SISTEMAS OPERACIONAIS

Gerência de Memória

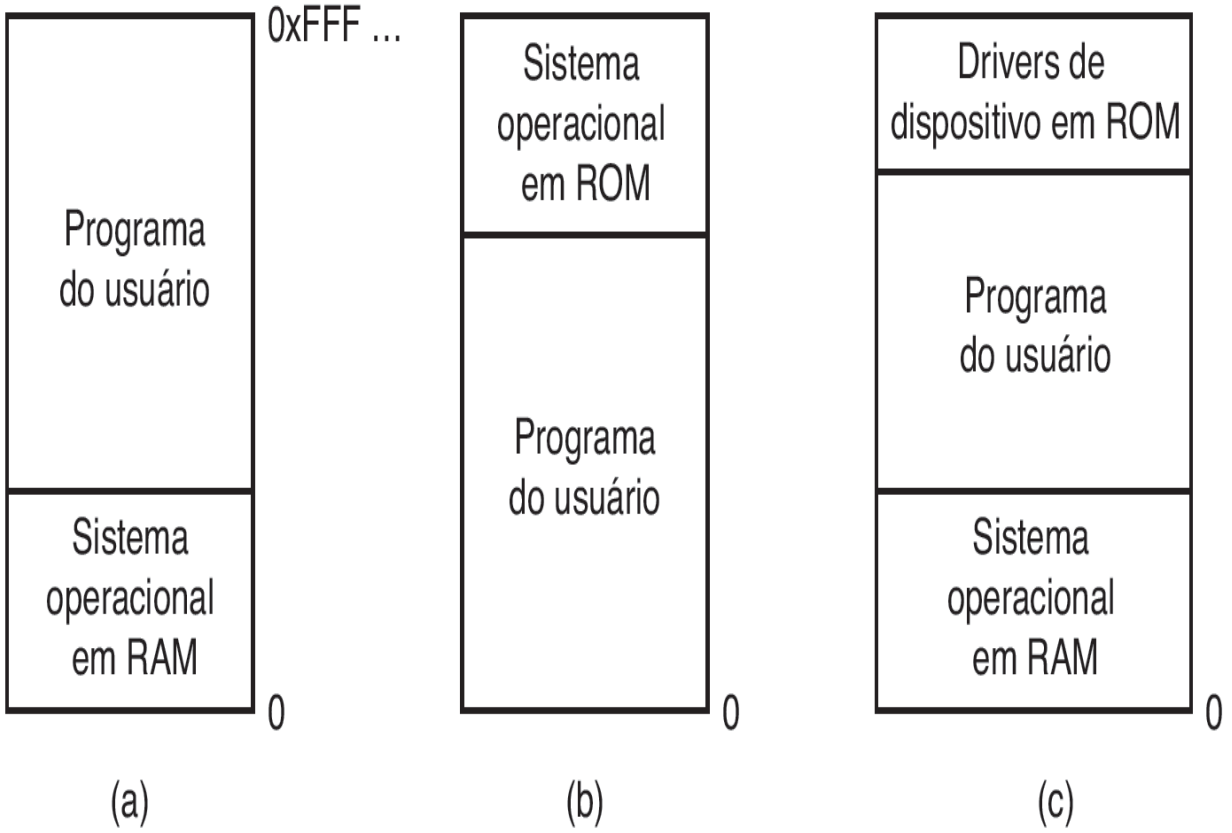
Andreza Leite
andreza.leite@univasf.edu.br

Plano da Aula

- Introdução
  - Necessidade gerenciador de memória
- Sistemas gerenciais de memória
  - Alocação contínua
    - Máquina simples
    - Monitor residente
    - Participações múltiplas
      - Tamanho fixo
      - Tamanho variável

