Aritmética Computacional e Conjunto de Instruções

Aritmética Computacional

Introdução

• Aritmética de computador realiza operações em dois diferentes tipos numéricos:

- ✓ Inteiros;
- ✓ Ponto Flutuante;

Introdução

• Cada uma tem uma característica particular e é tratada de forma diferente.

• Quem realiza essas operações de fato, é a Unidade Lógica/Aritmética.

Unidade Lógica/Aritmética (ULA)

• É quem realmente desempenha as operações lógicas e artiméticas;

• Os outros componentes, como Unidade de Controle, registradores, memória, dispositivos de E/S, realizam uma tarefa principal, que é trazer os dados para dentro da ULA.

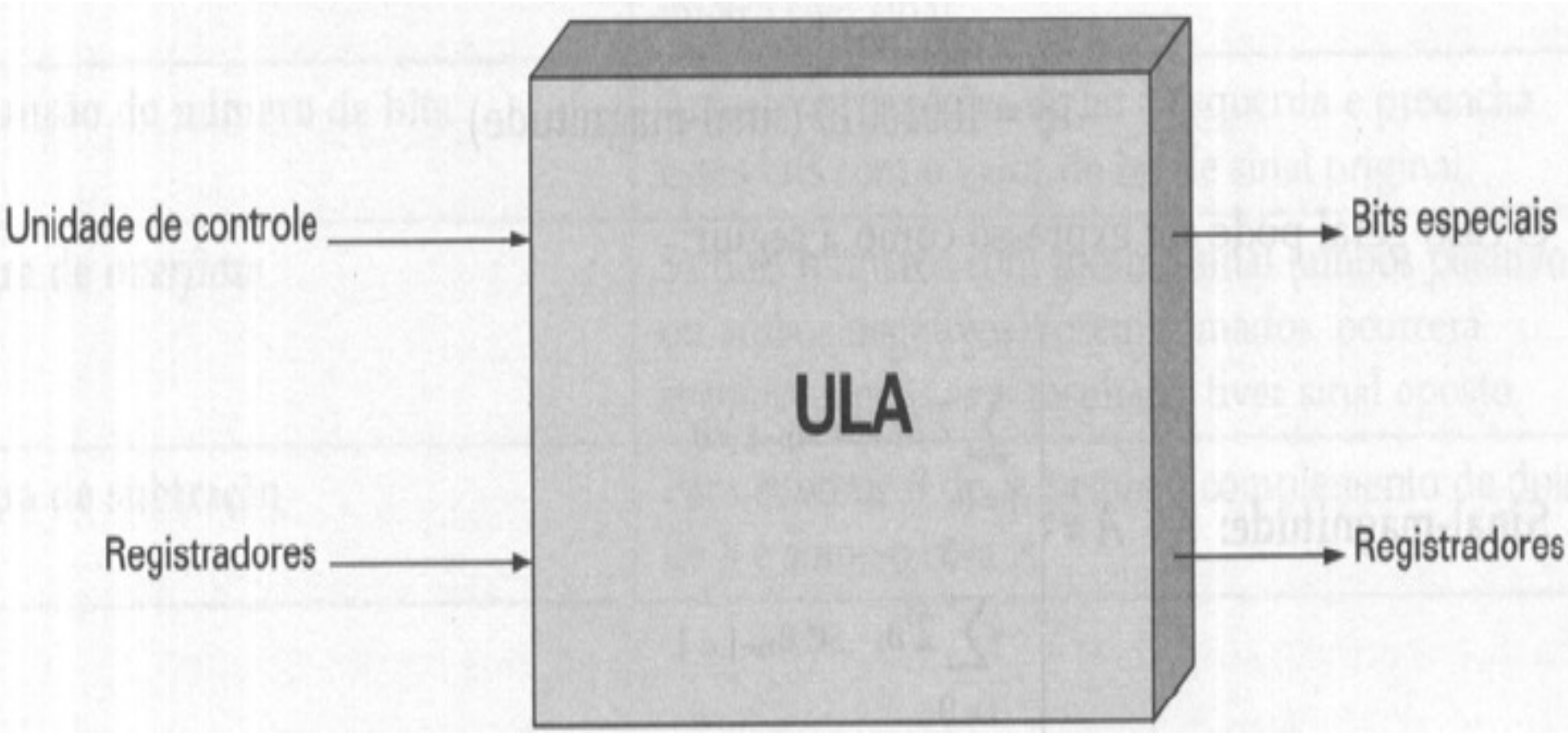
Unidade Lógica/Aritmética

• Esses mesmo componentes esperam para levar os resultados de volta;

• Quando fala-se de ULA, passa-se a tocar no núcleo do computador;

• A ULA, assim como os demais componentes eletrônicos no computador, baseiam-se na realização de operações de lógica booleana simples.

ULA



Fonte: Arquitetura e Organização de Computadores – William Stallings

Operação da ULA

• Dados são apresentados à ULA através de registradores;

• Os resultados das operações também são guardados em registradores;

• Esses registradores são localizações de armazenamento temporário dentro do processador.

8

Operação da ULA

• A ULA também pode acionar algumas *flags* como resultado de uma operação;

• Os valores das *Flags* geralmente são guardados nos registradores dentro do processador;

• A Unidade de Controle provê sinais que controlam a operação e o movimento dos dados para dentro e para fora da ULA.

9

Algumas operações

• Conhecimento prévio de operações com números binários;

• As operações utilizadas para números reais, também são utilizadas para números binários, como: Adição, subtração, multiplicação e divisão;

• Todas essas operações podem ser realizadas tanto com números inteiros como com números em ponto flutuante.

