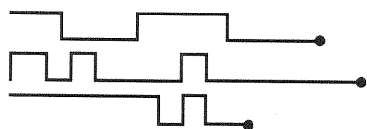

Introdução ao Microprocessador e ao Microcomputador



■ SUMÁRIO

- | | |
|--|--|
| 13-1 O que É um Computador Digital? | 13-6 Palavras |
| 13-2 Como os Computadores “Pensam”? | 13-7 Instruções |
| 13-3 Agente Secreto 89 | 13-8 Executando um Programa em Linguagem de Máquina |
| 13-4 Organização de um Sistema Computacional Básico | 13-9 Estrutura Típica de um Microcomputador |
| 13-5 Elementos Básicos de um Microcomputador (μ C) | 13-10 Comentários Finais |

■ OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Descrever a função e a operação de cada um dos cinco elementos básicos de qualquer computador.
- Compreender que um computador repete continuamente uma seqüência de operações de busca e execução.
- Entender a diferença entre um microprocessador e um microcomputador.
- Analisar os ciclos de busca e execução durante a execução de um programa em linguagem de máquina.
- Compreender a função dos diferentes tipos de barramentos e seus sinais em um microcomputador.
- Citar as principais funções realizadas por um microprocessador.
- Descrever os diferentes tipos de palavras utilizadas nos computadores.
- Atualizar seu conhecimento aprendendo mais detalhes e conceitos avançados sobre sistemas baseados em microprocessadores.

■ INTRODUÇÃO

Não é exagero dizer que o microprocessador e o microcomputador revolucionaram a indústria eletrônica e tiveram um enorme impacto em diversos aspectos de nossas vidas. O desenvolvimento de CIs de altíssima densidade reduziu tão drasticamente o tamanho e o custo dos microcomputadores que os projetistas rotineiramente consideram utilizar suas capacidades e versatilidade em uma grande variedade de produtos e aplicações.

Neste capítulo vamos estudar os princípios básicos de operação de um microcomputador. Apesar de focarmos no microcomputador, a maioria dos conceitos e idéias se aplica aos computadores de todos os portes. Para ilustrar de maneira representativa os diversos aspectos operacionais dos microcomputadores, utilizamos um microprocessador específico, o Intel 8051.

A família 8051 representa um ramo da árvore da família dos microprocessadores. Conforme a tecnologia de microprocessadores e microcomputadores evoluiu desde as origens de 4 bits (Intel 4004), no início dos anos 1970, para os processadores de 8 bits (8008, 8080, 8085, 6800, 6502, Z80 etc.) no final dos anos 1970, as aplicações se dividiram em dois ramos distintos. Um destes ramos foi a infância dos computadores pessoais. Estes computadores baseados em microprocessadores pretendiam ser ferramentas versáteis que poderiam carregar e executar vários programas, tais como processadores de texto, planilhas, banco de dados e jogos. Eles poderiam ser facilmente programados pelo usuário para realizar o que o programador imaginasse. Os descendentes de 16 e 32 bits (8086, as séries 80x86 e 680x0) simplesmente evoluíram daí. A arquitetura básica

do microprocessador não mudou substancialmente desde os primeiros sistemas, embora a velocidade e a quantidade de memória endereçada diretamente tenham aumentado bastante.

O segundo ramo que se desenvolveu a partir de 1970 foi a utilização de computadores baseados em microprocessadores como unidade de controle embutida num produto comercial. Este tipo de microcomputador é construído com os mesmos elementos de um computador pessoal, mas é programado apenas uma vez pelo fabricante. Deste modo, ele passa sua "vida" realizando suas tarefas específicas, tais como esperar que teclas sejam pressionadas e ligar e desligar dispositivos tais como luzes, motores, campainhas etc. A família 8051 se enquadra nesta categoria geral conhecida como microcontroladores *dedicados*. Esta família de dispositivos foi desenvolvida no início dos anos 1980 juntamente com o 68HC11, mas as aplicações desses dispositivos de oito bits orientados para controle se tornaram tão difundidas que eles ainda são muito utilizados hoje em dia. Muitas versões diferentes do 8051 original foram desenvolvidas com características especiais, mas elas ainda utilizam o mesmo conjunto básico de instruções e a arquitetura do original. Muitos outros microcontroladores, mais complexos e de maior poder de processamento, foram desenvolvidos para aplicações que exigem mais do que a capacidade de um computador de oito bits.

A família 8051 foi escolhida porque é muito utilizada atualmente, e além disso é capaz de demonstrar como os computadores funcionam de um modo adequado para um estudante iniciante. Assim que você possuir uma compreensão básica de um dos primeiros microprocessadores, tal como o 8051, é uma tarefa mais fácil evoluir para os chips mais avançados.

Seria necessário um livro completo para cobrir todos os detalhes importantes da operação de computadores e microprocessadores. O material neste capítulo pretende apenas fornecer uma base para estudos posteriores.

13-1 O QUE É UM COMPUTADOR DIGITAL?

Um computador digital é uma combinação de circuitos e dispositivos digitais que podem realizar uma seqüência *programada* de operações com mínima intervenção humana. A seqüência de operações é chamada de **programa**. O programa é um conjunto de instruções codificadas que é armazenado na memória interna do computador juntamente com todos os dados de que o programa necessita. Quando o computador é comandado a executar o programa, ele executa as instruções na ordem em que foram armazenadas na memória até que o programa termine. Ele faz isso numa velocidade extremamente alta.

13-2 COMO OS COMPUTADORES "PENSAM"?

Computadores não pensam! O **programador** do computador fornece um *programa* de instruções e dados que especificam cada detalhe sobre o que fazer e quando deve fazer. O computador é simplesmente uma máquina de alta

velocidade que pode manipular dados, resolver problemas e tomar decisões, sempre sob o controle de um programa. Se o programador cometer um erro no programa ou entrar com os dados errados, o computador produzirá resultados incorretos. Um ditado popular no campo da computação diz “lixo na entrada/lixo na saída” (“*garbage in/garbage out*”).

Talvez a melhor questão a formular neste ponto é: Como um computador executa um programa de instruções? Geralmente, esta pergunta é respondida mostrando-se um diagrama de uma arquitetura computacional (organização de seus vários elementos) e então realizando-se passo a passo o procedimento que o computador realiza na execução do programa. Nós faremos isto, mas não agora. Primeiro, vamos procurar uma analogia um tanto artificial que contenha muitos dos conceitos relacionados com a operação dos computadores.

13-3 AGENTE SECRETO 89

O Agente Secreto 89 está tentando descobrir quantos dias antecederam o assassinato de um certo líder mundial. Seu contato lhe revelou que esta informação está localizada numa série de caixas postais. Para garantir que ninguém mais obtenha a informação, ela é espalhada por dez caixas diferentes. Seu contato forneceu a ele dez chaves juntamente com o seguinte conjunto de instruções:

1. A informação em cada uma das caixas está escrita em código.
2. Abra a caixa 1 primeiro e execute a instrução localizada lá.
3. Continue pelas caixas restantes em seqüência, a menos que seja instruído do contrário.
4. Uma das caixas está preparada para explodir quando for aberta.

O Agente 89 pega as dez chaves e se dirige ao correio com o livro de código nas mãos.

A Fig. 13-1 mostra os conteúdos das dez caixas após terem sido decifradas. Suponha que você seja o Agente 89; comece na caixa 1 e prossiga pela seqüência de operações para encontrar o número de dias que antecederam a tentativa de assassinato. Naturalmente, não deve ser tanto trabalho para você como seria para o Agente 89, pois você não precisa decifrar as mensagens codificadas. A resposta está no próximo parágrafo.

Se você proceder corretamente, deve terminar na caixa 6 com uma resposta de 17. Se cometeu um erro, você pode ter aberto a caixa 7, e neste caso você não estaria mais conosco. Conforme percorreu a seqüência de operações, você essencialmente duplicou os tipos de operação e encontrou muitos dos conceitos que são parte de um computador. Vamos discutir agora estas operações e conceitos no contexto da analogia do agente secreto e ver como eles estão relacionados com os computadores reais.

No caso de você não ter adivinhado ainda, as caixas postais são similares à **memória** em um computador, onde **instruções** e **dados** são armazenados. As caixas postais de 1 a 6 contêm instruções a serem executadas pelo agente secreto, e as caixas 8, 9 e 10 contêm os dados referenciados pelas instruções. (O conteúdo da caixa 7, segundo nosso


1 Some o número armazenado na caixa 9 ao seu número de código de agente secreto.	2 Divida o resultado anterior pelo número armazenado na caixa 10.
3 Subtraia o número armazenado na caixa 8.	4 Se o resultado anterior não for igual a 30, vá para a caixa 7. Caso contrário, continue para a próxima caixa.
5 Subtraia 13 do resultado anterior.	6 Retorne para o quartel-general para receber mais instruções.
7 	8 20
9 11	10 2

Fig. 13-1 Dez caixas postais com mensagens codificadas para o Agente 89.

conhecimento, não tem equivalente nos computadores.) Os números de cada caixa são como os **endereços** das posições de memória.

Três classes diferentes de instruções são apresentadas nas caixas de 1 a 6. As caixas 1, 2, 3 e 5 são instruções que demandam *operações aritméticas*. A caixa 4 contém uma instrução de *decisão* denominada *salto condicional* ou *ramificação condicional*. Esta instrução solicita que o agente (ou computador) decida se deve saltar para o endereço 7 ou prosseguir para o endereço 5, dependendo do resultado da operação aritmética anterior. A caixa 6 contém uma instrução de controle simples que não requer dados e não se refere a nenhum outro endereço (número de caixa). Esta instrução de *retorno* informa ao agente que o procedimento está terminado (o programa acabou) e ele não deve ir adiante. Conforme você estudar microprocessadores mais profundamente, você descobrirá como ele sabe para onde deve retornar.

