1. Introdução

Os computadores nos auxiliam em diversas tarefas do dia a dia, sejam profissionais, acadêmicas, de diversão, etc. Essa generalidade se deve ao fato de poderem ser programados conforme a aplicação desejada. Um programa é uma sequência de instruções que realiza uma determinada tarefa.

Os computadores só conseguem entender instruções muito básicas como somar dois números, copiar um dado da memória para um dispositivo de saída, ou verificar se resultado de uma operação foi negativo. Essas instruções básicas são chamadas de instruções de máquina.

Qualquer aplicação precisa ser escrita, ou convertida através de compiladores, para linguagem de máquina para que possa ser executada. Neste material, veremos como esses programas em linguagem de máquina são executados ou, em outras palavras, como o computador e, em especial, a CPU (microprocessador) funcionam. Para isto estudaremos um processador genérico simples.

1. A CPU e o Computador

A unidade central de processamento (CPU – Central Processing Unit) é o componente principal de um computador. Ela é composta por três blocos principais e o devido sistema interno de interconexão. Esses blocos são:

● ULA – Unidade Lógica e Aritmética — Encarregada de executar todas as operações dessas naturezas.

• Operações lógicas: E, OU, NÃO, OU Exclusivo, Coincidência, etc.;

• Operações aritméticas: soma, subtração, multiplicação, divisão e deslocamentos. E aqui, é sem “et cetera”, já que os computadores não fazem muito mais do que isso.

● Registradores — São a unidade de armazenamento mais básica, tanto para os dados que serão processados, como para os dados resultantes desse processamento. Alguns exemplos de registradores são o de endereços e o de instruções, o acumulador e o contador de programa.

● Unidade de Controle (ou Unidade de Temporização e Controle, conforme a referência) — Controla a perfeita execução das sequências de instruções (programas), garantindo o sincronismo (tempo) e a ordenação correta de cada passo; bem como, interpretando e gerando os sinais de controle externos à CPU.

Para compor um computador, a CPU é interligada a um sistema de armazenamento (memórias) e a dispositivos de entradas e saídas (E/S). Essa interligação é feita através de um sistema de barramentos.

De uma forma simplificada, o sistema de armazenamento constitui-se da memória RAM, onde ficam instruções (programas) e dados, e de dispositivos externos como, por exemplo, discos rígidos (HD), ópticos (CD e DVD) e dispositivos baseados em memória Flash (pen drives e cartões de memória).

Os dispositivos de entradas e saídas (sistema de E/S) permitem a internalização de dados, para armazenamento ou, processamento; a externalização dos resultados e o acesso aos dados armazenados. Ele é constituído por módulos de E/S, os chamados controladores e interfaces, e pelos periféricos. Alguns exemplos são: placa de vídeo, NIC (Network Interface Card, a placa de rede) e controladora USB, para o primeiro grupo, e teclado, mouse, monitor, impressora, para o segundo grupo.

Os barramentos podem ser definidos como um conjunto de sinais ou, fisicamente, como um agrupamento de trilhas em uma placa de circuito impresso (PCI, ou PCB no inglês). Obviamente esses sinais e trilhas conectam os diversos circuitos e dispositivos que fazem parte do computador e, em geral, são vias de comunicação compartilhadas. Cada barramento segue uma especificação lógica, elétrica e mecânica adequadas ou, em outras palavras, tipos de sinais, velocidade, níveis de tensão, padrão de pinagem dos circuitos integrados (“chips”) e dos conectores, etc. Alguns exemplos de barramentos são: PCI Express, PCI, USB, SATA, AGP, DDR, etc. Podemos classificar os barramentos em três tipos principais:

● de endereço — seleção da posição de memória ou dispositivo de E/S a ser acessado;

● de dados — trafega os dados que são lidos e escritos de, e para, a memória e de, e para, os dispositivos de E/S;

● de controle — trafega todos os sinais de temporização e controle.

