

Software Básico

Endereçamento de Memória

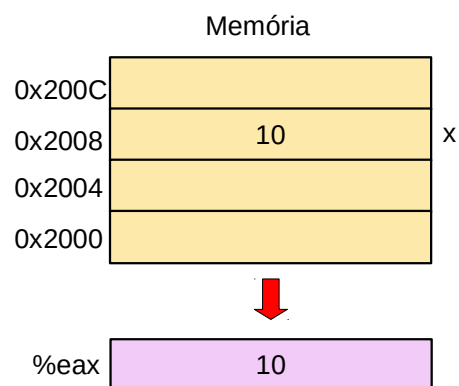


Memória: Modo Direto

- O endereço de memória é especificado por uma constante ou rótulo (*label*)

```
.data
    x:    .int    10

.text
main:
    movl  x, %eax
```

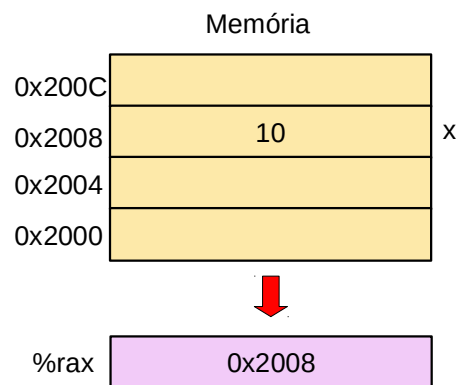


Memória: Modo Direto

- **Atenção:** se usarmos \$x, colocamos o endereço da variável no registrador
- Endereços são 64-bits

```
.data
    x:    .int    10

.text
main:
    movq  $x, %rax
```

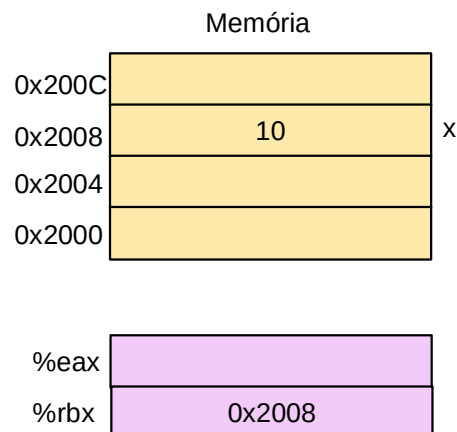


Memória: Modo Indireto

- O endereço de memória está em um registrador
- Formato: (R_{Base})

```
.data
    x:    .int    10

.text
main:
    movl  $1, (%rbx)
    movl  (%rbx), %eax
```

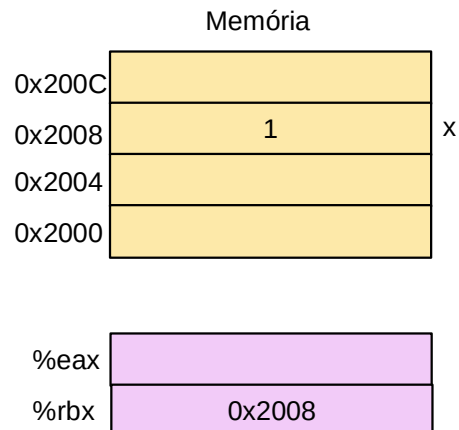


Memória: Modo Indireto

- O endereço de memória está em um registrador
- Formato: (R_{Base})

```
.data
    x:    .int    10

.text
main:
    movl    $1, (%rbx)
    movl    (%rbx), %eax
```

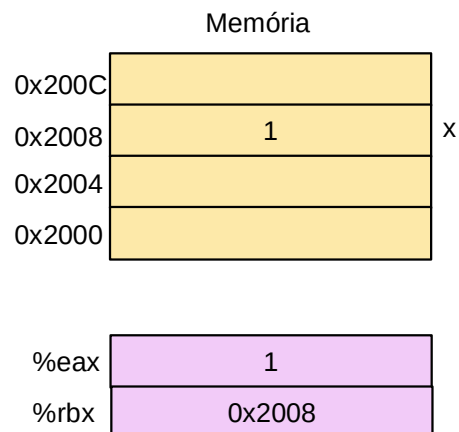


Memória: Modo Indireto

- O endereço de memória está em um registrador
- Formato: (R_{Base})

```
.data
    x:    .int    10

.text
main:
    movl    $1, (%rbx)
    movl    (%rbx), %eax
```

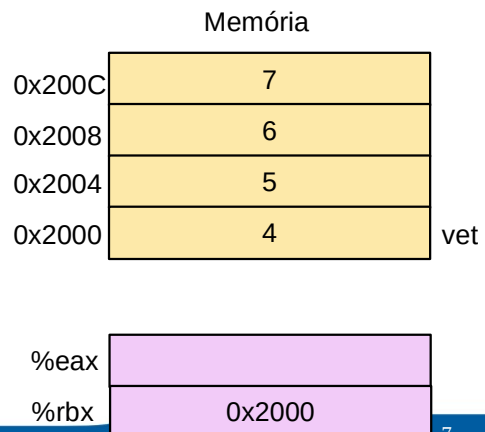


Memória: Base-Deslocamento

- Base é um registrador tem um endereço de memória
- Deslocamento é um número inteiro (positivo ou negativo)
- Formato: $\text{Im}(\text{R}_{\text{Base}}) \rightarrow \text{Addr} = \text{Im} + \text{R}_{\text{Base}}$

```
.data
vet: .int 4, 5, 6, 7

.text
main:
    movl    $1, 4(%rbx)
    movl    8(%rbx), %eax
```