Cada uma das instruções aritméticas ou de salto condicional consiste em duas partes: uma **operação** e um **endereço**. Por exemplo, a primeira parte da primeira instrução especifica a operação de adição. A segunda parte fornece o endereço (caixa 9) do dado a ser utilizado na adição. Estes dados usualmente são chamados de **operandos**, e seus endereços, de **endereços do operando**. A instrução na caixa 5 é um caso especial no qual nenhum endereço de operando é especificado. Em vez disso, o operando (dado) a ser usado na operação de subtração está incluído como parte da instrução.

Um computador, como o agente secreto, decodifica e então executa as instruções armazenadas *seqüencialmente* na memória, começando na primeira posição. As instruções

são executadas na ordem, a não ser que algum tipo de instrução de *desvio* (tal como na caixa 4) provoque um desvio ou salto na operação para um novo endereço para obter a próxima instrução. Após o desvio, as instruções são executadas seqüencialmente a partir do novo endereço.

Essas são as informações que podemos obter da analogia com o agente secreto. Cada um dos conceitos encontrados será revisto novamente no material apresentado adiante. Esperamos que a analogia tenha fornecido noções preliminares que serão úteis à medida que você inicia um estudo mais técnico sobre os computadores.

13-4 ORGANIZAÇÃO DE UM SISTEMA COMPUTACIONAL BÁSICO

Todos os computadores contêm cinco elementos ou unidades essenciais: a **unidade lógica e aritmética (ULA)**, a **unidade de memória**, a **unidade de controle**, a **unidade de entrada** e a **unidade de saída**. A interconexão básica destas unidades é apresentada na Fig. 13-2. As setas neste diagrama indicam a direção na qual os dados, informações ou sinais de controle estão fluindo. Setas de dois tamanhos diferentes são utilizadas. As setas mais largas representam dados ou informação que na verdade consistem em um número relativamente grande de linhas paralelas. As setas mais estreitas, por sua vez, representam sinais de controle que normalmente consistem em uma ou poucas linhas. As várias setas também estão numeradas para permitir que nos refiramos a elas facilmente nas descrições a seguir.

Unidade Lógica e Aritmética

A ULA é a área do computador na qual as operações lógicas e aritméticas são realizadas sobre os dados. O tipo de

operação realizada é determinado pelos sinais da unidade de controle (seta 1). Os dados a serem operados pela ULA podem ser oriundos da unidade de memória (seta 2) ou da unidade de entrada (seta 3). Os resultados das operações realizadas na ULA podem ser transferidos tanto para a unidade de memória para armazenamento (seta 4) como para a unidade de saída (seta 5).

Unidade de Memória

A memória armazena grupos de dígitos binários (palavras) que podem representar instruções (programa), que o computador vai executar, e os dados que serão manipulados pelo programa. A memória também serve para o armazenamento de resultados intermediários ou finais das operações aritméticas (seta 4). A operação da memória é controlada pela unidade de controle (seta 6), que sinaliza uma operação de leitura ou uma operação de escrita. Uma determinada posição de memória é acessada pela unidade de controle que fornece o código de endereço apropriado (seta 7). Informações oriundas da ULA ou da unidade de entrada (seta 8) podem ser escritas na memória, também sob o controle da unidade de controle. Informações podem ser lidas da memória para a ULA (seta 2) ou para a unidade de saída (seta 9).

Unidade de Entrada

A unidade de entrada consiste em todos os dispositivos utilizados para obter informações e dados externos ao computador e colocá-los na unidade de memória (seta 8) ou na ULA (seta 3). A unidade de controle determina para onde a informação de entrada é enviada (seta 10). A unidade de entrada é utilizada para colocar programas e dados na unidade de memória antes de iniciar o processamento. Esta unidade também é usada para a entra-

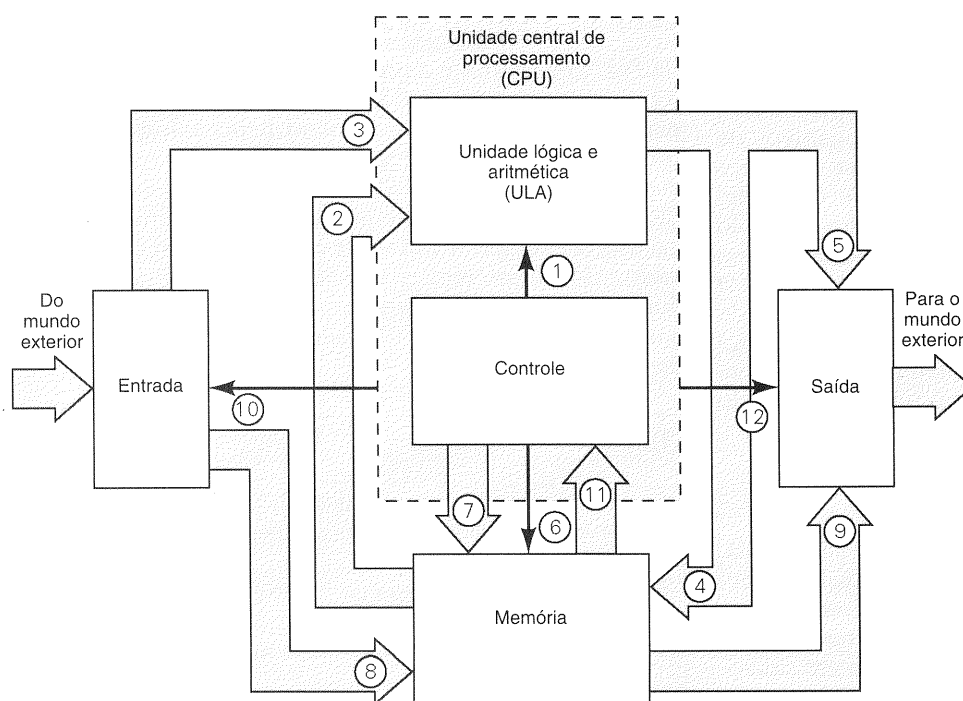


Fig. 13-2 Organização básica de um computador.

da de dados de dispositivos externos na ULA durante a execução de um programa. Alguns dispositivos de entrada comuns são: teclados, chaves de comutação, modems, leitores de cartões magnéticos, unidades de disco magnético, unidades de fita magnética e conversores analógico-digitais (A/D).

Unidade de Saída

A unidade de saída consiste em dispositivos utilizados para transferir dados e informações do computador para o “mundo exterior”. Os dispositivos de saída são acionados sob o comando da unidade de controle (seta 12) e podem receber dados da memória (seta 9) ou da ULA (seta 5); então os dados são colocados na forma apropriada para utilização externa. Exemplos de dispositivos de saída comuns são: displays a LEDs, lâmpadas indicadoras, impressoras, unidades de fita ou disco, monitores de vídeo e conversores D/A.

Enquanto o computador executa um programa, ele usualmente tem resultados ou sinais de controle que deve apresentar ao mundo exterior. Por exemplo, um sistema computacional pode ter uma impressora como um dispositivo de saída. Neste caso, o computador envia sinais para imprimir os resultados no papel. Um pequeno microcomputador pode apresentar seus resultados em lâmpadas indicadoras ou em displays a LEDs.

Interfaceamento

Os dispositivos que fazem parte das unidades de entrada e de saída são chamados de **periféricos** porque são externos ao resto do computador. O aspecto mais importante dos periféricos está relacionado com a interface. **Interfaceamento** em computadores é definido como a transmissão digital de informação entre um computador e seus periféricos de modo compatível e sincronizado.

Muitos dispositivos de entrada/saída não são diretamente compatíveis com o computador devido a diferenças em características como velocidade de operação, formato dos dados (por exemplo: BCD, ASCII, binário), modo de transmissão dos dados (por exemplo: serial, paralelo) e nível lógico dos sinais. Estes dispositivos de E/S requisitam circuitos especiais de interface que lhes permitem comunicarem-se com as áreas de controle, memória e ULA do sistema computacional. Um exemplo típico é o terminal de vídeo, que opera tanto como um dispositivo de entrada como um dispositivo de saída. O terminal transmite e recebe dados serialmente (um bit de cada vez) enquanto a maioria dos computadores manipula os dados em paralelo. Deste modo, o terminal necessita de circuitos de interface para enviar ou receber dados do computador.

Unidade de Controle

A função da unidade de controle agora deve estar óbvia. Ela comanda a operação de todas as outras unidades fornecendo sinais de controle e temporização. De um certo modo, a unidade de controle é como o maestro que é responsável por manter cada um dos membros da orquestra em sincronismo. Esta unidade contém circuitos lógicos e de tempo-



Fig. 13-3 Um computador continuamente busca e executa instruções.

rização que geram os sinais apropriados necessários para executar cada instrução em um programa.

A unidade de controle **busca** uma instrução da memória enviando um endereço (seta 7) e um comando de leitura (seta 6) para a unidade de memória. A palavra da instrução armazenada na posição de memória é então transferida para a unidade de controle (seta 11). Esta palavra da instrução, que está em alguma forma de código binário, é então decodificada pelos circuitos lógicos na unidade de controle para determinar que instrução está sendo invocada. A unidade de controle utiliza esta informação para enviar os sinais apropriados para as unidades restantes de modo a **executar** a operação específica.

Esta sequência de busca de um código de instrução e de execução da operação indicada é repetida indefinidamente pela unidade de controle (vide Fig. 13-3). Esta sequência repetitiva de busca/execução continua até que o computador seja desligado ou até que o RESET seja ativado. O RESET sempre faz o computador buscar sua primeira instrução no programa, normalmente no endereço 0000.

Então, como vimos, na verdade um computador repete indefinidamente as mesmas operações básicas: busca, executa, busca, executa e assim por diante. Naturalmente, os diversos ciclos de execução serão diferentes para cada tipo de instrução à medida que a unidade de controle envia sinais diferentes para as outras unidades para a execução de uma instrução em particular.

Unidade Central de Processamento (CPU)

Na Fig. 13-2, a ULA e a unidade de controle são mostradas combinadas em uma unidade chamada **unidade central de processamento (CPU — central processing unit)**. Isto comumente é feito para separar o “cérebro” real do computador das outras unidades. Em um microcomputador, a CPU usualmente é implementada em um único chip, o microprocessador. A CPU também contém um conjunto de registradores que realiza funções especiais. Estes registradores também podem fornecer armazenamento temporário para os dados dentro da CPU sem necessidade de acessar a memória externa.