Memória

- Sem abstração de memória os programas consideravam memória física



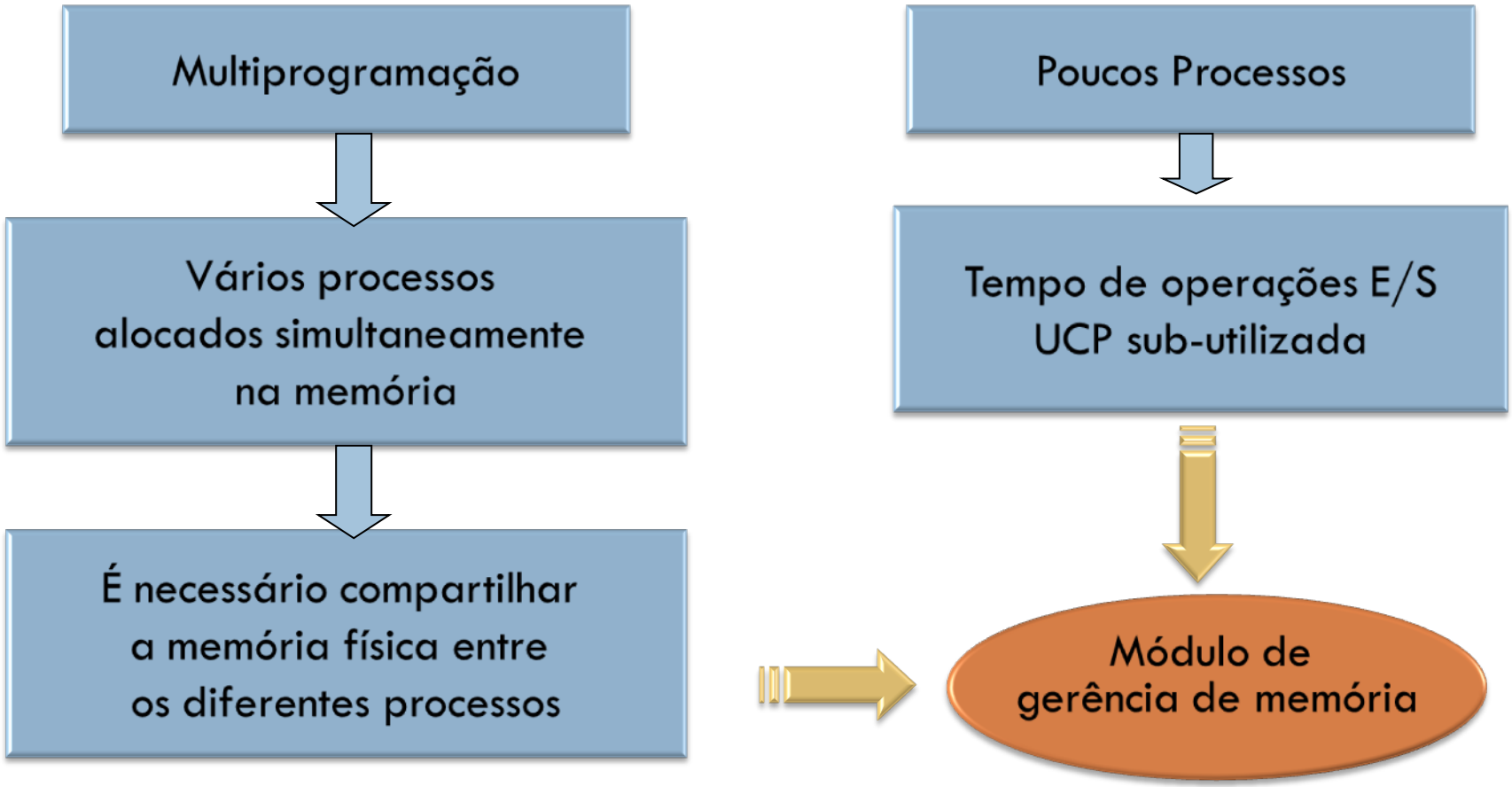
Memória

- Considerações:
  - Recurso caro e escasso;
  - Programas só executam se estiverem na memória principal;
  - Quanto mais processos residentes na memória principal, melhor será o compartilhamento do processador;
  - Necessidade de uso otimizado;
  - O S.O. não deve ocupar muita memória;
  - "É um dos fatores mais importantes em um projeto de S.O.".