10

Conjunto de Instruções

O que é um conjunto de instruções?

- A coleção completa de instruções que são entendidas por uma UCP.
- Código de máquina.
- Binário.
- Normalmente, representado por códigos em *assembly*.

Elementos básicos de uma instrução

- Código de operação (Op code);
- Referência a operando fonte;
- Referência a operando de destino;
- Referência à próxima instrução.

Considerações sobre ISA (*Instruction Set Architecture*)

• Para um programador de alto nível, não se mostra interessante ter a visibilidade das camadas mais baixas da arquitetura;

• Um limitador entre o Projetista de Computadores e o Programador de Computadores é conjunto de instruções de máquina;

Considerações sobre ISA (*Instruction Set Architecture*)

• Para o Projetista, a o conjunto de instruções prove as necessidades funcionais para a UCP, por exemplo implementar a UCP é, em grande parte, implementar os conjuntos de instruções de máquina;

• Diferente do programador que já tem o que ele quer fazer e passa essas instruções para a máquina.

Características das Instruções de Máquina

- A operação da UCP é determinada pelas instruções que ela executa;

- Essas instruções são chamadas de instruções de máquina;

- Elas são colocadas no Conjunto de Instruções da UCP.

Elementos de uma instrução de máquina

- Cada instrução deve conter os elementos necessários para que a UCP a execute;

- Os elementos de uma instrução de máquina são:

  ✓**Código de Operação:** Especifica a operação a ser desempenhada (ADD, I/O). É especificado por um código binário também chamado de *opcode.*

17

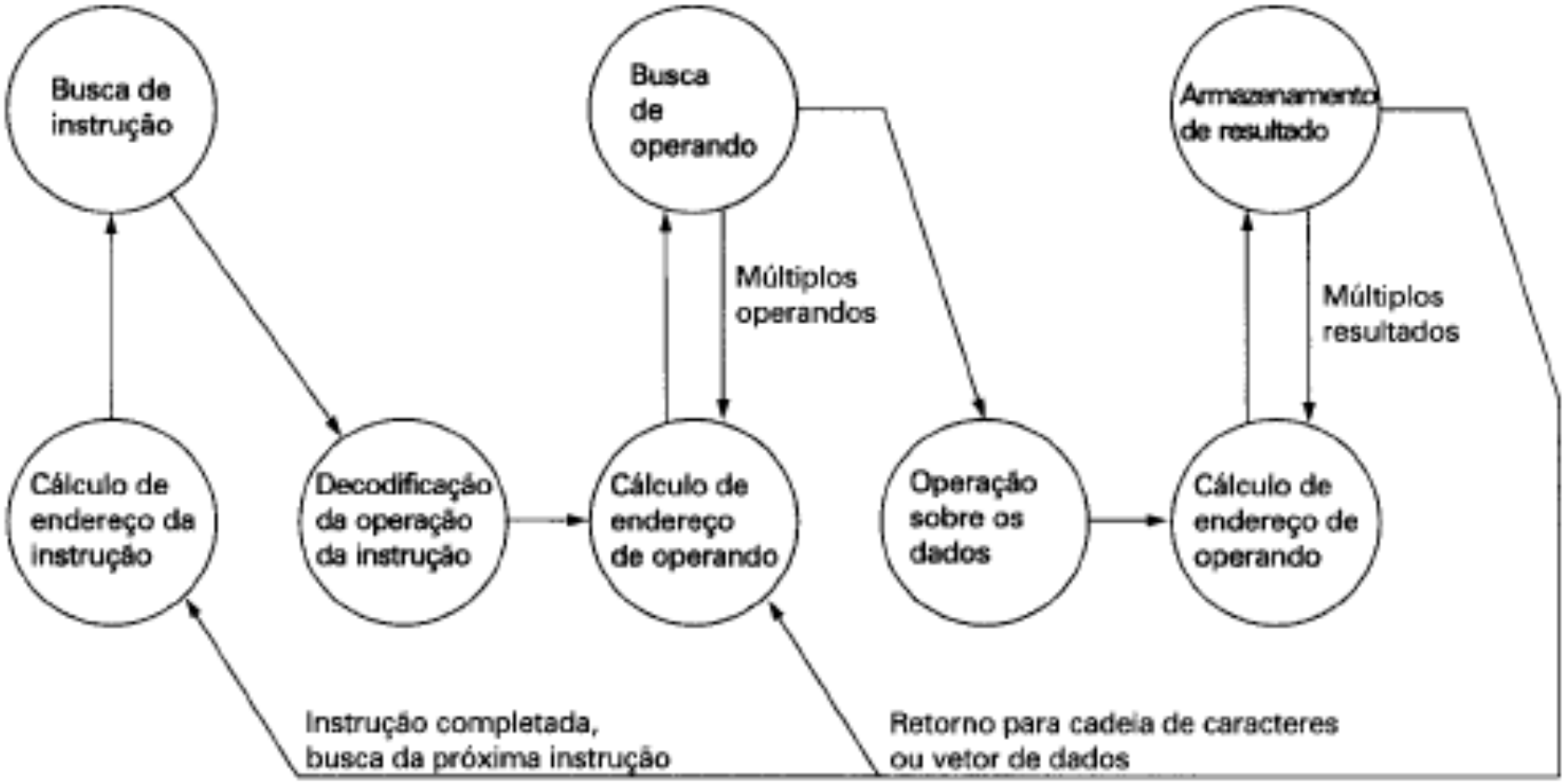
Elementos de uma instrução de máquina

✓ **Referencia ao operando fonte:** A operação pode envolver um ou mais operandos fonte. Assim, serve para indicar quais são as entradas para uma determinada operação;

✓ **Referência ao operando destino:** Resultado da operação realizada;

✓ **Referência para a próxima instrução:** Essa referência indica para a UCP onde carregar a próxima instrução depois que a atual for terminada.