Questões de Revisão

1. Relacione as cinco unidades básicas de um computador e descreva as principais funções de cada uma.
2. O que é CPU?
3. O que significa *interfaceamento* em um sistema computacional?
4. Que operações básicas ocorrem repetidamente em um computador?

13-5 ELEMENTOS BÁSICOS DE UM MICROCOMPUTADOR (μ C)

É importante compreender a diferença entre o microcomputador (μ C) e o microprocessador (μ P). Um microcomputador contém muitos elementos, e o mais importante é o microprocessador. O microprocessador tipicamente é um único CI que contém todos os circuitos das unidades de controle e lógica e aritmética, em outras palavras, a CPU. É comum chamar o microprocessador de **MPU (microprocessor unit — unidade microprocessadora)**, já que ele é a CPU do microcomputador. Isto é ilustrado na Fig. 13-4, onde os elementos básicos de um microcomputador estão mostrados.

A unidade de memória mostra dispositivos RAM e ROM. A seção RAM consiste em um ou mais chips LSI interligados para fornecerem a capacidade de memória projetada. Esta seção da memória é utilizada para armazenar programas e dados que serão freqüentemente alterados durante a operação. Também é usada como armazenamento para resultados intermediários ou finais das operações realizadas durante a execução de um programa.

A seção ROM contém um ou mais chips de ROM para armazenar instruções e dados que não mudam e não devem ser perdidos quando o sistema for desligado. Por exemplo, ela armazena o programa de carga inicial que o microcomputador executa ao ser ligado, ou ela poderia armazenar uma tabela de códigos ASCII necessários para o envio de informações para um terminal ou para uma impressora.

As seções de entrada e saída contêm os circuitos de interface necessários para permitir que os periféricos se co-

munique adequadamente com o restante do computador. Em alguns casos, estes circuitos de interface são chips LSI projetados pelo fabricante do microprocessador para interfaceá-lo com uma variedade de dispositivos de E/S. Em outros casos, os circuitos de interface podem ser tão simples quanto um registrador. Em muitas aplicações de controle dedicado, todos os elementos básicos de um microcomputador são integrados em um único CI, o microcomputador em um único chip.

0 Microprocessador (MPU)

O microprocessador é o coração de todo microcomputador. Ele realiza várias funções, incluindo:

1. Fornecimento de sinais de temporização e controle para todos os elementos do μ C.
2. Busca de instruções e dados da memória.
3. Transferência de dados de e para a memória e dispositivos de E/S.
4. Decodificação de instruções.
5. Realização de operações lógicas e aritméticas indicadas pelas instruções.
6. Resposta aos sinais de controle gerados pela E/S, tais como RESET e INTERRUPT.

A MPU contém todos os circuitos lógicos para realizar essas funções, mas sua lógica interna geralmente não é acessível externamente. Em vez disso, podemos controlar o que ocorre dentro da MPU pelo *programa de instruções* que colocamos na memória para a MPU executar. Isto é o que torna o microprocessador tão versátil e flexível; quando desejamos

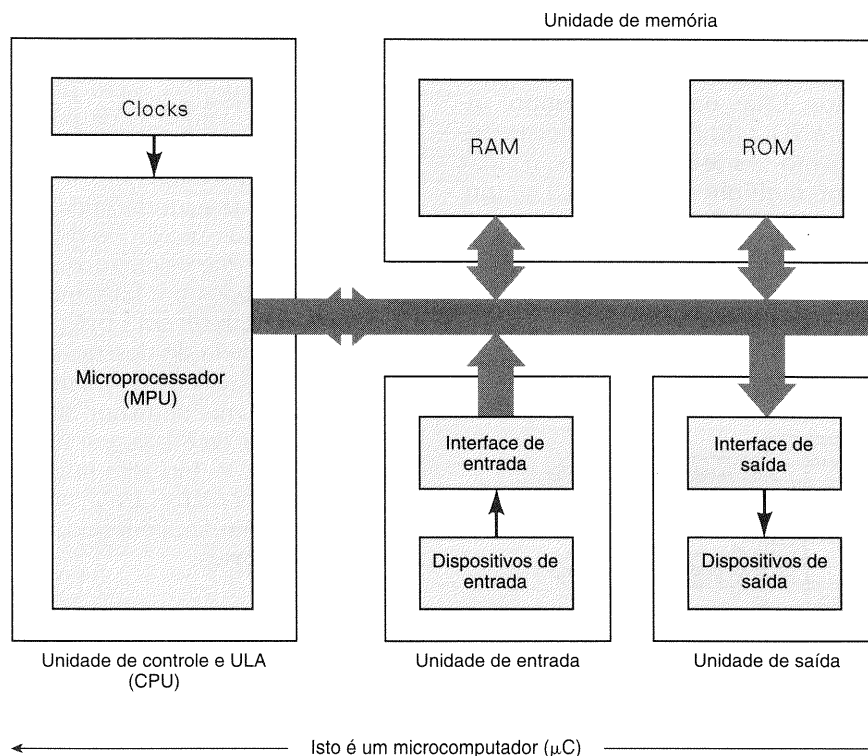


Fig. 13-4 Elementos básicos de um microcomputador.

mudar sua operação, simplesmente alteramos o programa armazenado na RAM (software) ou na ROM (firmware), em vez de alterar os circuitos eletrônicos (hardware).

A lógica interna da MPU é extremamente complexa, mas pode ser imaginada como sendo composta de três seções básicas: a seção de *controle e temporização*, a seção de *registradores* e a *ULA* (vide Fig. 13-5). Embora existam interações entre estas três seções, cada uma tem funções específicas.

A principal função da seção de temporização e controle é buscar e decodificar (interpretar) códigos de instruções da memória de programa e então gerar os sinais de controle necessários requisitados pelas outras seções da MPU, de modo a realizar a execução das instruções. Esta seção também gera os sinais de temporização e controle (por exemplo: *R/W*, clock) que são necessários pela RAM e ROM externas e pelos dispositivos de E/S.

A seção de registradores contém vários registradores (internos à MPU), e cada um deles realiza uma função especial. O mais importante é o **program counter (PC)**, que armazena os endereços dos códigos das instruções à medida que são buscados da memória. Utilizaremos o PC na descrição da execução de programas. Outros registradores da MPU são usados para realizar funções tais como: armazenamento de códigos de instrução à medida que vão sendo decodificados (registrador de instrução, IR na Fig. 13-5), manutenção do dado sendo operado pela ULA (A), armazenamento de endereços de dados a serem lidos da memória (*data pointer* — ponteiro de dados, DPTR), e muitas funções de armazenamento e contagem (R0-R7). Todos os microprocessadores têm um registrador em especial que é muito utilizado chamado de **acumulador** ou registrador A. Ele armazena um operando para qualquer instrução lógica ou matemática, e o resultado é armazenado no acumulador depois de a instrução ser executada. Algumas vezes ele é abreviado como Acc.

A ULA realiza uma diversidade de operações lógicas e aritméticas sobre os dados. Estas operações sempre incluem adição, subtração, AND, OR, EX-OR, deslocamento, incremento e decremento. As MPUs mais avançadas possuem ULAs que podem multiplicar e dividir. Durante a operação de um microcomputador, as operações que uma ULA deve realizar estão sob o controle da seção de temporização e controle, que, naturalmente, faz o que é informado pelos códigos de instrução que ela lê da memória.

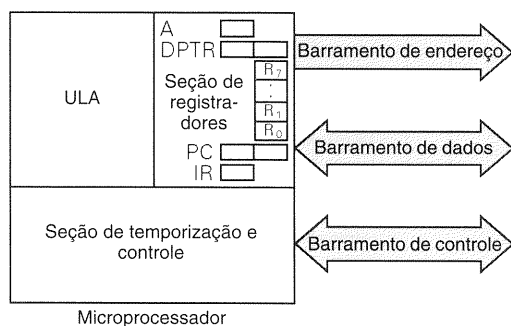


Fig. 13-5 Principais áreas funcionais de um chip μ P.

Questões de Revisão

1. Explique a diferença entre um microprocessador e um microcomputador.
2. Relacione os elementos básicos de um microcomputador.
3. Relacione as três seções principais de uma MPU.
4. Qual é a função do PC?

13-6 PALAVRAS

A menor unidade de informação em um computador é o bit. Um único bit isolado, entretanto, carrega muito pouca informação. Por isso, a principal unidade de informação em um computador é um grupo de bits denominado **palavra**. O número de bits que forma uma palavra é o **tamanho da palavra** do computador. O tamanho da palavra é uma maneira comum de descrever um computador. Computadores freqüentemente são descritos em termos do seu tamanho de palavra, tal como um computador de 8 bits, um computador de 16 bits e assim por diante. Por exemplo, um computador de 16 bits é um computador no qual instruções e dados são armazenados na memória como unidades de 16 bits e são processadas pela CPU em unidades de 16 bits. O tamanho da palavra também indica o tamanho do barramento de dados que carrega dados entre a CPU e a memória e entre a CPU e os dispositivos de E/S.

Os computadores de maior porte apresentam tamanhos de palavra que estão usualmente na faixa de 32 a 64 bits. Os minicomputadores têm tamanhos de palavra de 8 a 32 bits. Microcomputadores apresentam tamanhos de palavra de 4 a 32 bits. Os novos microcomputadores usam tamanhos de palavra de 16 a 32 bits. De um modo geral, um computador com um tamanho de palavra maior pode executar programas numa taxa mais elevada porque mais dados e mais instruções estão presentes em uma palavra. Quanto maior o tamanho da palavra, entretanto, significa que mais linhas compõem o barramento de dados, e portanto existem mais interconexões entre a CPU, a memória e os dispositivos de entrada/saída.