Memória

- Sistema operacional deve:
  - controlar quais regiões de memória são utilizadas e por qual processo
  - decidir qual processo deve ser carregado para a memória, quando houver espaço disponível
  - alocar e desalocar espaço de memória

Memória


Multiprogramação
Vários processos alocados simultaneamente na memória
É necessário compartilhar a memória física entre os diferentes processos
Poucos Processos
Tempo de operações E/S
UCP sub-utilizada
Módulo de gerência de memória

Memória

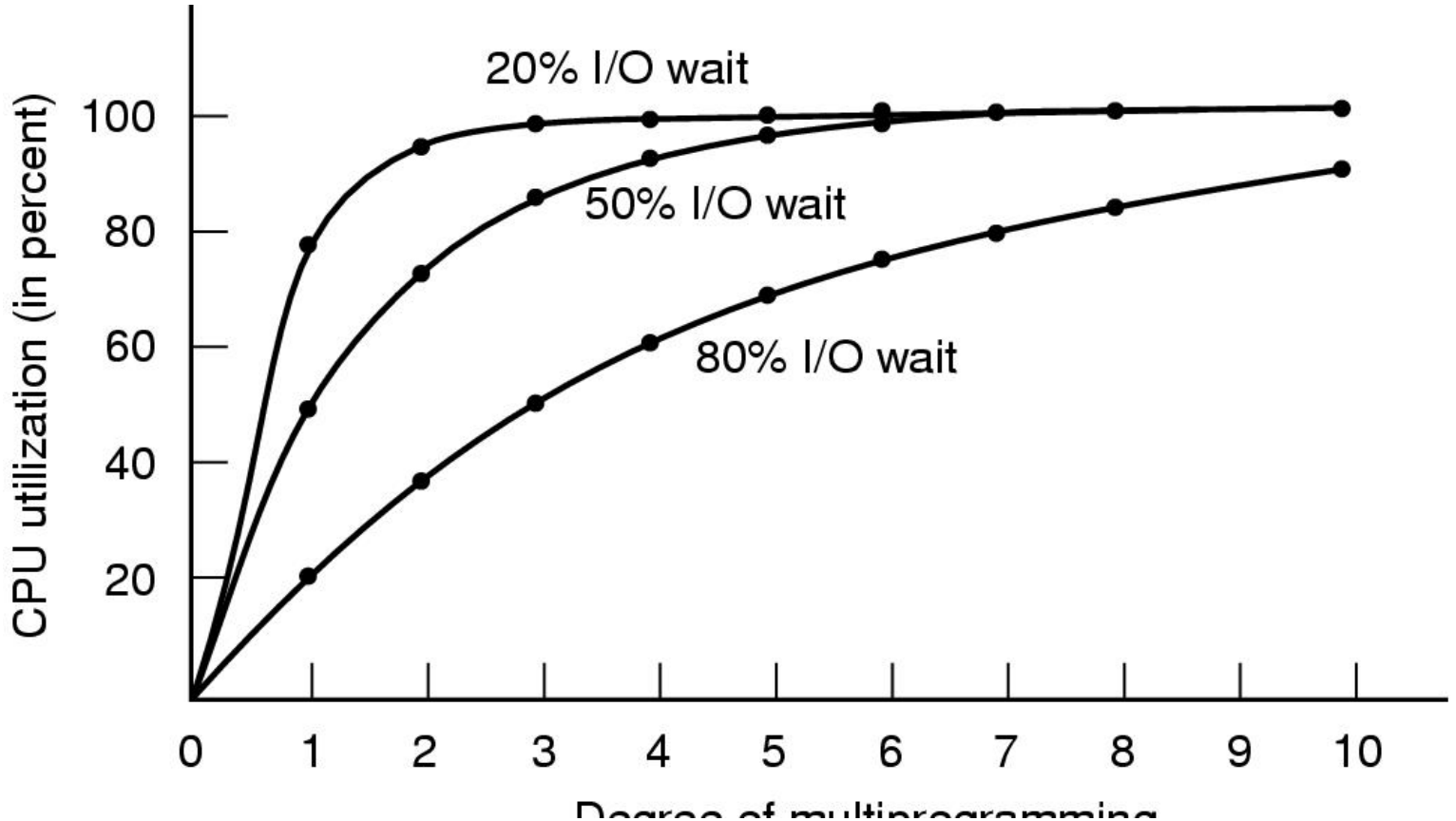
- Algumas funções do Gerenciador de memória:
  - Controlar quais as unidades de memória estão ou não estão em uso, para que sejam alocadas quando necessário;
  - Liberar as unidades de memória que foram desocupadas por um processo que finalizou;
  - Tratar do Swapping entre memória principal e memória secundária.
    - Transferência temporária de processos residentes na memória principal para memória secundária.

