Linguagem de Montagem

Modos de Endereçamento
Aula 08

Edmar André Bellorini

Ano letivo 2016

Instruções

- É uma determinada operação executada pelo processador
  - Transferência de dados

    ```
    MOV destino, fonte
    ```
  - Aritmética

    ```
    ADD destino, fonte
    ```
  - Lógica

    ```
    AND destino, fonte
    ```
  - Controle

    ```
    JMP local
    ```
  - Entre outras

Instruções

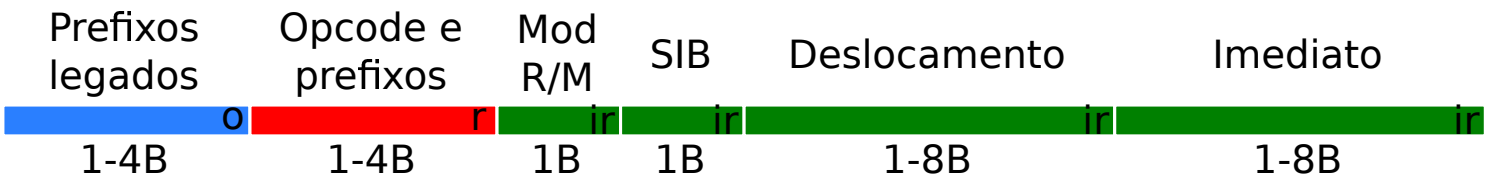
- Elementos de uma Instrução
  - Código da operação (*opcode*)

    Especifica a operação a ser executada
  - Referência à operando A (destino)

    Operando de retorno

    Também pode ser um operando fonte
  - Referência à operando B (fonte)

    Operando de entrada
- Instruções x86_x64 são complicadas
  - Pode conter até 15 bytes de extensão



Exemplos a08e01a/b.asm

- Código da Operação (*Opcode*)

a08e01a.asm:

```
1 section .text
2     global _start
3 _start:
4     mov eax, ebx
5 fim:
6     mov eax, 1
7     mov ebx, 0
8     int 0x80
```

a08e01b.asm:

```
1 section .text
2     global _start
3 _start:
4     add eax, ebx
5 fim:
6     mov eax, 1
7     mov ebx, 0
8     int 0x80
```

Exemplos a08e01a/b.asm

- Diferença entre os arquivos **.asm**:
  - Comando no terminal:

    ```
    $ diff a08e01a.asm a08e01b.asm
    ```
  - Saída:

    ```
    4c4
    <     mov eax, ebx
    ---
    >     add eax, ebx
    ```
- Diferença entre os arquivos montados/ligados **.x**:
  - Comando no terminal:

    ```
    cmp -l a08e01a.x a08e01b.x
    ```
  - Saída:

    ```
    129 211   1
    720 141 142
    d.  a.x b.x ; significado das colunas
    ```

Exemplos a08e01.a/b.asm

- A saída do comando *cmp* não é tão intuitiva, pois usa base decimal, por este motivo, alteramos a linha de comando anterior para:

```
cmp -l a08e01a.x a08e01b.x |
 gawk '{printf "%08X %02X %02X\n",
 $1-1, strtonum(0$2), strtonum(0$3)}'
```

- Saída:

```
00000080   89   01
000002CF   61   62
   desl.  a.x  b.x ; significado das colunas
```

  - Para facilitar a vida dos alunos, existe o script *cmpASM.sh*, em anexo, que contém linha de comando.
    Se não tiver permissão para executar, use:
    `chmod +x cmpASM.sh`

