Organização de Computadores Digitais

Conjunto de Instruções:
Características e Funções

Modos de Endereçamento e Formatos

Características das Operações de Máquina

- As operações da CPU são determinadas pelas instruções que ela própria executa
- Estas operações são referidas como instruções de máquina ou instruções do computador
- A coleção de diferentes instruções que a CPU e' capaz de executar e' conhecida como conjunto de instruções da CPU

Características das Operações de Máquina

Elementos de uma instrução máquina:

- **Código de operação** (OPCODE) – especifica a operação a ser executada através de código binário;
- **Referência de operando fonte** – a operação pode envolver um ou mais operandos que são os inputs da operação;
- **Referência ao resultado do operando** –a operação pode produzir um resultado
- **Referência à própria instrução** -indica a CPU onde fazer a busca da próxima instrução, quando a execução tiver sido completa

Características das Operações de Máquina

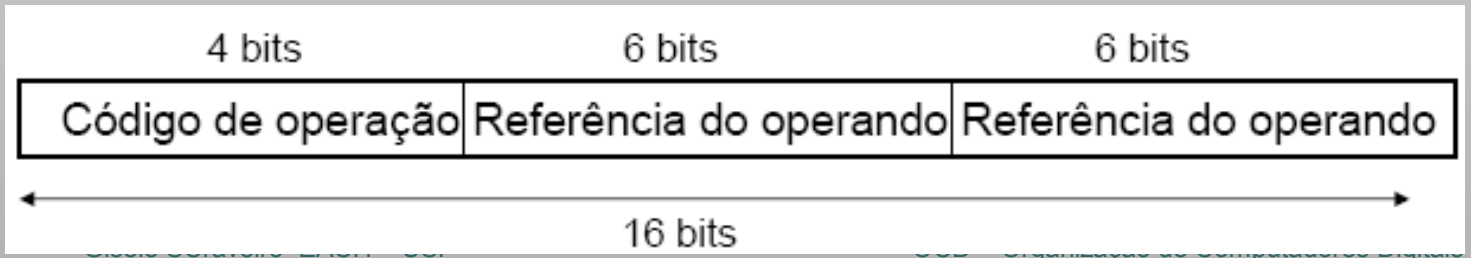
Os dados requeridos pelas instruções, como fonte ou resultado de operandos pode ser lida ou escrita em uma das seguinte áreas:

- Memória principal
- registradores da CPU
- Dispositivos de I/O

Características das Operações de Máquina

Representação das Instruções

- Cada instrução é representada como uma sequência de bits
- A instrução é dividida em pequenos campos, cada um deles é um elemento da instrução
- Durante a sucessão de uma instrução, a instrução é colocada no IR (Instruction Register) da CPU.A CPU encarrega-se da interpretação dos bits



Características das Operações de Máquina

- E' difícil para o programador lidar com representações binárias de instruções máquina
- Por isso, tornou-se prática comum usar uma representação simbólica para instruções de máquina
- Mnemônicas
  - ADD – adição
  - SUB – subtração
  - MUL – multiplicação
  - LOAD – Carregar dados da memória
  - STORE – Armazenar dados da memória
  - Ex ADD R, Y #Addicionar o valor contido na posição Y com o conteúdo do registrador R

Características das Operações de Máquina

- Considere uma instrução de alto nível: X = X + Y
- Suponha que as variáveis X e Y correspondem as posições de memória de endereços 513 e 514
- Se consideraremos um conjunto simples de instruções de máquina, esse comando pode ser implementado com três instruções :
  - Carregar um registrador com o conteúdo da posição 513
  - Adicionar o conteúdo da posição 514 ao registrador
  - Armazenar o conteúdo do registrador na posição de memória 513

Tipos de instruções

- Um computador deve ter um conjunto de instruções que permita ao usuário formular qualquer tarefa de processamento de dados
- Podemos catalogar os tipos de instruções de máquina :
  - Processamento de dados -Instruções aritméticas e lógicas
  - Armazenamento de dados -Instruções de memória
  - Movimento -Instruções de I/O
  - Controle -Teste e instruções de salto

Número de endereços

- Poderíamos dizer que as instruções mais comuns teriam obrigatoriedade de ter 4 endereços de referência:
  - 2 operandos, 1 resultado e o endereço da instrução seguinte
  - Na prática, 4 endereços é uma situação extremamente rara
  - Muitas CPU possuem 1,2, ou 3 endereços na instrução, com o endereço da próxima instrução estando explícito através do PC

Número de Endereços

3

Instruction		Comment
SUB	Y, A, D	Y ← A − B
MPY	T, D, E	T ← D × E
ADD	T, T, C	T ← T + C
DIV	Y, Y, T	Y ← Y ÷ T

(a) Three-Address Instructions

2

Instruction		Comment
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

(b) Two-Address Instructions

1

Instruction		Comment
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

(c) One-Address Instructions

Figure 9.3 Program to Execute Y = (A −B) ÷ (C + D × E).