Memória

- Em um ambiente multiprogramado, é necessário:
  - Subdividir a memória para acomodar múltiplos processos
    - Se poucos processos estão na memória, em boa parte do tempo estarão esperando por E/S:
      - UCP sub-utilizada
    - Então, deve-se alocar memória de forma eficiente ao maior número de processos

Memória

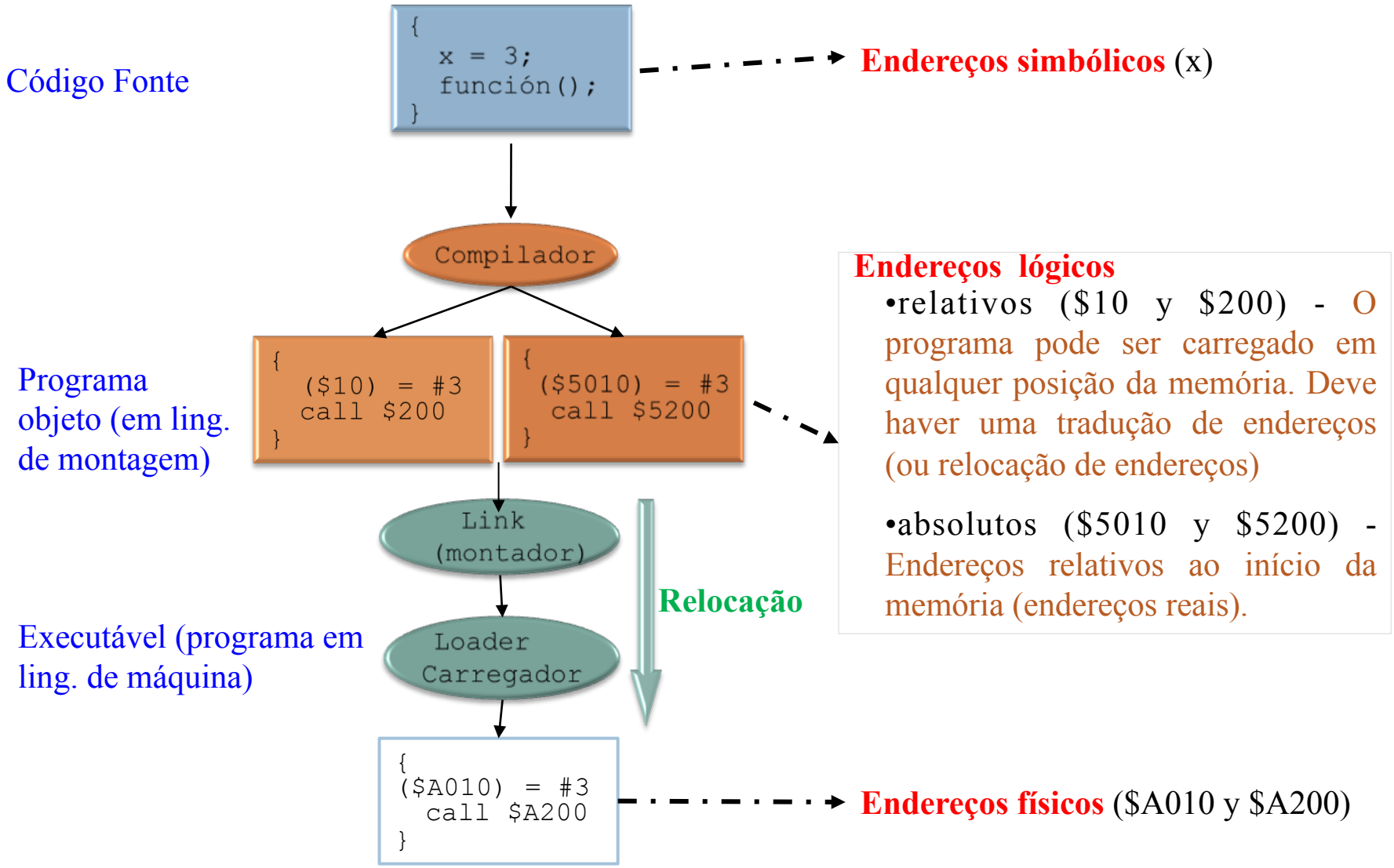
- Em um ambiente multiprogramado, é necessário:



