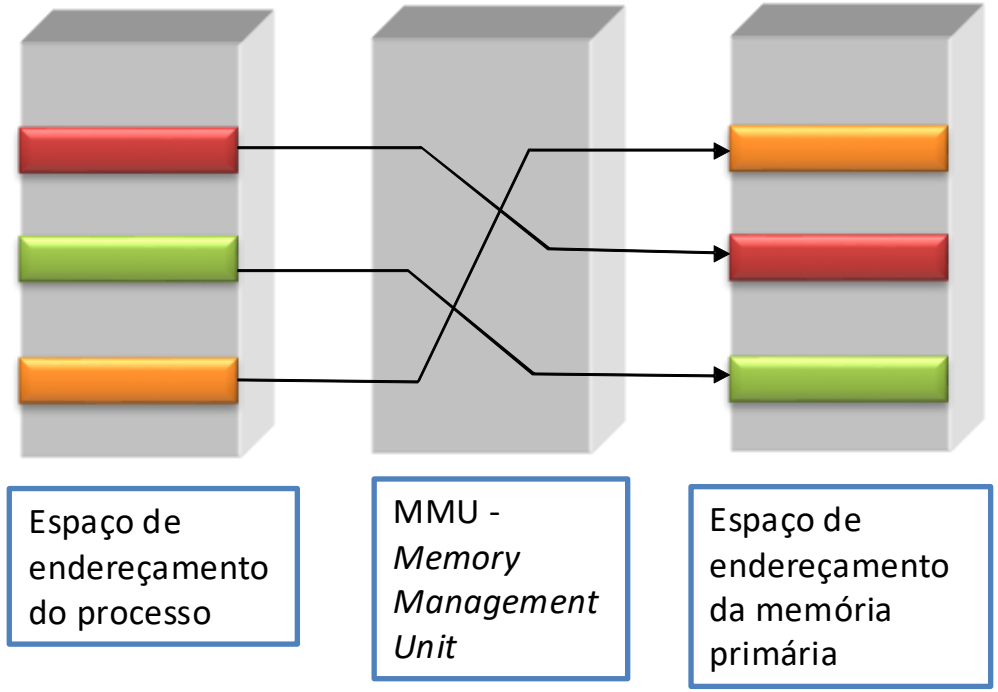
Funcionamento do Endereçamento Virtual

Gestão de Memória – Objetivos do Sistema Operativo

- Gerir o espaço de endereçamento dos processos
  - Assegurar que cada processo dispõe da memória que necessita
  - Garantir que cada processo só acede à memória a que tem direito (proteção)
  - Optimizar o desempenho dos acessos a memória

Como Traduzir um Endereço Virtual num Endereço Físico?



- A solução mais flexível seria utilizar uma tabela que, para cada endereço virtual, indicasse qual o endereço físico correspondente.
- Neste caso, *cada byte* do espaço de endereçamento de um processo poderia ficar em qualquer endereço da memória física.
- No entanto, ter uma tabela com uma entrada para cada *byte* faria com que fosse ocupada mais memória com a tabela de tradução de endereços do que com a informação propriamente dita.

Endereços virtuais de 32 bits e endereços físicos também de 32 bits podemos endereçar até 4 *Gbytes*
A tabela teria pelo menos a dimensão de $2^{32} \times 2^{2} = 2^{34}$ *bytes (16 Gbytes)*!