Número de endereços

Poucos endereços por instrução resultam em instruções mais primitivas:

- Requerem uma **menor complexidade** da CPU
- Instruções de **tamanho menor**
- Por outro lado, programas que contêm um maior número de instruções, em geral resultam num **maior tempo de execução** e **programas mais complexos**

Projeto do conjunto de instruções

- O projeto de um conjunto de instruções é muito complexo, uma vez que ele afeta muitos aspectos do sistema computacional
- Os elementos mais usados no projeto de instruções são:
  - Repertório de operações
  - Tipos de dados
  - Formato das instruções
  - Registradores
  - Modos de endereçamento

Projeto do conjunto de instruções

- Repertório de operações -quantas e quais as operações que são necessárias e como quão complexas elas podem ser
- Tipos de dados – quais os tipos de dados sobre os quais as operações são efetuadas
- Formato das instruções -comprimento das instruções em bits, número de endereços, tamanho dos vários campos
- Registradores -nº de registradores da CPU que podem ser usados e o propósito de cada um
- Modos de endereçamento – de que modo o endereço de um operando pode ser especificado

Tipos de Operandos

- Endereços
  - Diferentes modos de endereçamento
- Números
  - ponto, fixo, ponto flutuante e decimais
- Caracteres
  - ASCII, EBCDIC
- Dados Lógicos
  - bits 1

Tipos de Operações

- Operações de Transferência de Dados
- Operações Aritméticas
- Operações Lógicas
- Operações de Conversões
- Operações de I/O
- Operações de controle do sistema
- Operações de Transferência de Controle

Tipos de Operações

- Operações de Transferência de dados
  - Especifica a localização da fonte e destino
  - O comprimento de dados a transferir tem de ser especificado
  - E' necessário referir o modo de endereçamento para cada operando

Operações de Transferência de dados

O tipo mais fundamental de instrução de máquina é a instrução de transferência de dados

- Deve especificar os endereços dos operandos fonte e de destino da operação. Cada endereço pode indicar uma posição de memória, um registrador ou o topo da pilha.
- Deve indicar o tamanho dos dados a serem transferidos.
- Como em todas as instruções com operandos, deve especificar o modo de endereçamento de cada operando

Operações de Transferência de dados

- As operações de transferência de dados são o tipo mais simples de operação, em termos da ação tomada pela CPU
- Se o operando fonte e de destino são registradores, a CPU simplesmente transfere dados de um registrador para outro; essa é uma operação interna da CPU

Operações de Transferência de dados

- Se um ou ambos os operandos estão na memória, a CPU tem de efetuar algumas ou todas as ações a seguir:

1. Calcule o endereço de memória, com base no modo de endereçamento especificado
2. Se o endereço se refere à memória virtual, traduza esse endereço para um endereço de memória real
3. Determine se o item endereçado está na memória cache
4. Se não estiver, emita um comando para o módulo de memória

Operações Aritméticas

- A maioria das máquinas fornece operações aritméticas básicas para soma, subtração, multiplicação e divisão
- Essas operações são oferecidas para números inteiros com sinal (de ponto fixo)
- Muitas vezes, elas são também oferecidas
- para números na representação decimal empacotada e números de ponto flutuante

Operações Aritméticas

- Outras possíveis operações incluem uma variedade de instruções com um único operando
- Por exemplo:
  - Tomar o valor absoluto do operando
  - Negar o operando
  - Incrementar o operando de 1
  - Decrementar o operando de 1
- A execução de uma instrução aritmética pode envolver transferência de dados, para fornecer os valores dos operandos como entrada para a ULA e para armazenar na memória o valor obtido como saída da ULA

Operações Lógicas

- A maioria das máquinas fornece também uma variedade de operações para manipular bits individuais de uma palavra ou de qualquer unidade endereçável
- Essas operações são baseadas em operações booleanas :
  - A operação NOT (NÃO) inverte um bit
  - As operações AND (E), OR (OU) e XOR (ou-exclusivo) são as funções lógicas mais comuns com dois operandos
  - A operação EQUAL é um teste de igualdade binária, bastante útil

Operações de Conversões

- Instruções de conversão são aquelas que mudam ou operam sobre o formato de dados
- Um exemplo simples é a conversão de um número decimal para binário e um exemplo mais complexo, a instrução Translate (TR) do S/370
- Essa instrução pode ser usada para converte um código de 8 bits para outro (EBCDIC para ASCII) e tem três operandos: TR R1, R2, L
- O operando R2 contém o endereço do início de uma tabela de códigos de 8 bits Os L bytes a partir do byte especificado em R1 são traduzidos, cada byte sendo substituído pelo conteúdo da entrada na tabela indexada por esse byte

Operações de I/O

- As instruções de entrada/saída (E/S) foram discutidas com algum detalhe anteriormente
- Como vimos, existe uma variedade de abordagens, incluindo E/S programada, E/S mapeada na memória, DMA e uso de processadores de E/S
- Muitas implementações fornecem apenas algumas instruções de E/S, com ações específicas determinadas por meio de parâmetros, códigos ou palavras de comando

Operações de Controle do Sistema

- Instruções de controle de sistema são aquelas que apenas podem ser executadas quando o processador está no estado privilegiado ou está executando um programa carregado em uma área especial da memória, que é privilegiada
- Tipicamente, elas são reservadas para uso pelo sistema operacional

Operações de Transferência de Controle

- Para todos os tipos de operação discutidos até agora, a próxima instrução a ser executada é aquela que segue imediatamente, na memória, a instrução corrente
- No entanto, uma fração significativa das instruções de qualquer programa tem como função alterar a sequência de execução de instruções
- Nessas instruções, a CPU atualiza o contador de programa com o endereço de alguma outra instrução armazenada na memória

Operações de Transferência de Controle

- Essas operações são requeridas por diversas razões. Entre as mais importantes estão as seguintes:

1. No uso prático de computadores, é essencial poder **executar um conjunto de instruções mais de uma vez** e talvez milhares de vezes.
2. Quase todos os programas envolvem a tomada de algumas decisões, isto é, o computador deve **executar uma determinada sequência** de operações, se uma determinada condição é satisfeita, e uma outra sequência de operações, se essa condição não se verifica.
3. Implementar corretamente um programa de computador de grande porte é uma tarefa complexa. Por isso, é útil dispor de mecanismos para **dividir o programa em partes** menores, que possam ser programadas separadamente