Como sabemos, um grupo de 8 bits é chamado um *byte*. Tendo em vista que os microcomputadores de 8 bits foram amplamente usados por bastante tempo e porque os códigos ASCII se encaixam em 1 byte, o byte continua a ser usado como uma unidade de descrição para o tamanho da palavra e para a capacidade de memória, mesmo em computadores com palavras de maior tamanho. Pode-se dizer que um computador de 8 bits tem uma palavra de 1 byte. Um computador de 16 bits tem um tamanho de palavra de 2 bytes e assim por diante. Uma memória que armazena 128K palavras de 16 bits também pode ter sua capacidade descrita como 256K bytes.

Tipos de Palavras

Uma palavra armazenada na memória de um computador pode conter dois tipos de informação: **instruções** ou **dados**. Os dados podem ser informações numéricas ou caracteres que devem ser processados por um programa que a CPU está executando. Os dados podem estar em muitos

formatos, incluindo: números binários sinalizados ou não, BCD, ponto flutuante (algo parecido com a notação de engenharia) ou códigos ASCII para caracteres, entre outros. Segue um exemplo de como o valor numérico $+86_{10}$ será armazenado numa palavra de 8 bits:

01010110

A seguir é apresentado como o código ASCII para o caracter "V" seria armazenado numa palavra de 8 bits:

01010110

Note que as duas palavras são iguais. O computador não sabe a diferença entre as duas. É responsabilidade do programador saber o tipo de dados que está sendo armazenado e garantir que o programa interprete e processe os dados adequadamente.

A seguir temos um novo exemplo, no qual uma palavra de 16 bits é usada para armazenar dois caracteres codificados em ASCII:

0101011001010111
V W

Esta mesma palavra de 16 bits poderia ser a representação para $+22103_{10}$. Obviamente, palavras de maior tamanho podem armazenar mais caracteres e números maiores.

Palavras de instruções são mais complexas do que palavras de dados. Discutiremos as palavras de instruções em detalhe na seção seguinte.

Questões de Revisão

1. Quais são os dois tipos de informação que são armazenadas em palavras de computadores?
2. Qual é a vantagem de um computador com tamanho de palavra maior?

13-7 INSTRUÇÕES

O formato utilizado para as palavras de **dados** sofre pequenas variações de computador para computador, especialmente entre aqueles que possuem o mesmo tamanho de palavra. Entretanto, isto não é verdade para o formato das palavras de **instruções**. Estas palavras contêm as informações necessárias para um computador executar suas várias operações, e o formato e os códigos destas operações podem variar bastante de computador para computador. Dependendo do tipo do computador, a informação contida em uma palavra de instrução pode ser diferente. Entretanto, para a maioria dos computadores, as palavras de instrução são formadas por duas unidades básicas de informação: a *operação* a ser executada e o *endereço* do *operando* que deve ser usado na operação.

A Fig. 13-6 mostra um exemplo de uma *palavra de instrução de endereço único* para o 8051. Os oito bits da palavra de instrução estão divididos em duas partes. A primeira parte da palavra (bits 7 a 3) contém os cinco bits do **código de operação (opcode*** para abreviar). O opcode representa

a operação que o computador está sendo instruído para executar, como por exemplo adição, subtração ou transferência de dados. A segunda parte (bits 2 a 0) contém o endereço do operando, que indica a localização na memória onde o operando está armazenado.

Se todas as instruções do 8051 tivessem a mesma forma daquela mostrada na Fig. 13-6 (cinco bits de opcode e três bits de endereço), o número total de instruções possíveis (opcodes) seria $2^5 = 32$. O número total de registradores ou posições na memória seria $2^3 = 8$. Isto iria limitar severamente a capacidade do microprocessador. Como veremos na próxima seção, existem maneiras de aumentar o número de instruções possíveis sem que seja necessário aumentar o tamanho da palavra.

O ponto importante que deve ser compreendido sobre esse formato é que os primeiros cinco bits dizem que operação o micro (microprocessador) deve executar. Os últimos três bits dizem onde o micro deve obter os dados para utilizar nesta operação. Como um exemplo, a instrução do 8051 "MOVE para o acumulador os dados do registrador 3" é codificada da seguinte maneira:

11101	011
opcode	endereço do registrador

O opcode representa mover um byte de dados de um registrador para o A (acumulador). Os últimos três bits especificam que o byte de dados que deve ser transferido está no registrador 3 (011_2). Este código de instrução representa uma operação única. Em vez de utilizarmos esta representação em binário, vamos preferir utilizar a representação hexadecimal EB. Esta instrução poderia ser armazenada na memória de um computador juntamente com um certo número de instruções que formariam um programa. Em um instante apropriado, esta instrução seria lida da memória e depois executada. O resultado de sua execução seria:

A palavra de dados de 8 bits contida no registrador R3 é transferida (na verdade copiada) para o acumulador. O conteúdo original do acumulador é perdido.

Examinaremos esta e outras instruções em detalhe mais adiante.

Instruções de Múltiplos Bytes

Vimos um formato de instrução que contém o opcode e o endereço do operando em uma *única* palavra. Em outras palavras, uma instrução completa, como aquela mostrada na Fig. 13-6, é armazenada em uma *única* posição de memória. Se todas as instruções fossem codificadas desse modo, necessitaríamos de um tamanho de palavra maior para armazenar todas as instruções possíveis e os endereços dos operandos. Os computadores com tamanhos de palavra maior usam este formato sempre que possível. O 8051 está limitado a um tamanho de palavra de oito bits. Para que um maior número de instruções esteja disponível e para proporcionar maior flexibilidade no acesso aos dados, a maioria dos computadores com tamanho de palavra menor usa mais de uma palavra para descrever cada instrução.

* O termo opcode é consagrado na terminologia técnica e foi mantido por conveniência. (N. T.)

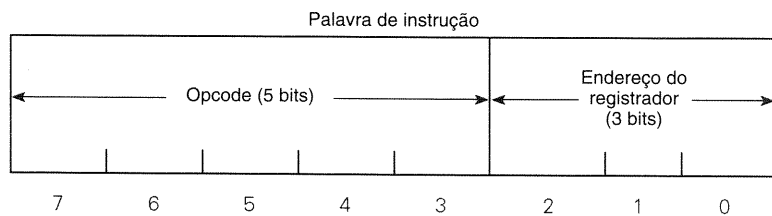


Fig. 13-6 Típica palavra de instrução de endereço único.

Geralmente, existem três formatos básicos de instrução: um byte, dois bytes e três bytes. Estes formatos estão ilustrados na Fig. 13-7.

A instrução da Fig. 13-6 é uma instrução de um byte. Outras instruções de um byte não têm necessidade de um operando. Um exemplo é a instrução para limpar o acumulador (CLR A), que instrui o computador a limpar todos os flip-flops do acumulador. Para esta instrução, todos os oito bits são considerados parte do opcode, uma vez que não existe instrução para limpar outros registradores que não o acumulador.

Em uma instrução de dois bytes, o primeiro byte sempre contém o opcode. O propósito do segundo byte (operando) é especificar os dados que serão utilizados na instrução. Existem vários métodos de especificar estes dados. Estes métodos são chamados **modos de endereçamento**. Discutiremos neste livro apenas dois modos de endereçamento que são bastante populares. O primeiro é semelhante ao método descrito anteriormente, no qual o operando representa o endereço que indica ao computador onde os dados estão armazenados. Este é o chamado **endereçamento**

direto. O outro modo de endereçamento inclui um valor existente como parte de uma instrução. Neste modo o operando é o próprio valor, em vez do endereço do valor. Uma vez que este valor de dados segue imediatamente o opcode na memória de programa, este método é chamado de modo de **endereçamento imediato**.

Uma instrução de três bytes é necessária quando o operando deve ser um número de 16 bits. Para estas instruções que possuem mais de um byte, eles devem estar armazenados em posições de memória sucessivas. Isto é ilustrado na Tabela 13-1 para uma instrução de três bytes. A coluna à esquerda relaciona as posições de memória onde cada byte (palavra) é armazenado. Estes endereços são fornecidos em código hexadecimal. A segunda coluna fornece a palavra, em representação binária, que está armazenada na memória, e a terceira coluna apresenta o equivalente em hexadecimal desta palavra. Examine esta tabela com cuidado, antes de continuar a leitura, e tente descobrir o que ela representa.

Por exemplo, considere a instrução que comanda o computador a fazer um desvio para a primeira instrução e inici-

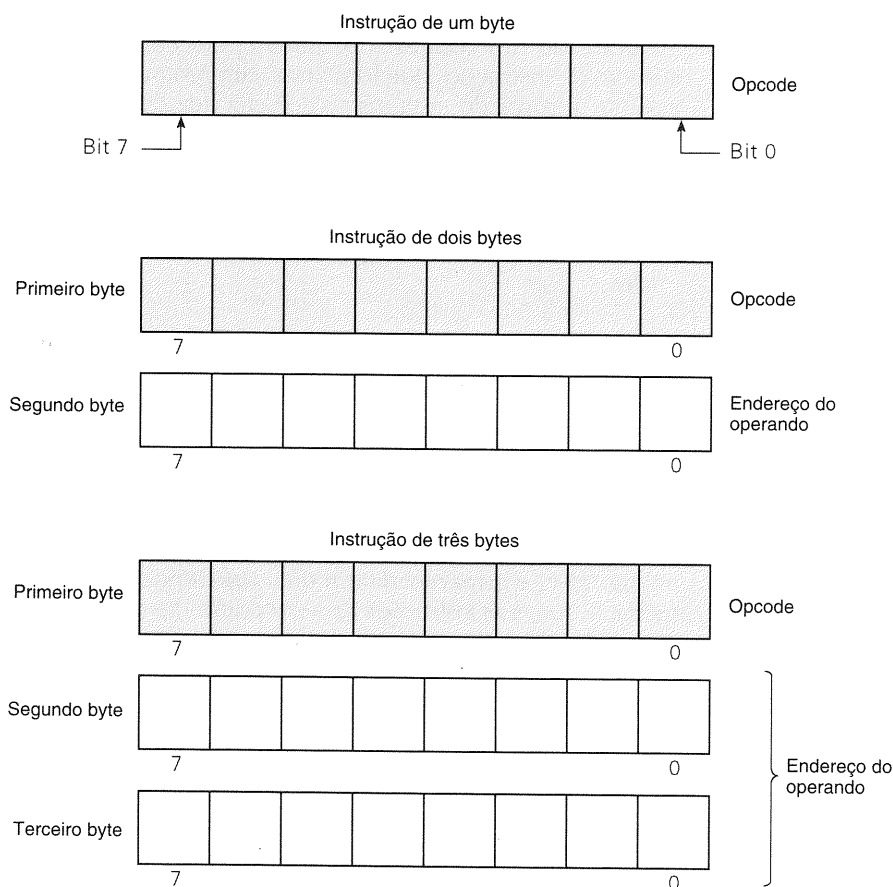


Fig. 13-7 Formatos de instruções usadas em microcomputadores de oito bits.