Exemplos a08e01.a/b.asm

- 00000080 89 01
  - Indica que no deslocamento 0x80, o arquivo a08e01a.x contém o valor 0x89, enquanto que o arquivo a08e01b.x contém o valor 0x01
    0x89 é o parte do código da instrução MOV
- 000002CF 61 62
  - Indica que no deslocamento 0x2CF, o arquivo a08e01a.x contém o valor 0x61, enquanto que o arquivo a08e01b.x contém o valor 0x62
    0x61 e x062 representam os caracteres 'a' e 'b' na tabela ASCII
- Editor Hexadecimal
  - É altamente recomendado procurar as diferenças entre os arquivos **.x** manualmente através do uso de um editor hexadecimal
    Editor Hexadecimal Online: HexEd.it

Instruções

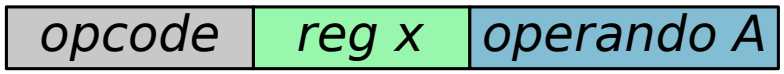
- Referência à Operandos (MOD R/M)
  - Determina se os operandos da instrução são registradores (r), memória (m) ou imediato (i)
    Obs.: Definição extremamente simplificada
- As instruções, em sua maioria, utilizam 1 ou 2 operandos, que devem ser buscados em algum local de armazenamento ($r/m/i$), para então serem processados.
  - O local de armazenamento do operando é referenciado através de **Endereçamento**
  - Existem diversos **Modos de Endereçamento**
    - A adoção dos Modos de Endereçamento por uma arquitetura é decisão de projeto de instruções

Modos de Endereçamento

- Imediato
- Direto
- Indireto
- por Registrador
- Indireto por Registrador
- por Deslocamento
- por Pilha (aula 09)

Endereçamento Imediato

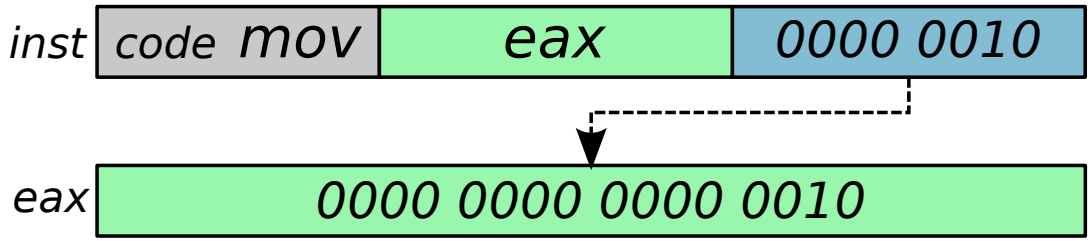
- Operando faz parte da instrução



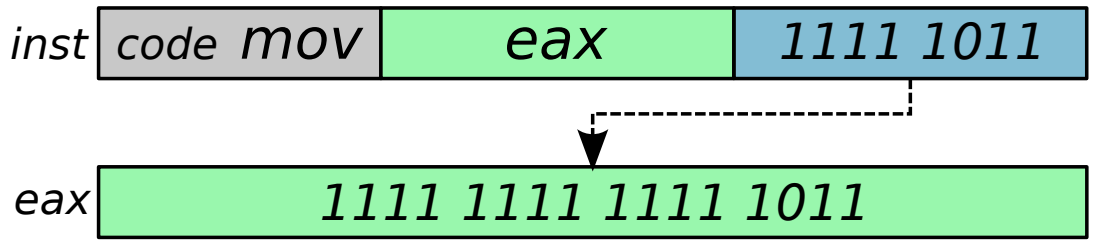
  - Operando pronto no ciclo de busca à instrução
  - Usado na especificação de constantes
  - Forma mais simples de endereçamento
  - Não contém referência à memória
    - Operando limitado ao campo da instrução