Operações de Transferência de Controle

A seguir, discutimos as operações de transferência de controle encontradas mais frequentemente num conjunto de instruções:

- As operações de desvio,
- de salto e
- de chamada de procedimento

Instrução de Desvio

- Uma instrução de desvio tem como um de seus operandos oendereço da próxima instrução a ser executada.
- Com frequência, essa instrução é um desvio. condicional, isto é, o desvio será feito (o contador de programa é atualizado com o endereço especificado no operando) apenas se uma dada condição for satisfeita.
- Caso contrário, será executada a próxima instrução da sequência de instruções (o contador de programa é incrementado)

Instrução de Desvio

Por exemplo, uma máquina pode ter vários tipos de desvio condicional:

- BRP X -Desviará para a instrução de endereço X se o resultado for positivo
- BRN X -Desviará para a instrução de endereço X se o resultado for negativo
- BRZ X -Desviará para a instrução de endereço X se o resultado for zero
- BRO X -Desviará para a instrução de endereço X se ocorrer overflow

Instruções de Salto

- Outra forma comum de instrução de transferência de controle é a instrução de salto
- Instruções desse tipo incluem um endereço de desvio implícito
- Tipicamente, um salto indica que a execução de uma instrução da sequência de instruções deve ser omitida; portanto, o endereço da próxima instrução a ser executada é obtido somando o endereço da instrução corrente com o tamanho de uma instrução

Instruções de Salto

- Um exemplo típico é uma instrução para incrementar o valor contidoem um registrador e saltar caso o resultado dessa operação seja igual a zero **(ISZ -increment-and-skip-if-zero)**
- Considere o seguinte fragmento de programa:

301

...

309 **ISZ** R1

310 BR 301

311

Instruções de chamada de procedimento

- O conceito de procedimento foi talvez uma das mais importantes inovações nodesenvolvimento de linguagens de programação
- Um procedimento é um subprograma autocontido, que é incorporado em um programa maior
- Um procedimento pode ser invocado, ou chamado, em qualquer ponto do programa
- Uma chamada a um procedimento instrui o processador a executar todo o procedimento e, então, retornar ao ponto em que ocorreu a chamada

Instruções de chamada de procedimento

- O mecanismo de controle de procedimentos envolve duas instruções básicas:
  - uma instrução de chamada, que desvia a execução da instrução corrente para o início do procedimento,
  - e uma instrução de retorno, que provoca o retorno da execução do procedimento para o endereço em que ocorreu a chamada
- Ambas constituem formas de instrução de desvio

Instruções de chamada de procedimento

- Considere uma instrução CALL X, em linguagem máquina, que representa uma chamada ao procedimento de endereço X
- Se o endereço de retorno for armazenado num registrador, a instrução CALL X causará as seguintes ações:
  - RN<-PC + Delta
  - PC<-X
- onde RN é o registrador usado para armazenar o endereço de retorno da chamada de procedimento, PC é o contador de programa e Delta é o tamanho de uma instrução
- O procedimento chamado pode, então, salvar o conteúdo de RN, para que ele seja usado posteriormente, no retomo do procedimento

Modo de Endereçamento

- Virtualmente todos os computadores possuem mais que um modo de endereçamento
- A questão está em como é que a unidade de controle vai escolher qual o modo de endereçamento que está sendo usado numa instrução em particular
- Muitas aproximações podem ser tomadas, geralmente essa distinção é conseguida pela utilização do opcode
- Do mesmo modo, 1 ou mais bits podem ser usados no formato de instruções para fazer essa distinção

As Técnicas de Endereçamento

- Imediato
- Direto
- Indireto
- Registrador
- Indireto por Registrador
- Deslocamento (Indexado)
- Pilha