Diagrama de um ciclo de instrução



Fonte: Arquitetura e Organização de Computadores – William Stallings

Os operandos

Eles podem ser encontrados em:

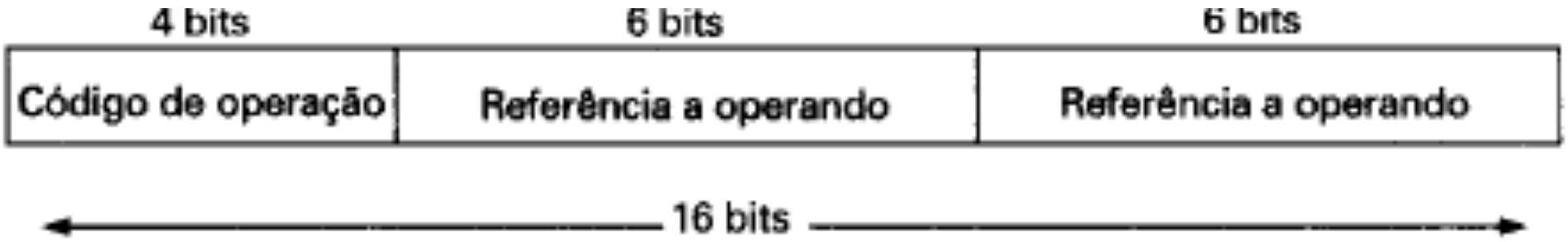
• Memória principal e memória virtual;

• Registrador da UCP: Geralmente usa um único registrador de instrução, mas caso tenha mais de um registrador, o seu endereço deve ser informado;

• Dispositivo de E/S;

Representação de instruções

Formato da representação de uma instrução



Fonte: Arquitetura e Organização de Computadores – William Stallings

Representação simbólica

- Devido a grande dificuldade dos programadores e dos leitores entenderem a linguagem de máquina, passou-se a utilizar uma representação simbólica;

- Representação simbólica das instruções de máquina;

- Códigos de operação, por exemplo, são representados por abreviações chamadas *mnemônicos*.

Representação simbólica

Exemplos de mnemônicos:

ADD – Adição
SUB – Subtração
MPY – Multiplicação
DIV – Divisão
LOAD – Carrega dados da memória
STORE – Guarda dados na memória
NOP – Ausência de operação

Representação simbólica

Exemplo:

ADD R, Y

Adicionar o conteúdo de dados localizado em **Y** ao conteúdo do registrador **R**.

Tipos de Instruções

- Tome a operação: X = X + Y;

- Assumindo que X corresponda à posição 513 da memória e que Y corresponda à posição 514;

- Essa operação pode ser descrita:
  - ✓ Carrega um registrador com o conteúdo da posição 513;
  - ✓ Adicione o conteúdo da memória à posição 514;
  - ✓ Guarde o conteúdo do registrador na posição 513.

Tipos de Instruções

- Nota-se na expressão anteriormente descrita, as diferenças entre a linguagem de alto-nível e a linguagem de máquina;

- Linguagem de alto-nível descreve uma operação de uma forma algébrica e usando variáveis;

- Na linguagem de máquina as operações ocorrem com o movimento de dados de e para os registradores.

Tipos de Instruções

• Um conjunto de instruções em um computador, deve permitir que ele execute qualquer tarefa;

• Uma linguagem de programação de alto-nível deve suprir todas as suas necessidades;

• Com isso, um conjunto de instruções de máquina deve ser criado para que as instruções traduzidas da linguagem de alto-nível possam ser executadas.

Tipos de Instruções

Categorias:

- **Processamento:** Instruções lógicas e aritméticas;
- **Armazenamento:** Instruções de memória;
- **Movimento:** Instruções de E/S;
- **Controle:** Instruções de teste.

Número de endereços

• Uma outra forma de descrever uma determinada arquitetura, através da quantidade de endereços que uma instrução vai necessitar;

• No entanto, essa preocupação com o número de endereços tem diminuído com a complexidade do projeto das UCPs atuais.

Número de endereços

• A instrução ideal mais interessante para se trabalhar, seria aquela com quatro endereços: dois operandos, um resultado e o endereço para próxima instrução;

• Geralmente não se trabalha com esse tipo de instrução;

• As mais usadas são as instruções com um, dois ou três endereços.

Exemplo de uma instrução

$Y = (A - B) / (C + D \times E)$

Execução com três endereços:

Instrução		Comentário
SUB	Y, A, B	$Y \leftarrow A - B$
MPY	T, D, E	$T \leftarrow D \times E$
ADD	T, T, C	$T \leftarrow T + C$
DIV	Y, Y, T	$Y \leftarrow Y \div T$

Exemplo de uma instrução

Execução com dois endereços:

Instrução		Comentário
MOVE	Y, A	Y ← A
SUB	Y, B	Y ← Y − B
MOVE	T, D	T ← D
MPY	T, E	T ← T × E
ADD	T, C	T ← T + C
DIV	Y, T	Y ← Y ÷ T

Exemplo de uma instrução

Execução com um endereço:

Instrução		Comentário
LOAD	D	AC ← D
MPY	E	AC ← AC × E
ADD	C	AC ← AC + C
STOR	Y	Y ← AC
LOAD	A	AC ← A
SUB	B	AC ← AC − B
DIV	Y	AC ← AC ÷ Y
STOR	Y	Y ← AC

Comentários sobre as execuções

- Para três endereços, o seu uso não é tão comum, pois requer um longo formato de instrução;

- Para dois endereços, o espaço reservado para as instruções é reduzido, porém começa a introduzir algumas complicações.