Endereçamento Imediato - Exemplo

```
1 mov eax, 0x10
```

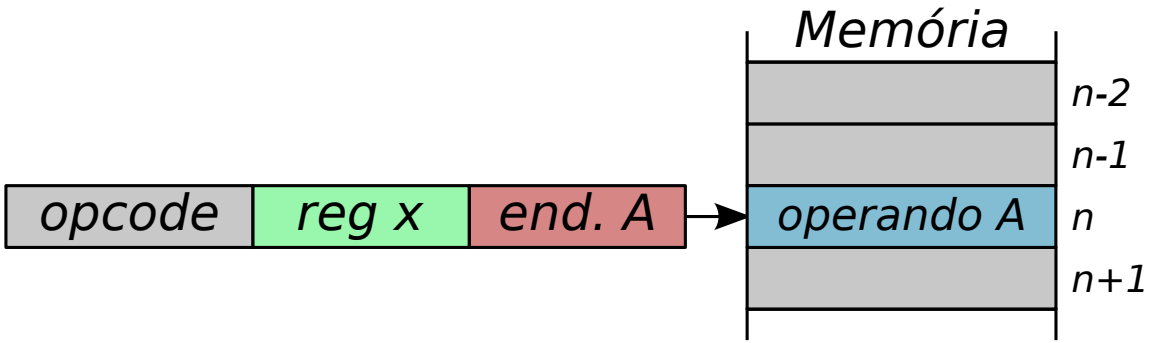



```
1 mov eax, -0x05
```



Endereçamento Direto

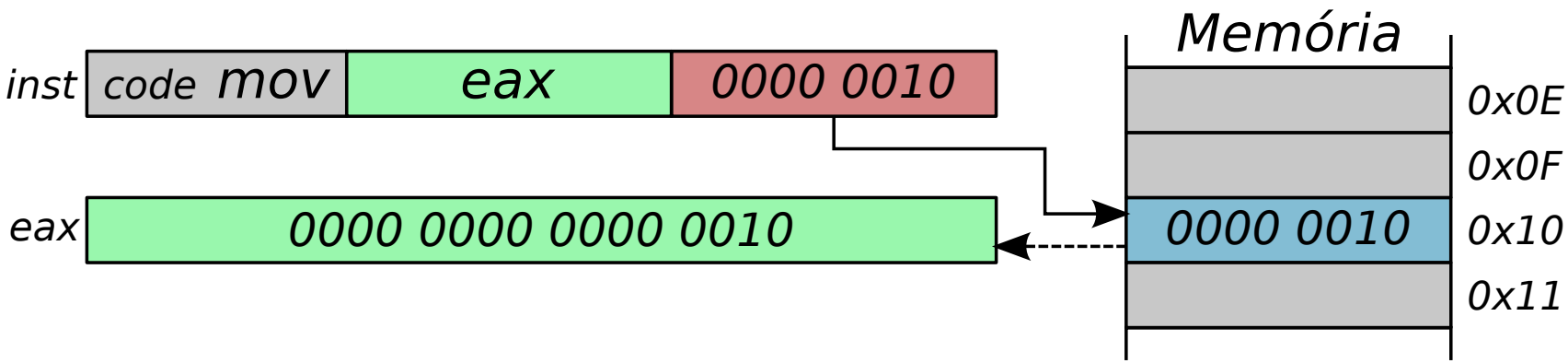
- Operando está na posição de memória indicado no campo da instrução



  - Contém 1 referência à memória
    - O intervalo referenciado é limitado pelo campo da instrução

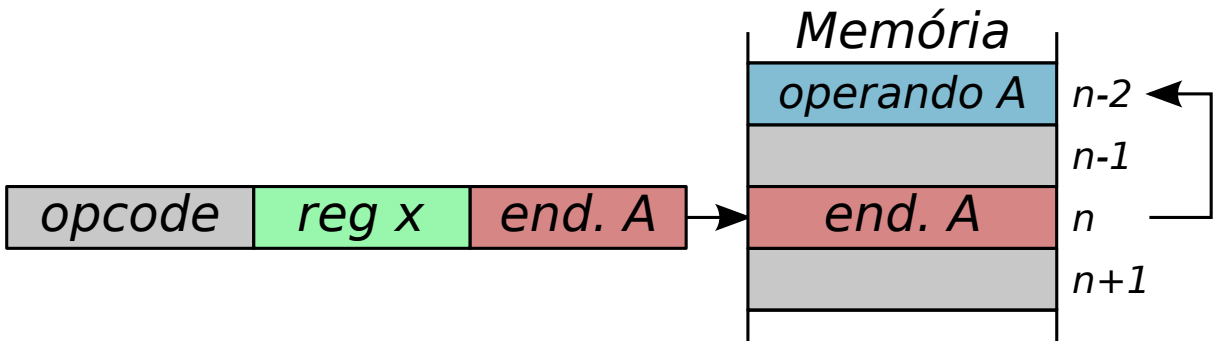
Endereçamento Direto - Exemplo

```
1  mov eax, [0x10]
```