A Técnicas de Endereçamento

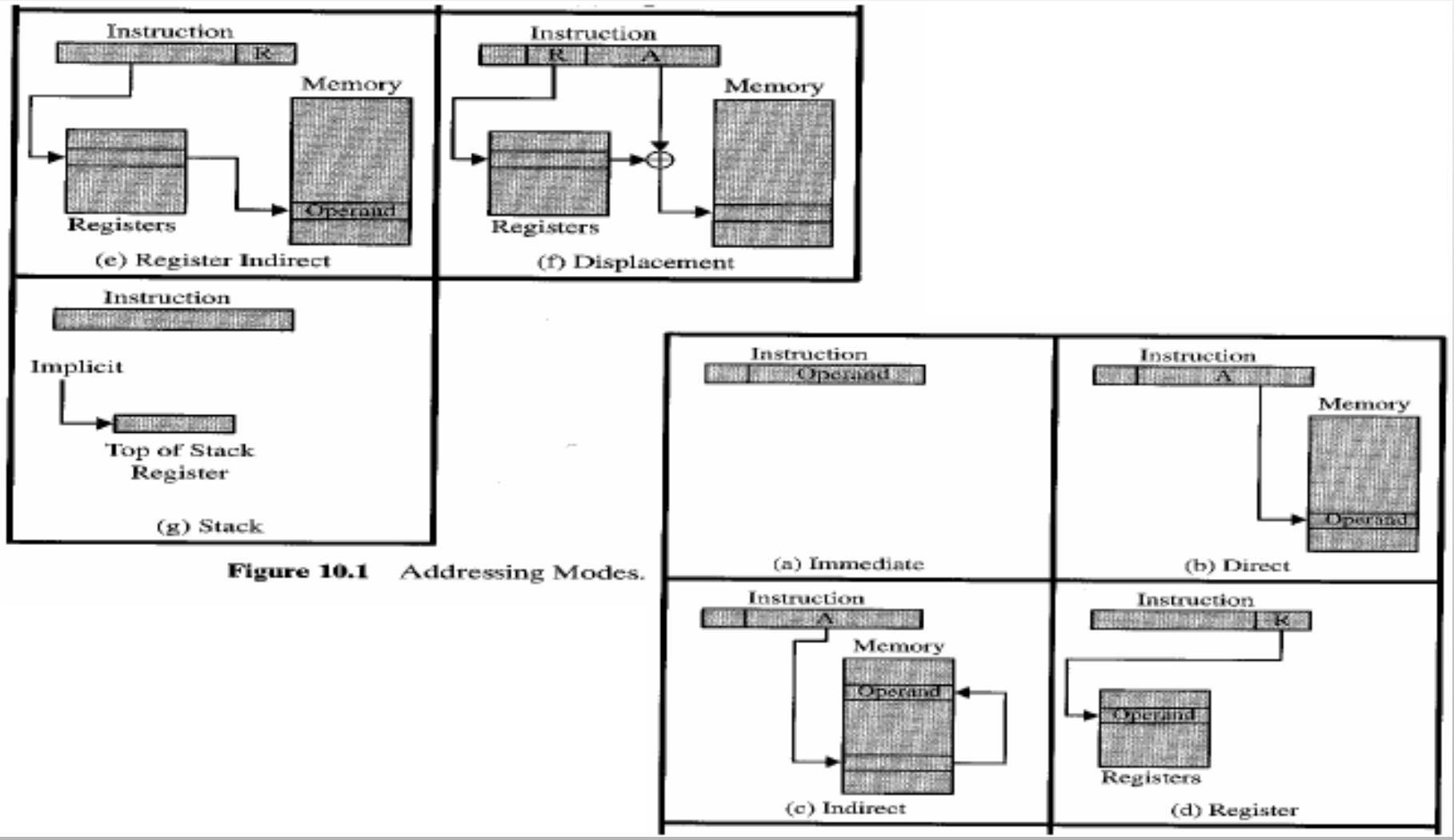


Figure 10.1 Addressing Modes.

As Técnicas de Endereçamento

- A
  - Conteúdo do campo de endereço na instrução
- EA
  - Endereço efetivo (effective address) da localização
- X
  - Conteúdo da localização X

Endereçamento Imediato

- É a forma mais simples de endereçamento em que o operando está presente na instrução (operando –a)
- Este modo de endereçamento pode ser utilizado para definir e utilizar constantes e inicializar variáveis
- Vantagem
  - Não e' necessário adicionar memória para obter os operandos, salvando ciclo de acesso à memória
- Desvantagem
  - O número referindo o operando, esta' restrito ao tamanho do campo de endereço, o que em muitos casos é inferior comparado com o comprimento da palavra

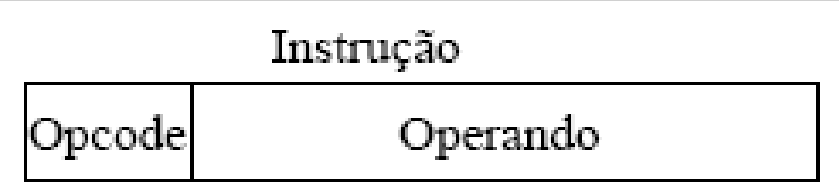
Endereçamento Imediato

- O operando faz parte da instrução
- Operando = campo de endereço
- exemplo ADD 5
  - Adiciona 5 ao conteúdo do acumulador
  - 5 e' o operando
- Não existe referencia a memória para identificar dados
- Rápido
- Leque limitado

Endereçamento Imediato

Exemplo ADD 5

- Adiciona 5 ao conteúdo do acumulador
- 5 e' o operando



Endereçamento Direto

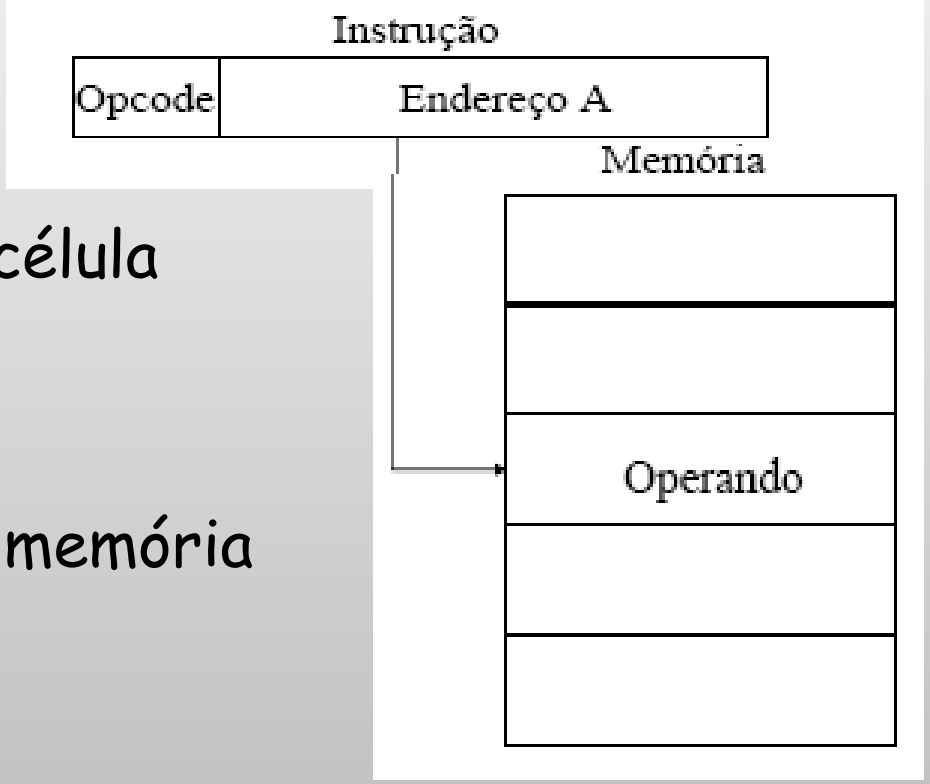
- O campo de endereçamento contém o endereço efetivo do operando