TABELA 13-1 Instruções de um programa armazenadas na memória.

Endereço de Memória	Binário	Hexa	Descrição
2050	XXXXXXXX	XX	Primeira instrução do programa
2051			
2052			
.			
.			
2377	00000010	02	Opcode para instrução de desvio (LJMP)
2378	00100000	20	Parte alta do endereço de desvio
2379	01010000	50	Parte baixa do endereço de desvio
237A			
.			
.			

ar o programa novamente. Esta instrução é mostrada na Tabela 13-1 do mesmo modo como ela apareceria na memória. O primeiro byte é o opcode (02) que diz ao computador que ele deve desviar. Este opcode também informa a unidade de controle que é de três bytes, e, para que ela possa saber para onde desviar, ela deve buscar na memória os dois próximos bytes. Juntos, estes bytes formam o endereço de 16 bits da próxima instrução a ser buscada na memória. Lembre-se de que o contador de programa sempre contém o endereço da próxima instrução. Quando o contador de programa tiver avançado para 2377, o computador irá buscar na memória o opcode 02. Ele então busca os próximos dois bytes (20 e 50) e os carrega no contador de programa. O novo endereço do contador de programa significa que a

próxima instrução será buscada no endereço 2050, em vez de 237A.

Questões de Revisão

1. O que é um opcode?
2. O que é o endereço de um operando?
3. Que informação está contida no primeiro byte de uma instrução de múltiplos bytes?

13-8 EXECUTANDO UM PROGRAMA EM LINGUAGEM DE MÁQUINA

As palavras de instrução apresentadas até agora são chamadas de instruções em linguagem de máquina porque elas são representadas por 0s e 1s, a única linguagem que uma máquina (computador) compreende. Muitas outras linguagens são usadas para programar computadores, e você deve estar familiarizado com algumas delas, como por exemplo BASIC ou C. Estas **linguagens de alto nível** são projetadas para facilitar a elaboração de um programa. É importante compreender que estes programas de alto nível devem ser convertidos em instruções em linguagem de máquina e colocadas na memória do computador antes que ele possa executá-las. Em um computador típico, a conversão de C para linguagem de máquina é feita por um programa chamado **compilador**. O programa compilador, tipicamente, roda em um computador pessoal. Para usar um compilador, um programador cria um arquivo texto com instruções em uma determinada linguagem, como por exemplo C. O compilador então traduz estas instruções de alto nível em um conjunto de instruções binárias (linguagem de máquina) que podem ser carregadas na memória do microcomputador.

Para ilustrar como um microcomputador executa um programa em linguagem de máquina, usaremos as instruções descritas na Tabela 13-2. Estas instruções fazem parte

TABELA 13-2 Algumas instruções do 8051.

Mnemônico	Opcode			Descrição da Operação
	Binário	Hexa		
LJMP endereço	0000	0010	02	Desvio longo. Desvia da posição de memória atual do programa para o endereço especificado pelo operando de dois bytes.
MOV A, direto	1110	0101	E5	Move de um endereço direto para Acc. O valor contido no endereço direto é movido para Acc.
MOV direto, A	1111	0101	F5	Move do Acumulador para um endereço direto. O valor em Acc é transferido para o endereço direto especificado na instrução.
DEC A	0001	0100	14	Decrementa o Acumulador. Subtrai 1 do valor do Acc.
JNZ endereço	0111	0000	70	Salta se Acc não é zero. Se Acc \neq 0, a próxima instrução vai ser buscada na memória no endereço do operando. Caso contrário, a próxima instrução na sequência é buscada.
JZ endereço	0110	0000	60	Salta se Acc é zero. Se Acc = 0, a próxima instrução vai ser buscada na memória no endereço do operando. Caso contrário, a próxima instrução na sequência é buscada.
LCALL endereço	0001	0010	12	Chamada Longa de uma sub-rotina. Faz com que um pequeno programa chamado <i>sub-rotina</i> seja executado. Quando ela termina, o programa retorna para a instrução seguinte ao LCALL.
RET	0010	0010	22	Retorno de uma sub-rotina. Manda o computador de volta ao lugar de onde ele foi chamado.

do conjunto de instruções do 8051, mas são também exemplos típicos dos tipos de instrução que a maioria dos microprocessadores pode executar. Cada instrução é acompanhada por seu **mnemônico** ou abreviação, que é mais fácil de lembrar do que o opcode. O conjunto completo de mnemônicos para o conjunto de instruções de um computador é chamado de sua **linguagem de montagem (assembly)**. Leia a descrição de cada instrução com cuidado, e tenha em mente que apenas os opcodes são mostrados.

Usaremos algumas das instruções do 8051 da Tabela 13-2 para escrever um programa em linguagem de máquina que começa no endereço 0000H e faça o seguinte:

1. Desvie para o endereço correto para iniciar o programa.
2. Decida se o acumulador está com valor 0.
3. Se $A \neq 0$, mostre o valor do acumulador na porta 1.
4. Aguarde 1 segundo.
5. Decrementa o valor do acumulador de 1.
6. Repita o passo 3 até que o valor do acumulador seja 0.

A Tabela 13-3 mostra um programa que estaria na memória do computador. Na verdade, as duas primeiras colunas são o programa em linguagem de máquina; as outras foram incluídas para auxiliar a descrição. A primeira coluna relaciona os endereços em hexadecimal de cada posição de memória que está sendo usada pelo programa. A segunda coluna fornece o equivalente em hexadecimal da palavra armazenada em cada posição de memória. Lembre-se de que estes valores hexadecimais representam endereços binários e códigos de instruções que o computador compreende.

A terceira coluna mostra os mnemônicos da linguagem assembly e o endereço do operando (se existir algum) associado com cada instrução. A última coluna descreve a operação realizada pela instrução. Por exemplo, a primeira instrução do programa possui três bytes. O primeiro byte armazenado na posição de memória 0000H é o opcode 02. O segundo e terceiro bytes armazenados nos endereços 0001H e 0002H formam o endereço do operando 0100. O mnemônico para esta instrução é LJMP 0100H. LJMP é a abreviação para uma operação de desvio (*long jump* — grande salto, desvio longo) e 100H é o endereço de desvio. O H é geralmente usado para indicar que o endereço está representado em hexadecimal.

Uma aplicação típica de um microprocessador dedicado é o sistema de controle de um forno de microondas. Um diagrama de blocos deste sistema é mostrado na Fig. 13-8. Se tentássemos analisar todas as instruções em linguagem de máquina que fazem parte de um programa de um microondas na realidade, logo veríamos que isto é bastante complexo e trabalhoso. Nosso objetivo aqui é compreender como uma pequena parte deste programa funciona e dar a você uma visão geral do que o programa faz para controlar o sistema. No exemplo da Tabela 13-3, apenas uma parte do programa é mostrada de um modo simplificado para que você não tenha dificuldade em compreendê-la. Seu objetivo é determinar se um valor diferente de zero foi colocado no acumulador. O valor colocado no acumulador representa o número de segundos durante o qual o microondas deve cozinhar a comida. Se um valor diferente de zero está no acumulador, o programa mostra este valor em uma porta de saída e o decrementa em intervalos de 1 segundo, até que este seja igual a zero. A partir deste ponto, continua-se a executar o resto do programa. O programa começa a ser executado no endereço 0000 quando a alimentação é ligada, o que faz com que o sistema seja ressetado. A instrução que geralmente está armazenada no endereço de reset é uma instrução de desvio, que faz com que o micro vá para o programa principal. O programa principal, neste caso, começa em 0100 onde se toma a decisão de ir executar o resto do programa em 010A ou de executar as instruções de 0102 a 0109. Qualquer que seja o caso, o micro em algum momento executará o resto do programa, a partir de 010A, até que a execução do programa seja desviada para 0100 e o micro comece a executar o programa novamente. Gaste algum tempo examinando a Tabela 13-3 até que você a compreenda.

A Execução do Programa

Prosseguiremos agora com a execução completa deste programa e descreveremos o que o processador faz a cada passo. Nosso objetivo aqui é traçar um perfil apenas das operações principais, sem nos preocuparmos com todos os detalhes de tudo que está acontecendo no computador. Em

TABELA 13-3 Um programa simples em linguagem de máquina.