Endereçamento Indireto

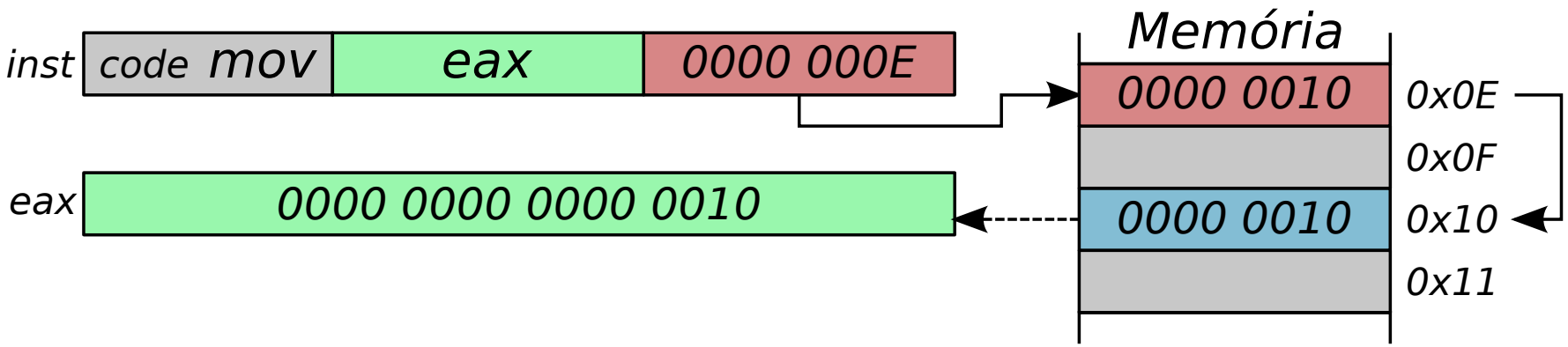
- Operando está na posição de memória indicado na posição de memória indicado no campo da instrução



  - Contém *n* referências à memória
    - O intervalo referenciado é limitado pelo campo da instrução na primeira indireção
    - O intervalo referenciado é limitado pela palavra de memória a partir da segunda indireção
  - **Não é suportado** pelo NASM

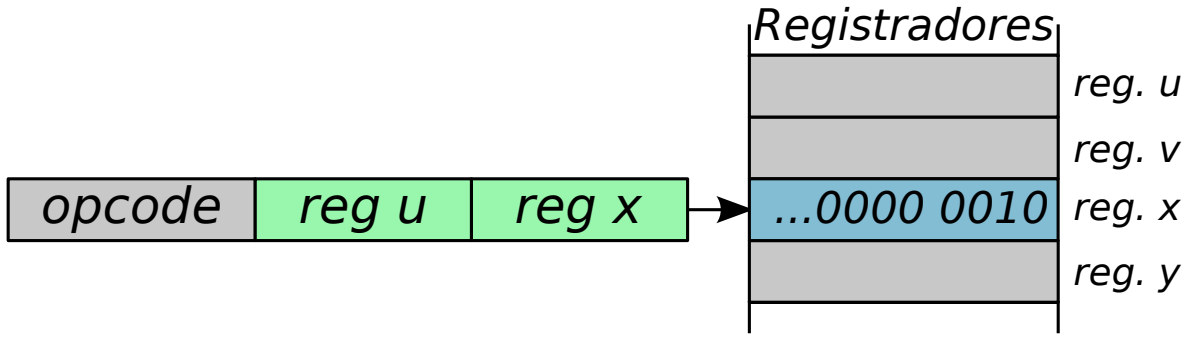
Endereçamento Indireto - Exemplo

```
1  mov eax, [[0x10]]
```