- Uso de instruções de movimento para realizar as operações.

Comentários sobre as execuções

• Na execução com um endereço, aparece a figura do Acumulador (AC);

• Ele armazena o resultado da operação;

• Existe ainda possibilidade de realizar operações com nenhum endereço, utilizando a pilha de execução.

Projeto do Conjunto de Instruções

• Deve ser levado em conta, de forma a reduzir a complexidade da UCP;

• As necessidades do programador devem ser levadas em conta no momento de projetar esse conjunto de instruções.

Projeto do Conjunto de Instruções

Alguns pontos principais a serem analisados:

- Diversidade de operações a serem implementadas;
- Os tipos de dados;
- O formato das instruções;
- Número de registradores a serem referenciados;
- Modos de endereçamento.

Tipos de operandos

Mais importantes:

- **Endereços;**

- **Números:** Tipos de dados numéricos, como Inteiros e Ponto Flutuante;

- **Caracteres:** Tipo comum em textos, também chamado de cadeia de caracteres;

- **Lógicos.**

Modos de endereçamento

Introdução

- Toda faixa de memória necessita de receber algum tipo de endereçamento para guardar ou recuperar as informações;

- Para isso existe uma variedade muito grande de modos de endereçamento;

Introdução

• Tudo isso envolve duas características, uma é o **balanceamento entre a faixa de endereçamento e a flexibilidade desse endereçamento**, e o **balanceamento entre o número de referências a memória e a complexidade de cálculo do endereço.**

Considerações

• Como a Unidade de Controle identifica qual o modo de endereçamento a ser utilizado;

• Nas arquiteturas convencionais, mais de um modo de endereçamento pode ser utilizado;

• Existe um *bit* que vai identificar o modo de endereçamento a ser utilizado.

Considerações

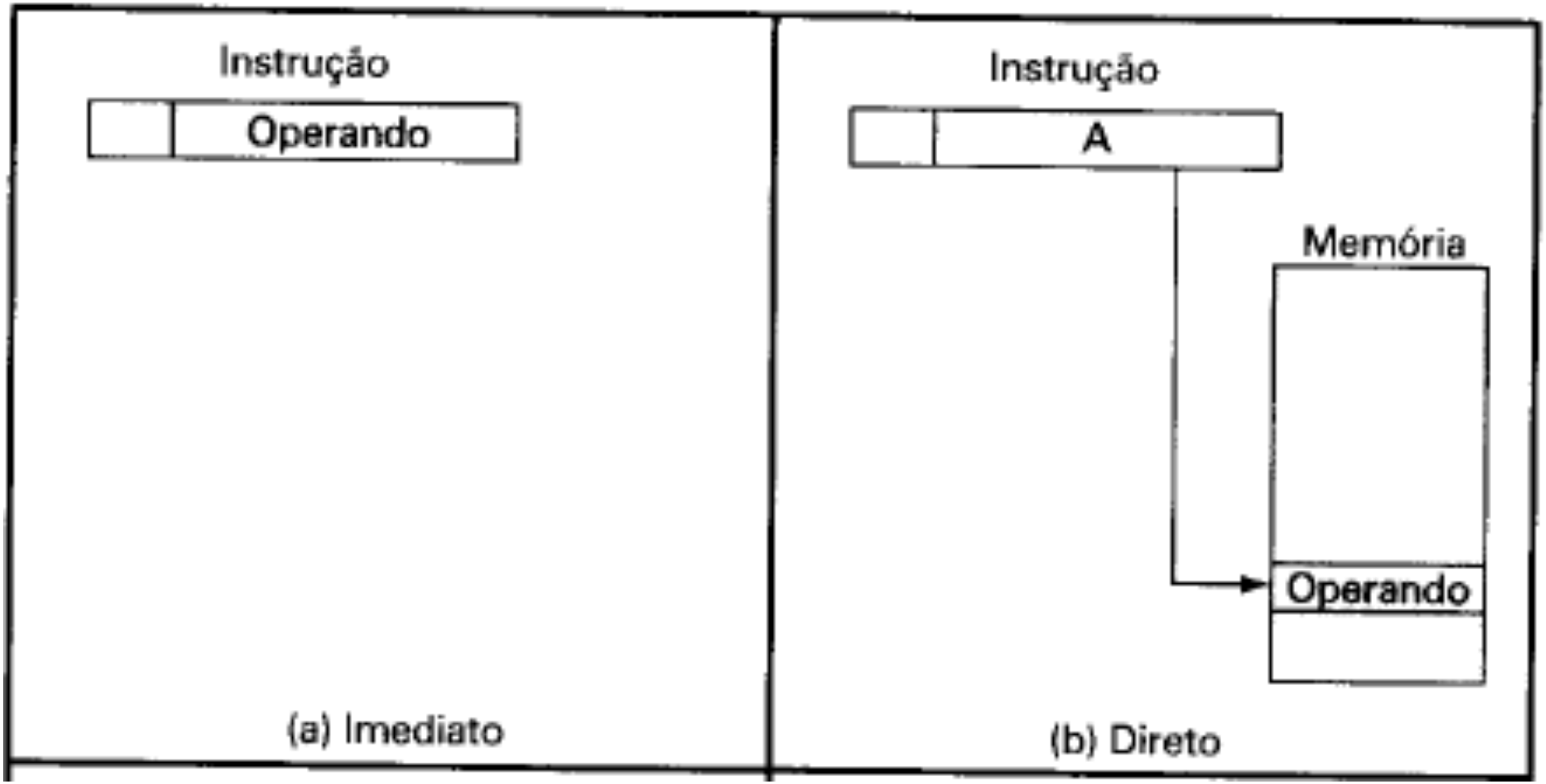
- Em sistemas que não possuam memória virtual, o endereço efetivo, ou seja aquele que guarda o operando de fato, estará ou no memória principal, ou no registrador;

