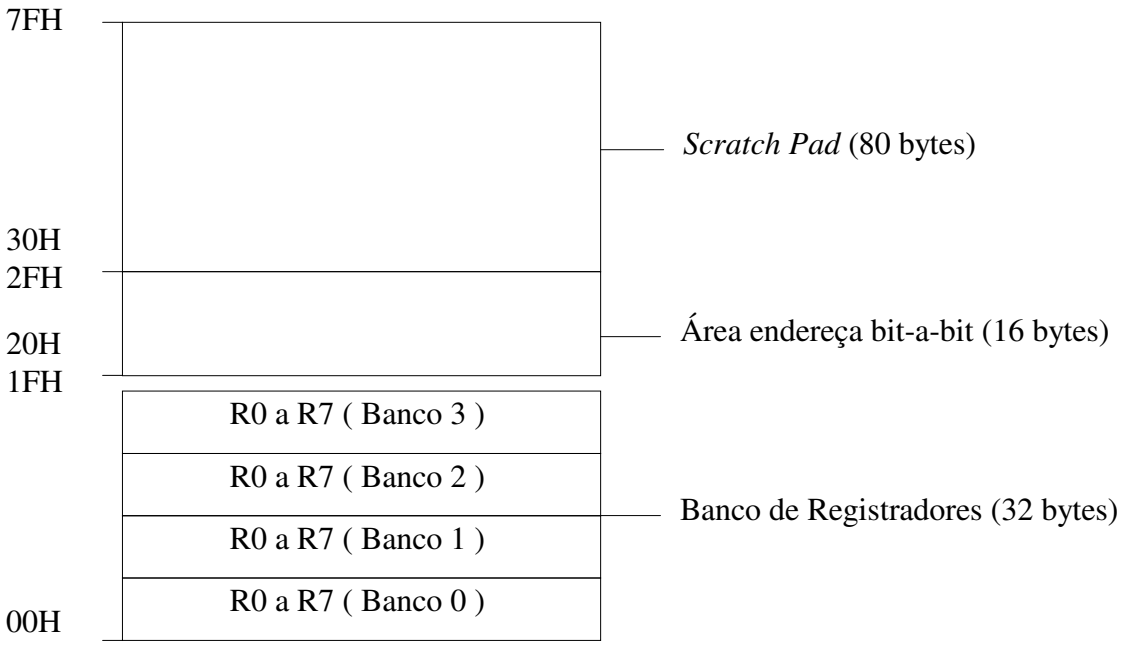


Figura 5. Bloco inferior da Memória Interna de Dados

O bloco superior (área dos SFRs 80H a FFH) está mostrado na figura abaixo.

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

Figura 6. Bloco superior da Memória Interna de Dados - Área dos SFRs

CAPÍTULO 2 - ARQUITETURA INTERNA

A família MCS-51 é baseada no seguinte diagrama em blocos:

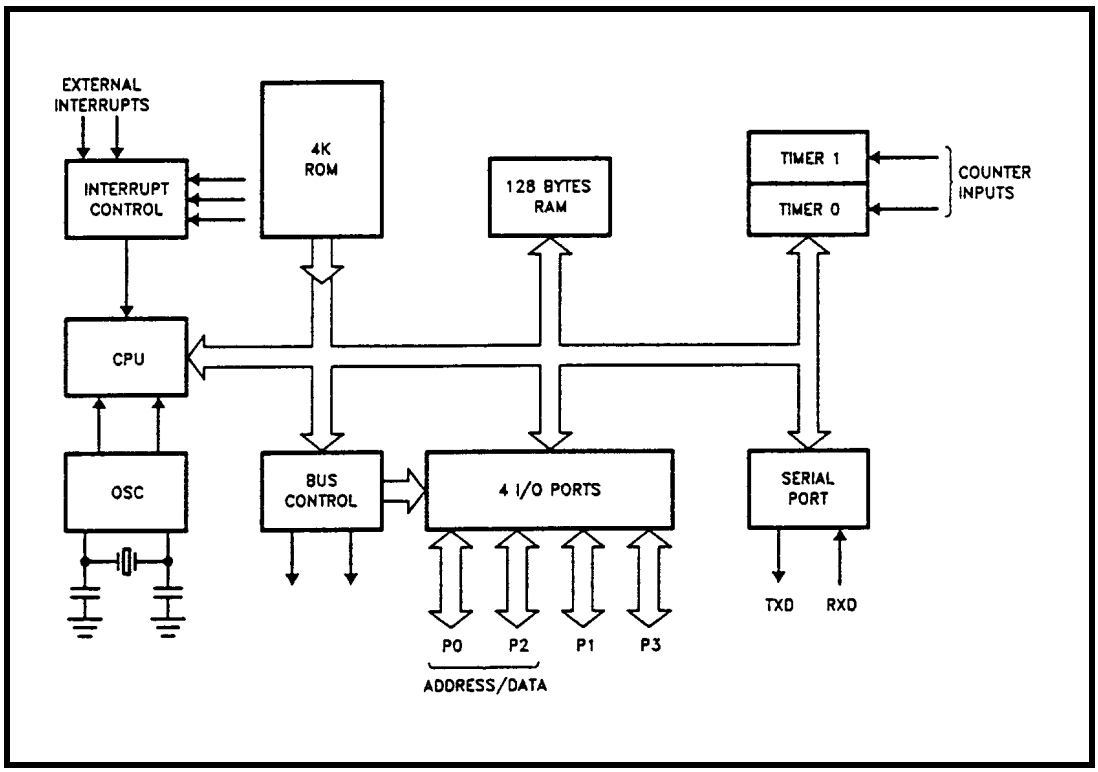


Figura 7. Diagrama em Blocos da CPU do 8051

2.1) Registradores da CPU

Os registradores internos do 8051, com exceção do PC (*Program Counter*), estão alocados na parte superior da Memória Interna de Dados. Este espaço é conhecido como SFR (*Special Function Register*).

- ACUMULADOR (ACC) - O ACC é usado como acumulador. Nas instruções específicas do acumulador, ele é simplesmente referenciado como A.

- Registrador B - O registrador B é utilizado durante operações de multiplicação e divisão e serve como fonte e destino dos dados.

- *Stack pointer* (SP) - É o ponteiro de pilha da CPU. Tem largura de 8 bits e é incrementado antes de um PUSH ou CALL e decrementado após um POP ou RET.

- DPTR (*Data Pointer*) - O DPTR é um registrador de 16 bits usado para o endereçamento da memória externa de dados. O DPTR ocupa dois endereços sucessivos no SFR e pode ser usado separadamente como 2 registradores de 8 bits (DPL e DPH).

- Portas 0 a 3 - Os registradores P0, P1, P2 e P3 são os *latchs* das portas 0 ... 3 de E/S.

- SBUF (*Serial Data Buffer*) - O SBUF é formado por dois registradores separados, um registrador de transmissão e um de recepção. Quando um dado é transferido para SBUF, ele é levado à porta serial para ser transmitido. Na recepção o SBUF é carregado com o byte recebido.

- PSW (*Program Status Word*) - O registrador PSW contém os *flags* da CPU. A tabela a seguir mostra os *flags* disponíveis.

	(MSB)							(LSB)
PSW	CY	AC	F0	RS1	RS0	OV	-------	P

Símbolo	Bit	Descrição
CY	PSW.7	*Flag* de *Carry*
AC	PSW.6	*Flag* de *Carry* Auxiliar (operações BCD)
F0	PSW.5	Disponível ao Usuário
RS1	PSW.4	Seleção do Banco de Registradores
RS0	PSW.3	Seleção do Banco de Registradores
OV	PSW.2	Flag de *Overflow*
--------	PSW.1	Reservado
P	PSW.0	*Flag* de Paridade

Tabela 2. Bits do registrador PSW

- Registradores de Temporização - Os pares de registradores (TH0, TL0), (TH1, TL1), e (TH2, TL2) são os contadores/temporizadores do 8051 e do 8052. Cada par corresponde a um registrador de 16 bits do temporizador/contador correspondente (0, 1, e 2) respectivamente.

- Registradores de Captura * (apenas no 8052) - O par de registradores (RCAP2H, RCAP2L) são registradores de captura para o Timer 2 "Modo de Captura". Neste modo, em resposta a uma transição do pino T2EX do 8052, TH2 e TL2 são copiados para o RCAP2H e RCAP2L. Timer 2 também tem o modo de 16 bits auto-*reload*, e RCAP2H e RCAP2L armazena o valor de *reload* para este modo.

- Registradores de Controle - Os registradores IP, IE, TMOD, TCON, SCON e PCON são os registradores de controle e estado do sistema de interrupção, da seção de temporização/contagem e da porta serial.

A figura 8 (abaixo), mostra o diagrama em blocos a arquitetura interna da família MCS-51, constando todos os componentes internos da CPU.

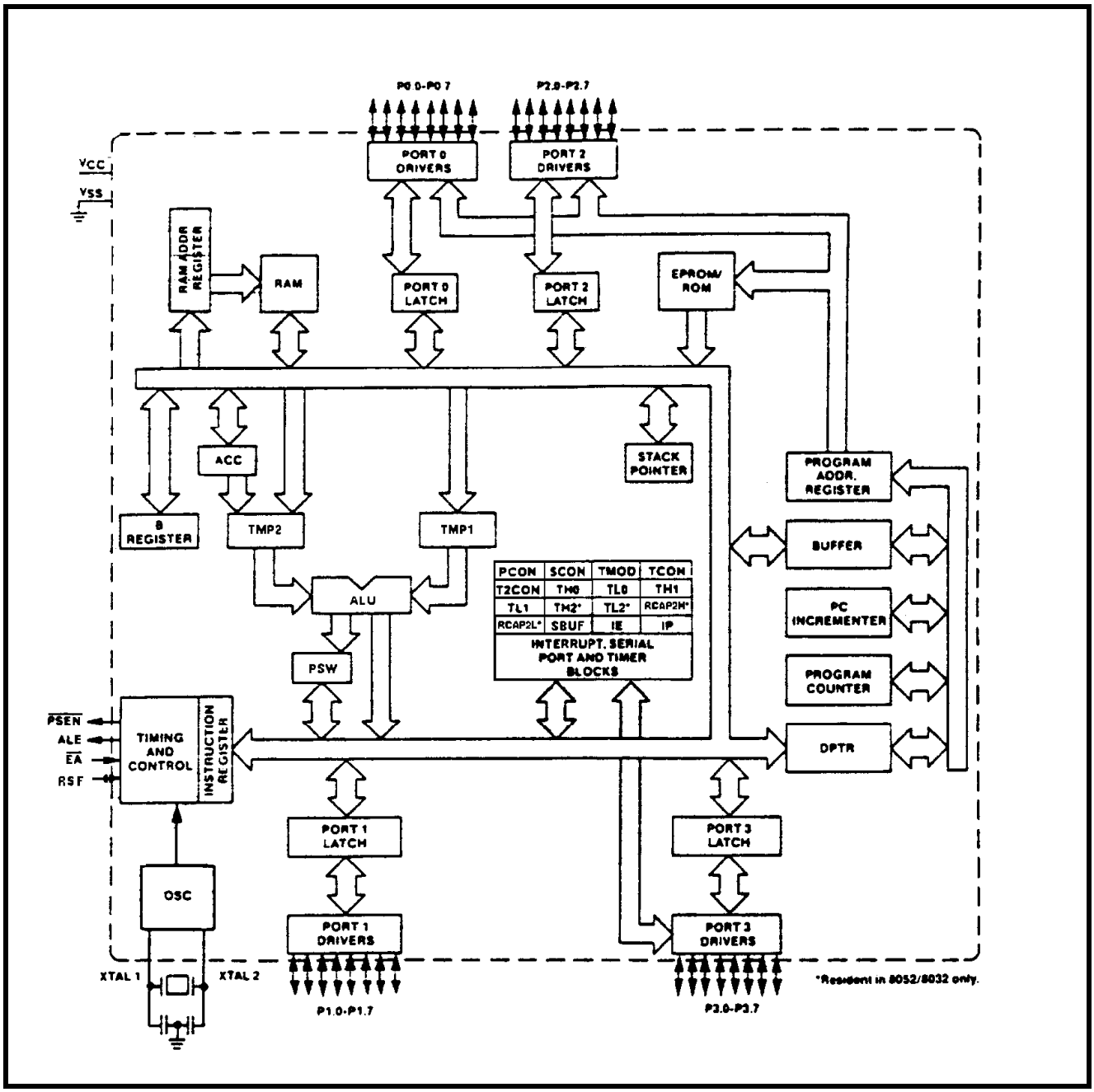


Figura 8. Diagrama em Blocos da Arquitetura Interna da Família MCS-51

2.2) Portas de E/S

O 8051 possui 4 (quatro) portas de E/S bidirecionais de 8 bits. Cada porta consiste de um *latch* (P0 a P3), de um driver de saída e de um *buffer* de entrada. As portas de E/S do 8051 podem ser usadas para E/S ou para funções genéricas.