Simplificar a Tradução de um Endereço Virtual num Endereço Físico

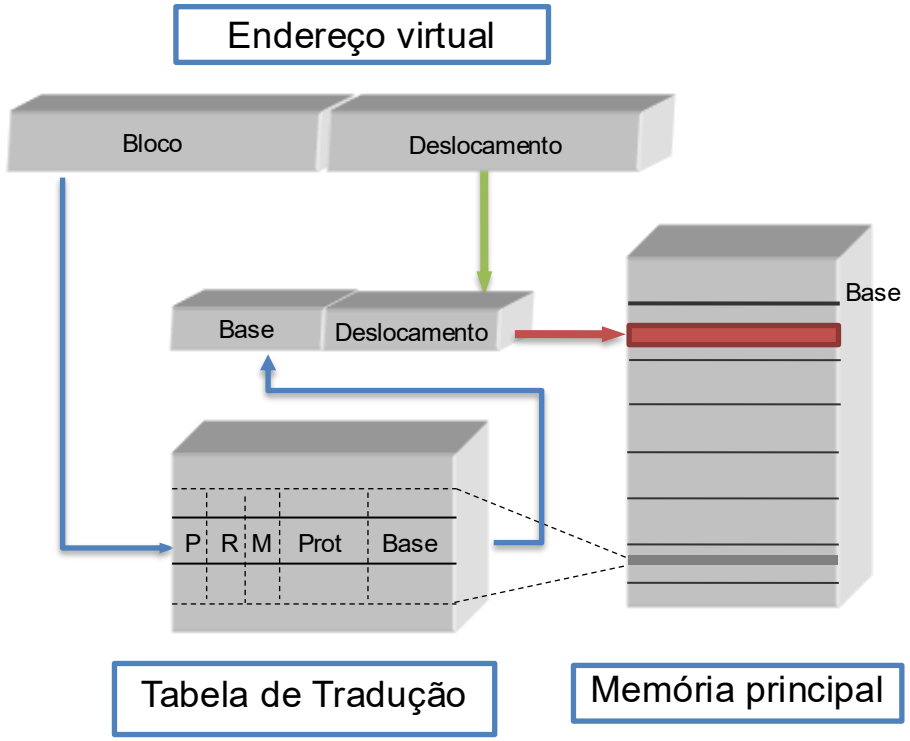
- **Uma gestão em blocos** simplifica muito a tradução porque só é preciso traduzir o numero de bloco na base do endereço físico onde o bloco se encontra em memória

- Os blocos tornam mais eficiente a localidade espacial

- Os blocos também otimizam as transferências de memoria de massa para memória primária e vice-versa

- Um endereço virtual tem a seguinte formato:

Tradução do Endereço Virtual

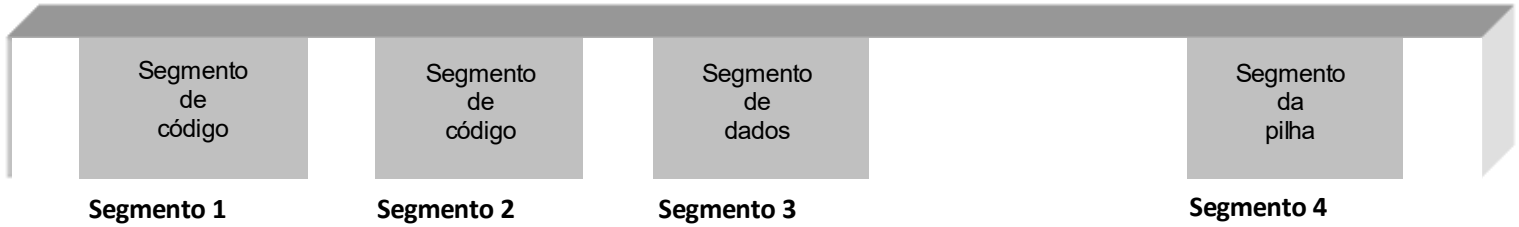


- Problemas a resolver
  - Como definir os blocos
  - Qual a dimensão dos blocos
- A definição dos blocos tem duas soluções
  - Segmentos - dimensão variável.
  - Páginas - dimensão constante
- A dimensão tem de ter em conta a fragmentação e arquitetura dos dispositivos de memória de massa

Endereços virtuais de 32 bits e endereços físicos também de 32
Deslocamento 12 bits -> 20 bits para numero do bloco
A tabela teria pelo menos a dimensão de $2^{20} \times 2^2 = 2^{22}$ *bytes (4 Mbytes)*

Segmentação

- Divisão dos programas em segmentos lógicos que refletem a sua estrutura funcional:
  - Rotinas, módulos, bibliotecas, dados, *heap*, pilha, zonas de memória partilhadas, etc.