Endereçamento por Registrador

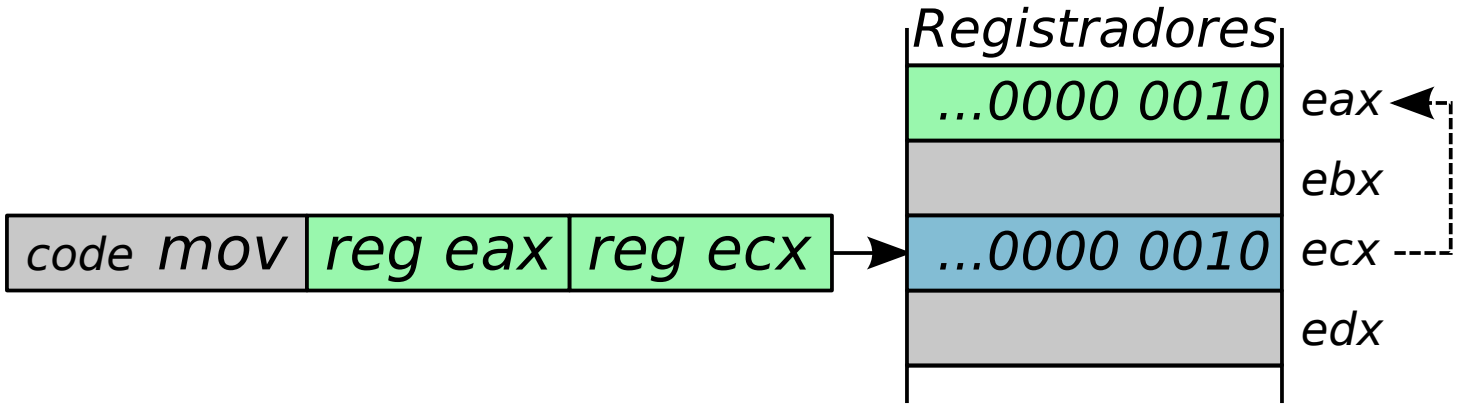
- Operando está no registrador indicado no campo da instrução



  - Não contém referência à memória
    - Operando deve ter sido buscado/calculado previamente
    - Espaço de endereçamento é limitado ao número de registradores

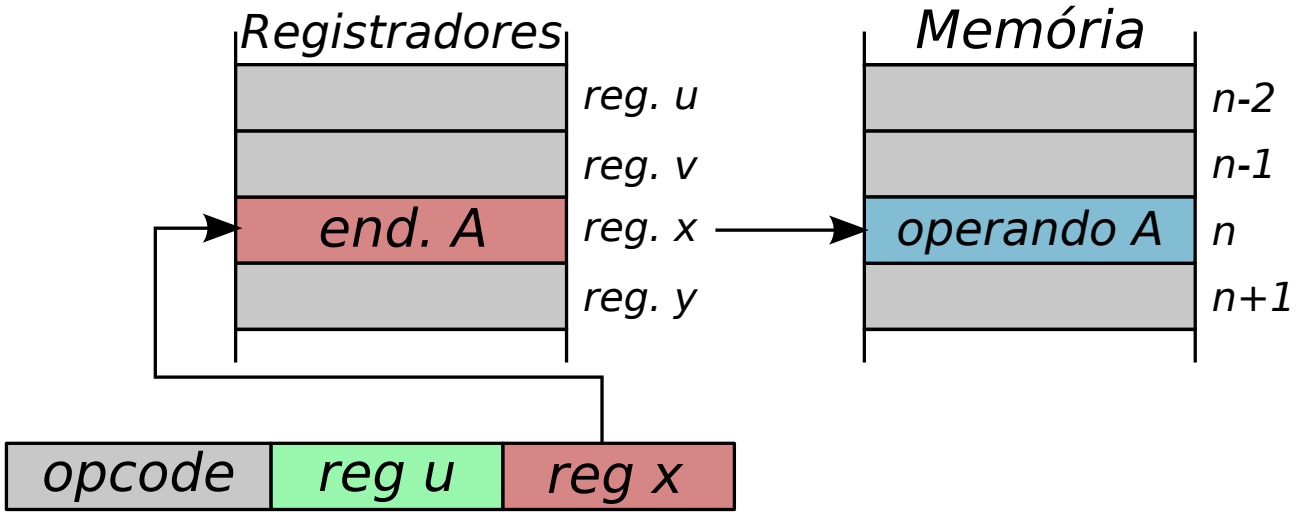
Endereçamento por Registrador - Exemplo

```
1  mov eax, ecx
```