- Porta P0

A porta P0 é usada como barramento multiplexado (ADD/DATA) no acesso à Memória Externa de Programa e Dados. Em um acesso à Memória de Programa ou Dados, a porta P0 contém os 8 LSBs do endereço (no início do acesso) e contém os 8 bits de dados lidos ou escritos na memória.

Para sistemas que não utilizam memória externa, a porta P0 pode ser usada como porta de E/S para propósitos gerais, mostrada na figura 10a.

- Porta P1

A porta P1 é usada exclusivamente como porta de E/S. Sua estrutura está mostrada na figura 10b. Para o 8052, os pinos P1.0 e P1.1, tem outra função especial, como segue:

P1.0	T2 (Timer/Contador 2 entrada externa)
P1.1	T2EX (Timer/Contador 2 Capture/*Reload trigger*)

- Porta P2

A porta P2 é usada para acomodar o byte mais significativo do endereço externo (a porta P0 acomoda o byte menos significativo deste endereço). As linhas de E/S de P2 podem ser usadas para propósitos gerais se não forem afetados pela CPU durante o acesso à Memória Externa. Sua estrutura está mostrada na figura 10c.

- Porta P3

A porta P3 pode ser usada para E/S de uso genérico ou para acomodar as linhas externas da CPU (interrupção, temporizadores, porta serial e sinais de RD/WR). A estrutura é mostra na figura 10d. e a configuração de pinos da porta P3 estão mostradas na figura abaixo.

P3.0	RXD	(Entrada Serial)
P3.1	TXD	(Saída Serial)
P3.2	INT0	(Interrupção Externa 0)
P3.3	INT1	(Interrupção Externa 1)
P3.4	T0	(Entrada temp/cont 0)
P3.5	T1	(Entrada temp/cont 1)
P3.6	WR	Sinal de Ativação de Escrita
P3.7	RD	Sinal de Ativação de Leitura

Figura 9. Descrição dos pinos da porta P3

A figura 10 mostra as internamente as portas P0 a P3, apresentando a sua estrutura em termos de dispositivos digitais.

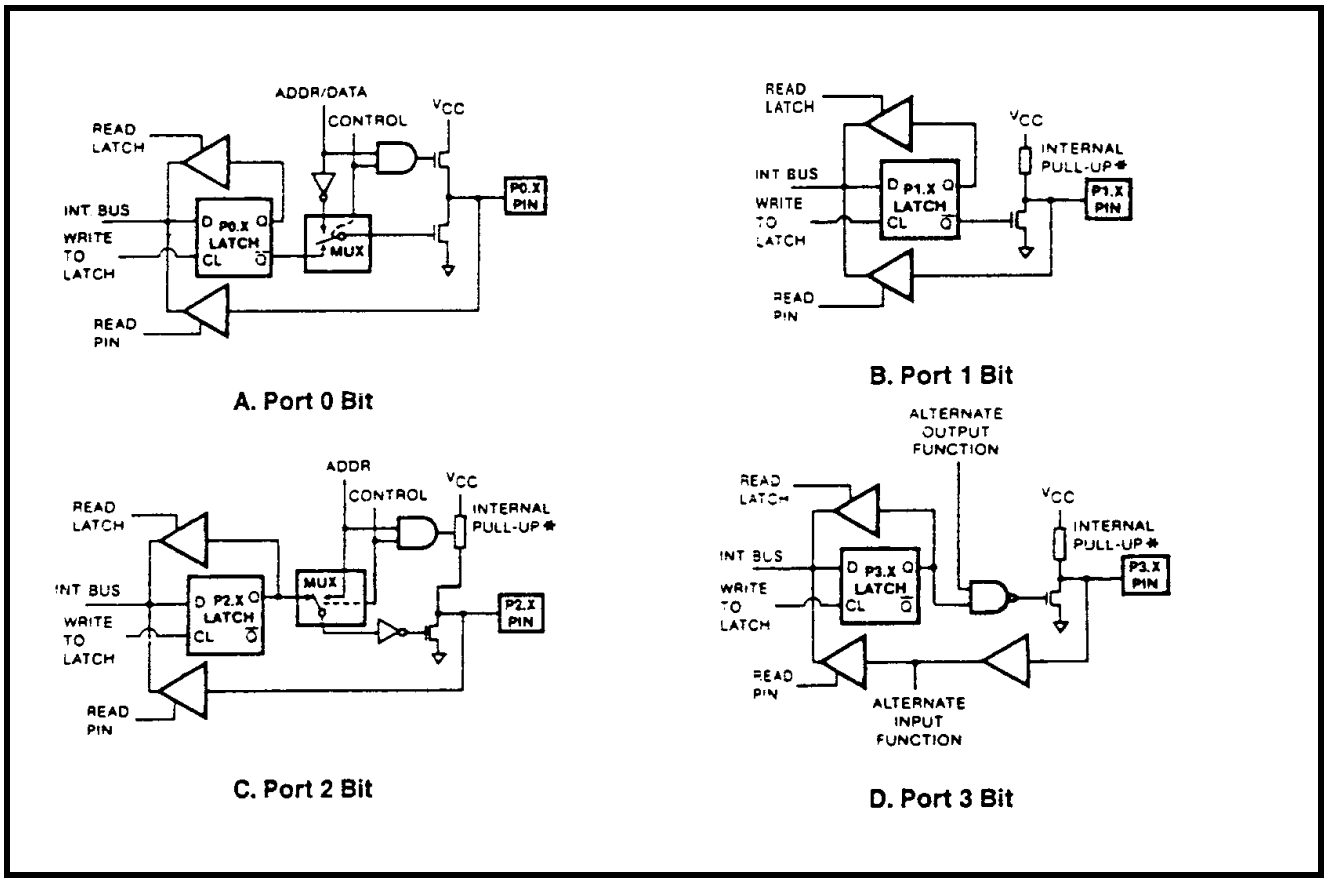


Figura 10. Estrutura das portas P0 a P3

2.3) Acesso à Memória Externa

Os acessos externos podem ser feitos à Memória de Programa e à Memória de Dados. Os acessos à Memória Externa de Programa utilizam o sinal PSEN (*Program Strobe Enable*) como sinal de "*strobe*" para leitura.

Os acessos à Memória Externa de Dados utilizam os sinais RD e WR (P3.7 e P3.6) para habilitar a leitura e escrita durante a instrução MOVX. O acesso à Memória de Programa utiliza um endereço de 16 bits, enquanto que, na memória de dados, podem ser usados endereços de 8 ou 16 bits (MOV @Ri ou MOV @DPTR).

Em um acesso externo com endereço de 16 bits, o byte de mais alta ordem do endereço é emitido na porta P2 e permanece válido durante todo o ciclo de leitura/escrita. O byte menos significativo do endereço, em qualquer caso, é emitido na porta P0.

2.4) Timing *da CPU*

No 8051, o ciclo de máquina é composto por 12 períodos do oscilador. A cada 2 períodos, definimos um "estado". Logo, um Ciclo de Máquina (CM) é composto por 6 (seis) estados (S1 a S6), cada qual definido por dois períodos P1 a P2.

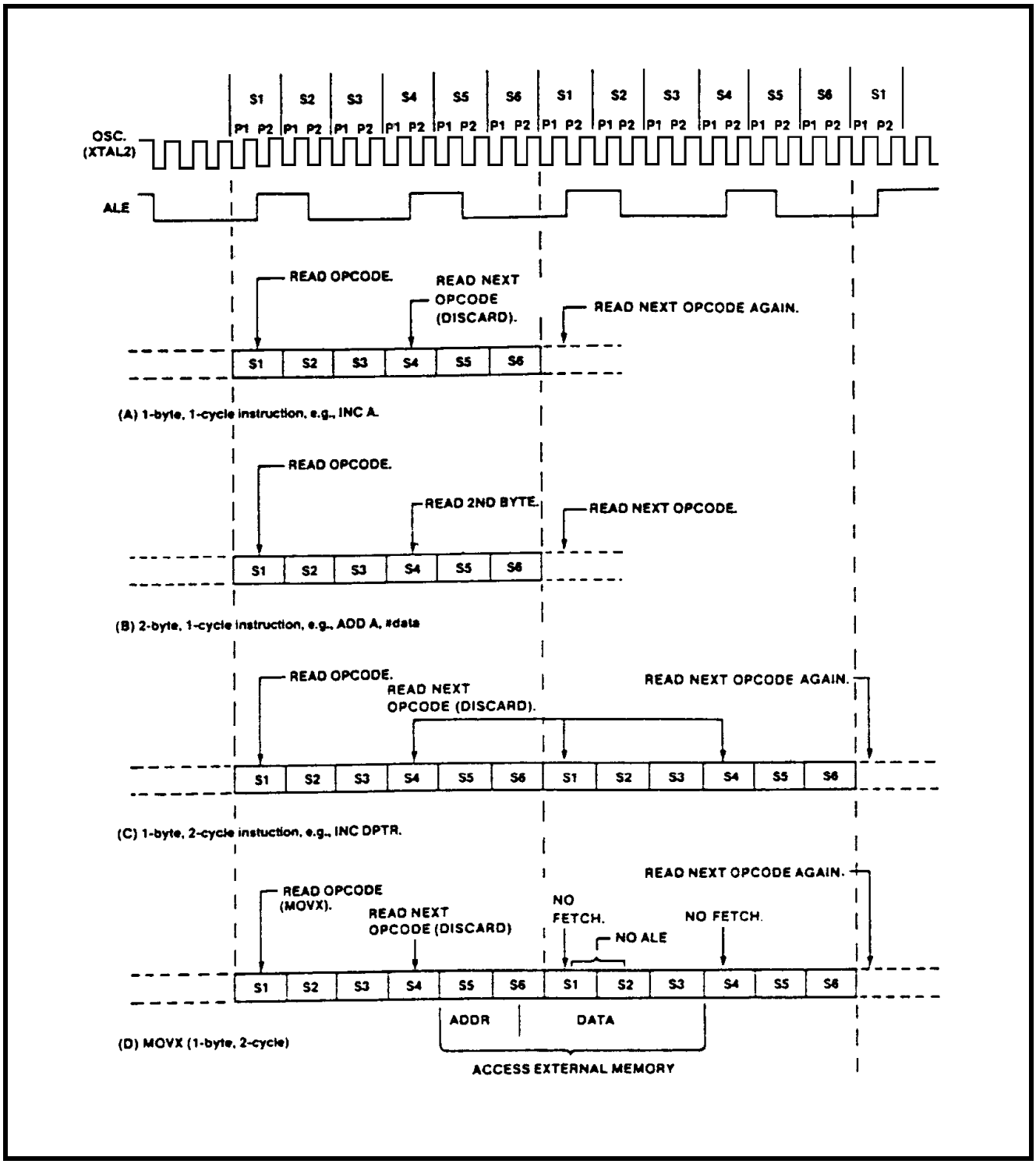


Figura 11. *Timing* da CPU

A execução de uma instrução de 1 ciclo começa em S1P2 quando o *opcode* é armazenado no Registrador de Instrução (IR). No caso de uma instrução de 2 bytes, o segundo byte é lido em S4 no mesmo ciclo de máquina.