- Segmento é a unidade de:
  - Carregamento em memória (eficiência)
  - Proteção

- Dimensão dos segmentos:
  - limitada pela arquitetura (numero de bits do deslocamento) e obviamente não pode exceder a dimensão da memória principal

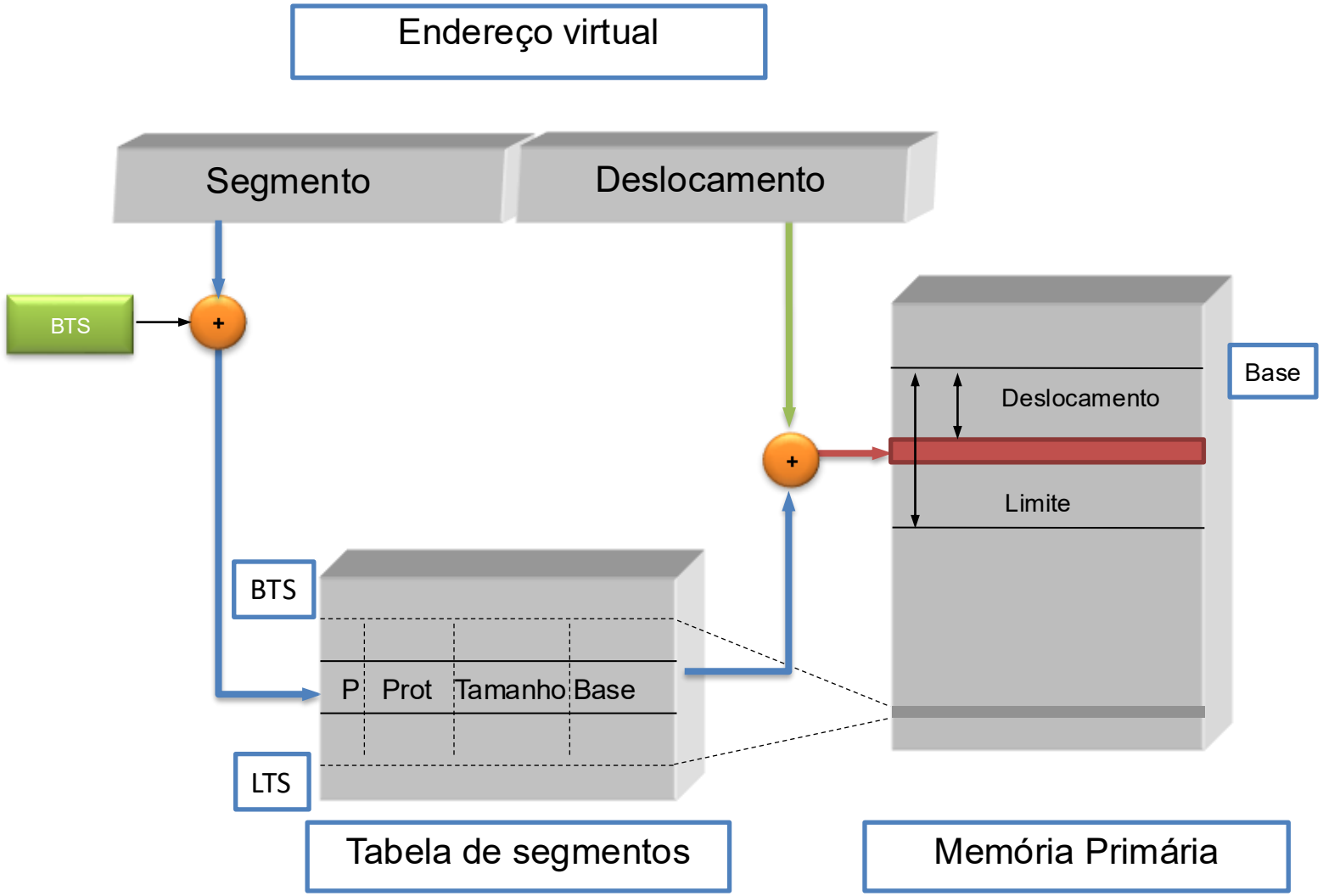
Segmentação

- Quando um programa se executa o seu espaço de endereçamento é representado por um mapa que indica todos os segmentos que o constituem