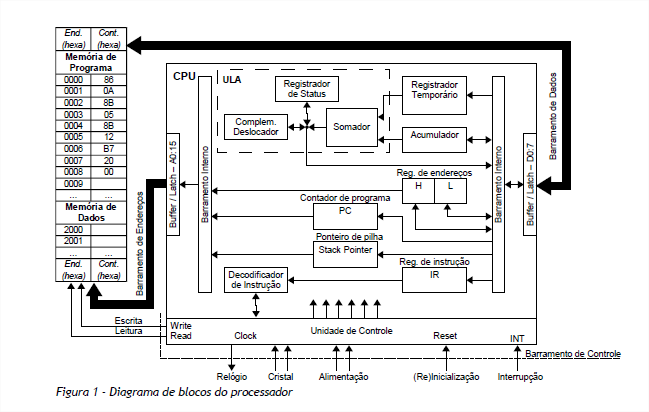
Alguns autores consideram um quarto barramento, o de alimentação, responsável pela energização dos circuitos, mas esse seria um barramento que não trafegaria informação, então não o consideraremos como tal.

Os microprocessadores são circuitos integrados que abrigam em seu encapsulamento uma CPU completa. Posteriormente foram incorporados memória cache e outros circuitos como controladores de E/S e de Memória dentro do mesmo encapsulamento (do mesmo chip físico). Atualmente é bastante comum encontrar diversas CPUs e mesmo GPUs (Unidade de Processamento Gráfico) em um mesmo chip.

Além dos microprocessadores, existem também os microcontroladores que em um mesmo encapsulamento abrigam uma CPU completa, sistema de armazenamento, controladores de dispositivos de E/S e os devidos barramentos. Os microcontroladores são muito utilizados em computadores dedicados, como: eletrodomésticos (forno de micro-ondas, lavadoras); eletrônicos de consumo (TV, som, vídeo); automação industrial (controle de máquinas e instrumentação); etc.

1. A Descrição do computador

A Figura 1 mostra o diagrama de blocos do processador genérico que utilizaremos. Trata-se, na verdade, de uma simplificação dos microprocessadores 8080 e 8085 da Intel, muito similar às encontradas em Tokheim (1985) e Malvino (1985). A estrutura encontrada neste nosso exemplo é muito próxima a utilizada até hoje nos processadores comerciais. O 8080, lançado pela Intel em 1974, foi o primeiro microprocessador de 8 bits de propósito geral, sendo utilizado nos primeiros microcomputadores.



Fora da CPU, notamos a memória, que está dividida em programa (instruções) e dados (usuário), muito embora em um mesmo chip. Notamos também a presença de dois sinais de controle, um para habilitar a leitura e outro para habilitar a escrita. Esses sinais são gerados pela unidade de controle da CPU e fazem parte do barramento de controle.

Ainda na memória vemos a conexão do barramento de endereços e do barramento de dados. A informação no barramento de endereços indica a posição de memória a ser acessada; enquanto através do barramento de dados trafegam os dados que são lidos, ou escritos, na memória. O barramento de endereços desse processador é de 16 bits, o que permite 2 16 (65536, ou 64k) posições de memória. Porém, cada uma dessas posições armazena oito bits, ou um byte (B), que é também a largura do barramento de dados (tamanho). Assim, este computador pode ter até 64k posições de 1 byte, ou 64kB (kilobytes) de memória RAM.

O acesso aos dispositivos de E/S é similar, utilizando sinais de controle apropriados. É muito comum que o endereçamento desses dispositivos seja feito através de uma faixa dedicada de posições de memória. Um exemplo poderia ser a faixa de endereços de FF00 a FFFF, o que permitiria endereçar 256 dispositivos. O circuito encarregado de interpretar essa faixa de endereços e acionar cada dispositivo é chamado de decodificador de endereços, mas não aparece no diagrama.

3.1.Descrição da CPU

Ainda com referência à Figura 1, podemos observar os diversos sinais que integram o barramento de controle, além das entradas de alimentação e de cristal, que é um componente eletrônico utilizado para gerar uma base de tempo de alta precisão. Essa base de tempo é subdividida dentro da CPU, para gerar um sinal de relógio (clock) responsável por sincronizar todas as atividades da CPU e do computador.

Além do sinal de relógio e dos sinais de leitura e escrita comentados anteriormente, há um sinal de inicialização da CPU (reset), que faz com que a CPU retorne a um estado inicial definido e conhecido, para que possa começar a executar os programas. Há também uma entrada de sinal de interrupção (INT) que será detalhada posteriormente.