No caso de uma instrução de 1 byte, o 8051 efetua o segundo "*fetch*", mas o byte lido é ignorado e o contador de programa (PC) não é incrementado. De qualquer forma, para uma instrução de 1 ciclo, a execução termina em S6P2, isto é, no último período do ciclo de máquina.

No 8051, a maior parte das instruções é executada em 1 Ciclo de Máquina (CM). As instruções MUL (multiplicação) e DIV (divisão) são as únicas que necessitam de mais de 2 ciclos (elas são executadas em 4 ciclos de máquina).

A instrução MOVX (acesso à Memória Externa) é uma instrução de 1 byte e 2 ciclos. A sua execução envolve o endereçamento e acesso à Memória Externa entre S5 do primeiro CM e S3 do segundo CM, intervalo no qual, o 8051 não efetua nenhum "*fetch*".

2.5) Contadores / Temporizadores

A seção de temporização/contagem do 8051 consiste de 2 temporizadores/contadores programáveis de 16 bits (Timer 0 e Timer 1). O 8052 possui um temporizador adicional (Timer 2). Os temporizadores podem ser configurados para operar como:

- Temporizadores ou
- Contadores de eventos.

Na função de "temporizador", o registrador é incrementado a cada ciclo de máquina, isto é, a cada 12 períodos do oscilador. Isto nos leva a refletir sobre a diferença conceitual entre um temporizador e um contador de eventos. Em um temporizador o registrador executa uma contagem que é determinada por um evento periódico (o ciclo de máquina).

Na função de "contador", o registrador é incrementado em resposta a uma transição 1 -> 0 em seu pino externo correspondente (T0, T1 ou T2 (8052)). Neste caso, o pino externo é amostrado durante S5P2 de cada Ciclo de Máquina. Quando a amostragem detectar um nível alto em um ciclo e um nível baixo no ciclo seguinte, o registrador é incrementado. Desta forma, o 8051 precisa de 2 CMs para reconhecer uma transição 1 -> 0, o que impõe a condição de taxa mínima para a contagem de eventos = 1/24 períodos de oscilador. Além da seleção "temporizador/contador", os temporizadores T0 e T1 podem operar em 4 modos diferentes.

- Modo 0

Este modo configura o Timer 0/1 como um contador de 8 bits (THx) com um "*prescaler*" de 5 bits. Quando houver *overflow*, isto é, quando a contagem passar de "todos 1s" para "todos 0s", o *flag* de interrupção TF1 é setado.

A entrada de contagem é habilitada pelos bits TR1 (Timer 1 Run 1) e GATE e pela entrada INT1. Quando o bit GATE = 0, o contador será habilitado somente pelo bit TR1. Quando GATE = 1, o contador será habilitado por TR1 durante o tempo em que INT1 = 1. Isto possibilita a medição da largura do pulso em INT1 diretamente.

Os bits GATE e TR1 pertencem aos registradores TCON e TMOD respectivamente. A figura abaixo mostra o formato destes registradores.

	(MSB)							(LSB)
TMOD	GATE	C/T	M1	M0	GATE	C/T	M1	M0

- GATE:
  - Se GATE = 1, o temporizador/contador "x" é habilitado somente quanto Intx = 1.
  - Se GATE = 0, o temporizador/contador "x" é habilitado somente quanto TRx = 1.
- C/T: Define se a operação será de contagem (entrada no pino Tx) ou temporização (entrada pelo oscilador).
- M1, M0: Modo de Operação (0 ... 3)

	(MSB)							(LSB)
TCON	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

- TF1: *Flag* de *overflow* de Timer 1. Setado por hardware no *overflow* do Timer 1. Resetado por hardware quando a CPU atender a interrupção.
- TR1: Bit de "RUN" do Timer 1. Setado/resetado por software para ativar/desativar o Timer 1.
- TF0: *Flag* de *Overflow* do Timer 0.
- TR0: Bit de "RUN" do Timer 0.
- IE1: *Flag* de borda da interrupção externa 1. Setado por hardware quando a CPU detectar uma borda em INT1. Resetado quando a CPU atender a interrupção.

- IT1: Bit de tipo de interrupção 1. Setado/resetado por software para especificar ativação por borda de descida/nível baixo em INT1.
- IE0: *Flag* de borda da interrupção 0.
- IT0: Bit de tipo de interrupção 0.

  Se IT0 = 1 -> borda de descida

  Se IT0 = 0 -> nível baixo

- Modo 1

O modo 1 é semelhante ao Modo 0, com a diferença de que o contador, neste modo, opera com todos os 16 bits.

- Modo 2

O modo 2 configura o Timer 0/1 como um contador de 8 bits (TL1) com recarga automática (auto-*reload*). O modo 2 é o mesmo para o Timer 0 e 1.

O registrador TH1 (8 bits) contém o valor de recarga, que é transferido ao TL1 a cada *overflow*. O bit TF1 (interrupção interna) é setado a cada *overflow*. O valor de recarga é inicializado em TH1 por software.

- Modo 3

Este modo só tem efeito sobre o Timer 0. O Timer 1, quando configurado neste modo, simplesmente mantém seu valor (como se TR1=0). O Timer 0, quando configurado no modo 3, opera como dois contadores de 8 bits separados (TL0 e TH0). O registrador TL0 utiliza os bits de controle de Timer 0 (GATE, C/T, TR0), a entrada INT0 e ativa o flag de interrupção TF0. O TH0, por outro lado, opera somente como temporizador (contador de Ciclos de Máquina) e utiliza o bit TR1 (do Timer 1) para seu controle. Além disso, o TH0 ativa o *flag* TF1 em seu *overflow*. Quando o Timer 0 estiver programado no modo 3, o Timer 1 poderá ser programado nos modos 0, 1 e 2, com a restrição de não setar o *flag* TF1 e, portanto, não gerar interrupções de contagem. O Timer 1, neste caso, poderá gerar interrupções para a porta serial, definindo a taxa de transmissão (*baud-rate*).

2.6) Porta Serial

O 8051 possui uma interface serial "*full-duplex*" duplamente "*bufferizada*" que pode ser programada para operar nos modos síncrono ou assíncrono com taxa de transmissão variável e quadro de transmissão de 8, 10 ou 11 bits.

A transmissão/recepção serial é feita através dos pinos TXD (P3.1) e RXD (P3.2). O pino RXD é amostrado periodicamente e quando a porta serial detecta um "*start-bit*" válido, o dado correspondente é carregado no registrador SBUF.

O *flag* de interrupção serial (RI) é, então, setado e a rotina de atendimento da porta serial (se estiver habilitada) será executada.

A CPU tem acesso ao byte recebido, executando uma operação de leitura em SBUF. Na transmissão, a CPU escreve o dado a ser enviado em SBUF e a porta serial encarrega-se de transmiti-lo através do pino TXD.

A porta serial do 8051 pode operar em quatro diferentes modos:

- **Modo 0**: Transmissão síncrona com taxa de transmissão fixa. O pino RXD é usado como entrada e saída dos dados enquanto o pino TXD emite o sinal de *clock* para a transmissão.

- **Modo 1**: Transmissão/Recepção assíncrona com quadro de 10 bits (1 *start* bit, 8 bits de dados, e 1 *stop* bit). Os bits são transmitidos através de TXD e recebidos por RXD. Na recepção, o *stop* bit é carregado no bit RB8 do registrador SCON. A taxa de transmissão é variável.

- **Modo 2**: Transmissão/Recepção assíncrona com quadro de 11 bits (1 *start* bit, 8 bits de dados, 1 bit de paridade, e 1 *stop* bit). O bit de paridade (TB8 do registrador SCON) pode ser atribuído por software antes da transmissão fazendo TB8 <- PSW.0. Na recepção, o bit de paridade é carregado em RB8 e o *stop* bit é ignorado. A taxa de transmissão pode ser programada para 1/32 ou 1/64 da freqüência do oscilador.

- **Modo 3**: Transmissão/Recepção assíncrona com quadro de 11 bits. O modo 3 é igual ao modo 2 exceto pela taxa de transmissão, que é variável.

Em todos os modos, a transmissão serial inicia por uma instrução que utiliza o SBUF como registrador de destino. A recepção inicia, no modo 0, quando RI = 0 e REN = 1 e nos outros modos quando for detectado um *start* bit e REN = 1. O registrador SCON contém os bits de controle da porta serial.

	(MSB)							(LSB)
SCON	SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0	SM1	MODO	DESCRIÇÃO	TAXA TRANSMISSÃO
0	0	0	*Shift Register*	Osc./12
0	1	1	UART 8 bits	Variável
1	0	2	UART 9 bits	Osc./64 ou osc./32
1	1	3	UART 9 bits	Variável

Tabela 3. Combinações de SM0 e SM1, determinando o modo e a taxa de transmissão

- SM2: Habilita a comunicação entre processadores nos modos 2 e 3. Nestes modos, se SM2 = 1, o *flag* RI não será ativado se o 9º bit recebido (RB8) for zero. No modo 1, se SM2 = 1, o *flag* RI não será ativado se não for detectado um *stop* bit válido. No modo 0, o SM2 deve ser 0.
- REN: Habilita a recepção serial. Setado/resetado para habilitar/desabilitar a recepção.
- TB8: Representa o 9º bit de dados a ser transmitido nos modos 2 e 3. Setado/resetado por software pelo programador.
- RB8: Nos modos 2 e 3, recebe o 9º bit de dados da recepção. No modo 1, se SM2 = 0, o bit RB8 recebe o *stop* bit recebido. No modo 0, o RB8 não é usado.
- TI: *Flag* de interrupção de transmissão. Setado por hardware ao final do 8º bit transmitido (modo 0) ou ao final do *stop* bit (nos outros modos). O *flag* TI deve ser resetado por software.
- RI: *Flag* de interrupção de recepção. Setado por hardware ao final do 8º bit recebido (modo 0) ou durante a recepção do *stop* bit (outros modos). Deve ser resetado por software.