- Para cada segmento é necessário saber
  - Se está armazenado no disco ou presente na memória
  - Dimensão
  - Tipo de acesso permitido

- A descrição deste mapa do espaço de endereçamento tem de estar numa tabela de tradução durante a execução: **tabela de segmentos**

Tradução de Endereços em Memória Segmentada



P	Presente na memória
Prot	Leitura/escrita/ execução
Tamanho	Dimensão do segmento (inferior a $2^{\text{bits deslocamento}}$)
Base	Endereço base na memória primária

BTS	Base da tabela de segmentos
LTS	Limite da tabela de segmentos

Memória Virtual Segmentada

- O endereço físico é calculado somando a base com o deslocamento
- Proteção:
  - verificação de limites de endereçamento intra-segmento (tamanho)
  - limitação dos tipos de acesso ao segmento: leitura, escrita e execução
  - processos diferentes têm tabelas de segmentos diferentes: espaços de endereçamento disjuntos e inacessíveis a terceiros
- O programador pode ter que se preocupar com a gestão de memória quando escreve um programa devido a dimensão máxima dos segmentos. Normalmente através de diretivas para o compilador

Fragmentação

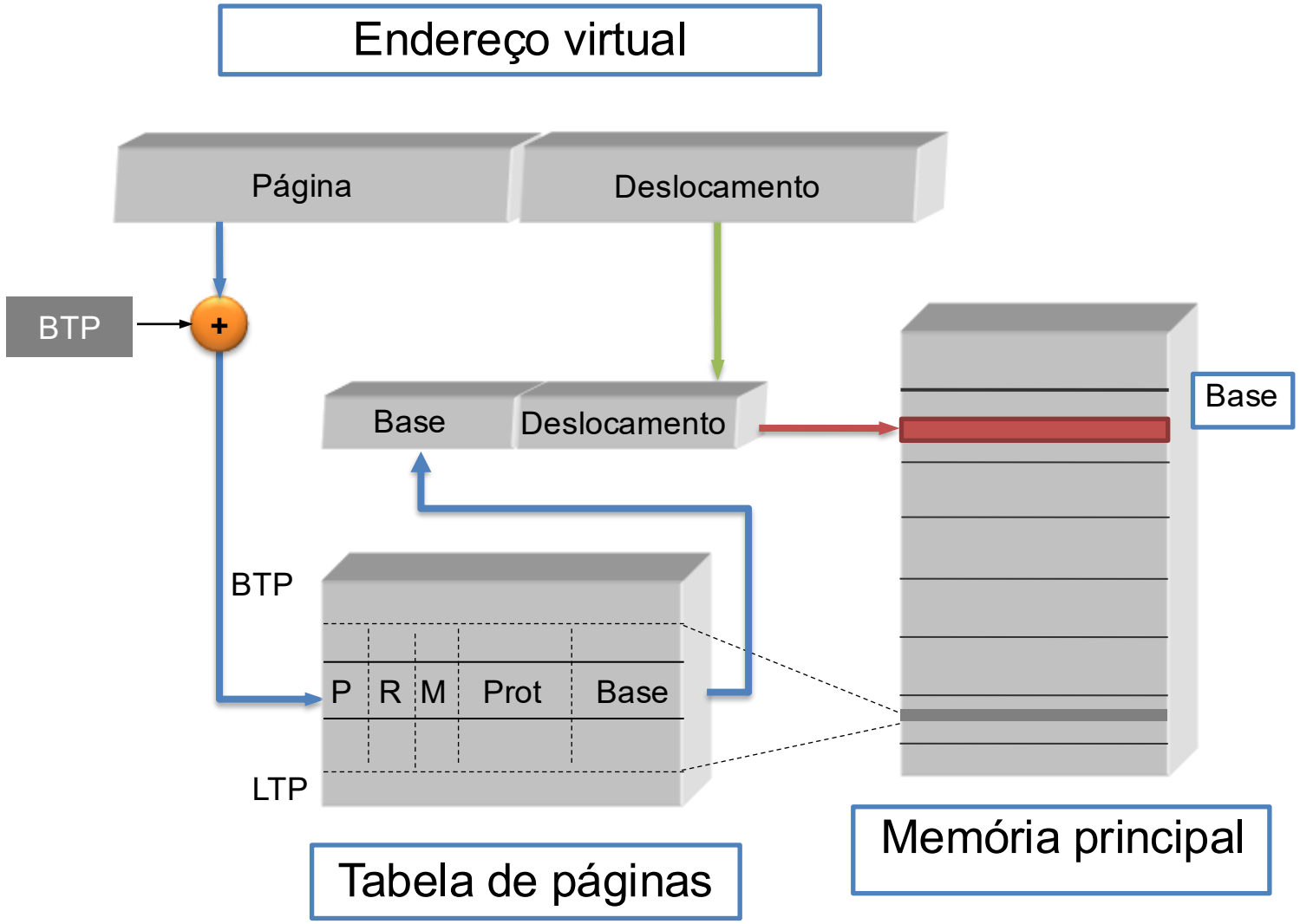
- Percentagem do espaço de memória primária não utilizado devido a gestão em blocos
  - **Fragmentação interna** – o bloco é maior que a informação e fica desocupada uma parte
  - **Fragmentação externa** – os blocos são colocados na memória primária e quando são substituídos só um bloco menor irá ocupar esse espaço (probabilidade de encontrar um igual é mínima). Ao fim de algum tempo a memória terá vários pequenos blocos desocupados
- Na segmentação como os blocos tem dimensão variável teremos fragmentação externa