Endereçamento Indireto por Registrador

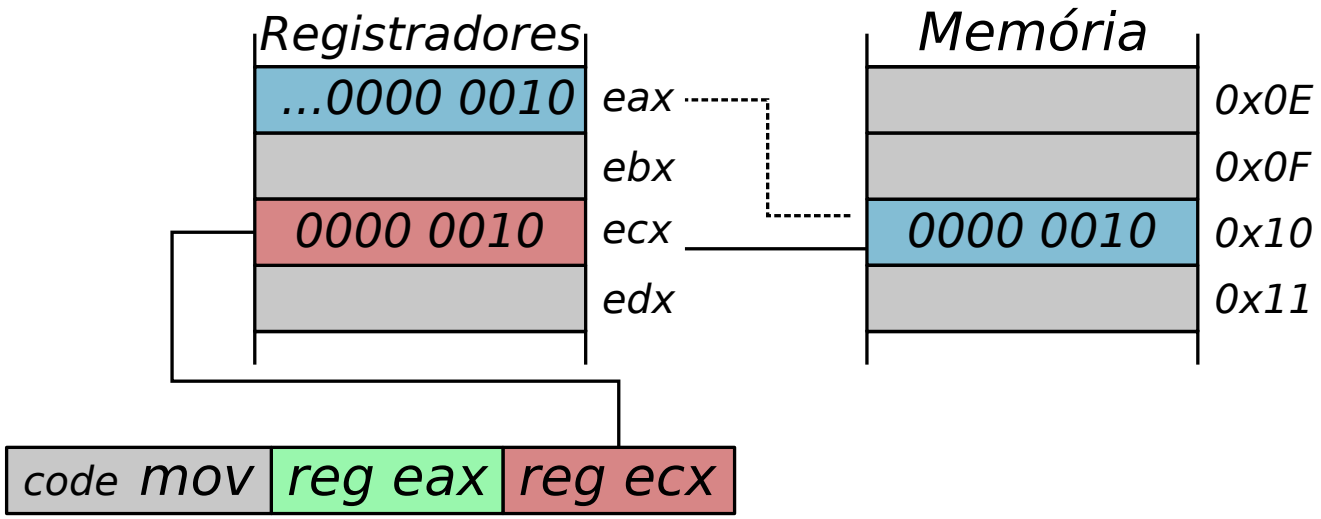
- Operando está na posição de memória indicado no registrador indicado no campo da instrução



  - Contém 1 referência à memória
    - Endereço deve ter sido buscado/calculado previamente