- O mapeamento da memória virtual é invisível para o programador e é realizado pelo mecanismo de paginação.

Modos de Endereçamento

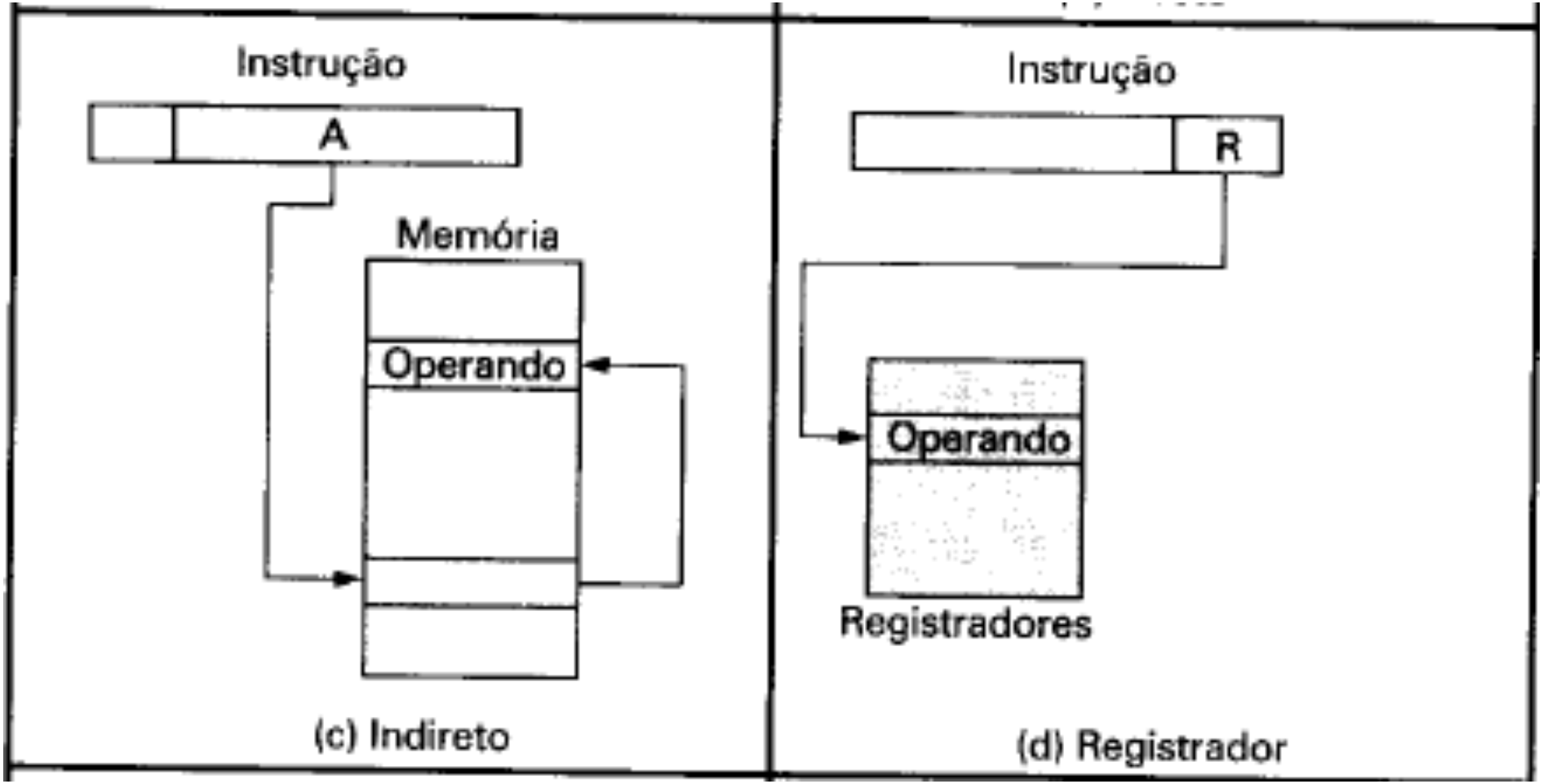
- Imediato;
- Direto;
- Indireto;
- Via Registrador;
- Via Registrador Indireto;
- Deslocamento;
- Pilha.