$$EA = A$$

- Esta técnica, muito utilizada nas primeiras gerações de computadores
- Implica um acesso à memória
- Limita o espaço de endereçamento

Endereçamento Direto

Exemplo, ADD A



- Adiciona o conteúdo da célula

  Aao acumulador

- Procura no endereço de memória

  A pelo operando

Endereçamento Indireto

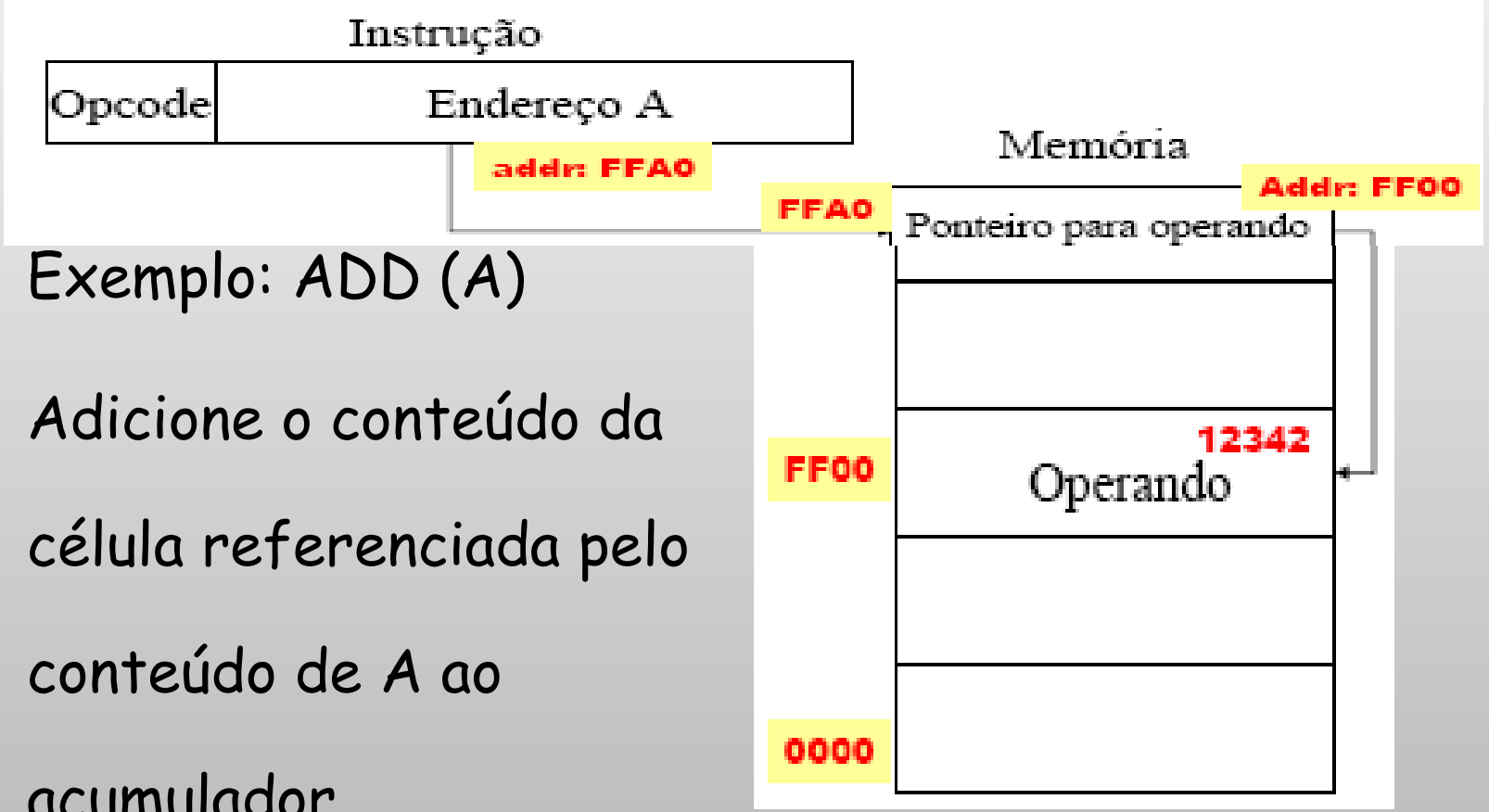
- Esta técnica procura solucionar o problema do espaço de endereçamento da técnica anterior colocando no campo do endereço da instrução uma referência à memória que contém o endereço completo da operação

$$EA = (A)$$

Endereçamento Indireto

- Vantagem
  - Para um campo da palavra n está disponível um espaço de endereçamento 2^N
- Desvantagem
  - São necessários 2 acessos para buscar o operando, um para o endereço, outro para o valor do operando