Memória

- Em um ambiente multiprogramado, é necessário:
  - Tradução de endereços
  - Relocação
  - Proteção
  - Compartilhamento
  - Organização física / lógica

Memória


Código Fonte
{
x = 3;
función();
}
Endereços simbólicos (x)
Compilador
Programa objeto (em ling. de montagem)
{
($10) = #3
call $200
}
{
($5010) = #3
call $5200
}
Endereços lógicos
•relativos ($10 y $200) - O programa pode ser carregado em qualquer posição da memória. Deve haver uma tradução de endereços (ou relocação de endereços)
•absolutos ($5010 y $5200) - Endereços relativos ao início da memória (endereços reais).
Link (montador)
Relocação
Executável (programa em ling. de máquina)
Loader Carregador
{
($A010) = #3
call $A200
}
Endereços físicos ($A010 y $A200)

Memória – Tradução de endereços

- Uma das funções mais importantes do sistema de **gerenciamento de memória** é a **tradução** dos endereços lógicos utilizados pelos processos **para** endereços físicos reais.
- A **MMU** (*Memory-Managment Unit*) é o modulo hardware responsável por realizar a conversão entre os endereços lógicos e físicos

Memória – Tradução de endereços

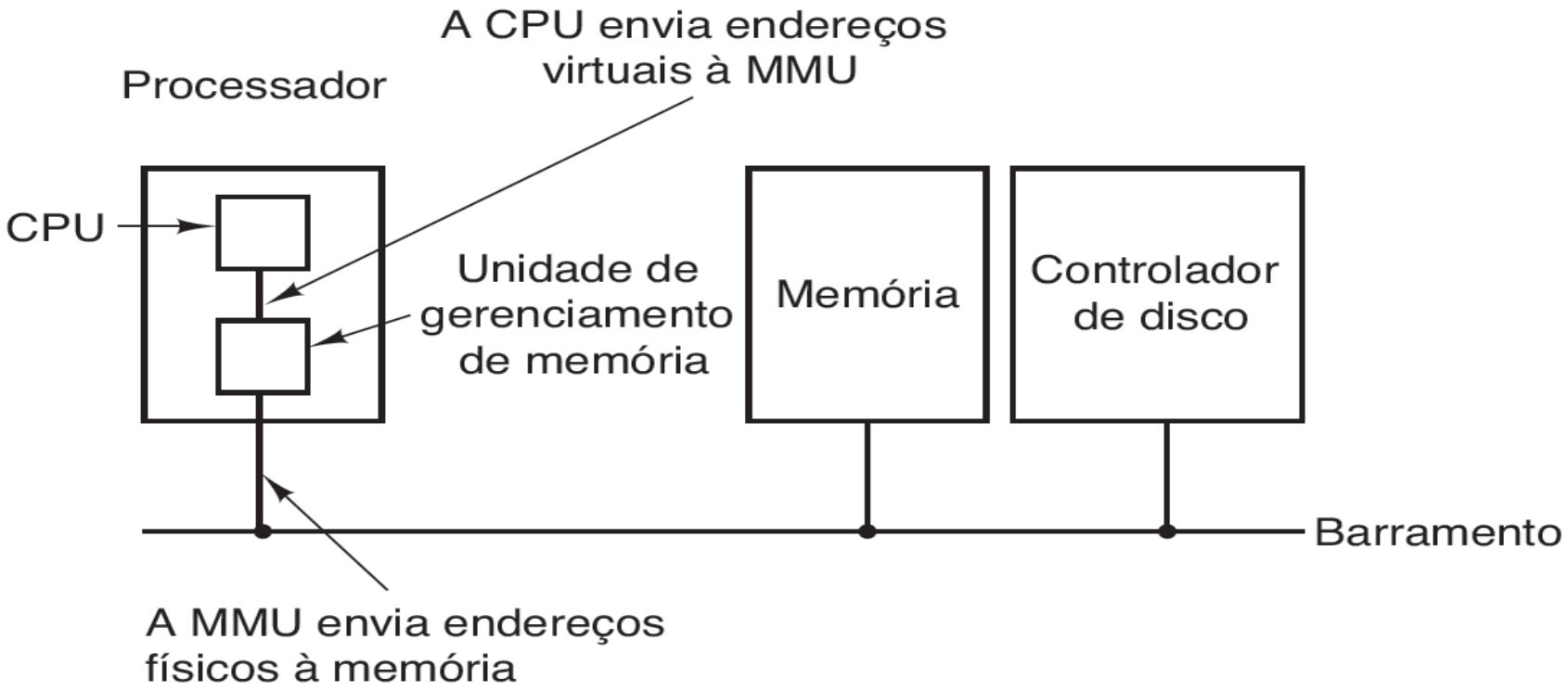
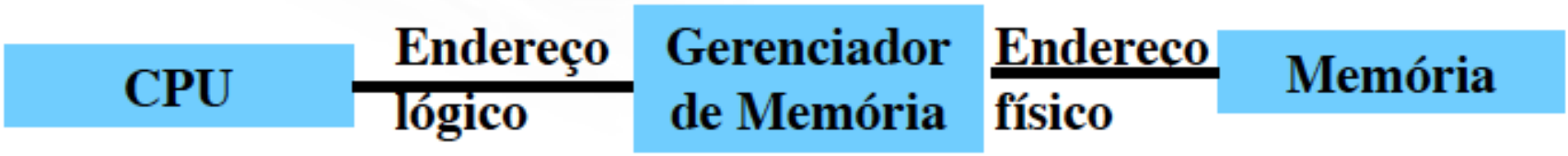


Figura 3.8 A posição e a função da MMU. Aqui a MMU é mostrada como parte do chip da CPU (processador) porque isso é comum atualmente. Contudo, em termos lógicos, poderia ser um chip separado, como ocorria no passado.