Modos de Endereçamento



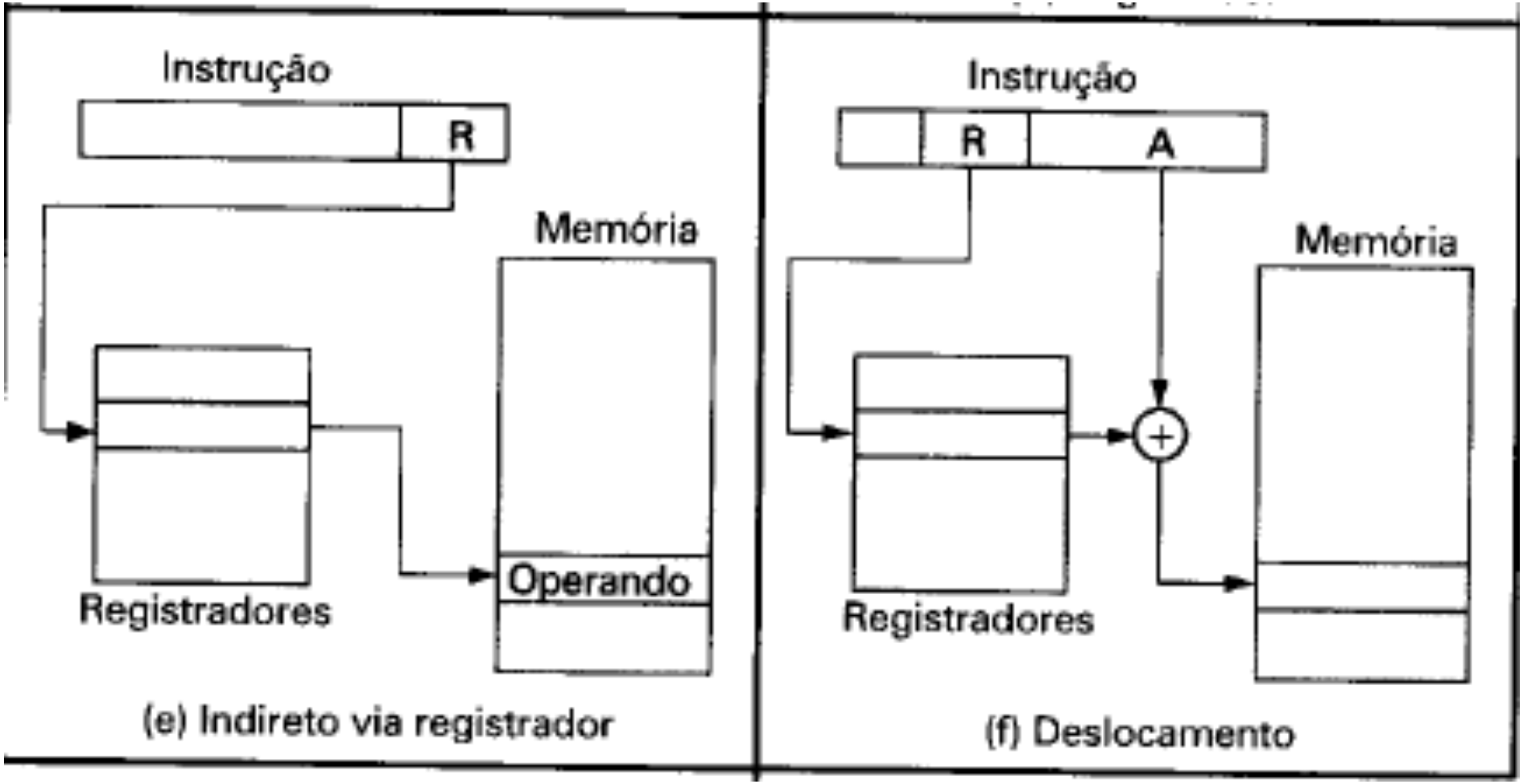
Fonte: Arquitetura e Organização de Computadores – William Stallings

Modos de Endereçamento



Fonte: Arquitetura e Organização de Computadores – William Stallings

Modos de Endereçamento



Fonte: Arquitetura e Organização de Computadores – William Stallings

Modos de Endereçamento

Fonte: Arquitetura e Organização de Computadores – William Stallings

Símbolos

- A = Conteúdo do campo de endereço na instrução;

- R = Conteúdo do campo de endereço na instrução que referencia um registrador;

- EA = Endereço real da posição que contém o operando;

- (X) = Conteúdo da posição do endereço X.

Endereçamento Imediato

- Forma mais simples de se endereçar;

- A vantagem desse modo é que não é necessário realizar uma referência a memória para armazenar ou buscar um operando, economizando um ciclo de instrução;

- A desvantagem é que o número a ser armazenado fica restrito ao tamanho do campo de endereço.