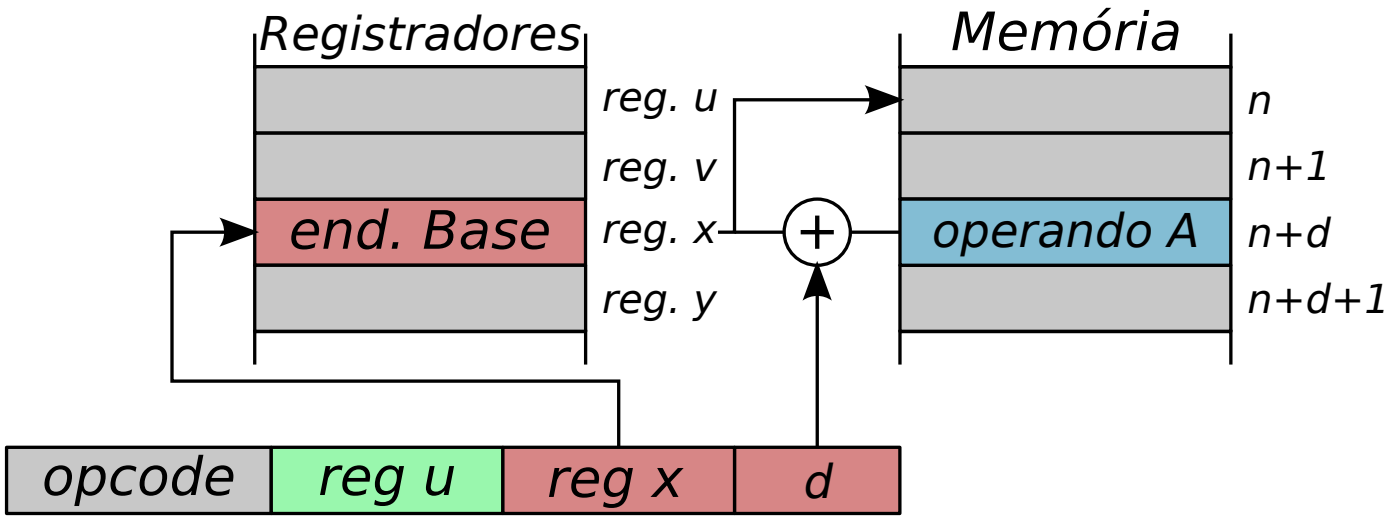
Endereçamento Indireto por Registrador - Exemplo

```
1  mov eax, [ecx]
```



Endereçamento por Deslocamento

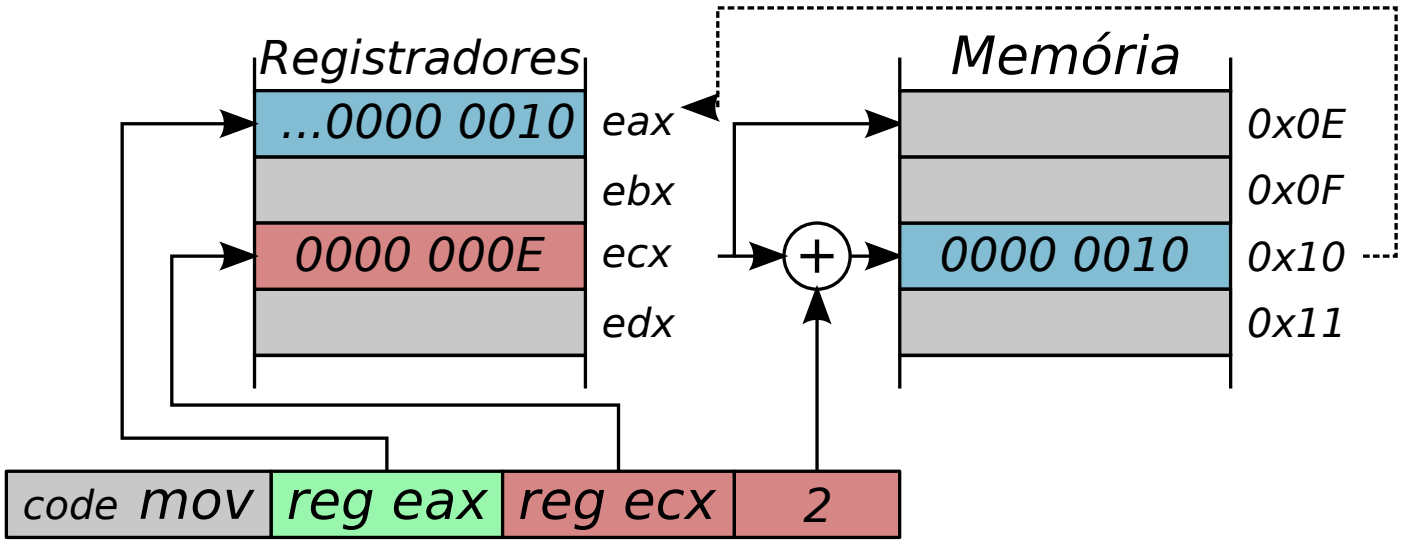
- Operando está na posição de memória resultante do cálculo entre o registrador e um deslocamento definidos nos campos da instrução