Endereçamento Indireto

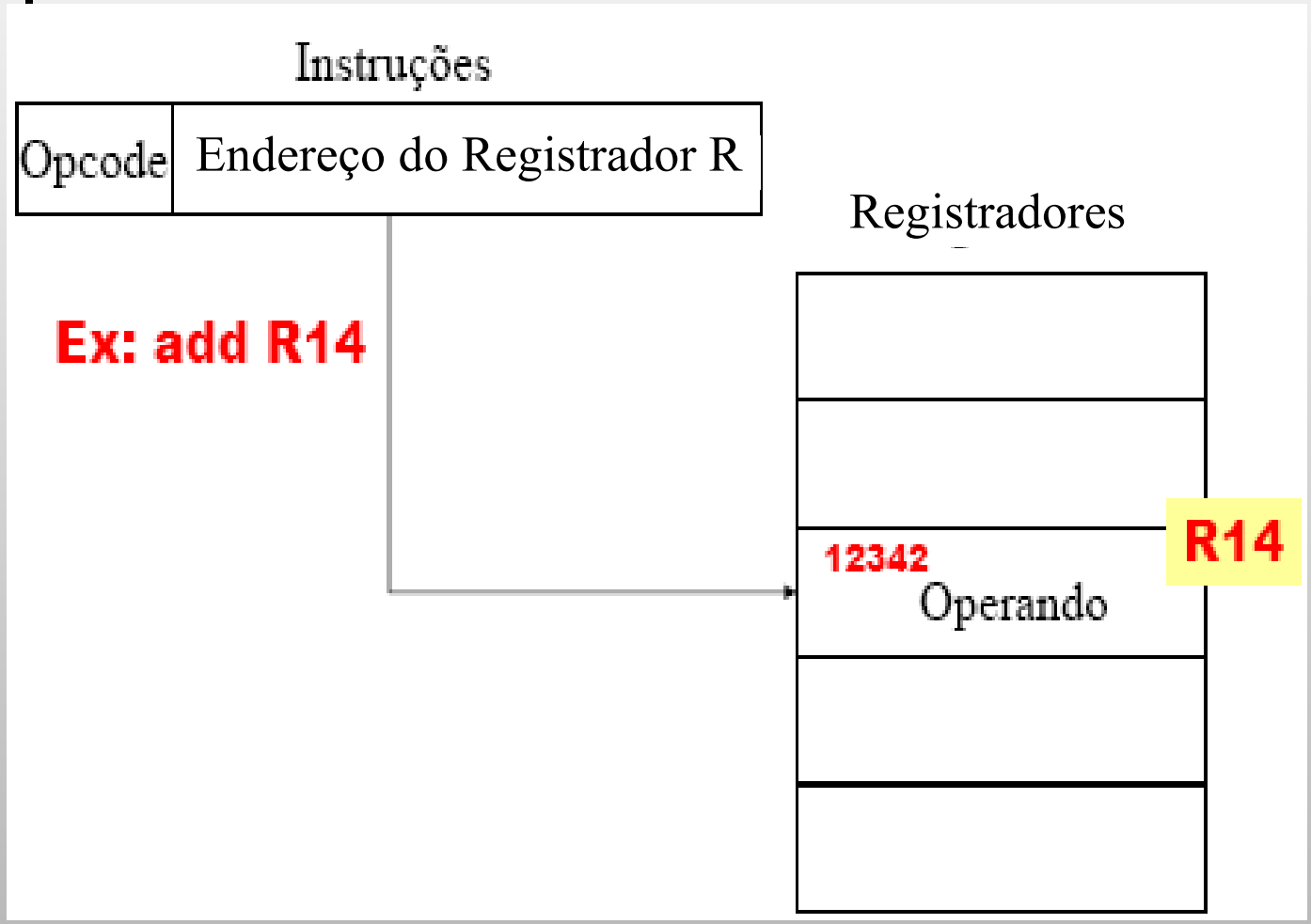


- Exemplo: ADD (A)
- Adicione o conteúdo da célula referenciada pelo conteúdo de A ao acumulador

Endereçamento por Registrador

- Esta técnica é semelhante ao endereçamento direto, a única diferença é que o campo refere um registrador em vez de uma posição de memória
- Tipicamente o campo de endereço que referênciaregistradores tem 3 a 4 bits para referenciar um conjunto de 8a 16 dígitos
- Vantagens
  - Apenas um pequeno campo de endereço é necessário e não são necessárias referências à memória
  - Execução muito rápida
- Desvantagens
  - Nu'mero limitado de registradores