Memória – Tradução de endereços

- **Memória Lógica** - é aquela que o processo enxerga, o processo é capaz de acessar.
- **Memória Física** - é aquela implementada pelos circuitos integrados de memória, pela eletrônica do computador (memória real)



Memória – Tradução de endereços

- Funções MMU:
  - Verificar a posibilidade de conversão.
  - Comprovar as permissões de acesso do processo ao endereço de memória.
  - Converter endereços lógicos em físicos.
- O mecanismo de tradução da MMU depende do sistema de gerência de memória utilizado e vice-versa.

Memória – Relocação

- Capacidade de **mover** um programa de uma região da memória principal para uma outra **sem invalidar** as referências de memória dentro do programa;
- O programador não deve se preocupar com o local onde o programa (processo) será carregado para execução.
- Durante a execução, o processo poderá sair da memória e retornar para um local diferente.
- Referências devem ser resolvidas para **endereços de memória física**.
- O hardware do processador e o SO devem ser capazes de traduzir os endereços de referência de memória no código do programa para o endereço físico da memória

Memória – Relocação

- A relocação pode ocorrer sempre:
  - No momento de lançar o programa em memória.
  - Dinamicamente, durante a execução do processo.
- A MMU deve ser reajustada para cada processo de forma que a relocação possa ocorrer corretamente.