Paginação

- Blocos de tamanho fixo, chamados páginas

- Espaço de endereçamento virtual linear, i.e., contíguo
  - o programador não se apercebe da gestão de memória virtual
- A memória primária é gerida em blocos do mesmo tamanho das paginas virtuais
  - No cálculo do endereço físico basta concatenar a base com o deslocamento
- Protecção:
  - Verificação dos tipos de acesso: leitura, escrita e execução.
  - Processos diferentes têm tabelas de páginas diferentes: espaços de endereçamento disjuntos e inacessíveis a terceiros

Memoria Virtual Paginada



P	Presente na memória
R	Referenciado
M	Modificada
Prot	Leitura/escrita/execução
Base	Endereço base na memória primária

BTP	Base da tabela de páginas
LTP	Limite da tabela de páginas

Qual a dimensão certa para as páginas?

- A dimensão da página influencia:
  - A fragmentação interna (quanto maiores, maior a quantidade de memória não ocupada na última pagina)
  - O número de faltas de páginas (paginas maiores conduzem a menos faltas de paginas)
  - Tempo de transferência – cresce com a dimensão das páginas
  - A dimensão das tabelas de páginas e listas de páginas mantidas pelo sistema operativo

- Valor típico hoje em dia: 4 Kbytes

Falta de Página

- A tabela de páginas tem o **bit de presença** (bit P) que indica se a página está ou não em memória primária.

- Se o bit P estiver a zero, o *hardware* de gestão de memória (MMU) gera uma exceção que interrompe a instrução. Diz-se que ocorreu uma **falta de página** (*page fault*).

- O sistema operativo analisa a causa da exceção e começa o processamento adequado.
  - Em primeiro lugar, terá de alocar uma página livre em memória primária
  - Se é uma página nova (resultante do crescimento da pilha, ou uma página de dados não inicializados), basta preenchê-la com zeros;
  - Se existe uma cópia da página em memória secundária, é necessário lê-la do disco.