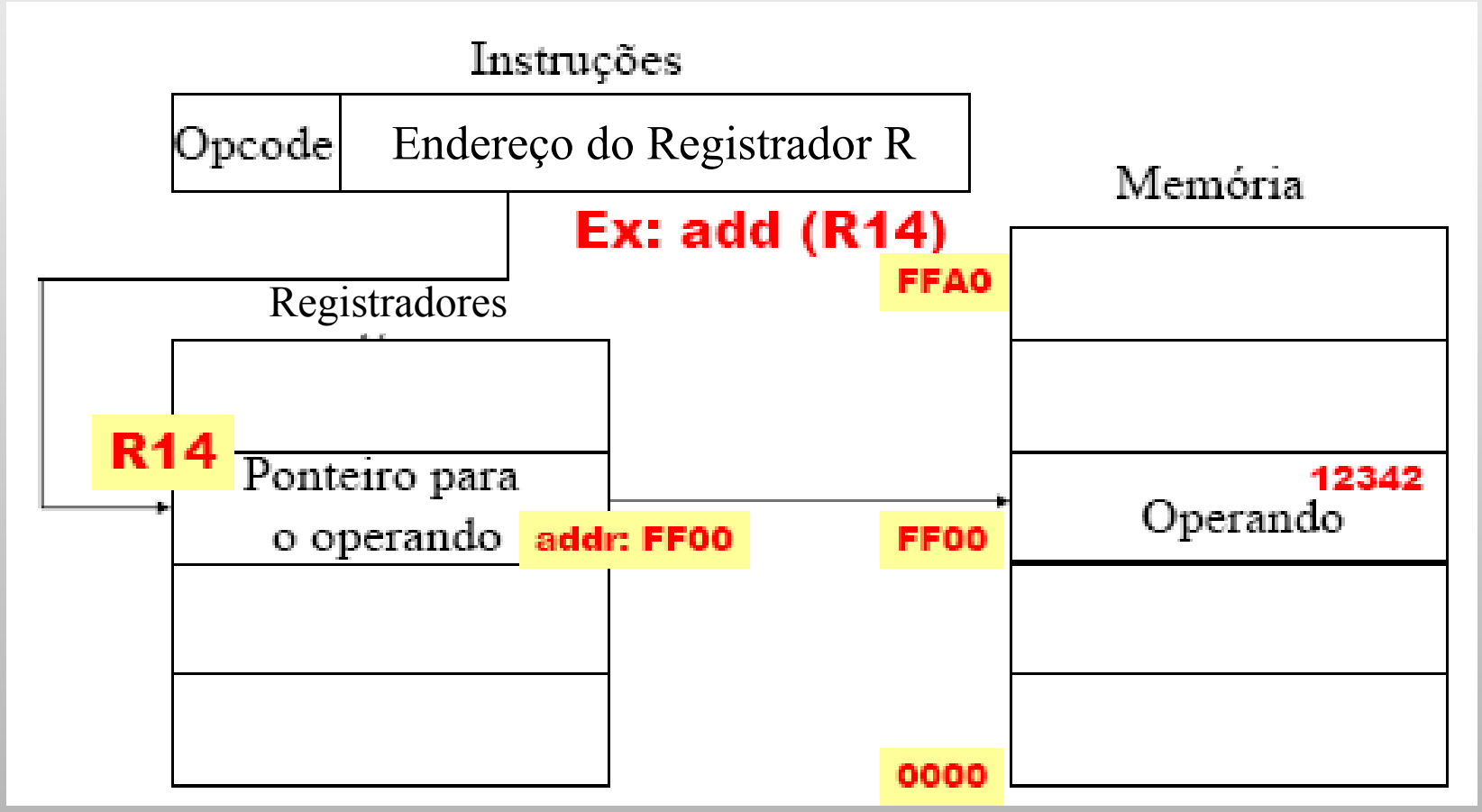
Endereçamento por registrador



Endereçamento Indireto por registrador

- Esta técnica é parecida ao endereçamento indireto variando apenas na referência do campo de endereço ser a de um registrador e não uma posição da memória
- $EA = R$
- As vantagens e limitações são basicamente as mesmas que o endereçamento indireto
- Uma outra vantagem deste tipo de endereçamento, reside no fato deste usar uma menos uma referência à memória que o endereçamento indireto

Endereçamento Indireto por Registrador



Endereçamento por Deslocamento

- Esta técnica muito poderosa combina as técnicas de endereçamento direto e do endereçamento indireto por registrador

$$EA = A+(R)$$

- Esta técnica obriga a que a instrução tenha dois campos de endereço
  - Sendo pelo menos um explícito (valor A usado diretamente)
  - O outro campo de endereçamento, referencia implicitamente um registrador cujo conteúdo é adicionado a A para produzir o endereçamento deslocado

Endereçamento por Deslocamento

1-Endereçamento Relativo

- Usa-se o PC (Program Counter)
- O campo de endereço da instrução corrente é um deslocamento em relação ao PC
- Esta técnica explora o princípio da localidade permitindo poupar bits de endereço na instrução

2-Endereçamento via Registrador Base

- O registrador de referência contém a posição da memória e o endereço da instrução contém o deslocamento a partir duma posição de memória