  - Contém 1 referência à memória
  - Requer cálculo do endereço do operando

Endereçamento por Deslocamento - Exemplo

```
1  mov eax, [ecx+2]
```



Exemplo a08e02.asm

```
18      ...
19  laco:
20      mov al , [r8]       ; end. indireto por registrador
21
22      mov bl , [v1+r15d]     ; end. por deslocamento
23              ; offset + deslocamento (R)
24
25      ;mov ecx, [r9+r15d*4] ; end. eh invalido
26
27      mov edx, [v2+r15d*4]   ; end. por desloc. indexado
28              ; offset + desloc. * posicionamento
29      ...
```

Exercícios de Fixação

- EF0801: Utilize os exemplos a08e01a/b.asm para descobrir o código das seguintes instruções:
  - SUB r/r
  - OR r/r
  - AND r/r
  - XOR r/r

Exercícios de Fixação

- EF0802: Inversão de vetor de caracteres.
  - Dado um vetor de 10 posições de caracteres, inverta a ordem dos seus elementos
  - O vetor de caracteres deve ser lido do teclado
    - Considere que o usuário é esperto o suficiente para sempre entrar com 10 caracteres + < *enter* >
    - O < *enter* > não deve ter a posição alterada
    - ambos os vetores devem estar alocados ao mesmo tempo (use 2 variáveis não inicializadas)
  - Mostre a sequência de caracteres com a inversão
    - Exemplo de entrada/saída:

```
$: ./EF0802.x
entre com o vetor: abcdefghij<enter>
jihgfedcba eh inversao de abcdefghij
$:
```

Exercícios de Fixação

- EF0803: Ordenação de vetor de inteiros
  - Dado um vetor de 10 posições de inteiros (4 bytes cada), ordene seus elementos em ordem crescente
  - O vetor de inteiros deve ser criado como uma variável inicializada
    - Utilize 10 valores fora de ordem (inclusive iguais)
    - Pode ser utilizado qualquer algoritmo de ordenação
    - A ordenação deve acontecer no mesmo vetor (e não em um vetor auxiliar)
    - O programa não tem iteração com o usuário, assim deve ser *debuggado* para confirmar a corretude.

Relatório

- Somente relatório
  - O modelo de relatório para a disciplina de LM está disponível em anexo da Aula 01
    - Arquivo **modeloRelatorioLM.odt**
    - A data do relatório é a data de entrega (ver moodle)
  - Somente serão aceitos os relatórios em formato .pdf com nome do arquivo seguindo o padrão:
    TY.PXX.nome.sobrenome.pdf
    - TY é o número da turma prática (1, 2, 3 ou 4)
    - PXX é o número da prática, neste caso: P08
  - Ex.: aluno Warren Robinett da turma prática 9 (de 1979):
    - T9.P08.Warren.Robinett.pdf