Aula 13

Gerência de Memória - Segmentação

1.1 Limitações da paginação - o contexto da segmentação
1.2 O que é a segmentação
1.3 Implementação
1.4 Referências: Capítulo 9 (9.5)

O problema com a paginação

A visão do usuário é diferente da visão do S.O.

Como o usuário vê o espaço de endereçamento do processo

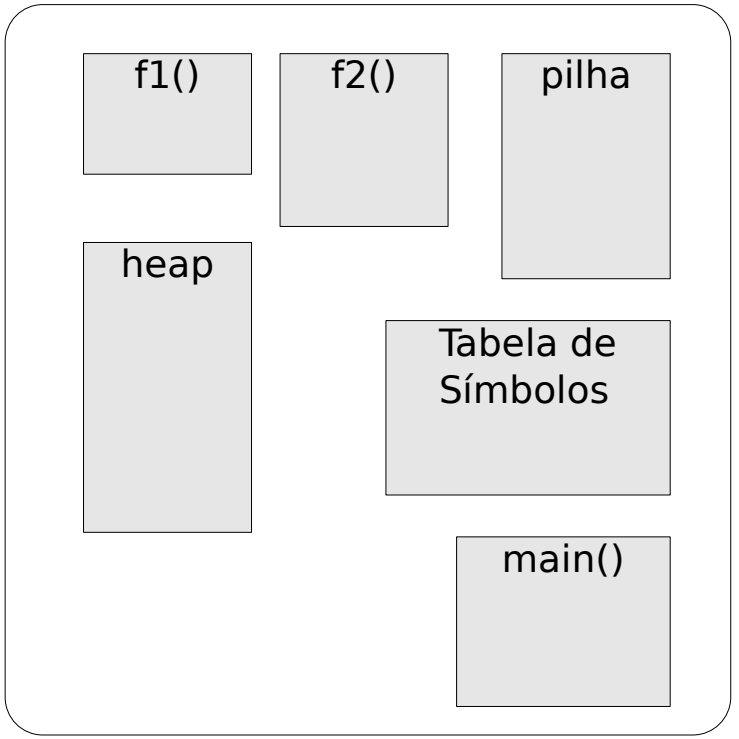
Pag 0
Pag 1
Pag 2
Pag 3
Pag 4
⋮

Como um conjunto homogêneo de endereços e posições de memória

Já o S.O. ...

... vê um conjunto de áreas cada uma com uso e dimensão próprios

O S.O vê o programa de forma **segmentada**

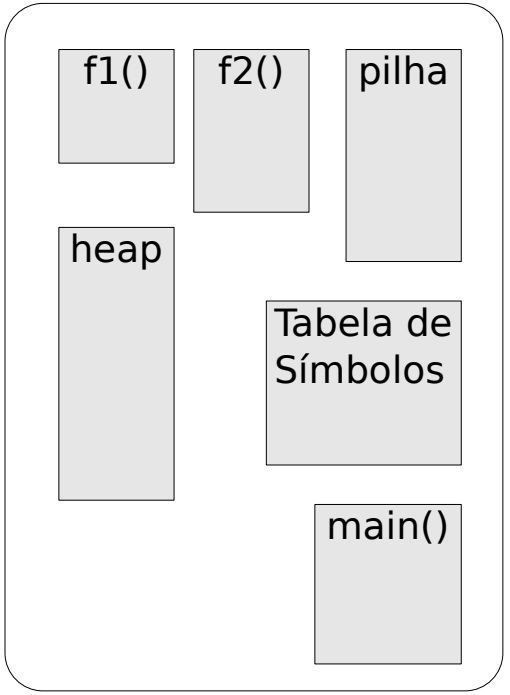


Segmentação

Muitos SOs implementam uma visão **segmentada** do espaço de endereçamento:

- Cada “pedaço” do programa é tratado como um **segmento**
- Todo endereçamento usa:
  <Nro do segmento,Offset>
- Desta forma, a visão do usuário replica a do S.O.