Endereçamento Imediato

OPERANDO = A

51

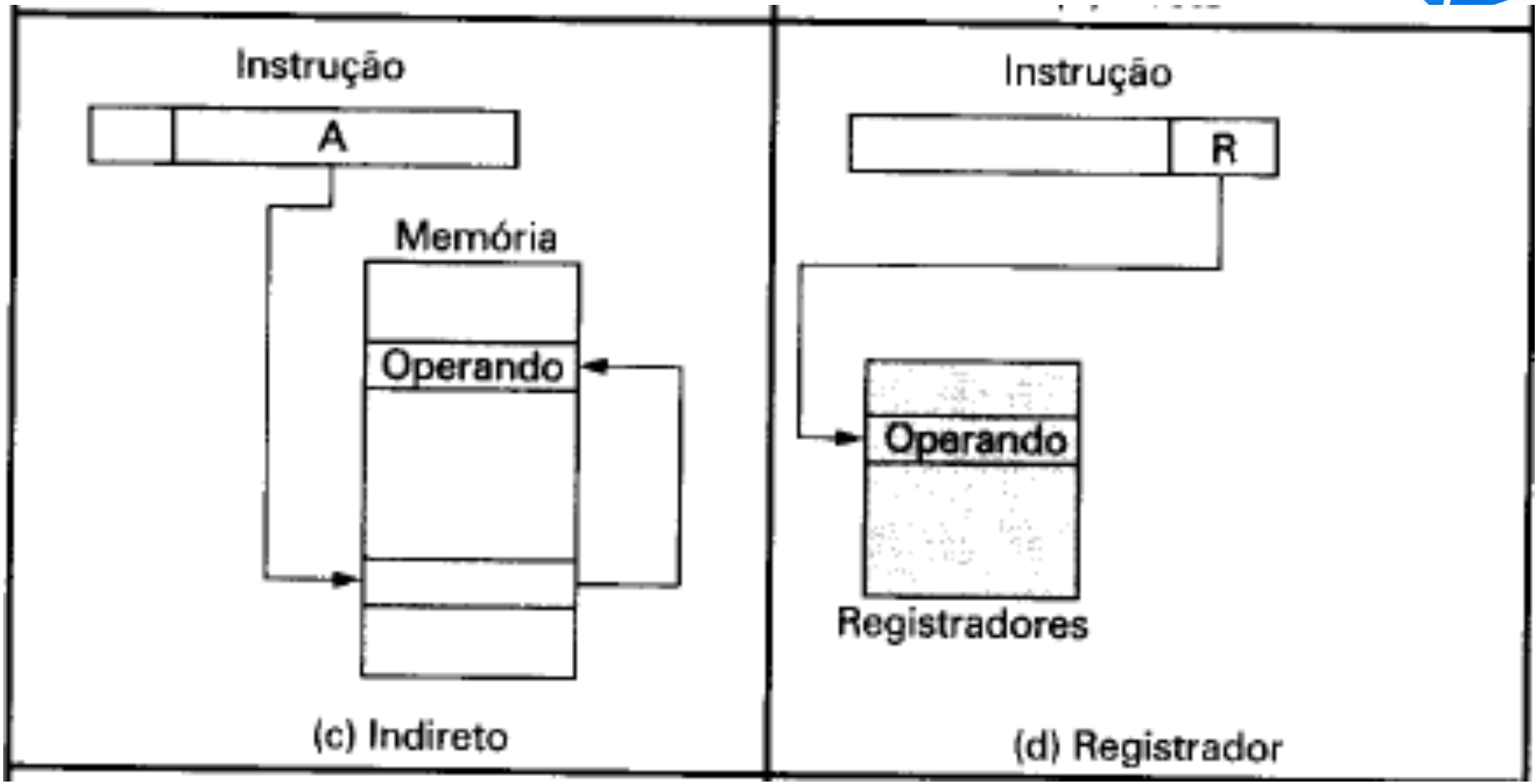
Endereçamento Direto

- Uma forma muito simples de endereçamento também;
- Foi comumente utilizada nos primeiros computadores;
- Necessita apenas de uma referência à memória;
- Limitada pelo espaço de endereçamento.