A Porta Serial no Modo 0

O modo 0 é usado para expandir a capacidade de E/S do 8051, transferindo o conteúdo de SBUF serialmente através de RXD e permitindo que um registrador de deslocamento externo utilize o sinal de *clock*, emitido em TXD, para armazenar o dado transmitido. Da mesma forma na recepção, RXD recebe os dados serialmente e um registrador de deslocamento interno coloca estes dados no barramento interno do 8051.

A Porta Serial no Modo 1

No modo 1 são transmitidos e recebidos 10 bits através de TXD e RXD. O quadro de transmissão é composto por:

- 1 *start* bit;
- 8 bits de dados;
- 1 *stop* bit.

A taxa de transmissão serial é determinada pela taxa de *overflow* do Timer 1.

A Porta Serial nos Modos 2 e 3

Nos modos 2 e 3, são transmitidos e recebidos 11 bits através de TXD e RXD. O quadro de transmissão/recepção é formado por:

- 1 *start* bit;
- 8 bits de dados;
- 1 (9º) bit (normalmente paridade);
- 1 *stop* bit.

Na transmissão, o bit TB8 é carregado pelo programador a partir do bit de paridade (PSW.0). Na recepção, o 9º bit é levado a RB8 (SCON.2). O trecho de programa abaixo mostra a carga do bit de paridade em TB8 e a transmissão do conteúdo do ACC pela porta serial.

```
MOV A , < dado >     ; dado a ser transmitido é levado ao ACC
```

```
MOV C , P          ; CY:= Paridade
MOV TB8 , C        ; TB8:= Paridade
MOV SBUF , A       ; Transmissão serial do dado
```

No modo 2, a taxa de transmissão é definida por 1/32 ou 1/64 da freqüência do oscilador (dependendo do bit SMOD). No modo 3, a taxa de transmissão serial é determinada pela taxa de *overflow* do Timer 1.

Taxas de Transmissão

No modo 0, a taxa de transmissão é fixa e igual a OSC/12.

No modo 2, a taxa de transmissão depende do bit SMOD (PCON.7). Se SMOD=0 (que é o valor de *reset*), a taxa é de OSC/64. Se SMOD=1, a taxa é OSC/32.

Usando o Timer 1 para Geração da taxa de transmissão

Nos modos 1 e 3, a taxa de transmissão depende da taxa de *overflow* do Timer 1. A taxa de *overflow* do Timer 1, por sua vez, depende do valor inicial carregado e da freqüência do oscilador. A taxa de transmissão nos modos 1 e 3 pode ser calculada por:

```
Taxa TX = [ 2 ^ SMOD * (Taxa Overflow Timer 1)] / 32
```

Supondo que o Timer 1 seja configurado para operar no modo auto-*reload*, a taxa de transmissão pode ser calculada por:

```
Taxa TX = ( 2 ^ SMOD * fosc) / (32 * 12 * (256 - TH1))
```

Usando o Timer 2 para Geração da taxa de transmissão (apenas no 8052)

No 8052, o Timer 2 é selecionado como o gerador da taxa de transmissão pelo "*setting*" TCLK e/ou RCLK em T2CON. Nota-se então que a taxa de *baud* para transmitir e receber podem ser

simultaneamente diferentes. O modo de geração da taxa de *baud* é similar para o modo auto-*reload*, na que um "*rollover*" no TH2 torna os registros do Timer 2 para ser recarregado com o valor de 16 bits nos registradores RCAP2H e RCAP2L, os quais são pré-setados por software.

Agora, a taxa de *baud* nos Modos 1 e 3 são determinadas pela taxa de *overflow* do Timer 2 como segue:

Taxa TX = [Taxa Overflow Timer 2] / 16

O Timer pode ser configurado para qualquer dos dois modos de operação "timer" ou "contador". Em aplicações típicas, é configurado para operação "timer" (C/T2 = 0). A operação "Timer" é uma pequena diferença para Timer 2 quando iniciamos usando como um gerador de taxa de baud. Normalmente, como um timer será incrementado a cada ciclo de máquina (desta forma 1/12 a freqüência de oscilação). Como a geração da taxa de *baud*, entretanto, incrementamos cada um dos tempos de estado (*state time*) (assim ½ a freqüência de oscilação). Nestes casos a taxa de *baud* é dada pela fórmula:

Taxa TX = (fosc) / (32 * [65536 - (RCAP2H, RCAP2L)])

onde (RCAP2H, RCAP2L) é o conteúdo de RCAP2H e RCAP2L tendo como um inteiro sem sinal de 16 bits.

O seguinte trecho de programa, faz parte do programa do BASIC residente do 8052, que ajusta automaticamente a taxa de transmissão via porta serial com outro sistema com o qual está interagindo.

```
                MOV     R3,#00H     ; INITIALIZE the AUTO BAUD COUNTERS
                MOV     R1,#00H     ;
                MOV     R0,#04H     ;
                JB      RXD,$       ; LOOP UNTIL a START BIT is RECEIVED
                ;
                ;
RESET5:         DJNZ    R0,$        ; WASTE 8 CLOCKS INITIALLY, SIX CLOCKS
                                    ; IN THE LOOP (16) TOTAL
                CLR     C           ; 1 CLOCK (1)
                MOV     A,R1        ; 1 CLOCK (2)
                SUBB    A,#1        ; 1 CLOCK (3)
                MOV     R1,A        ; 1 CLOCK (4)
                MOV     A,R3        ; 1 CLOCK (5)
```

```
SUBB    A,#00H          ; 1 CLOCK -- R3:R1 = R3:R1 - 1 (6)
MOV     R3,A            ; 1 CLOCK (7)
MOV     R0,#3           ; 1 CLOCK (8)
JNB     RXD,RESET5      ; 2 CLOCKs (10), WAIT FOR END OF SPACE
JB      RXD,$           ; WAIT FOR THE SPACE TO END (20H)
JNB     RXD,$           ; WAIT FOR THE STOP BIT
MOV     RCAP2K,R3       ; LOAD THE TIMER2 HOLDING REGISTERS
MOV     RCAP2L,R1       ;
;
;
```

Comunicação entre Processadores

Os modos 2 e 3 têm uma característica especial que torna-se muito útil na comunicação entre processadores. Nestes modos, são enviados 9 bits de dados. Na recepção, o 9º bit é armazenado em RB8. A porta serial pode ser programada de tal forma que, quando um dado for recebido, só haverá interrupção se o bit RB8=1. Esta programação é feita com o bit SM2 (SCON.5)=1. Esta característica pode ser aproveitada para a comunicação entre processadores da seguinte forma:

Quando o processador-Mestre deseja enviar uma mensagem a um dos processadores-escravos, ele envia primeiro o endereço do processador com quem ele deseja se comunicar. Este endereço deve diferir dos dados de comunicação pelo fato de que ele deverá interromper todos os processadores, enquanto que os dados não devem interromper os processadores que não fazem parte da comunicação. Logo, o endereço deverá ter o 9º bit = 1 para que todos os escravos sejam interrompidos, analisem o valor do endereço e reconheçam o destino correto. Quando o processador endereçado iniciar a comunicação com o processador-mestre, os dados envolvidos nesta comunicação (RB8=0) não causarão interrupção nos demais processadores.

O procedimento de comunicação pode ser sintetizado pelos seguintes passos:

1. O processador-mestre envia o byte de endereço com o 9º bit setado. Isto faz com que os processadores-escravos sejam todos interrompidos.
2. Os processadores-escravos recebem o byte de endereço e verificam se a comunicação refere-se a eles.
3. O processador-escravo endereçado reseta o bit SM2 para permitir que os dados, que virão na sequencia, causem interrupções na sua porta serial. Durante a transmissão dos dados entre o processador-mestre e o processador endereçado, o 9º bit deve ser sempre 0, fazendo com que os outros processadores não sejam interrompidos.

2.7) Estrutura de Interrupção

O 8051 possui uma estrutura de interrupção com 5 fontes e o 8052 provê 6 fontes de interrupção e dois níveis de prioridade. As fontes de interrupção são:

- Timer 0;
- Timer 1;
- Timer 2 (apenas no 8052);
- Porta Serial;
- Interrupção Externa 0 (INT0);
- Interrupção Externa 1 (INT1).

As interrupções externas INT0 e INT1 cada uma delas podem ser ativadas pelo nível ou pela transição de nível, dependendo dos bits IT0 e IT1 no registrador TCON. Os *flags* que atualmente gera estas interrupções são os bits IE0 e IE1 de TCON. Quando uma interrupção externa é gerada, o *flag* que geram é zerado pelo hardware quando a rotina de serviço é vetorizada para somente se a interrupção está ativada pela transição.

Se a interrupção está ativada pelo nível, então a origem de requisição externa é que controla o *flag* de requisição.

A interrupção Timer 0 e Timer 1 são geradas por TF0 e TF1, a qual são setadas por um "*rollover*" nos seus respectivos registradores Timer/Contadores (exceto Timer 0 no Modo 3). Quando uma interrupção de *timer* é gerada, o *flag* que gera é zerado pelo hardware "*on-chip*" quando a rotina de serviço é vetorizada.

A interrupção da porta serial é gerada por um OR lógico de RI e TI. Nenhum destes *flags* é zerado por hardware quando a rotina de serviço é vetorizado. De fato, a rotina de serviço normalmente tem determinado se estava RI ou TI que gerou a interrupção, e o bit terá de ser zerado em software.

Todos os bits que geraram interrupção podem ser setados ou zerados por software, com o mesmo resultado como embora tenha sido setado ou zerado por hardware. Isto é, as interrupções podem ser geradas ou interrupções pendentes podem ser cancelados em software.

Cada uma destas fontes de interrupções podem ser individualmente habilitadas ou desabilitadas pelo *setting* ou o zeramento de um bit no SFR (*Special Function Register*) IE. O IE contém também um bit de desabilitação global, EA, a qual desabilita todas as interrupções de uma vez.

Quando uma das fontes solicita interrupção, o 8051 (8052) executa um salto para o endereço de atendimento da fonte requisitante. Os endereços são:

FONTE	ENDEREÇO
INT 0	3 (0003 H)
TIMER 0	11 (000B H)
INT 1	19 (0012 H)
TIMER 1	27 (001B H)
PORTA SERIAL	35 (0023 H)
TIMER 2	43 (002B H) *

Tabela 4. Fontes de Interrupção e os seus respectivos endereços

Nos endereços de destino da tabela acima, normalmente encontram-se as instruções de JUMP para o início da rotina de atendimento.

O pedido de interrupção é feito setando o *flag* de interrupção correspondente nos registradores TCON e SCON. Os *flags* de interrupção são:

FONTE	FLAG	LOCALIZAÇÃO
INT0	IE0	TCON.1
TIMER0	TF0	TCON.5
INT1	IE1	TCON.3
SERIAL (TX)	TI	SCON.1
SERIAL (RX)	RI	SCON.0

Tabela 5. Fonte de Interrupção e respectiva localização

Aceitação do pedido de Interrupção

Um determinado pedido de interrupção será aceito se:

- a fonte requisitante estiver habilitada, e
- não houver outro pedido com maior prioridade sendo atendido.

O 8051 possui um registrador que armazena os bits de habilitação de cada fonte individualmente e um bit de habilitação geral. Este registrador (IE - *Interrupt Enable*) está mostrado abaixo.