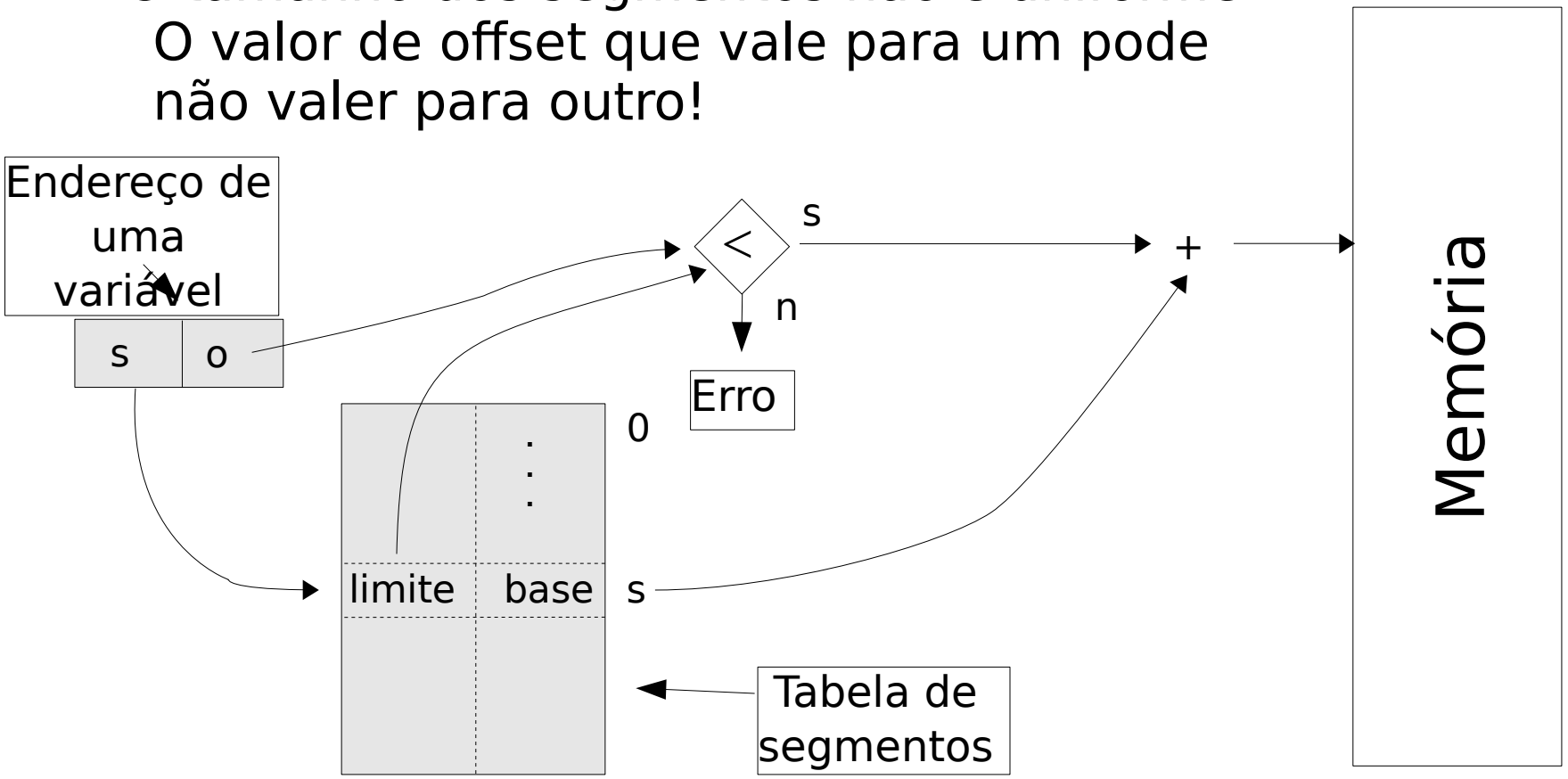
- Facilita a proteção, pois é impossível “errar” de segmento durante a execução:
  - Com a paginação é possível “errar”, porque o endereçamento dentro de uma tarefa é linear



Implementação (1)

O hardware não é o mesmo da paginação:

- O tamanho dos segmentos não é uniforme.
  O valor de offset que vale para um pode não valer para outro!