Dentro da CPU podemos ver a unidade de controle, a ULA (com sua estrutura interna), os diversos registradores e as interconexões entre eles. Note que, para maior clareza, foram representados diversos sinais saindo da unidade de controle, mas sem nenhuma conexão. O que ocorre de fato é que a unidade de controle é conectada à ULA e a todos os registradores da CPU, gerando os sinais de leitura, escrita, habilitação, clock, etc.

O decodificador de instruções contém os passos a serem executados para cada instrução. Ele “interpreta” a instrução armazenada no registrador de instrução e aciona os circuitos ou microprograma correspondente para a correta execução da instrução. É, na verdade, parte da unidade de controle.

Além da ULA e da unidade de controle (e decodificador de instrução), a CPU é composta por diversos registradores. As funções dos registradores da CPU são detalhadas a seguir:

● Acumulador (AC - accumulator) — Armazena o resultado das operações realizadas na ULA e um dos operandos;

• Embora Acumulador seja o nome mais comum, também pode aparecer como registrador Work (W) em algumas arquiteturas.

● Registrador temporário (T - temporary) — Armazena um dos operandos a serem processados na ULA;

• É necessário ao menos um registrador para cumprir esse papel, pode aparecer como um ou mais registradores auxiliares, ou com nomes como RB, RC, etc.;

• Nos microprocessadores 8080 e 8085 reais há 3 pares de registradores auxiliares, a saber: B e C, D e E, H e L (este par descrito adiante); além de mais um par de registradores de uso interno pela unidade de controle (W e Z).

● Contador de programa (PC – Program Counter) — Armazena o endereço (na memória) da próxima instrução a ser executada;

• É chamado de Instruction Pointer (ponteiro de instrução) nos processadores Intel posteriores ao 8080 e 8085.

● Registrador de instrução (IR – Instruction Register) — Armazena a instrução em execução (ou a ser executada no ciclo de execução);

● Registrador de endereços (H, L) — Podem ser utilizados como registradores de uso geral, mas normalmente funcionam como um par de registradores (16 bits) para armazenamento de um endereço (posição) de memória.

● Ponteiro de pilha (Stack Pointer) — Utilizado para indicar a base (ou topo) da área de memória utilizada como pilha.

● Registrador de status (estado) da ULA — Consiste-se de uma série de indicadores binários (flags) referentes aos resultados das operações ocorridas na ULA. Exemplos: zero, negativo, positivo, estouro (overflow), “vai um” (carry), etc.

• Em muitas arquieturas esse registrador é chamado de PSW – Program Status Word, armazenando, além das flags, outras importantes características de controle do computador, tais como clock, controle de interrupções e modo supervisor.

• Nos microprocessadores 8080 e 8085 reais há 5 flags: Zero, Sinal (positivo ou negativo), Transporte (Carry, o “vai um” ou “empresta um”), Paridade (quantidade par ou ímpar de bits “1” no número) e Transporte auxiliar (Auxiliary Carry, indica o transporte do 4º para o 5º bit, usado em algumas operações específicas).

● Latch de Endereços (Address Latch) – Armazena o endereço (posição) a ser acessado na memória. É a interface entre a CPU e a memória.

• O nome mais comum deste registrador é MAR — Memory Address Register.

● Latch de Dados (Data Latch) – Armazena o dado (conteúdo) a ser lido ou escrito na memória.

•O nome mais comum deste registrador é MBR — Memory Buffer Register.

Note que os latches de dados e endereços (MBR e MAR) são a interface entre a CPU e a memória e possuem tamanho idêntico à largura dos barramentos de dados e endereços aos quais são conectados.

Além dos registradores descritos é comum haver outros registradores, quer sejam para um uso específico, quer sejam de propósito geral. Por outro lado, os registradores absolutamente indispensáveis para o funcionamento de qualquer CPU são seis: o PC e o IR, para que se controle a execução do programa; o MAR e o MBR, para a comunicação com a memória; e o AC e um segundo registrador para o funcionamento da ULA (nesse caso dois ou três indicadores binários também seriam recomendados).