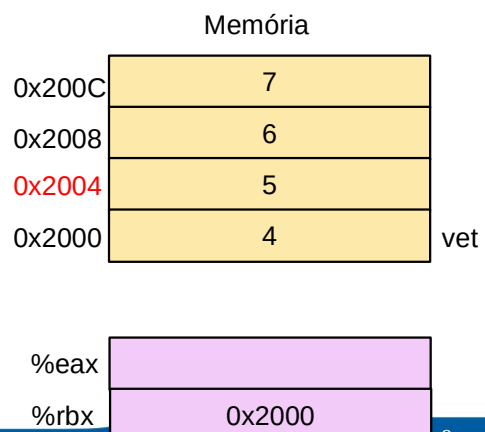


Memória: Base-Deslocamento

- Base é um registrador tem um endereço de memória
- Deslocamento é um número inteiro (positivo ou negativo)
- Formato: $\text{Im}(\text{R}_{\text{Base}}) \rightarrow \text{Addr} = \text{Im} + \text{R}_{\text{Base}}$

```
.data
vet: .int 4, 5, 6, 7

.text
main:
    movl    $1, 4(%rbx)
    movl    8(%rbx), %eax
```



Memória: Base-Deslocamento

- Base é um registrador tem um endereço de memória
- Deslocamento é um número inteiro (positivo ou negativo)
- Formato: $\text{Im}(\text{R}_{\text{Base}}) \rightarrow \text{Addr} = \text{Im} + \text{R}_{\text{Base}}$

```
.data
vet: .int 4,5,6,7

.text
main:
    movl    $1, 4(%rbx)
    movl    8(%rbx), %eax
```

Memória

0x200C	7	
0x2008	6	
0x2004	1	
0x2000	4	vet

%eax	
%rbx	0x2000

9

Memória: Base-Deslocamento

- Base é um registrador tem um endereço de memória
- Deslocamento é um número inteiro (positivo ou negativo)
- Formato: $\text{Im}(\text{R}_{\text{Base}}) \rightarrow \text{Addr} = \text{Im} + \text{R}_{\text{Base}}$

```
.data
vet: .int 4,5,6,7

.text
main:
    movl    $1, 4(%rbx)
    movl    8(%rbx), %eax
```

Memória

0x200C	7	
0x2008	6	
0x2004	1	
0x2000	4	vet

%eax	
%rbx	0x2000

10

Memória: Base-Deslocamento

- Base é um registrador tem um endereço de memória
- Deslocamento é um número inteiro (positivo ou negativo)
- Formato: $\text{Im}(\text{R}_{\text{Base}}) \rightarrow \text{Addr} = \text{Im} + \text{R}_{\text{Base}}$

```
.data
vet: .int 4, 5, 6, 7

.text
main:
    movl    $1, 4(%rbx)
    movl    8(%rbx), %eax
```

Memória

0x200C	7
0x2008	6
0x2004	1
0x2000	4

vet

%eax	6
%rbx	0x2000

11

Memória: Indexado (1)

- Base é um registrador tem um endereço de memória
- Índice é um registrador com deslocamento
- Formato: $(\text{R}_{\text{Base}}, \text{R}_{\text{Idx}}) \rightarrow \text{Addr} = \text{R}_{\text{Base}} + \text{R}_{\text{Idx}}$

```
.data
vet: .int 4, 5, 6, 7

.text
main:
    movl    $10, (%rbx, %rcx)
```

Memória

0x200C	7
0x2008	10
0x2004	5
0x2000	4

vet

%rbx	0x2000
%rcx	8

12

Memória: Indexado (2)

- Base é um registrador tem um endereço de memória
- Índice é um registrador com deslocamento
- Formato: $\text{Im}(R_{\text{Base}}, R_{\text{Idx}}) \rightarrow \text{Addr} = \text{Im} + R_{\text{Base}} + R_{\text{Idx}}$

```
.data
vet: .int 4, 5, 6, 7

.text
main:
    movl $10, 4(%rbx, %rcx)
```

Memória	
0x200C	10
0x2008	6
0x2004	5
0x2000	4

vet

%rbx	0x2000
%rcx	8

13

Memória: Indexado e Escalado (1)

- Registrador índice tem um endereço de memória
- A escala s pode ser 1, 2, 4 ou 8
- Formato: $(, R_{\text{Idx}}, s) \rightarrow \text{Addr} = s * R_{\text{Idx}}$

```
.data
vet: .int 4, 5, 6, 7

.text
main:
    movl $10, (, %rcx, 2)
```