Endereçamento Direto

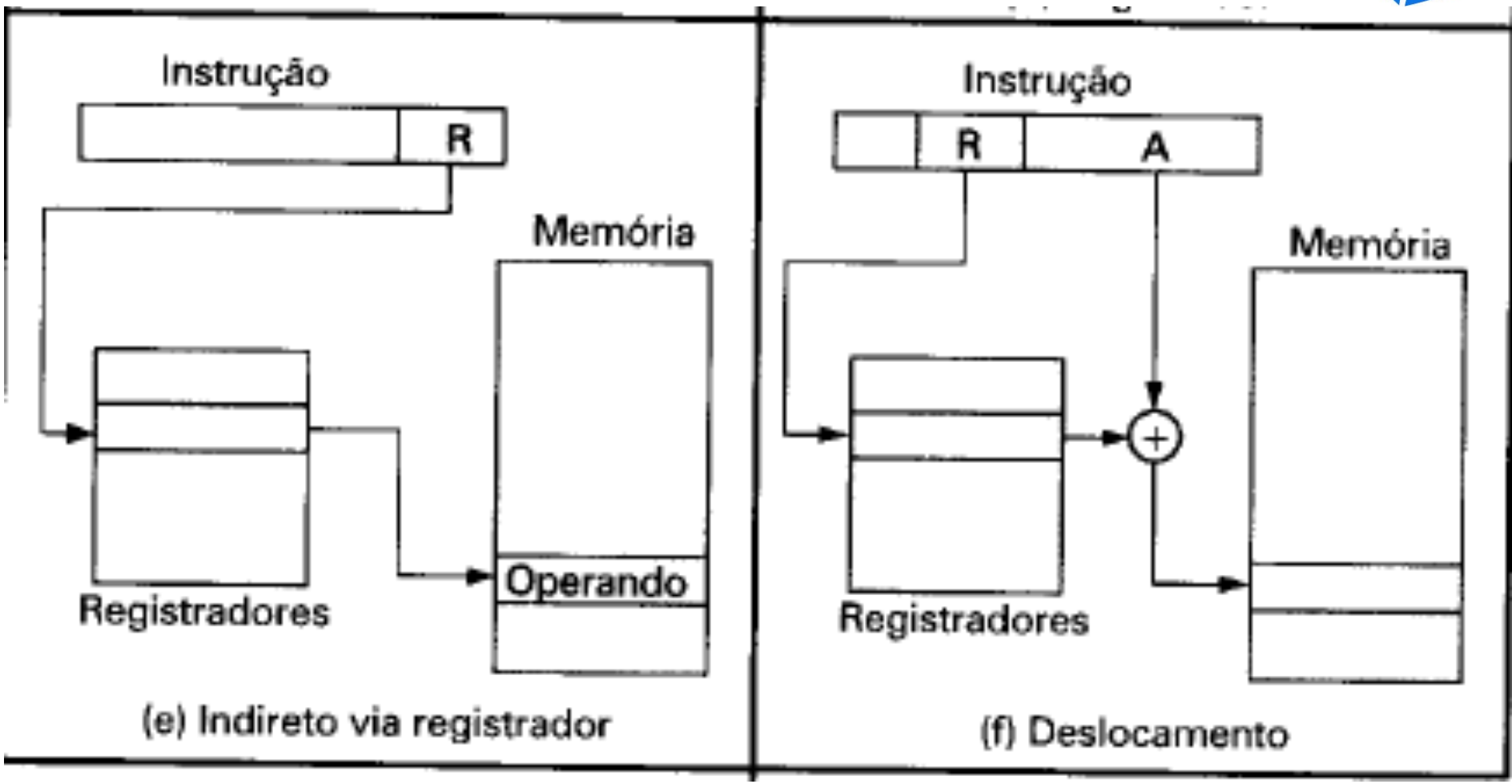
EA = A

Modos de Endereçamento



Fonte: Arquitetura e Organização de Computadores – William Stallings

Modos de Endereçamento



Fonte: Arquitetura e Organização de Computadores – William Stallings

Modos de Endereçamento

Fonte: Arquitetura e Organização de Computadores – William Stallings

Endereçamento Indireto

- Vem para suprir uma necessidade em relação ao endereçamento direto, que é a limitação da faixa de endereços;

- Uma solução é o endereçamento indireto;

- O campo de endereço refere-se a um endereço de uma palavra na memória, a qual contem o endereço completo do operando.

Endereçamento Indireto

Representação:

EA = (A)

Endereçamento Indireto

• **Vantagem**: para uma palavra de tamanho *N*, então, um espaço de endereçamento de 2^N fica disponível;

• **Desvantagem**: a execução da instrução requer duas referências a memória para carregar a instrução. Uma para pegar o endereço e outra para pegar o seu valor.

Endereçamento via Registrador

- Similar ao endereçamento direto;

- A única diferença é que o campo de endereço refere-se a um registrador e não a uma endereço da memória principal

Endereçamento via Registrador

Representação:

EA = R

Endereçamento via Registrador

Vantagens: apenas pequenos campos de endereços são necessários na instrução e não necessita de referência à memória;

Desvantagem: o espaço de endereçamento é bastante limitado.

Endereçamento via Registrador indireto

- É semelhante ao modo de endereçamento indireto;

- A única diferença é que o campo de endereço se refere a um registrador e não a uma posição de memória.

- EA = (R).

Endereçamento por Deslocamento

- Poderoso método de endereçamento;

- Combina as capacidades do endereçamento direto com as do indireto via registrador;

- Representação: EA = A + (R).

Endereçamento por Deslocamento

• Necessita de dois campos para endereçamento;

• O valor contido em um dos campos de endereçamento é usado diretamente (valor = A);

• O outro valor é implicitamente referenciado no OPCODE e se refere ao registrador (R).

Endereçamento por Pilha

- Uma pilha é um arranjo linear de posições;

- O itens são colocados no topo da pilha e retirados daquele topo;

- Sempre existe um ponteiro que indica onde é o topo da pilha.

Endereçamento por Pilha

- O ponteiro para a pilha é mantido em um registrador;

- Assim, referências às posições da pilha na memória, são de fato endereço indiretos do registrador.

Formato das instruções

• Define a disposição dos *bits* na instrução;

• O formato da instrução pode indicar a forma de endereçamento para cada operando;

• Para maioria dos conjuntos de instruções, mais do que um formato de instrução é usado.

Formato das instruções

• O projeto do formato de uma instrução é uma arte complexa;

• Muitos projetistas trabalham para conseguir a melhor forma de organizá-la.

69

Tamanho da instrução

- Primeiro ponto a ser analisado no projeto do formato de instruções;

- Essa decisão afeta diretamente: Tamanho da memória, Organização da memória, estrutura do barramento, complexidade da UCP e a velocidade dessa UCP.

Tamanho da instrução

• Programadores sempre desejam fazer o máximo de funções possíveis com o conjunto de instruções que possuem;

• Assim, quanto maior a quantidade de bits para o endereçamento, mas funções os programadores poderão realizar;

• No entanto, instruções de tamanho muito longo geram desperdícios;

Tamanho da instrução

Uma instrução de 32 *bits*, por exemplo, ocupa duas vezes mais espaço do que uma de 16 *bits*, mas não é provavelmente duas vezes mais útil.

Tamanho da instrução

- A instrução deve ser deve ser do mesmo tamanho da faixa de transferência da memória, ou deve ser múltipla da mesma;

- Analisar a taxa de transferência da memória;

Alocação de *bits*

- Importante saber como alocar os *Bits* de forma eficiente no formato usado;

- Saber realizar o balanceamento entre o número de OPCODES e o poder da capacidade de endereçamento.

Alocação de *bits*

Considerações a serem relacionadas:

- Número de modos de endereçamento;
- Número de operandos;
- Registrador versus memória;
- Número de conjuntos de registradores;
- Faixa de endereçamento;
- Granularidade do endereço.