As instruções tem de ser *restartable*

- As exceções provocadas pela Unidade de Gestão de Memória têm uma diferença importante em relação às interrupções que interrompem o processador **no fim** de uma instrução.

- Com a memória paginada, uma instrução pode aceder a múltiplos *bytes*, partidos entre duas páginas: se a primeira página estiver presente e a segunda não, haverá uma falta de página a meio da instrução.

- Se o processador não fosse capaz de interromper a instrução a meio e depois reexecutá-la, a instrução ficaria perdida e o programa não funcionaria corretamente.

- As exceções da gestão de memória têm, portanto, de **interromper a instrução a meio** e o processador tem de ser capaz de, mais tarde, completar a instrução que foi interrompida. Diz-se que as instruções têm de ser recomeçáveis (*restartable*). Esta facilidade implica obviamente uma maior complexidade da estrutura do processador.

Equipa UGM/núcleo do SO

- A maioria dos acessos a memória são traduzidos e servidos pela UGM
  - Se processo está em modo utilizador, mantém-se nesse modo
- Núcleo só se envolve na tradução nestes momentos:
  - Quando comuta para outro processo
  - Quando página acedida não está presente
  - Quando acesso é ilegal (endereço fora dos limites ou sem permissões)

3. [1,5 val] Considere um sistema de gestão de memória com 16 bits de espaço de endereçamento virtual, com páginas de 256B e onde existem dois processos, P1 e P2.

Suponha que os processos P1 e P2 tenham as seguintes tabelas de páginas:

P1:

Página	Presente	Protecção	Base
0	0	RW	
1	1	R	0x02
2	1	R	0x04

P2 :

Página	Presente	Protecção	Base
0	1	RW	0x02
1	0	R	
2	1	R	0x01

Na tabela abaixo, indique qual é o endereço físico correspondente aos seguintes endereços virtuais, caso possível. Caso o acesso der origem a alguma exceção, indique também o tipo da exceção.
Assuma que, em caso de falta de páginas, o SO aloca tramas (*page frames*) livres a partir do endereço 0x09 00.

Processo	Acesso	End. Virtual	End. Físico	Eventuais exceções
P1	Leitura	0x01 22		
P2	Escrita	0x00 22		
P1	Leitura	0x03 01		
P1	Leitura	0x00 FF		

3. [1,5 val] Considere um sistema de gestão de memória com 16 bits de espaço de endereçamento virtual, com páginas de 256B e onde existem dois processos, P1 e P2.

Suponha que os processos P1 e P2 tenham as seguintes tabelas de páginas:

P1:

Página	Presente	Protecção	Base
0	0	RW	
1	1	R	0x02
2	1	R	0x04

P2 :

Página	Presente	Protecção	Base
0	1	RW	0x02
1	0	R	
2	1	R	0x01

Na tabela abaixo, indique qual é o endereço físico correspondente aos seguintes endereços virtuais, caso possível. Caso o acesso der origem a alguma exceção, indique também o tipo da exceção.
Assuma que, em caso de falta de páginas, o SO aloca tramas (*page frames*) livres a partir do endereço 0x09 00.

Processo	Acesso	End. Virtual	End. Físico	Eventuais exceções
P1	Leitura	0x01 22	0x0222	
P2	Escrita	0x00 22	0x0222	
P1	Leitura	0x03 01	-	Endereço inválido
P1	Leitura	0x00 FF	0x09ff	Page fault

Resumo

- Vimos duas formas de organizar a memória virtual
  - Segmentos – baseados na estrutura do programa de dimensão variável
  - Paginação – divisão linear, tamanho fixo
- A maioria dos acessos a memória são traduzidos e servidos pela UGM usando as tabelas de segmentos ou páginas
  - Se processo está em modo utilizador, mantém-se nesse modo
- O núcleo do sistema operativo só se envolve na tradução quando:
  - a página acedida não está presente
  - o acesso é ilegal (endereço fora dos limites ou sem permissões)
  - Ou é necessário comutar para outro processo