	(MSB)							(LSB)
IE	EA	-----	ET2	ES	ET1	EX1	ET0	EX0

- EA: *Enable* ALL - Habilita/desabilita todas as interrupções independentemente dos bits IE.0 a IE.4.
- ET2: *Enable Timer* 2 - Habilita/desabilita a interrupção do temporizador/contador 2 (apenas 8052)
- ES: *Enable Serial Port* - Habilita/desabilita as interrupções (TI e RI) da porta serial.
- ET1: *Enable Timer* 1 - Habilita/desabilita a interrupção do temporizador/contador 1.
- EX1: *Enable External Interrupt* 1 - Habilita/desabilita a interrupção externa INT1.
- ET0: *Enable Timer* 0 - Habilita/desabilita a interrupção do temporizador/contador 0.
- EX0: *Enable External Interrupt* 0 - Habilita/desabilita a interrupção externa INT0.

Interrupções de Temporização/Contagem

As interrupções dos Timers 0 e 1 são geradas pelos *flags* TF0 e TF1 respectivamente, quando o *timer* correspondente sofrer *overflow*. Quando for gerada uma interrupção de temporização/contagem, o *flag* que gerou esta interrupção é resetado por hardware quando a rotina de atendimento for iniciada. Os *flags* TF0 e TF1 pertencem ao registrador TCON e estão localizados nas posições TCON.5 e TCON.7 respectivamente.

Interrupção da Porta Serial

As interrupções da porta serial são geradas por um OR lógico entre os *flags* RI (recepção serial) e TI (transmissão serial). Os *flags* RI e TI não são resetados pela CPU no atendimento a interrupções, e eles devem ser resetados pelo programador como parte da rotina de atendimento.

Interrupções Externas

As interrupções INT0 e INT1 podem ser ativadas por nível (baixo) ou borda (de descida) dependendo dos bits IT0 e IT1 no registrador TCON. Os *flags* que geram as interrupções externas são IE0 (TCON.1) e IE1 (TCON.3). Quando a CPU executa o atendimento a interrupções externas, os *flags* IE0 e IE1 são resetados somente se a interrupção em questão estiver programada para ativação por borda. No caso de ativação por nível, o *flag* de interrupção será controlado pela fonte externa.

Prioridade de Atendimento

Cada fonte de interrupções pode ser programada, individualmente, para operar em um dos dois níveis de prioridade através do registrador IP (*Interrupt Priority Register*).

	(MSB)							(LSB)
IP	-----	-----	PT2*	PS	PT1	PX1	PT0	PX0

- PT2*: Define a prioridade do Timer 2 (apenas 8052)
- PS: Define a prioridade (1 = alta; 0 = baixa) da porta serial
- PT1: Define a prioridade do Timer 1
- PX1: Define a prioridade da interrupção externa 1
- PT0: Define a prioridade do Timer 0
- PX0: Define a prioridade da interrupção externa 0

Uma interrupção com prioridade baixa pode ser interrompida por uma interrupção com prioridade alta mas não pode ser interrompida por outra interrupção com mesma prioridade. Uma interrupção com prioridade alta não pode ser interrompida por nenhuma outra interrupção.

Se duas interrupções com prioridades diferentes forem recebidas simultaneamente, a interrupção com prioridade alta será atendida primeiro. Se duas interrupções de mesma prioridade forem recebidas simultaneamente, a CPU realiza um "*polling*" por hardware para determinar a sequencia de atendimento. Sendo assim, cada nível de prioridade possui ainda uma segunda estrutura de prioridade determinada pela figura 12.

FONTE	PRIORIDADE ENTRE NÍVEIS
IE0	(Mais Alta)
TF0	
IE1	
TF1	
RI + TI	
TF2 + EXF2*	(Mais Baixa)

Figura 12. Fontes de Interrupção e prioridade entre os níveis

OBS: A prioridade mostrada acima só é válida para o caso de pedidos simultâneos de fonte com mesma prioridade.

Atendimento a Interrupções

Os *flags* de interrupção são armazenados em S5P2 de cada Ciclo de Máquina. Os *flags* são examinados pelo circuito de "*polling*" no Ciclo de Máquina seguinte. Se um dos *flags* estava setado em S5P2 do ciclo anterior, o circuito de "*polling*" detectará e o Sistema de Interrupção gerará uma instrução LCALL (*long* CALL) para as rotina de atendimento em questão. Esta instrução gerada por hardware será inibida se:

1. Uma interrupção com igual ou maior nível de prioridade estiver em execução; ou
2. O ciclo de máquina no qual o "*polling*" foi executado não é o último CM da instrução atual; ou
3. A instrução atual é o RETI (retorno da interrupção) ou qualquer acesso aos registradores IE ou IP.

A condição 2 assegura que a instrução atual será completada antes do atendimento à interrupção. A condição 3 assegura que ao menos uma instrução será executada depois de um RETI antes do atendimento a uma outra interrupção e permite que as instruções que alteram a prioridade e habilitação das interrupções sejam processadas.

A instrução forçada (LCALL) empilha o PC (contador de programa) mas não empilha o PSW (*Program Status Word*). Depois, PC é carregado com o endereço da rotina de atendimento.

FONTE	VETOR DE INTERRUPÇÃO
IE0	0003 H
TF0	000B H
IE1	0013 H
TF1	001B H
RI + TI	0023 H
TF2 +EXF2*	002B H

Tabela 6. Interrupção e o endereço da rotina de interrupção

A rotina de interrupção inicia no endereço correspondente e prossegue até que seja encontrada uma instrução RETI. A instrução RETI informa ao processador que a rotina da interrupção terminou e

o valor do PC recebe os dois bytes do topo da pilha para retornar ao programa no ponto onde foi interrompido.

Interrupções por Borda

As interrupções externas (INT0 e INT1) podem ser programadas para ativação por borda de descida (transição 1 -> 0). Neste caso, a entrada externa deve permanecer em nível baixo por 12 períodos de oscilador para que possa ser reconhecida pelo Sistema de Interrupção. A transição 0 -> 1 poderá ocorrer a qualquer momento após os 12 períodos de oscilador, mas deverá permanecer em nível alto pelo mesmo período antes da nova ativação.

Interrupção por Nível

Quando as interrupções externas são programadas para ativação por nível, a fonte de interrupção deve manter a entrada ativa (em nível baixo) até que a interrupção seja atendida e deverá desativá-la antes do final da rotina de atendimento, para evitar um novo atendimento em seguida.

2.8) Reset da CPU e Redução de Consumo

O *reset* da CPU é feito através do pino RST mantendo-se um nível alto por, no mínimo, 24 períodos de oscilador. A CPU responde executando um *reset* interno que é iniciado no 2º ciclo após o nível alto em RST e repetido em cada ciclo de máquina até que seja aplicado um nível baixo em RST. O *reset* da CPU coloca os seguintes valores nos registradores:

REGISTRADOR	CONTEÚDO
PC	0000 H
ACC	00 H
B	00 H
PSW	00 H
SP	07 H
DPTR	0000 H
P0 - P3	0FF H
IP(8051)	XXX00000 B
IP(8052)	XX000000 B
IE(8051)	0XX00000 B
IE(8052)	0X000000 B
TMOD	00 H
SCON	00 H

REGISTRADOR	CONTEÚDO
TCON	00 H
TH0	00 H
TL0	00 H
TH1	00 H
TL1	00 H
SBUF	INDETERMINADO
PCON (CMOS)	0XXX0000 B
PCON (NMOS)	0XXXXXXX B
TH2(8052)	00 H
TL2(8052)	00 H
RCAP2H(8052)	00 H
RCAP2L(8052)	00 H

Tabela 7. Valores de r*eset* para os registradores da CPU

Power - On Reset

O *reset* automático é obtido quando a tensão Vcc de alimentação é conectada ao pino RST através de um capacitor de 10 μF e o mesmo ponto é ligado ao terra através de um resistor de 8,2KΩ.

Redução de Consumo

Para aplicações onde o consumo de potência é critico, as versões NMOS e CMOS possuem um modo de operação para redução de consumo.

- Redução de Consumo em Versões NMOS

Nas versões NMOS, a tensão Vcc de alimentação pode ser reduzida a zero enquanto a RAM interna é mantida por uma fonte de alimentação externa conectada ao pino RST. Para colocar a CPU no modo de redução de consumo, o programador deve primeiro habilitar a fonte externa no pino RST antes de retirar a tensão Vcc. Quando o 8051 for novamente alimentado, a fonte externa deverá permanecer ligada durante o tempo de *reset* e então poderá ser retirada.

- Redução de Consumo em Versões CMOS (*Idle* e *Power-Down*)

As versões CMOS possuem dois modos de operação para redução de consumo - o modo *Idle* e o modo *Power-Down*.

No modo *Idle*, o oscilador continua operando e os sistemas de interrupção, temporização e a porta serial continuam recebendo o sinal do oscilador. A CPU, no entanto não recebe o sinal do oscilador.

No modo "*Power-Down*", o oscilador é desativado completamente. Os modos "*Idle*" e "*Power-Down*" são ativados através do registrador PCON.

	(MSB)							(LSB)
PCON	SMOD	-----	-----	-----	GF1	GF0	PD	IDL

- SMOD: Controla a divisão por 2 da taxa de transmissão serial
- GF1: *Flag* de uso geral
- GF0: *Flag* de uso geral
- PD: Ativa o modo "*Power - Down*"
- IDL: Ativa o modo "*Idle*"

OBS: Se PD=IDL=1, a CPU é colocada no modo de "*Power-Down*".

CAPÍTULO 3 – CONJUNTO DE INSTRUÇÕES

Todos os membros da família MCS-51 executam o mesmo conjunto de instruções. O conjunto de instruções do 8051 proporciona uma ampla gama de modos de endereçamento para o acesso da RAM interna e fornece grande facilidade para as operações de byte em estruturas de dados pequenas. O 8051 fornece um sub-conjunto de operações de manipulação de bit, o que facilita o processamento de variáveis booleanas em aplicações de controle.

3.1) Modos de Endereçamento

A CPU do 8051 oferece os seguintes modos de endereçamento:

- Direto;
- Indireto via Registrador;
- Imediato;
- Indexado;
- Direta via Registrador;
- Implícito.

• Endereçamento Direto

No endereçamento direto, o operando é especificado por um endereço de 8 bits contido no *opcode* da instrução. Este modo só pode ser usado no acesso à RAM interna a aos SFRs.

• Endereçamento Indireto via Registrador

Neste modo, a instrução especifica um registrador que contém o endereço do operando. Para acessar operandos dentro do limite de 256 bytes, os registradores R0 e R1 do banco selecionado podem ser usados. Para acessar a Memória Externa em toda a sua extensão (64Kbytes), é usado o registrador DPTR. Nas operações de pilha (PUSH e POP), o registrador SP (*Stack Pointer*) é usado como ponteiro.

- **Endereçamento Imediato**

No endereçamento imediato, o operando faz parte do *opcode*. Por exemplo, a instrução MOV A,#100, a qual carrega o acumulador com o número decimal 100, que está contido no 2º byte.