1. O Funcionamento do Microprocessador
   1. O Programa

O programa é uma sequência de instruções. Cada instrução é constituída de um código de operação (op code), que indica o que será feito, e de operandos (dados), que podem ser números, caracteres, ou endereços a serem operados.

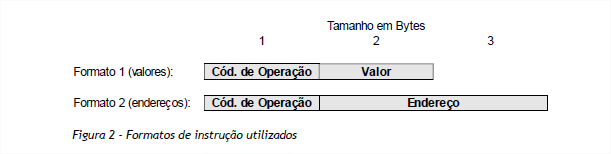
Acontece que os microprocessadores só “entendem” instruções em linguagem de máquina, isto é, em códigos binários específicos. Usamos o sistema hexadecimal para compactar a exibição desses binários, mas, de qualquer forma, é algo ininteligível ao ser humano.

Para facilitar o trabalho dos programadores é adotada uma série de mnemônicos (códigos que remetem à operação realizada, formados por contrações ou siglas dos termos) que tornam os códigos de máquina mais acessíveis. Assim, para mover uma informação podemos usar o mnemônico “MOV” e para efetuar uma adição, o mnemônico “ADD”. Esta linguagem de mnemônicos é chamada de linguagem de montagem, ou assembly.

Essa linguagem de máquina mais “inteligível”, que é o assembly, passa por um processo de montagem para gerar um programa em linguagem de máquina (código de máquina). Isto é feito por um programa montador, ou assembler.

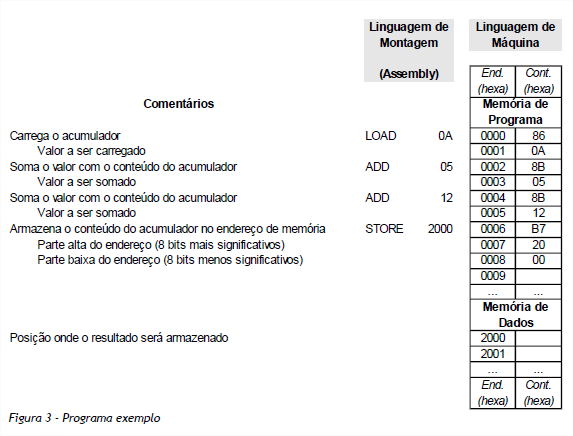
Não se esqueça e não confunda: assembly é o mesmo que linguagem de montagem e é usado pelo ser humano, assembler é um programa que converte linguagem de montagem em linguagem de máquina e a linguagem de máquina é utilizada no computador, para tanto, precisa ser carregada na memória dos mesmos.

Em nosso processador, cada operação possui um código de oito bits, ou seja, há capacidade para 256 instruções. Se o operando da instrução for um valor (número ou caractere), ele ocupará a posição de memória imediatamente posterior a do código de operação. Se o operando for um endereço, ele ocupará as duas posições de memória imediatamente posteriores a do código de operação. Assim, temos dois formatos de instrução para nosso processador, que são representados conforme a Figura 2.



Na verdade, a linguagem de montagem (assemby) utilizada em nosso processador exemplo não é a original do 8080/8085, mas sim uma simplificação. O item 4.1.1 apresenta como seria o programa exemplo no assembly original do 8080/8085.

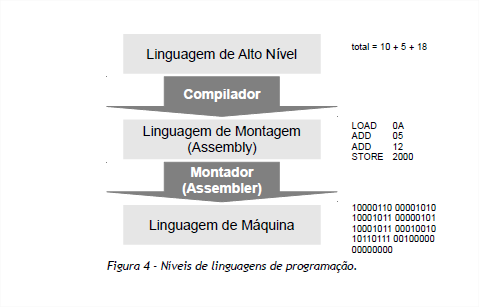
O programa exemplo a ser executado é o que está carregado na memória do computador na Figura 1. Nela, tanto as instruções, como os dados estão indicados em hexadecimal, que como vimos, é apenas uma forma compacta da linguagem de máquina binária. Na Figura 3, reproduzimos o programa em linguagem de máquina com as devidas explicações (comentários) e o programa equivalente em linguagem de montagem (assembly).