Memória	
0x200C	7
0x2008	6
0x2004	5
0x2000	10

vet

%rcx	0x1000
------	--------

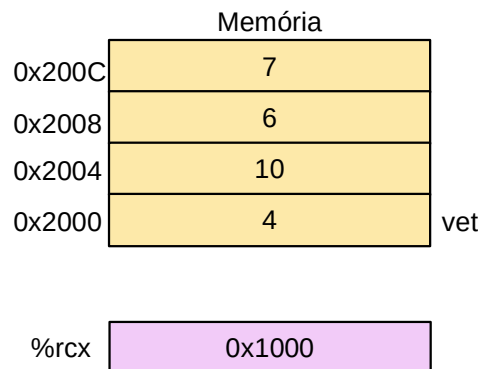
14

Memória: Indexado e Escalado (2)

- Registrador índice tem um endereço de memória
- A escala s pode ser 1, 2, 4 ou 8
- Formato: $\text{Im}(\text{ , } R_{\text{Idx}}, s) \rightarrow \text{Addr} = \text{Im} + s * R_{\text{Idx}}$

```
.data
vet: .int 4,5,6,7

.text
main:
    movl $10, 4( , %rcx, 2)
```



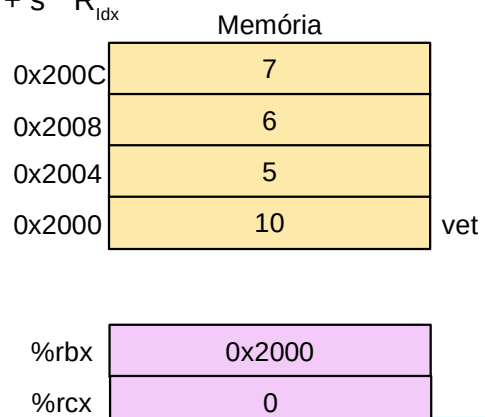
15

Memória: Indexado e Escalado (3)

- Registrador base tem um endereço de memória
- Registrador índice é usado para deslocamento
- A escala s pode ser 1, 2, 4 ou 8
- Formato: $(R_{\text{Base}}, R_{\text{Idx}}, s) \rightarrow \text{Addr} = R_{\text{Base}} + s * R_{\text{Idx}}$

```
.data
vet: .int 4,5,6,7

.text
main:
    movl $10, (%rbx, %rcx, 4)
    incq %rcx
    movl $11, (%rbx, %rcx, 4)
```



16

Memória: Indexado e Escalado (3)

- Registrador base tem um endereço de memória
- Registrador índice é usado para deslocamento
- A escala s pode ser 1, 2, 4 ou 8
- Formato: $(R_{Base}, R_{Idx}, s) \rightarrow Addr = R_{Base} + s * R_{Idx}$

```
.data
vet: .int 4,5,6,7

.text
main:
    movl $10, (%rbx, %rcx, 4)
    incq %rcx
    movl $11, (%rbx, %rcx, 4)
```

Memória	
0x200C	7
0x2008	6
0x2004	5
0x2000	10

vet

%rbx	0x2000
%rcx	1

17

Memória: Indexado e Escalado (3)

- Registrador base tem um endereço de memória
- Registrador índice é usado para deslocamento
- A escala s pode ser 1, 2, 4 ou 8
- Formato: $(R_{Base}, R_{Idx}, s) \rightarrow Addr = R_{Base} + s * R_{Idx}$

```
.data
vet: .int 4,5,6,7

.text
main:
    movl $10, (%rbx, %rcx, 4)
    incq %rcx
    movl $20, (%rbx, %rcx, 4)
```

Memória	
0x200C	7
0x2008	6
0x2004	20
0x2000	10

vet

%rbx	0x2000
%rcx	1

18

Memória: Indexado e Escalado (4)

- Registrador base tem um endereço de memória
- Registrador índice é usado para deslocamento
- A escala s pode ser 1, 2, 4 ou 8
- Formato: $\text{Im}(\text{R}_{\text{Base}}, \text{R}_{\text{Idx}}, \text{s}) \rightarrow \text{Addr} = \text{Im} + \text{R}_{\text{Base}} + \text{s} * \text{R}_{\text{Idx}}$ Memória

```
.data
vet: .int 4, 5, 6, 7

.text
main:
    movl $10, 4(%rbx, %rcx, 4)
```

0x200C	7
0x2008	10
0x2004	5
0x2000	4

vet

%rbx	0x2000
%rcx	1

19

Instrução “leaq”

- A instrução “leaq” resolve (calcula) o endereço final e o armazenando em um registrador
- Não há nenhum acesso à memória

```
leaq 4(%rbx), %rax      /* rax = %rbx + 4      */
leaq (%rbx, %rcx, 4), %rdi /* rdi = %rbx + 4 * %rcx */
leaq 4( , %rcx, 2), %rax /* rax = 4 + 2 * %rcx */
```