Implementação (2)

Mais complicado!

Mais caro porque a tabela de segmentos tem o dobro do tamanho da tabela de páginas

Mas não é mais lento:

- A comparação do offset com o limite pode ser feita simultâneamente com a soma da base

Fragmentação

Fragmentação é um problema da segmentação:
- Fragmentação externa

Normalmente, não é um problema grave, porque a execução do programa é dinâmica por natureza:

- Se não houver espaço para alocar um segmento, pode-se simplemente esperar até que o espaço seja liberado

Segmentação e Paginação

“To page or to segment, that is the question”

Guess what... Do both!.

Como se já não fôsse confuso o suficiente:
- Existem processadores que implementam segmentação com paginação

Felizmente, são processadores obscuros que podem ser ignorados sem problema:

Todos os processadores da Intel a partir do 386

Endereçamento no 386 (1)

- Número máximo de segmentos por processo: 16K
- Tamanho de cada segmento: 4G
- Tamanho da página 4K

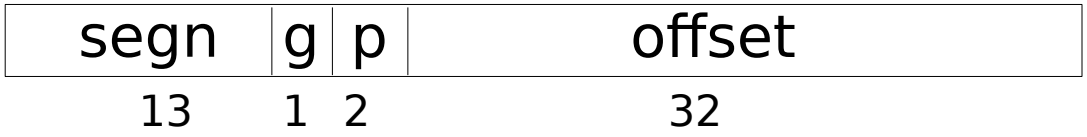
Endereçamento no 386 (2)

Espaço de endereçamento se divide em duas partições com 8K segmentos cada:

- Local, descrita na LDT
- Global, descrita na GDT

Valores na LDT e GDT têm 8 bytes contendo informações sobre o segmento, tais como base e limite

O endereço é:



segn: número do segmento
g: LDT ou GDT
p: proteção
offset: deslocamento dentro do segmento

Implementação

Existem 6 registradores de segmento, que permitem até 6 segmentos serem acessados "simultâneamente"

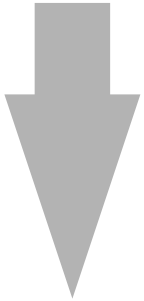
- Funciona como a TLB, mas menor
- Não é problema, porque os segmentos podem ser grandes

- Paginação em dois níveis de tabelas

Endereçamento

Endereço lógico:

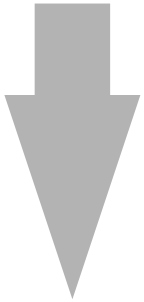
segn	g	p	offset
13	1	2	32



Cálculo do segmento

Endereço linear:

p1	p2	page offset
10	10	12



Cálculo da página

Endereço físico:

Complicado... (1)

Porquê os processadores da Intel usam este método?

Razões históricas – **compatibilidade**

- 8085: 16 bits de endereçamento – linear
- 8088: 20 bits de endereçamento (?) –
  para manter a compatibilidade considera que progra-
  mas antigos rodam dentro de um segmento.

  Segmentação:
  - 16 bits são o offset
  - 4 bits são o número do segmento

Complicado... (2)

Razões históricas – **compatibilidade**

- 8086: Primeiro true 16 bits. Endereçamento pode ser linear, mas já era tarde, e os segmentos já fazem parte dos programas

- 80386: Segmentação ineficiente leva à introdução da **paginação**

Conclusão: compatibilidade tem seu preço!

OBS: Fonte dos dados: *vaga lembrança*. Precisão dos números não é garantida

Conclusão

Segmentação: técnica que permite uma visão uniforme do espaço de endereçamento pelo usuário e pelo S.O.

Processadores Intel: esquema complexo de segmentação e paginação, mantido para garantir compatibilidade com processadores anteriores