Endereço de Memória (Hexa)	Conteúdo de Memória (Hexa)	Linguagem Assembly	Descrição
0000	02	LJMP 0100H	;Desvia para o início do programa
0001	01		
0002	00		
0100	60	JZ 010AH	;Devemos cozinhar a comida?
0101	08		
0102	F5	MOV P1, A	;Mostra o tempo de cozimento na porta 1
0103	90		
0104	12	LCALL 1_SEC_DELAY	;Gasta um segundo
0105	28		
0106	55		
0107	14	DEC A	;Subtrai um segundo do tempo
0108	70	JNZ 0102H	;A comida está pronta?
0109	F8		
010A	***** Aqui é onde o resto do programa continua*****		

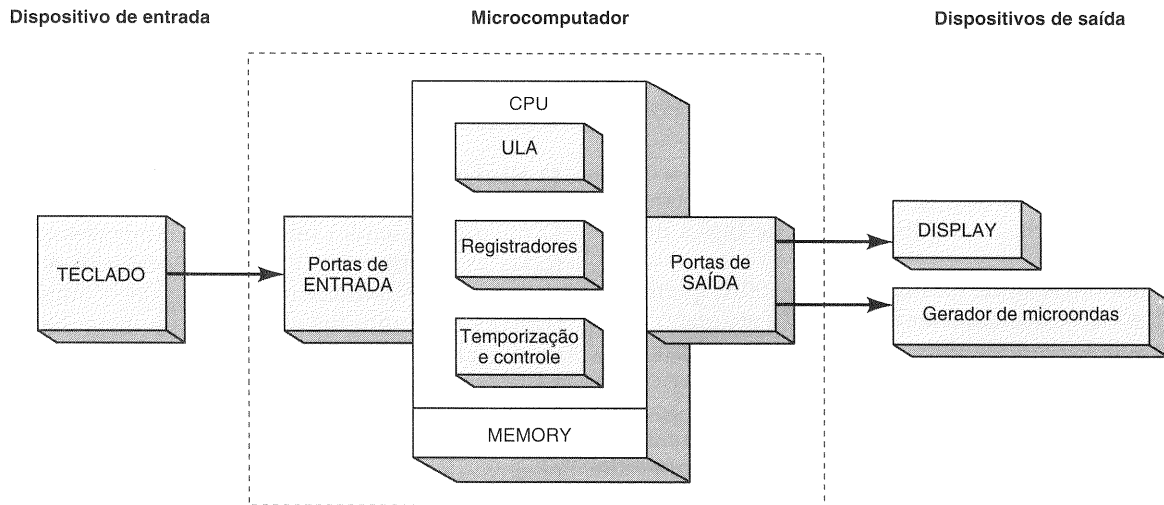


Fig. 13-8 Diagrama de blocos de um forno de microondas.

particular, veremos que o computador está sempre em um dos dois tipos de ciclo de operação: (1) um **ciclo de busca** durante o qual a unidade de controle busca os códigos de instrução (o opcode e o endereço do operando) da memória e (2) um **ciclo de execução** durante o qual a unidade de controle realiza a operação representada pelo opcode.

A operação inicia quando o operador ativa o pino de RESET ligando a alimentação. Isto vai inicializar o contador de programa (PC) para uma contagem inicial de 0000. Como foi mencionado antes, o PC é um contador da unidade de controle que guarda os endereços do programa à medida que o computador vai executando este programa.

1. A unidade de controle busca o primeiro byte no endereço 0000 conforme é determinado pelo contador de programa (PC). Este byte é 02H, que é o opcode da primeira instrução. Os circuitos na unidade de controle determinam que este opcode pede que o programa deva ser desviado, e os dois próximos bytes formam o endereço de desvio. Então, a unidade de controle busca a parte alta e depois a parte baixa do endereço de destino.
2. Para executar a instrução, os 16 bits do endereço de destino (0100H) são carregados no contador de programa. A próxima instrução a ser executada estará no endereço 0100H.
3. O opcode 60H é buscado em 0100H e é decodificado como um desvio condicional (JZ). A unidade de controle também sabe que esta instrução possui dois bytes, e então busca também o seu operando (08H). Ele indica para a unidade de controle de quanto deve ser o desvio, caso as condições para que ele ocorra sejam alcançadas. O desvio deverá ocorrer apenas se o valor do acumulador for zero. A unidade de controle examina o acumulador, e se $\text{Acc} = 0$, ela avança o contador de programa para o endereço 010AH. Se $\text{Acc} \neq 0$, a unidade de controle vai buscar o próximo código de operação, neste caso, no endereço 0102H. Como valores diferentes de zero chegam ao acumulador? Uma outra parte do programa seria responsável por colocar o intervalo de tempo fornecido através de um teclado no acumulador. Deixaremos isto para a sua imaginação para manter este exemplo simples. Por enquanto, admita que o microprocessador já executou este loop de programa várias vezes e que um determinado valor foi colocado em A.
4. A execução da instrução JZ simplesmente avança o contador de programa para o endereço 0102H uma vez que $\text{Acc} \neq 0$.
5. Um outro opcode é lido em 0102H. A instrução F5 diz a seção de controle para transferir o valor que está no acumulador. Para saber para onde ele deve ser transferido, o próximo byte de memória deve ser lido. Neste caso, o endereço (90H) significa que o valor deve ser escrito na porta #1.
6. A execução desta instrução copia o conteúdo do acumulador para o registrador da porta de saída. Este valor é escrito na porta de saída, que poderia estar conectada a um circuito que permitisse ao usuário visualizar o número.
7. O PC é incrementado para 0104H, e um novo ciclo de busca é realizado. Esta instrução é uma chamada para uma sub-rotina. Isto significa que o micro deve deixar esta parte do programa e ir executar algumas instruções que estão armazenadas em outro lugar na memória. Os dois próximos bytes (28H, 55H) dizem à CPU onde a sub-rotina está localizada.
8. Para executar uma instrução de chamada de sub-rotina, a CPU deve guardar o lugar no programa ao qual ela deve retornar após esta sub-rotina (endereço 0107H), e então carregar o PC com o endereço 2855H. Neste endereço, ela encontra a primeira instrução de um programa de espera de 1 segundo. A última instrução na sub-rotina seria um RET que faz com que o PC retorne ao endereço 0107H 1 segundo depois.
9. O opcode no endereço 0107H é lido e decodificado como um comando de decremento do acumulador. Esta é uma instrução de apenas um byte, e portanto nenhum operando é necessário. A execução da instrução ocorre dentro da CPU, e o número armazenado no acumulador é diminuído de 1. O PC é incrementado para apontar para a próxima instrução no endereço 0108H.

10. No endereço 0108H, o opcode da próxima instrução (70H) é lido. A unidade de controle reconhece o opcode como sendo um JNZ (salte, se não for zero), e ela sabe que deve tomar uma decisão baseada no conteúdo do acumulador.
11. Se o valor do acumulador chegar a zero, então a comida está pronta, e a CPU deve continuar executando o resto do programa, fazendo um ciclo de busca no endereço 010AH.
12. Se o valor em A ainda for maior do que 00H, queremos repetir o laço e decrementar os segundos até que o valor em A seja 0. Isto significa que devemos retornar ao endereço 0102H. Para saber de quanto é o desvio, um ciclo de busca deve ser realizado para pegar o operando que está no endereço 0109H. Este valor é combinado com o valor atual do PC para que o novo conteúdo do PC seja 0102H.

Este programa simples ilustra os diversos tipos de operações que acontecem quando um computador executa um programa em linguagem de máquina; entretanto, ele nem ao menos começa a mostrar os recursos e a versatilidade de um computador. É importante compreender que a execução de um programa é realizada passo a passo, de modo seqüencial, iniciando no primeiro endereço do programa (0000 no nosso exemplo). É claro que, apesar de o computador executar uma instrução de cada vez, elas são executadas com bastante rapidez. Por exemplo, cada instrução deste programa demora entre 1 e 2 microssegundos para ser executada em um microcontrolador típico.

Questões de Revisão

1. Qual a diferença entre um programa em linguagem de máquina e um programa de alto nível?
2. O que é o mnemônico de uma instrução?
3. O que, de modo geral, acontece em um ciclo de busca? E durante um ciclo de execução?
4. Qual é a função do contador de programa?

13-9 ESTRUTURA TÍPICA DE UM MICROCOMPUTADOR

Agora estamos preparados para estudar com mais detalhe a organização de um microcomputador. Muitas estruturas são possíveis, e elas têm essencialmente os mesmos princípios, embora variem em termos do tamanho do barramento de dados, de endereço e do tipo de sinais de controle que utilizam. Uma boa maneira de aprender bem os princípios de operação de um microcomputador é escolher um único tipo e estudá-lo em detalhe. Uma vez obtido um sólido entendimento do funcionamento de um microcomputador típico, é relativamente simples aprender outros tipos. Escolhemos a estrutura que está mostrada na Fig. 13-9. Ela é baseada no microprocessador 8051 mas tem essencialmente todos os elementos típicos de um microcomputador baseado em um microprocessador de 8 bits. A CPU mostrada, na verdade, é composta do chip 8051 e de vários chips de suporte para produzir as estruturas de barramento desejadas. Não vamos nos preocupar com os detalhes destas conexões. Nosso

enfoque será nos vários barramentos que conectam a CPU a outros elementos do microcomputador e como estes barramentos serão usados durante a operação de um microcomputador.

O Sistema de Barramentos

O microcomputador tem três barramentos que conduzem todas as informações e sinais necessários à operação do sistema. Estes barramentos conectam o microprocessador (CPU) a cada um dos elementos de memória e de E/S, de modo que dados e informações possam ser trocados entre a CPU e qualquer um destes elementos. Em outras palavras, a CPU está envolvida continuamente no envio e na recepção de informação com uma posição de memória, um dispositivo de entrada ou um dispositivo de saída.

Em um microcomputador, todas as transferências de informação são referenciadas à CPU. Quando a CPU está enviando dados para outro elemento do computador, isto é chamado de uma operação de *escrita*, e a CPU está escrevendo no elemento selecionado. Quando a CPU está recebendo dados de um outro elemento, isto é chamado de operação de *leitura*, e a CPU está lendo o elemento selecionado. É muito importante perceber que os termos “leitura” e “escrita” sempre se referem a operações realizadas pela CPU.

Os barramentos envolvidos em todas as transferências de dados têm suas funções descritas a seguir:

■ **Barramento de Endereço** Este barramento é *unidirecional*, porque a informação flui apenas em uma direção, da CPU para a memória ou para os elementos de E/S. A CPU pode colocar níveis lógicos nas linhas de endereço e portanto gerar $2^{16} = 65.536$ endereços diferentes possíveis. Cada um destes endereços corresponde a uma posição de memória ou a um elemento de E/S. Por exemplo, o endereço $20A0_{16}$ poderia ser uma posição de memória em uma ROM ou RAM onde uma palavra de 8 bits estaria armazenada, ou poderia ser um registrador buffer de 8 bits que fosse parte de um circuito de interface de um display de cristal líquido ou de um conversor D/A.