- **Endereçamento Indexado**

O endereçamento indexado só é permitido para acessar a Memória de Programa. Este modo de endereçamento é utilizado para acessar tabelas de constantes residentes em ROM. Os registradores utilizados como base são o DPTR ou o PC, o Acumulador é usado como valor de "*offset*". O endereço efetivo do operando, neste modo, é calculado pela soma do registrador de base (PC ou DPTR) com o conteúdo do Acc.

- **Endereçamento Direto via Registrador**

Os bancos de registradores, que contém R0 a R7, podem se acessados por instruções que possuem um campo de 3 bits para especificar o registrador endereçado. O operando da instrução é o valor contido no registrador. Desta forma, eliminamos a necessidade de incluir um byte completo no *opcode*, o que torna a instrução eficiente do ponto de vista de espaço de memória. A seleção do banco de registradores a ser acessado é definida pelos bits RS0 e RS1 do registrador PSW. Por exemplo, a instrução MOV A,Rn.

- **Endereçamento Implícito**

Neste modo, o operando é único, ou seja, não há necessidade de especificá-lo no *opcode* da instrução. No 8051, algumas instruções sempre utilizam o Acumular ou o DPTR, de forma que a codificação da instrução resume-se a um único byte. Por exemplo, a instrução DIV AB.

3.2) Instruções Aritméticas

O conjunto de instruções aritméticas oferecido pelo 8051 está contido na tabela abaixo.

MNEMÔNICO		OPERAÇÃO
ADD	A,< byte >	A = A + < byte >
ADDC	A,< byte >	A = A + < byte > + C
SUBB	A,< byte >	A = A - < byte > - C
INC	A	A = A + 1
INC	< byte >	< byte > = < byte > + 1
INC	DPTR	DPTR = DPTR + 1
DEC	A	A = A - 1
DEC	< byte >	< byte > = < byte > - 1
MUL	AB	B: A = B x A
DIV	AB	A = Inteiro [A/B]; B = resto [A/B]
DA	A	Ajuste Decimal

Tabela 8. Instruções Aritméticas

3.3) Instruções Lógicas/Manipulação de Variáveis Booleanas

As instruções lógicas do 8051 realizam operações Booleanas (AND, OR, XOR, ..) em operandos de 8 bits. As instruções que manipulam variáveis Booleanas realizam operações sobre operandos de 1 bit.

As instruções lógicas estão mostradas na tabela abaixo:

MNEMÔNICO		OPERAÇÃO
ANL	A ,< byte >	A = A AND < byte >
ANL	< byte >,A	< byte > = < byte > AND A
ANL	< byte >,#DATA	< byte > = < byte > AND #DATA
ORL	A,< byte >	A = A OR < byte >
ORL	< byte >,A	< byte > = < byte > OR A
ORL	< byte >,#DATA	< byte > = < byte > OR #DATA
XRL	A ,< byte >	A = A XOR < byte >
XRL	< byte >,A	< byte > = < byte > XOR A
XRL	< byte >,#DATA	< byte > = < byte > XOR #DATA
CLR	A	A = 00H
CPL	A	A = .NOT. A
RL	A	Rot. Acc p/ esq. 1 Bit
RLC	A	Rot. Acc p/ esq. com CY
RR	A	Rot. Acc p/ dir. 1 Bit
RRC	A	Rot. Acc p/ dir. com CY
SWAP	A	Troca *nibbles* do Acc

Tabela 9. Instruções Lógicas

As instruções que manipulam variáveis Booleanas estão mostradas na tabela abaixo:

MNEMÔNICO		OPERAÇÃO
ANL	C,bit	C = C AND bit
ANL	C,/bit	C = C AND NOT bit
ORL	C,bit	C = C OR bit
ORL	C,/bit	C = C OR NOT bit
MOV	C,bit	C = bit
MOV	bit,C	bit = C
CLR	C	C = 0
CLR	bit	bit = 0
SETB	C	C = 1
SETB	bit	bit = 1
CPL	C	C = NOT C
CPL	bit	bit = NOT bit
JC	rel	Jump se C = 1
JNC	rel	Jump se C = 0
JB	bit,rel	Jump se bit = 1
JNB	bit,rel	Jump se bit = 0
JBC	bit,rel	Jump se bit = 1; CLR bit

Tabela 10. Instruções Manipulam Variáveis Booleanas

Todos os acessos a bit são feitos com endereçamento direto. Os bits com endereços de 00H até 7FH estão na parte baixa (128 primeiros bytes da RAM interna) e os bits com endereços de 80H até FFH estão no espaço dos SFRs.

3.4) Instruções de Transferência de Dados

As instruções de transferência de dados realizam operações de movimentação de conteúdo entre a memória e o operando especificado pelo modo de endereçamento.

Estas instruções podem efetuar transferências de/para as Memórias Interna/Externa.

- **Transferência de e para a Memória Interna**

As instruções que transferem dados de/para a Memória Interna do 8051 estão apresentadas na tabela abaixo:

MNEMÔNICO		OPERAÇÃO
MOV	A,< src >	A = < src >
MOV	< dest >,A	< dest > = A
MOV	< dest > , < src >	< dest > = < src >
MOV	DPTR,#DATA16	DPTR = dado imed. 16 bits
PUSH	< src >	INC SP:MOV"@SP", <src>
POP	< dest >	MOV <dest>,"@SP": DEC SP
XCH	A,< byte >	ACC <--> < byte >
XCHD	A,@Ri	ACC(3-0) <--> @Ri(3-0)

Tabela 11. Instruções de Transferência de dados para Memória Interna

As instruções de PUSH e POP só aceitam endereçamento direto para especificar o operando a ser empilhado ou retirado da pilha, embora o endereço efetivo na pilha sempre é calculado indiretamente (usando SP como ponteiro). Sendo assim, a pilha não pode residir no espaço dos SFRs pois nesta área, só é permitido o acesso direto.

- **Transferência de/para a Memória Externa**

O 8051 permite que sejam feitas transferências de dados entre um operando e as memórias externas de Dados e Programa. As transferências de dados envolvendo a Memória de Programa, no entanto, só podem ser de leitura. As instruções que acessam a Memória Externa são MOVX (Memória Externa de Dados) e MOVC (Memória de Programa). A tabela abaixo apresenta as transferências de dados obtidas com as instruções MOVX.

Tamanho do Endereço	Mnemônico	Operação
8 bits	MOVX A,@Ri	Leitura RAM externa em @Ri
8 bits	MOVX @Ri,A	Escrita RAM externa em @Ri
16 bits	MOVX A,@DPTR	Leitura RAM externa em @DPTR
16 bits	MOVX @DPTR,A	Escrita RAM externa em @DPTR

Tabela 12. Instruções Transferência para Memória Externa

O acesso à Memória de Programa, feito pela instrução MOVC (*Move Code*), é usado para acessar tabelas de constantes em ROM. Estas instruções são conhecidas como instruções de "*Lookup Table*" e estão mostradas na tabela abaixo.

Mnemônico	Operação
MOVC A,@A+DPTR	Leitura da ROM em (A+DPTR)
MOVC A,@A+PC	Leitura da ROM em (A+PC)

Tabela 13. Instruções de Acesso à Memória de Programa

3.5) Instruções de Salto

As instruções de salto podem ser divididas em saltos condicionais e incondicionais. As tabelas abaixo mostram as instruções de salto do 8051.

SALTO CONDICIONAL

Mnemônico		Operação
JZ	rel	Jump se Acc = 0
JNZ	rel	Jump se Acc ≠ 0
DJNZ	< byte >,rel	Decrementa e salta se <byte> ≠0
CJNE	A,<byte>,rel	Compara e salta se A ≠ <byte>
CJNE	<byte>,#data,rel	Compara e salta se <byte> ≠ #data

Tabela 14. Instruções de Salto Condicional

SALTO INCONDICIONAL

Mnemônico		Operação
JMP	addr	Salta para addr
JMP	@A+DPTR	Salta para A+DPTR
CALL	addr	Chama subrotina em addr
RET		Retorna da subrotina
RETI		Retorna da interrupção
NOP		Nenhuma operação
LJMP	addr	*Long* Jump - (qualquer posição da Memória de Programas)
SJMP	rel	*Short* Jump - (Salto de ± 128 bytes)
AJMP	addr	*Absolute* Jump - (Salto dentro da pág. corrente 2K)
LCALL	addr	*Long* CALL - (Chama rotina - 64K da Memória Programa)
ACALL	addr	*Absolute* CALL - (Chama rotina na página corrente 2K)

Tabela 15. Instruções de Salto Incondicional

CAPÍTULO 4 – PROGRAMAÇÃO EM ASSEMBLY 8051

A seguir mostramos alguns exemplos de programas em Assembly para 8051, com base nas instruções do Anexo 1.

4.1) Programa de Extração da Centena-Dezena-Unidade

```
;---------------------------------------- ------------------------------------
;
; Programa: Extração da Centena-Dezena-Unidade
;
;---------------------------------------- ------------------------------------

                org 00h                 ;Posição de Reset

                sjmp inicio             ;Salto para o início do Código

                org 30h                 ;Endereço de início do programa

inicio:         mov     r1,#00          ;inicio do programa
                mov     r2,#00          ;Zera os registradores
                mov     r3,#00
                mov     a,p1            ;Leitura do valor
                mov     b,#100d         ;Armazenamento do valor 100d
                mov     r4,a
                subb    a,b
                mov     a,r4
                jc      dezena          ;Teste de valor >= 100d
                div     ab
                mov     r1,a            ;centena
                mov     a,b
dezena:         mov     b,#10d          ;Armazenamento do valor 10d
                mov     r4,a
                subb    a,b
                mov     a,r4
                jc      unidade         ;Teste de valor >= 10d
                div     ab
                mov     r2,a            ;Dezena
                mov     a,b
unidade:        mov     r3,a            ;Unidade
                sjmp    $               ;Fim do programa
```

4.2) Programa de Obtenção do Maior valor em um Vetor

```
;---------------------------------------- ------------------------------------
;
; Programa: Obtenção do Maior valor em um vetor (30h a 40h)
;
;---------------------------------------- ------------------------------------
                org     00h             ;Reset
                sjmp    inicio          ;Salta para inicio do programa
                org     30h             ;Endereco do inicio do programa
inicio:         mov     r0,#30h         ;Endereco do inicio do vetor
                mov     r3,#00          ;Maior valor <--- 00
salto1:         mov     a,@r0           ;Leitura da posicao do vetor
                mov     b,a
                mov     a,r3
                clr     c
                subb    a,b
                jnc     salto2          ;Teste para ver se eh maior
                mov     r3,b            ;Maior recebe valor
salto2:         inc     r0              ;Incrementa ponteiro
                mov     b,r0
                mov     a,#40h          ;Endereco de fim de vetor
                clr     c
                subb    a,b
                jnc     salto1          ;Teste de fim de vetor
                sjmp    $               ;Fim do programa
```