Portanto, a tarefa executada por este programa é a de somar três números e armazenar o resultado na posição de memória 2000.

Isso tudo poderia ter partido de um programa feito em linguagem de alto nível, que recebe esse nome por ser bastante próxima da “linguagem dos seres humanos”, especialmente, na maioria dos casos, da língua inglesa. Para converter o programa escrito em uma linguagem de alto nível em linguagem de baixo nível utiliza-se um programa compilador.

A Figura 4 mostra o relacionamento entre os diversos níveis de linguagem utilizados para a programação dos processadores e os programas utilizados para a conversão entre essas linguagens.



4.1.1.O Programa Real no 8080/8085

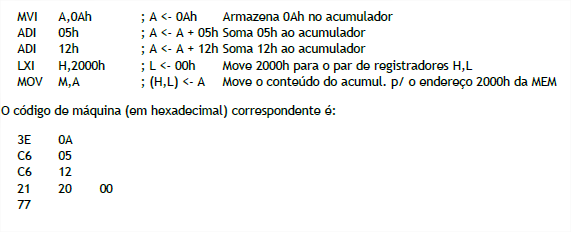
O programa exemplo foi escrito em uma linguagem de montagem (assembly) fictícia, ao mesmo tempo mais didática e mais próxima das linguagens reais predominantes atualmente em arquiteturas RISC, usando um padrão de 1 endereço.

O assembly do 8080/8085 mescla instruções de 0, 1 e 2 endereços, padrão comum em uma

arquitetura CISC e usado até hoje pela Intel.

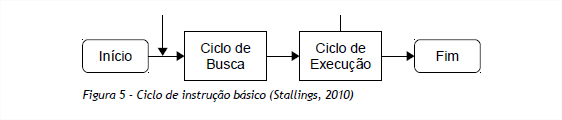
O código que corresponde ao programa exemplo Utilizando o assembly real do 8080/8085 é o

seguinte:

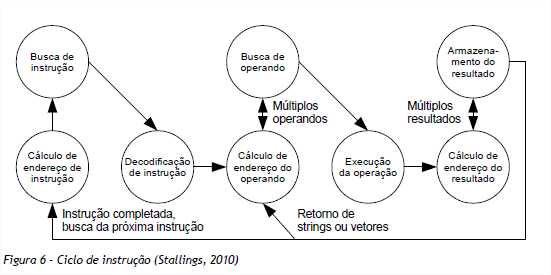


4.2.Ciclo de Instrução

Para que cada instrução seja executada, o processador, e consequentemente todo o computador, segue determinados passos, correspondentes ao ciclo de instrução. Resumidamente, o ciclo de instrução é composto dos ciclos de busca e de execução, conforme mostra a Figura 5.



A Figura 6 traz o ciclo de instrução detalhado, que pode ser aplicado a qualquer computador. O ciclo de busca é constituído dos três primeiros passos da Figura 6. Nele o processador cuida da instrução: onde está, qual é e como executá-la. O ciclo de execução é constituído pelos passos seguintes. Nele o processador cuida de descobrir quais são os operandos e obtê-los, executar a operação, descobrir onde armazenar o resultado e então fazê-lo.



As atividades realizadas em cada etapa do ciclo de instrução são (Stallings, 2010 e Tanenbaum, 2001):

● Cálculo do endereço de instrução — Verifica qual a próxima instrução a ser executada. Em geral, corresponde à leitura do PC.

● Busca de instrução — Obtém a instrução no endereço indicado (conteúdo da posição de memória indicada no PC).

● Decodificação de instrução — Interpreta a intrução acionando o microprograma correspondente.

● Cálculo de endereço do operando — Verifica onde encontrar o(s) operando(s) da instrução (memória, dispositivo de E/S, registrador).