Quando a CPU quer se comunicar com (ler de ou escrever em) uma certa posição de memória ou dispositivo de E/S, ela coloca um código de endereço de 16 bits nos pinos de saída A_0 a A_{15} , e no barramento de endereço. Estes bits de endereço são *decodificados* para selecionar a posição de memória ou o dispositivo de E/S desejado. Este processo de decodificação geralmente necessita de circuitos de decodificação que não estão mostrados neste diagrama.

■ **Barramento de Dados** Este é um barramento *bidirecional*, porque os dados podem ir para ou vir da CPU. Os oito pinos de dados da CPU, pinos D_0 a D_7 , podem ser tanto entradas quanto saídas, dependendo se a CPU está realizando uma operação de leitura ou de escrita. Durante uma operação de leitura, eles agem como entradas e recebem dados que foram colocados no barramento pela memória ou por um dispositivo de E/S selecionado pelo endereço colocado no barramento de endereço. Durante uma operação de escrita, os pinos de dados da CPU agem como saídas e colocam os dados no barramento de dados, que são enviados para a memória ou

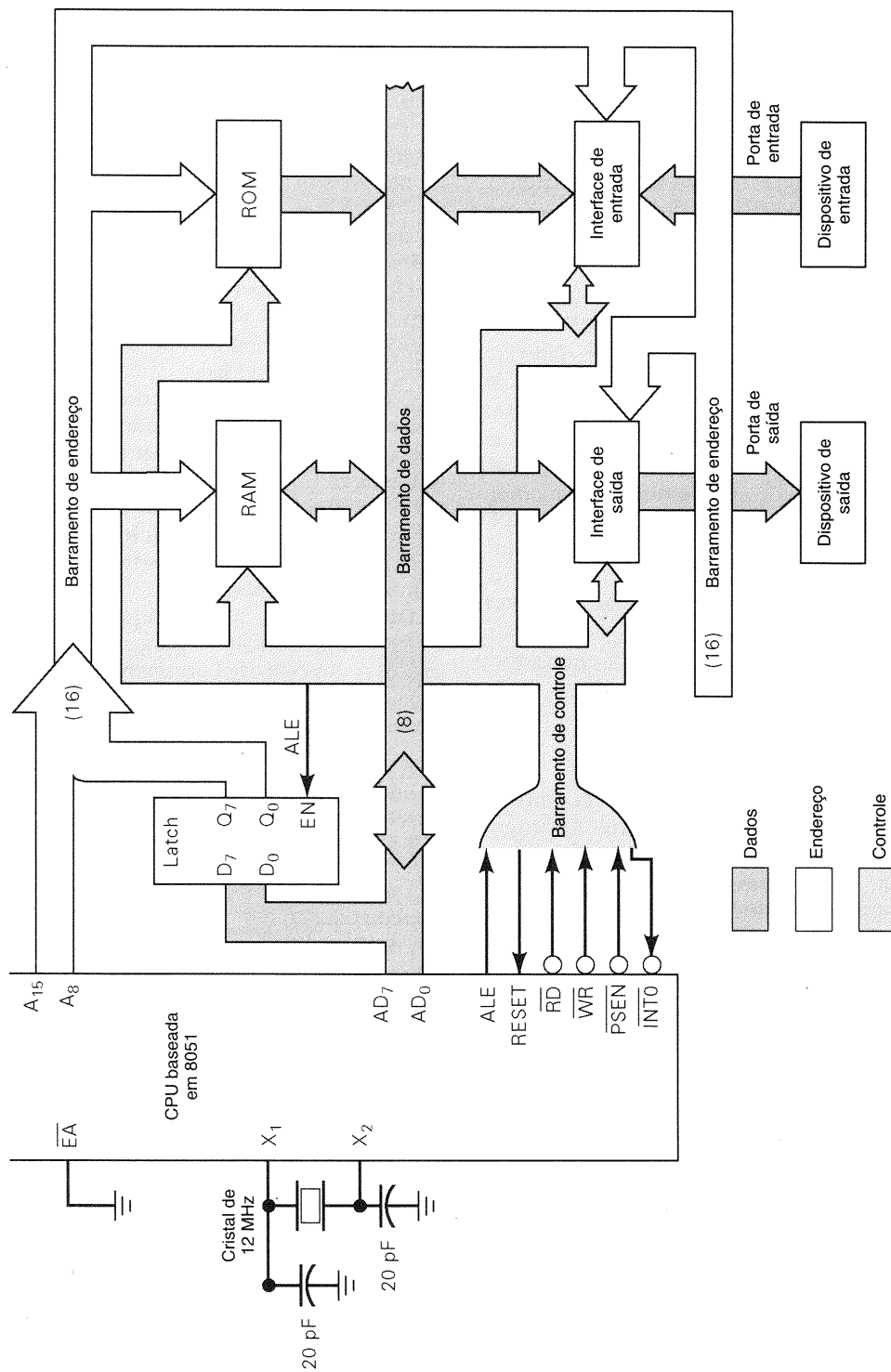


Fig. 13-9 Estrutura típica de um microcomputador de oito bits.

elemento de E/S selecionado. Em todos os casos, as palavras de dados transmitidas são de oito bits porque a CPU trabalha com dados de 8 bits, e portanto este é um computador de 8 bits.

■ **Barramento de Controle** Este conjunto de sinais é usado para sincronizar as atividades dos elementos do microcomputador. Alguns destes sinais de controle tais como *ALE*, *PSEN*, *RD* e *WR* são enviados pela CPU para outros elementos para informar a eles que tipo de operação está sendo executada. O sinal *ALE* (*Address Latch Enable* — habilitador do latch de endereço) está em nível ALTO enquanto a CPU está colocando a parte baixa do endereço em *AD₀* a *AD₇*. Este sinal habilita o latch de endereço a armazenar a parte baixa do endereço durante este intervalo. O sinal *PSEN* (*Program Store Enable* — habilitador da memória de programa) é BAIXO quando a CPU quer que a memória de programa coloque uma instrução no barramento de dados. O sinal *RD* está em BAIXO quando a CPU quer que a memória de dados ou uma porta de entrada externa coloque um byte de dados no barramento de dados. O sinal *WR* está em BAIXO quando a CPU está colocando um byte de dados no barramento de dados que ela quer escrever na memória de dados ou em uma porta de saída externa. Os elementos de E/S podem enviar sinais de controle para a CPU. Um exemplo é a entrada de RESET (*RST*) da CPU, que, quando em nível ALTO, faz com que a CPU vá para um estado inicial. Um outro exemplo é a entrada de interrupção (*INT0*), usada pelos dispositivos de E/S para chamar a atenção da CPU quando ela está executando outras tarefas.

Portas de E/S

Durante a execução de um programa, a CPU está constantemente lendo da ou escrevendo na memória. O programa também pode fazer com que a CPU leia um dos dispositivos de entrada ou escreva em um dos dispositivos de saída. Embora o diagrama do microcomputador de 8 bits mostre apenas um dispositivo de entrada e um de saída, pode existir um número qualquer destes dispositivos conectados ao sistema de barramentos do microcomputador. Cada dispositivo de E/S é normalmente conectado ao sistema de barramentos do microcomputador por meio de algum tipo de circuito de interface. A função deste circuito de interface é fazer o microcomputador e o dispositivo compatíveis, de modo que dados possam ser facilmente trocados entre eles. Um circuito de interface é necessário sempre que o dispositivo de E/S utilizar níveis, temporização ou formato de sinais diferentes daqueles do microcomputador.

Embora dispositivos de E/S sejam tratados como posições de memória, eles são bastante diferentes de uma memória em certos aspectos. Uma grande diferença é que dispositivos de E/S têm a capacidade de *interromper* a CPU enquanto ela está executando um programa. Isto significa que um dispositivo de E/S pode enviar um sinal para a entrada de interrupção da CPU (*INT0*) para indicar que ele deseja se comunicar com a CPU. A CPU então irá suspender a execução do programa que está sendo executado e irá realizar a operação esperada com o dispositivo de E/S que o está

interrompendo. As memórias do tipo RAM e ROM geralmente não possuem esta capacidade.

Temporização

O 8051 contém um oscilador no próprio chip que gera o sinal de clock principal para temporizar todas as suas operações. Um cristal externo é conectado a *X₁* e *X₂* para produzir uma frequência de clock precisa e estável (veja Fig. 13-9). Se soubermos que frequência é esta, podemos prever com precisão quanto tempo cada instrução demora para ser executada. Para o 8051, a frequência do cristal é geralmente próxima a 12 MHz. Todas as operações do sistema, tais como busca e execução de instruções, leitura e escrita de dados, acontecem em períodos chamados de **ciclos de máquina**. Cada ciclo de máquina é composto de 12 ciclos de clock, e portanto um ciclo de máquina tem a duração de 1 microssegundo a 12 MHz. Praticamente todas as instruções do 8051 necessitam de um ou dois ciclos de máquina para serem executadas.

Conforme dissemos anteriormente, o microprocessador está constantemente buscando e executando instruções. A execução da maioria das instruções envolve operações internas à CPU, como por exemplo transferência de dados entre registradores, soma de números e assim por diante. Entretanto, três tipos de **ciclos de barramento** distintos podem ser observados no sistema de barramento externo, como mostra a Tabela 13-4.

Durante um ciclo de máquina (12 períodos do clock), existe tempo suficiente para realizar duas buscas ou para executar um ciclo de barramento de leitura de dados ou um ciclo de barramento de escrita de dados. Cada ciclo de barramento é uma sequência de eventos previsíveis:

1. A CPU coloca um endereço estável no barramento de endereço (*ALE* está ALTO).
2. A CPU ativa os sinais de controle para sinalizar uma transferência de dados (*PSEN*, *RD* ou *WR* em BAIXO).