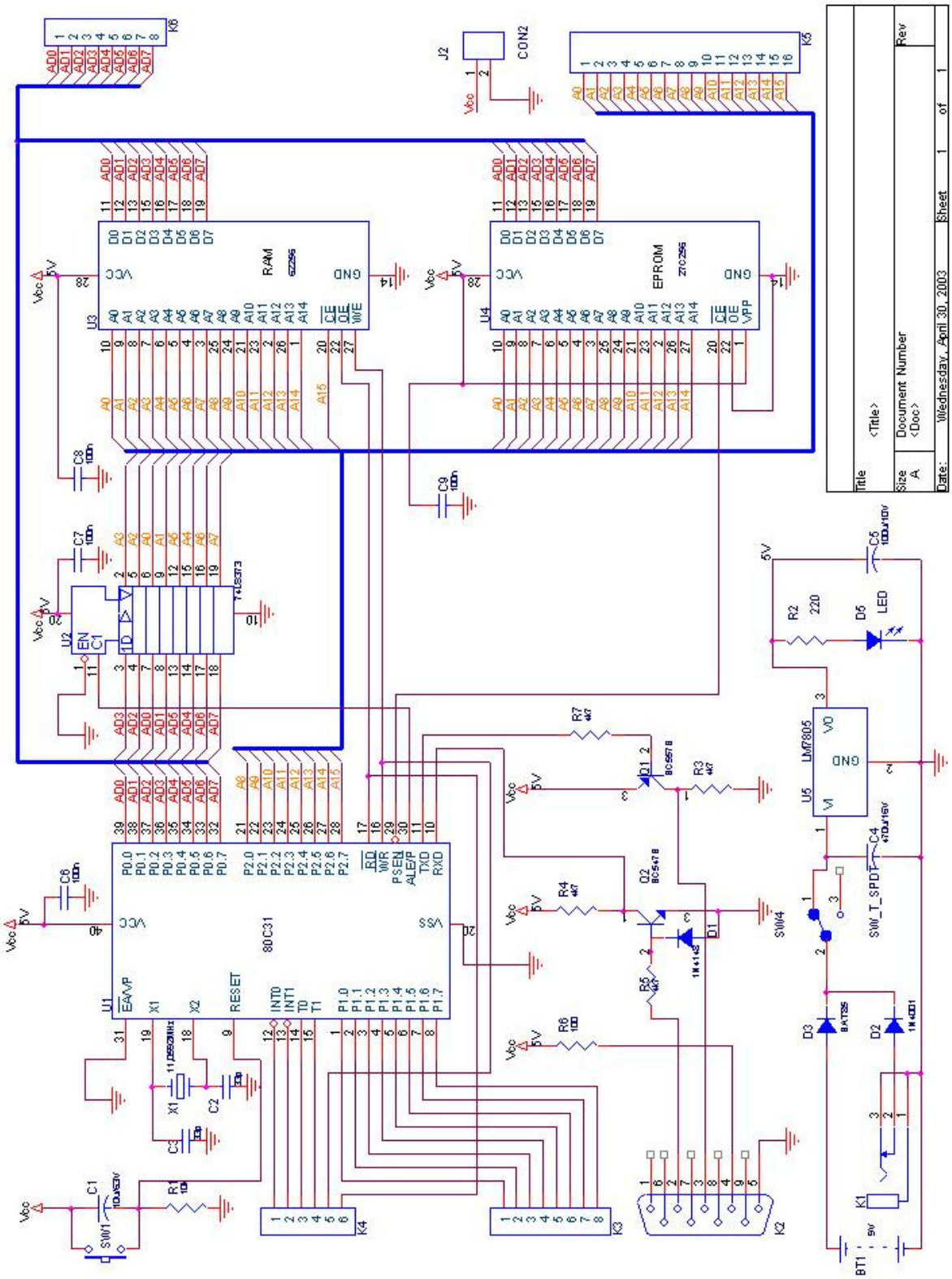
4.3) Programa de Potenciação de Po ^{P1}

```
;---------------------------------------- ------------------------------------
;
; Programa: Potenciação de P0^p1
;
;---------------------------------------- ------------------------------------
            org     00h       ;Reset
            sjmp    inicio    ;Salta para inicio do programa

            org     30h       ;Endereco de inicio do programa
inicio:     mov     a,p0      ;Leitura do numero
            mov     r0,a
            mov     a,p1      ;Leitura do expoente
            mov     r1,a
            mov     20h,#00   ;Posição de armazenamento
            mov     a,r0
            jz      fim       ;Testa se numero = zero
            mov     20h,#01h
loop:       mov     a,r1
            jz      fim       ;Testa se expoente = zero
            mov     a,20h
            dec     r1        ;Decrementa expoente
            mov     b,r0
            mul     ab        ;Multiplica
            mov     20h,a
            sjmp    loop      ;Volta para nova multiplicacao
fim:        sjmp    $         ;Fim do programa
```

4.4) Programa de Teste da Memória Principal

```
;---------------------------------------- ------------------------------------
;
; Programa: Teste de Memória Principal
;
;---------------------------------------- ------------------------------------
            org   00h
            sjmp  inicio
            org   30h


inicio:     mov   P1,#00            ; P1 <----- 00h
            mov   a,#10d
loop1:      nop
            djnz  a,loop1           ; delay

            mov   P1,#ffh           ; P1 <----- ffh

            mov   a,#10d
loop2:      nop
            djnz  a,loop2           ; delay

            mov   P1,#10101010b     ; P1 <----- 10101010b

            mov   DPTR,#0000h       ; Testa a memória com 0000h
zero:       mov   a,#00
            mov   b,#00
            movx  @dptr,a
            movx  a,@dptr
            cjne  a,b,erro
            inc   dptr
            mov   a,dph
            cjne  a,#ffh,zero
            mov   a,dpl
            cjne  a,#ffh,zero
            mov   DPTR,#0000h       ; Testa a memória com 10101010b
alter:      mov   a,#0101010b
            mov   b,#0101010b
            movx  @dptr,a
            movx  a,@dptr
            cjne  a,b,erro
            inc   dptr
            mov   a,dph
            cjne  a,#ffh,alter
            mov   a,dpl
            cjne  a,#ffh,alter
            mov   DPTR,#0000h       ; Testa a memória com ffh
uns:        mov   a,#ffh
            mov   b,#ffh
            movx  @dptr,a
            movx  a,@dptr
            cjne  a,b,erro
            inc   dptr
            mov   a,dph
            cjne  a,#ffh,uns
            mov   a,dpl
            cjne  a,#ffh,uns
            sjmp  $
erro:       mov   p1,#00h
            sjmp  $
```

CAPÍTULO 5 – PROJETOS DE SISTEMAS COM 8051



ANEXO 1 – Conjunto de Instruções do 8051

MCS®-51 INSTRUCTION SET

8051 Instruction Set Summary

Interrupt Response Time: Refer to Hardware Description Chapter.

Instructions that Affect Flag Settings(1)

Instruction	Flag			Instruction	Flag		
	C	OV	AC		C	OV	AC
ADD	X	X	X	CLR C	0		
ADDC	X	X	X	CPL C	X		
SUBB	X	X	X	ANL C,bit	X		
MUL	0	X		ANL C,/bit	X		
DIV	0	X		ORL C,bit	X		
DA	X			ORL C,bit	X		
RRC	X			MOV C,bit	X		
RLC	X			CJNE	X		
SETB C	1						

(1)Note that operations on SFR byte address 208 or bit addresses 209-215 (i.e., the PSW or bits in the PSW) will also affect flag settings.

Note on instruction set and addressing modes:

Rn — Register R7–R0 of the currently selected Register Bank.

direct — 8-bit internal data location's address. This could be an Internal Data RAM location (0–127) or a SFR [i.e., I/O port, control register, status register, etc. (128–255)].

@Ri — 8-bit internal data RAM location (0–255) addressed indirectly through register R1 or R0.

#data — 8-bit constant included in instruction.

#data 16 — 16-bit constant included in instruction.

addr 16 — 16-bit destination address. Used by LCALL & LJMP. A branch can be anywhere within the 64K-byte Program Memory address space.

addr 11 — 11-bit destination address. Used by ACALL & AJMP. The branch will be within the same 2K-byte page of program memory as the first byte of the following instruction.

rel — Signed (two's complement) 8-bit offset byte. Used by SJMP and all conditional jumps. Range is −128 to +127 bytes relative to first byte of the following instruction.

bit — Direct Addressed bit in Internal Data RAM or Special Function Register.

Mnemonic		Description	Byte	Oscillator Period
ARITHMETIC OPERATIONS				
ADD	A,Rn	Add register to Accumulator	1	12
ADD	A,direct	Add direct byte to Accumulator	2	12
ADD	A,@Ri	Add indirect RAM to Accumulator	1	12
ADD	A,#data	Add immediate data to Accumulator	2	12
ADDC	A,Rn	Add register to Accumulator with Carry	1	12
ADDC	A,direct	Add direct byte to Accumulator with Carry	2	12
ADDC	A,@Ri	Add indirect RAM to Accumulator with Carry	1	12
ADDC	A,#data	Add immediate data to Acc with Carry	2	12
SUBB	A,Rn	Subtract Register from Acc with borrow	1	12
SUBB	A,direct	Subtract direct byte from Acc with borrow	2	12
SUBB	A,@Ri	Subtract indirect RAM from ACC with borrow	1	12
SUBB	A,#data	Subtract immediate data from Acc with borrow	2	12
INC	A	Increment Accumulator	1	12
INC	Rn	Increment register	1	12
INC	direct	Increment direct byte	2	12
INC	@Ri	Increment direct RAM	1	12
DEC	A	Decrement Accumulator	1	12
DEC	Rn	Decrement Register	1	12
DEC	direct	Decrement direct byte	2	12
DEC	@Ri	Decrement indirect RAM	1	12

Tabela 16a. Conjunto de Instruções da família MCS-51

8051 Instruction Set Summary (Continued)

Mnemonic		Description	Byte	Oscillator Period
ARITHMETIC OPERATIONS (Continued)				
INC	DPTR	Increment Data Pointer	1	24
MUL	AB	Multiply A & B	1	48
DIV	AB	Divide A by B	1	48
DA	A	Decimal Adjust Accumulator	1	12
LOGICAL OPERATIONS				
ANL	A,Rn	AND Register to Accumulator	1	12
ANL	A,direct	AND direct byte to Accumulator	2	12
ANL	A,@Ri	AND indirect RAM to Accumulator	1	12
ANL	A,#data	AND immediate data to Accumulator	2	12
ANL	direct,A	AND Accumulator to direct byte	2	12
ANL	direct,#data	AND immediate data to direct byte	3	24
ORL	A,Rn	OR register to Accumulator	1	12
ORL	A,direct	OR direct byte to Accumulator	2	12
ORL	A,@Ri	OR indirect RAM to Accumulator	1	12
ORL	A,#data	OR immediate data to Accumulator	2	12
ORL	direct,A	OR Accumulator to direct byte	2	12
ORL	direct,#data	OR immediate data to direct byte	3	24
XRL	A,Rn	Exclusive-OR register to Accumulator	1	12
XRL	A,direct	Exclusive-OR direct byte to Accumulator	2	12
XRL	A,@Ri	Exclusive-OR indirect RAM to Accumulator	1	12
XRL	A,#data	Exclusive-OR immediate data to Accumulator	2	12
XRL	direct,A	Exclusive-OR Accumulator to direct byte	2	12
XRL	direct,#data	Exclusive-OR immediate data to direct byte	3	24
CLR	A	Clear Accumulator	1	12
CPL	A	Complement Accumulator	1	12