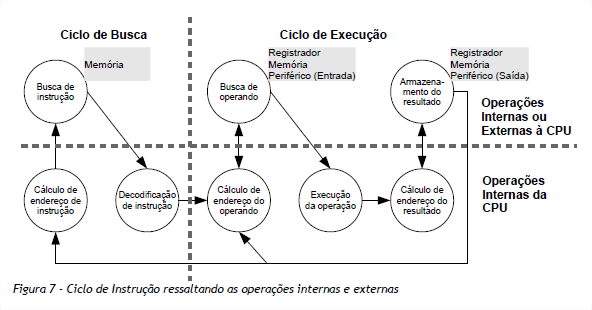
● Busca do operando — Obtém o(s) operando(s) especificados.

● Execução da operação

● Cálculo de endereço do resultado — Verifica onde o resultado da operação deve ser guardado.

● Armazenamento do resultado — Escreve o resultado no local indicado (memória, dispositivo de E/S, registrador).

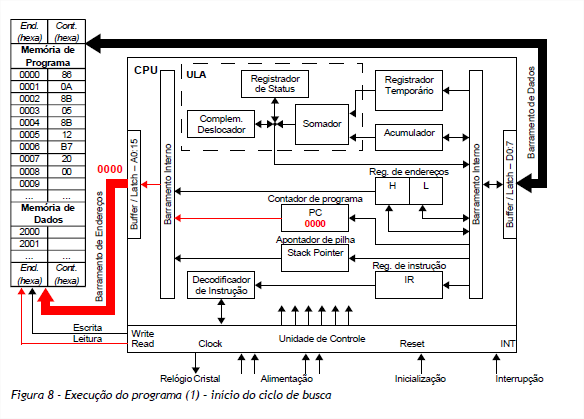
Note que, no ciclo de busca, a instrução estará sempre em uma posição de memória, ou seja, fora da CPU. Além disso, durante o ciclo de execução os operandos podem ser lidos ou escritos fora da CPU. Esses pontos são ressaltados na Figura 7 e implicam em operações de leitura e escrita em dispositivos externos, mais lentos, e no consequente “gargalo” para o desempenho na execução dos programas.



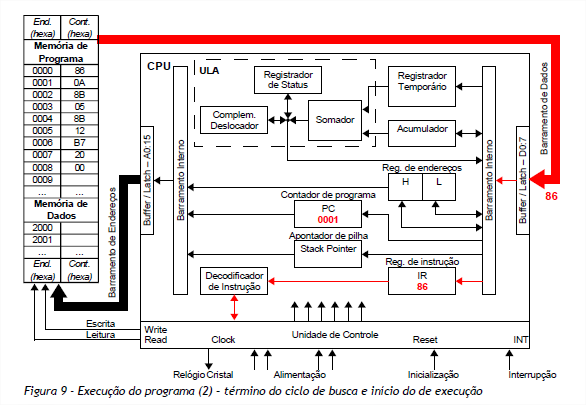
4.3.Execução do Programa

Agora que foram vistos o programa exemplo e o ciclo de execução de cada instrução, vejamos o programa exemplo sendo executado.

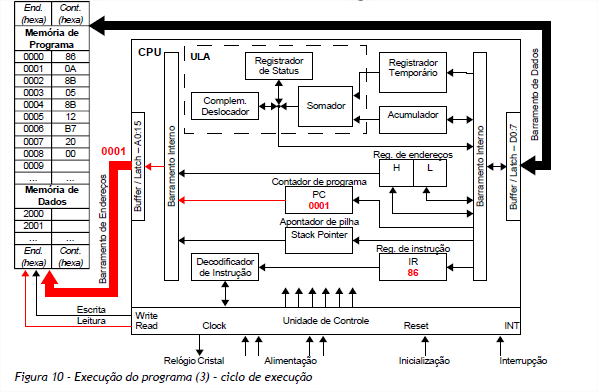
Quando inicializado o microprocessador genérico, o PC contém o endereço 0000. Isto significa que este será o endereço da primeira instrução. O processador coloca o valor 0000 no barramento de endereços e aciona o sinal de leitura da memória, conforme mostra a Figura 8, dando início ao ciclo de busca.