Do mesmo modo que você pode observar as luzes de um sinal de trânsito em um cruzamento e saber quem pode passar, você também pode olhar para esses quatro sinais de temporização e controle e saber que tipo de ciclo de barramento está acontecendo, o que a informação presente no barramento significa e de onde veio esta informação. O diagrama de tempo da Fig. 13-10 mostra duas instruções sendo buscadas e executadas. A primeira instrução carrega o número 3000H no registrador DPTR. A segunda instrução escreve o conteúdo do acumulador no endereço de memória externo especificado por DPTR. Sempre que o sinal de controle *ALE* está em ALTO, existe um endereço no barramento de dados (*AD₀*-

TABELA 13-4 Ciclos de barramento do 8051.

Ciclo de Barramento	Sinal de Controle	Transferência de Dados
Busca	\overline{PSEN}	CPU ← Memória de programa
Leitura de dados	\overline{RD}	CPU ← E/S ou memória de dados
Escrita de dados	\overline{WR}	CPU → E/S ou memória de dados

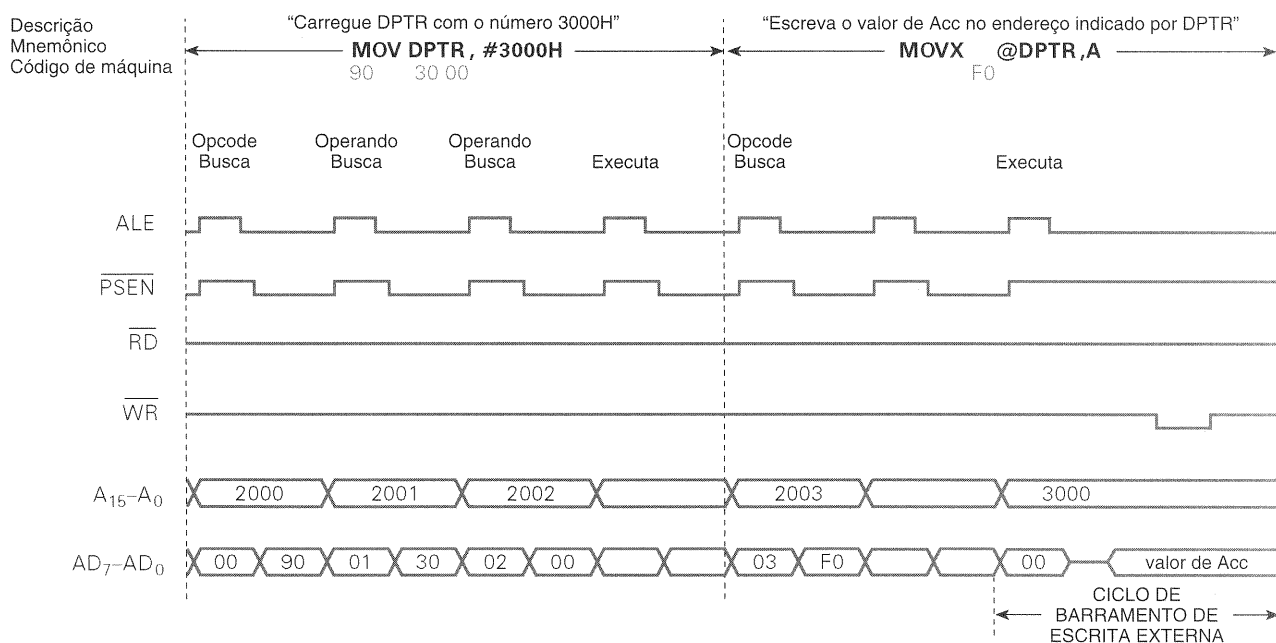


Fig. 13-10 Diagrama de tempo para a execução de duas instruções do 8051.

AD₇). Sempre que \overline{PSEN} está em BAIXO, existe um byte de instrução nas linhas AD₀-AD₇. Sempre que \overline{WR} está em BAIXO, existe um valor de dados válido que a CPU colocou em AD₀-AD₇ para escrever na memória externa. Neste exemplo, nada está sendo lido da memória externa de dados, e portanto \overline{RD} nunca vai para nível BAIXO.

Questões de Revisão

1. Descreva as funções dos três barramentos que fazem parte de um microcomputador típico.
2. O que é um barramento unidirecional?
3. O que deveria estar no barramento de dados quando:
 - (a) \overline{ALE} está em nível ALTO?
 - (b) \overline{PSEN} está em nível BAIXO?
 - (c) \overline{RD} está em nível BAIXO?
 - (d) \overline{WR} está em nível BAIXO?
4. Quantas posições de memória externa o 8051 pode acessar?

13-10 COMENTÁRIOS FINAIS

Nosso estudo foi, por necessidade, apenas uma breve introdução aos princípios básicos, terminologia e operações comuns da maioria dos microprocessadores e microcomputadores. Ainda nem começamos a compreender os recursos e as possibilidades de aplicação destes dispositivos. Quase todos os campos da tecnologia já começaram a tirar vantagem do controle por computador de baixo custo que os microprocessadores podem propiciar. Algumas das aplicações mais comuns são: fornos de microondas, videocassetes, CD players, sinais de trânsito, computadores pessoais, instrumentos eletrônicos de medida, controle de processos industriais, jogos eletrônicos, controle de emissão de

gases em automóveis, sistemas de freios automáticos e um número cada vez maior de novos produtos.

Com o impacto revolucionário que os microprocessadores tiveram na indústria da eletrônica, é de se esperar que qualquer um que esteja trabalhando com eletrônica e áreas afins tenha que se tornar um conhecedor da operação e do funcionamento de microprocessadores. Esperamos que nossa introdução sirva como uma base sólida para estudos mais aprofundados nesta importante área.

RESUMO

1. Um microprocessador é um circuito digital seqüencial sofisticado que é projetado para obedecer a uma seqüência de instruções denominada *programa*.
2. O programa é armazenado em dispositivos de memória que são conectados ao microprocessador.
3. Para rodar um programa, o microprocessador busca uma instrução da memória, executa-a, e então busca a próxima instrução, executa-a e assim por diante.
4. A execução de algumas instruções envolve a leitura de valores de dispositivos de entrada, tais como: teclados, chaves, conversores A/D, dentre outros. Outras instruções fazem o microcomputador escrever valores em dispositivos de saída, tais como: displays LCD, lâmpadas indicadoras, circuitos de controle de motor, conversores D/A e assim por diante.
5. Dados também podem ser armazenados e recuperados dos dispositivos de memória que estão conectados ao microprocessador.
6. O microprocessador está conectado nos dispositivos de memória e de E/S por um grupo de linhas chamado *barramento de dados*.
7. Um barramento de endereço também é conectado do microprocessador para os dispositivos de memória e de E/S. O barramento de endereço transporta um número que especifica a posição de memória da instrução ou do dado que o microprocessador está tentando acessar.

8. Um conjunto de sinais de controle forma o barramento de controle e temporização. Eles coordenam a transferência de dados e indicam que tipo de dados deveriam estar presentes no barramento de dados.
9. Um microprocessador, em conjunto com um sistema de memória e dispositivos de E/S, forma um microcomputador.
10. Compreendendo a temporização dos barramentos, podemos adicionar dispositivos periféricos num sistema de microcomputador e adaptar o hardware às nossas necessidades.
11. Entendendo a linguagem de programação de um microprocessador, podemos programá-lo para realizar diversos tipos de tarefas.
12. Microcomputadores podem ser sistemas modulares, versáteis, facilmente programáveis, como os computadores pessoais que são usados para os mais diferentes propósitos.
13. Microcomputadores podem estar contidos em um único chip que fica embutido num produto, tal como em um forno de microondas ou em um videocassete, com o objetivo de controlar a operação do dispositivo.

barramento de endereço
barramento de dados
barramento de controle
ciclo de máquina
ciclo de barramento

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 13-4

1. Entrada, saída, lógica e aritmética, controle, memória. Veja o texto para as funções
2. Unidades de controle e lógica e aritmética combinadas
3. A sincronização da transmissão de informação digital entre o computador e dispositivos de E/S externos
4. Busca e execução

SEÇÃO 13-5

1. O microprocessador (MPU) é a CPU do microcomputador
2. MPU, RAM, ROM, entrada, saída
3. Temporização e controle, registradores, ULA
4. Indicar os endereços das instruções

SEÇÃO 13-6

1. Dados e instruções
2. Executa programas numa taxa mais elevada

SEÇÃO 13-7

1. Um código binário que representa a operação a ser realizada pela CPU
2. O endereço do dado a ser manipulado quando a CPU executa a instrução indicada pelo opcode
3. Opcode

SEÇÃO 13-8

1. Um programa em linguagem de máquina consiste em códigos binários de instruções armazenados na memória do computador. Um programa em linguagem de alto nível é escrito numa linguagem conversacional que deve ser convertido para linguagem de máquina antes que o computador possa executá-lo. 2. Uma pequena abreviação para a operação. 3. Durante um ciclo de busca, a CPU busca o opcode e o endereço do operando da memória. Durante o ciclo de execução, a CPU executa a operação indicada pelo opcode. 4. O PC indica os endereços das instruções na memória.

SEÇÃO 13-9

1. Barramento de dados: transporta dados entre a CPU e a memória e os dispositivos periféricos. Barramento de endereço: transporta o código de endereço da CPU para a memória e para os dispositivos de E/S. Barramento de controle: transporta sinais de temporização e sincronização. 2. Barramento de endereço. 3. (a) byte inferior do endereço, (b) byte de instrução, (c) byte de dados, (d) byte de dados. 4. 65.536

TERMOS IMPORTANTES

programa
programador
memória
instruções
dados
endereço
operação
endereço
operando
endereço do operando
unidade lógica e aritmética (ULA)
unidade de memória
unidade de controle
unidade de entrada
unidade de saída
periférico
interfaceamento
busca
executa
unidade central de processamento (CPU)
unidade microprocessadora (MPU)
contador de programa (PC)
acumulador
palavra
tamanho de palavra
código de operação (opcode)
modo de endereçamento
endereçamento direto
endereçamento imediato
linguagem de máquina
linguagem de alto nível
compilador
mnemônico
linguagem de montagem
ciclo de busca
ciclo de execução