Mnemonic		Description	Byte	Oscillator Period
LOGICAL OPERATIONS (Continued)				
RL	A	Rotate Accumulator Left	1	12
RLC	A	Rotate Accumulator Left through the Carry	1	12
RR	A	Rotate Accumulator Right	1	12
RRC	A	Rotate Accumulator Right through the Carry	1	12
SWAP	A	Swap nibbles within the Accumulator	1	12
DATA TRANSFER				
MOV	A,Rn	Move register to Accumulator	1	12
MOV	A,direct	Move direct byte to Accumulator	2	12
MOV	A,@Ri	Move indirect RAM to Accumulator	1	12
MOV	A,#data	Move immediate data to Accumulator	2	12
MOV	Rn,A	Move Accumulator to register	1	12
MOV	Rn,direct	Move direct byte to register	2	24
MOV	Rn,#data	Move immediate data to register	2	12
MOV	direct,A	Move Accumulator to direct byte	2	12
MOV	direct,Rn	Move register to direct byte	2	24
MOV	direct,direct	Move direct byte to direct	3	24
MOV	direct,@Ri	Move indirect RAM to direct byte	2	24
MOV	direct,#data	Move immediate data to direct byte	3	24
MOV	@Ri,A	Move Accumulator to indirect RAM	1	12

Tabela 16b. Conjunto de Instruções da família MCS-51

8051 Instruction Set Summary (Continued)

Mnemonic		Description	Byte	Oscillator Period
DATA TRANSFER (Continued)				
MOV	@Ri,direct	Move direct byte to indirect RAM	2	24
MOV	@Ri, #data	Move immediate data to indirect RAM	2	12
MOV	DPTR, #data16	Load Data Pointer with a 16-bit constant	3	24
MOVC	A,@A + DPTR	Move Code byte relative to DPTR to Acc	1	24
MOVC	A,@A + PC	Move Code byte relative to PC to Acc	1	24
MOVX	A,@Ri	Move External RAM (8-bit addr) to Acc	1	24
MOVX	A,@DPTR	Move External RAM (16-bit addr) to Acc	1	24
MOVX	@Ri,A	Move Acc to External RAM (8-bit addr)	1	24
MOVX	@DPTR,A	Move Acc to External RAM (16-bit addr)	1	24
PUSH	direct	Push direct byte onto stack	2	24
POP	direct	Pop direct byte from stack	2	24
XCH	A,Rn	Exchange register with Accumulator	1	12
XCH	A,direct	Exchange direct byte with Accumulator	2	12
XCH	A,@Ri	Exchange indirect RAM with Accumulator	1	12
XCHD	A,@Ri	Exchange low-order Digit indirect RAM with Acc	1	12

Mnemonic		Description	Byte	Oscillator Period
BOOLEAN VARIABLE MANIPULATION				
CLR	C	Clear Carry	1	12
CLR	bit	Clear direct bit	2	12
SETB	C	Set Carry	1	12
SETB	bit	Set direct bit	2	12
CPL	C	Complement Carry	1	12
CPL	bit	Complement direct bit	2	12
ANL	C,bit	AND direct bit to CARRY	2	24
ANL	C,/bit	AND complement of direct bit to Carry	2	24
ORL	C,bit	OR direct bit to Carry	2	24
ORL	C,/bit	OR complement of direct bit to Carry	2	24
MOV	C,bit	Move direct bit to Carry	2	12
MOV	bit,C	Move Carry to direct bit	2	24
JC	rel	Jump if Carry is set	2	24
JNC	rel	Jump if Carry not set	2	24
JB	bit,rel	Jump if direct Bit is set	3	24
JNB	bit,rel	Jump if direct Bit is Not set	3	24
JBC	bit,rel	Jump if direct Bit is set & clear bit	3	24
PROGRAM BRANCHING				
ACALL	addr11	Absolute Subroutine Call	2	24
LCALL	addr16	Long Subroutine Call	3	24
RET		Return from Subroutine	1	24
RETI		Return from interrupt	1	24
AJMP	addr11	Absolute Jump	2	24
LJMP	addr16	Long Jump	3	24
SJMP	rel	Short Jump (relative addr)	2	24

Tabela 16c. Conjunto de Instruções da família MCS-51

8051 Instruction Set Summary (Continued)

Mnemonic		Description	Byte	Oscillator Period
PROGRAM BRANCHING (Continued)				
JMP	@A + DPTR	Jump indirect relative to the DPTR	1	24
JZ	rel	Jump if Accumulator is Zero	2	24
JNZ	rel	Jump if Accumulator is Not Zero	2	24
CJNE	A,direct,rel	Compare direct byte to Acc and Jump if Not Equal	3	24
CJNE	A, #data,rel	Compare immediate to Acc and Jump if Not Equal	3	24
CJNE	Rn, #data,rel	Compare immediate to register and Jump if Not Equal	3	24
CJNE	@Ri, #data,rel	Compare immediate to indirect and Jump if Not Equal	3	24
DJNZ	Rn,rel	Decrement register and Jump if Not Zero	2	24
DJNZ	direct,rel	Decrement direct byte and Jump if Not Zero	3	24
NOP		No Operation	1	12

Tabela 16d. Conjunto de Instruções da família MCS-51

ANEXO 2 – SAB 80C515/80C535

SAB 80C515/80C535 Microprocessadores CMOS de 8 bits

- **SAB 80C515/80C515-16:**

É um microprocessador CMOS com ROM programada por máscara.

- **SAB 80C535/80C535-16:**

É um microprocessador CMOS para ROM externa. Os integrados ACMOS SAB 80C515/80C535 são as últimas versões da família SAB 8051 da Siemens, e são 100% compatíveis com os SAB 80515/80535 fabricados com tecnologia MYMOS. OS novos integrados estão disponíveis em versões de 12MHz e 16MHz.

Cada integrado pode servir de processador e co-processador aritmético. Possui melhores características de aritmética binária e BCD e possui uma excelente capacidade de tratamento de bits solitários.

A utilização muito eficiente da memória resulta num conjunto de instruções com 44% de instruções de um só byte, 41% de instruções de 2 bytes e apenas 15% de instruções de 3 bytes. Controlado por um cristal de quartzo de 12 MHz, o que resulta em 58% das instruções são executadas em 1μs.

A arquitetura dos SAB 80C515/535 é baseada nas da família de microprocessadores SAB 8051/80C51. As seguintes características dos SAB 80C515/535 são 100% compatíveis com as dos SAB 80C51:

- Conjunto de instruções;
- Interface de extensão com memória externa (portas 0 e 2);
- Porta serial *Full-duplex;*
- Contadores/temporizadores 0 e 1;
- Funções alternativas da porta 3;
- Funções alternativas dos 128 bytes inferiores da RAM interna e dos 4Kbytes inferiores da ROM interna.

Estrutura Interna da família 80x535

A estrutura interna do 80535 é mostrada na figura 12 .

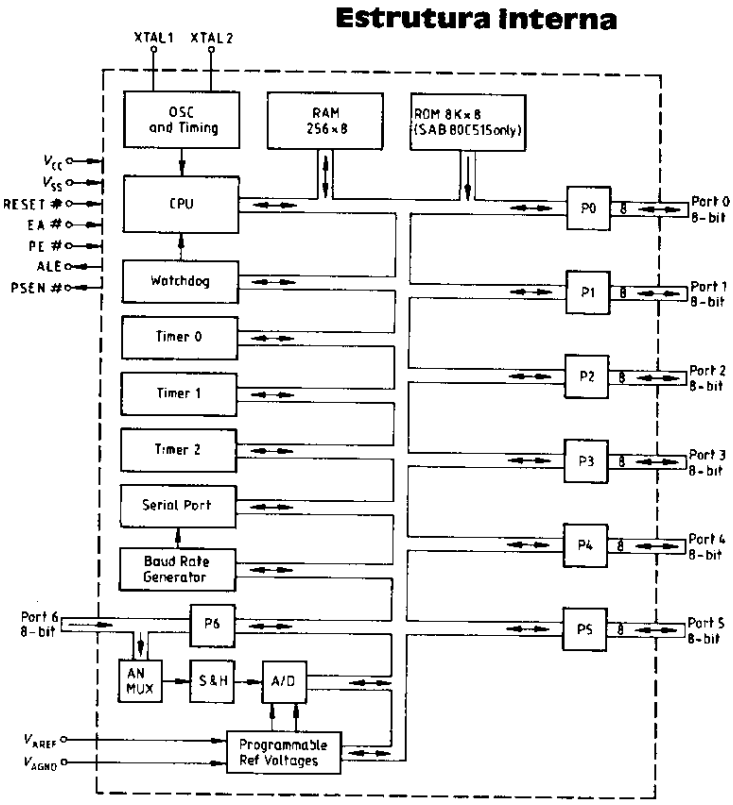


Figura 12. (Estrutura Interna do 80535)

Características Técnicas

As características técnicas do 80515 e 80535 são mostradas a seguir:

- ROM de 8x8kbits (só no 80C515);
- RAM de 8x256 bits;
- 6 portas de E/S de 8 bits;1 porta de entrada, digital ou analógica;
- Várias possibilidades de recarga, captura e comparação;
- Canal série *full-duplex*;
- 12 vetores de interrupção com 4 níveis de prioridade;

- Conversor A/D de 8 bits com de 8 entradas multiplexadas e tensões de referência internas programáveis;
- Temporizador "cão de guarda" de 16 bits (*watchdog*);
- Processador de álgebra de *Boole;*
- 256 posições de endereço por bit;
- Duração de execução da quase totalidade das instruções: 1µs (750ns);
- Multiplicação e divisão: 4µs (3µs);
- Memória externa extensível até 128Kbytes;
- Retro-compatibilidade com o SAB 8051;
- Funcionamento 100% compatível com o SAB 80515;
- Modos de funcionamento "*idle*" e "*power-down*".

BIBLIOGRAFIA

1. CIARCIA, Steve., ***Ciarcia's Circuit Cellar. Why Microcontrollers?***. Revista Byte Internacional. August. 1988. Pág. 239 a 247.
2. INTEL Corporation., ***Advance Information 8052AH-BASIC***. March 1985. Pág. 1 a 10
3. INTEL Corporation., ***MCS-51 Architectural Overview***. Pág. 5-1 a 5-19
4. INTEL Corporation., ***MCS-51 Programmer's Guide and Instruction Set***. Pág. 6-1 a 6-25
5. INTEL Corporation., ***Hardware Description 8051, 8052 e 80C51***. Pág. 7-1 a 7-33
6. INTEL Corporation., ***Application Note. AP-69***. An Introduction to the Intel MCS-51 Single-Chip Microcomputer Family. May 1980. Pág. 2-1 a 2-30
7. INTEL Corporation., ***Application Note. AP-70***. Using the Intel MCS-51 Boolean Processing Capabilities. April 1980. Pág. 2-31 a 2-71
8. INTEL Corporation., ***Application Brief. AB-40***. 32-bit Math Routines for the 8051. December 1987. Pág. 2-166 a 2-174
9. INTEL Corporation., ***Application Brief. AB-12***. Designing a Mailbox Memory for Two 80C31 Microcontrolers Using EPLDs. October 1987. Pág. 2-175 a 2-188
10. INTEL Corporation., ***Application Note. AP-252***. Designing With the 80C51BH. September 1987. Pág. 2-189 a 2-212
11. INTEL Corporation., ***Application Note. AP-410***. Enhanced Serial Port on the 83C51FA. November 1987. Pág. 2-213 a 2-220
12. INTEL Corporation., ***MCS-Basic-52***. Pág. 148 a 151
13. _____., **Microcontrolador 8051**. Traducomp Informática.