Endereçamento por Deslocamento

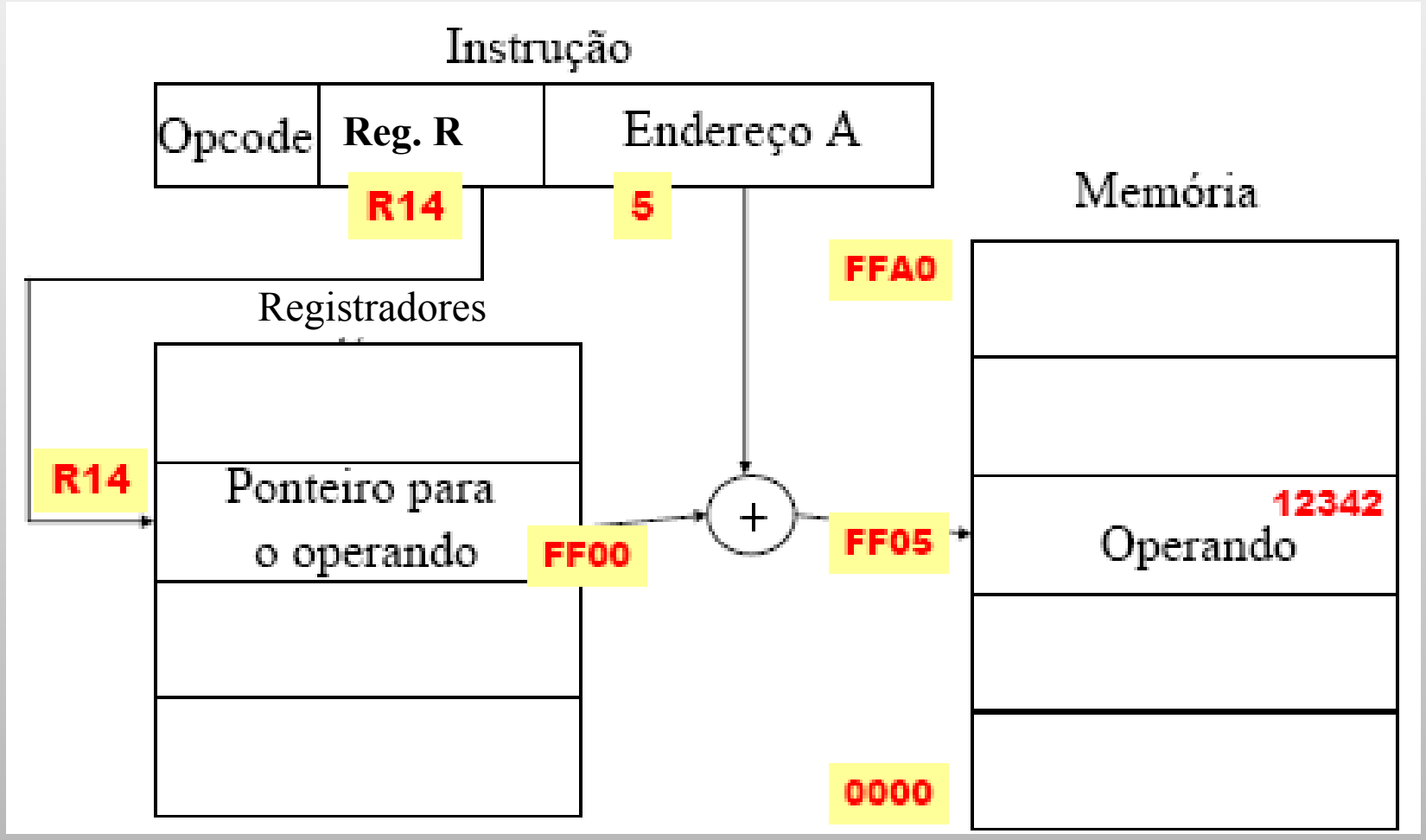
3-Endereçamento Indexado

- Neste caso o campo do endereço referencia uma memória principal e o registrador de referência contém um deslocamento positivo a partir dessa posição de memória
- Esta aproximação é exatamente contrária ao endereçamento baseado em registrador e tem grande utilidade na execução de instruções iterativas através de alterações sucessivas do registrador de referência

$$EA = A + (R) \qquad R = R + 1$$

- Por exemplo suponha-se que se quer adicionar um elemento aos elementos de uma lista O melhor seria começar com a base e somar um elemento a base A A+1 A+2 A+3,...

Endereçamento por Deslocamento



Endereçamento por Pilha

- A stack é um conjunto reservado de localizações de memória
- A pilha é gerida em filosofia LIFO (Last In First Out)
- Associado à pilha existe um apontador que é o endereço do topo da pilha
- O apontador para o topo da pilha é mantido num registrador especial (SP – Stack Pointer) que determina que o endereçamento por pilha é de fato endereçamento indireto por registrador

Endereçamento por Pilha

- O operando esta (implicitamente) no topo da pilha
- ex ADD
- remove dois elementos da pilha e adiciona-os e coloca o resulta na pilha

Formato das Instruções

- O formato das instruções define a forma como os campos são distribuídos
- Cada instrução deve ter um código de operação (OPCODE), e explícita ou implicitamente um ou mais operandos
- Cada operando explícito tem de ser referenciado utilizando um dos métodos descritos anteriormente (endereçamento)

Tamanho das Instruções

- É o fator mais básico do desenho do conjunto de instruções e esta decisão afeta e é afetada por:
  - Tamanho da memória
  - Organização da memória
  - Estrutura do bus
  - Velocidade do CPU
- O compromisso mais obvio é entre o desejo de ter um repertório de instruções mais poderoso, ou seja, mais OPCODES, mais operandos, mais modos de endereçamento, e a necessidade de poupar espaço

Tamanho das Instruções

Para além deste compromisso existem outras considerações:

- O tamanho das instruções deve ser igual ou múltiplo do bus do sistema, caso contrário as instruções não ficariam completas;
- O tamanho das instruções deve também ser múltiplo do tamanho dos caracteres (8 bits) e dos números da vírgula fixa para não haver desperdícios de bits nos cálculos a efetuar

Alocação de Bits

Os seguintes fatores determinam a utilização dos bits de endereçamento:

- Nº de modos de endereçamento
  - Os modos de endereçamento indicados explicitamente ocupam mais bits que os indicados implicitamente
- Nº de operandos
  - Menos endereços podem levar os programas mais longos e complexos
- Registrador vs Memória
  - Quantos mais registradores poderem ser utilizados para referenciar operandos menos bits são necessários

Alocação de Bits

- Nº de conjuntos de registradores
  - Atualmente a tendência tem sido para dividir os registradores em bancos especializados (Pe dados e deslocamentos), usando os opcode(s) da instrução para determinar sobre que banco de registradores deve a operação ser realizada
- Alcance do endereçamento
  - Está diretamente relacionado com o nº de bits para endereçamento na instrução
  - Quando o acesso é feito à memória principal