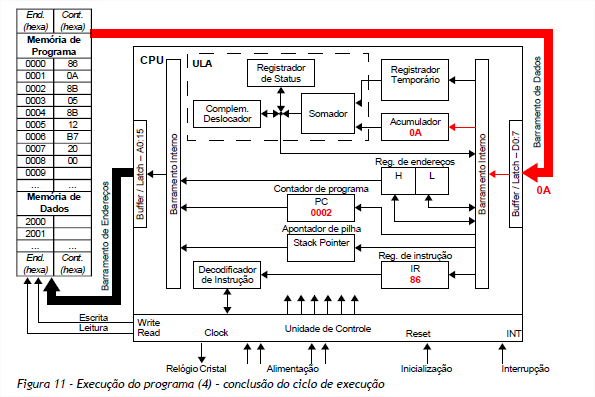
A seguir, na Figura 9, o conteúdo da posição de memória 0000, ou seja, a instrução, é trazida através do barramento de dados e armazenada no registrador de instrução. Daí ela passa ao decodificador de instruções que se comunica com a unidade de controle, finalizando o ciclo de busca. O contador de programa também é incrementado para que se possa obter o operando da instrução, ou seja, tem início o ciclo de execução.



O próximo passo do ciclo de execução é a obtenção do operando. O processador coloca o endereço do operando, correspondente ao PC já incrementado (0001) no barramento de endereço e aciona o sinal de leitura novamente, conforme mostra a Figura 10.

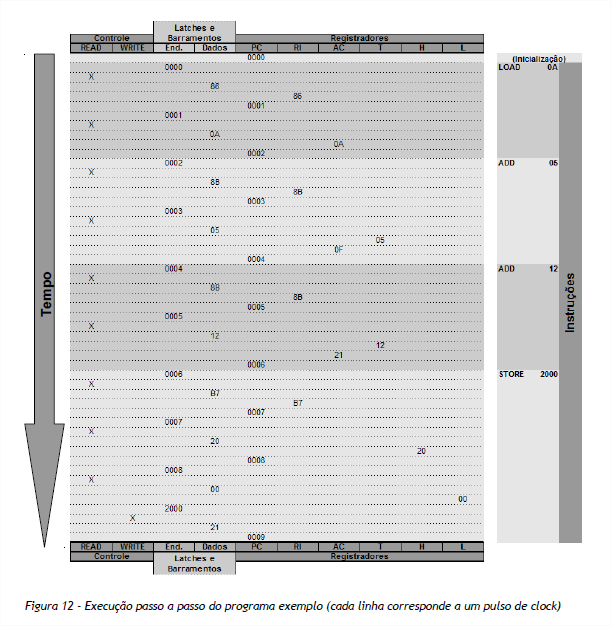


O conteúdo do endereço 0001 (0A) é trazido pelo barramento de dados e armazenado no acumulador. A primeira instrução do programa é concluída com o incremento do PC para 0002. Ou seja, os ciclos de execução e de instrução estão concluídos. Vide Figura 11.



O programa continua com a repetição dos passos (ciclo de instrução) para a execução das próximas instruções.

A Figura 12 mostra uma tabela com a execução passo a passo do programa exemplo ao longo do tempo, até a obtenção do resultado final. Cada linha da tabela corresponde a um instante (pulso de clock/relógio), exibindo os dados ou sinais gerados (ou armazenados) naquele instante. Os dados armazenados nos registradores permanecem armazenados até que sejam sobrescritos.



1. Referências Bibliográficas

• VALIANTE FILHO, F. Princípio do Funcionamento de um Microprocessador. v. 5, p. 1 - 14, set. 2013.

• VALIANTE FILHO, F. Desenvolvimento de Projetos Utilizando Microcontroladores. Revista Eletrônica Total, São

Paulo, v. 108, p. 8 - 11, 01 jun. 2005.

• CARVALHO, Claudio Sergio da Rós de. Microprocessador 8085. 2ª ed. Editora da Unicamp. 1990.

• MALVINO, Albert P. Microprocessadores e Microcomputadores. Editora McGraw-Hill. 1985.

• STALLINGS, William. Arquitetura e Organização de Computadores. 8ª edição. Pearson Education. 2010.

• TANENBAUM, Andrew S. Organização Estruturada de Computadores. 4ª edição. LTC Editora. 2001.

• TOKHEIM, Roger L. Introdução aos Microprocessadores (Coleção Schaum). Editora McGraw-Hill. 1985.