Memória – Relocação

- Relocação **estática** em tempo de **compilação**.
  - Se conhece/estabelece em que posição de memória se situará o processo, por tanto, o código gerado contém o endereço físico.
  - Endereços lógicos se igualam com os físicos.
- Relocação **estática** em tempo de **execução**.
  - O código é gerado utilizando endereços de memória lógicos uma vez que não é possível conhecer a posição final do processo na memória.
  - Quando o processo é alocado, se realizam tarefas de tradução de endereços lógicos à físicos.

Memória – Relocação

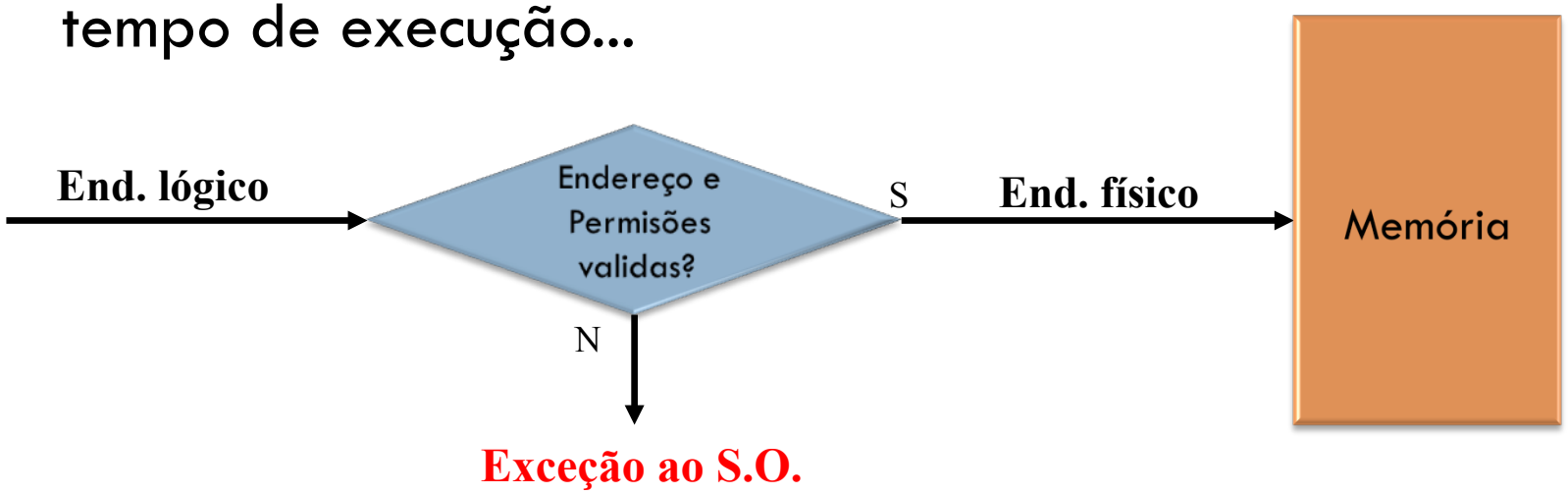
- Relocação **dinâmica** em tempo de **execução**.
  - A correspondência entre o espaço de endereços lógicos e físicos se realiza em tempo de execução.
  - Este sistema permite que o processo, durante sua execução possa mover-se de uma zona de memória à outra.

Memória – Proteção

- Proteção
  - O sistema de gerência de memória deve assegurar a proteção do **código** e dos **dados** dos processos contra acessos **acidentais ou intencionados** de outros processos.
  - Também deve proteger o **código e dados do S.O**.
  - Processos **não** devem poder referenciar posições de memória em outros processos sem permissão prévia.

Memória – Proteção

- Proteção
  - Em virtude da relocação, não é possível testar endereços em programas.
  - Logo, com suporte de ***hardware***, o teste deverá ser em tempo de execução...

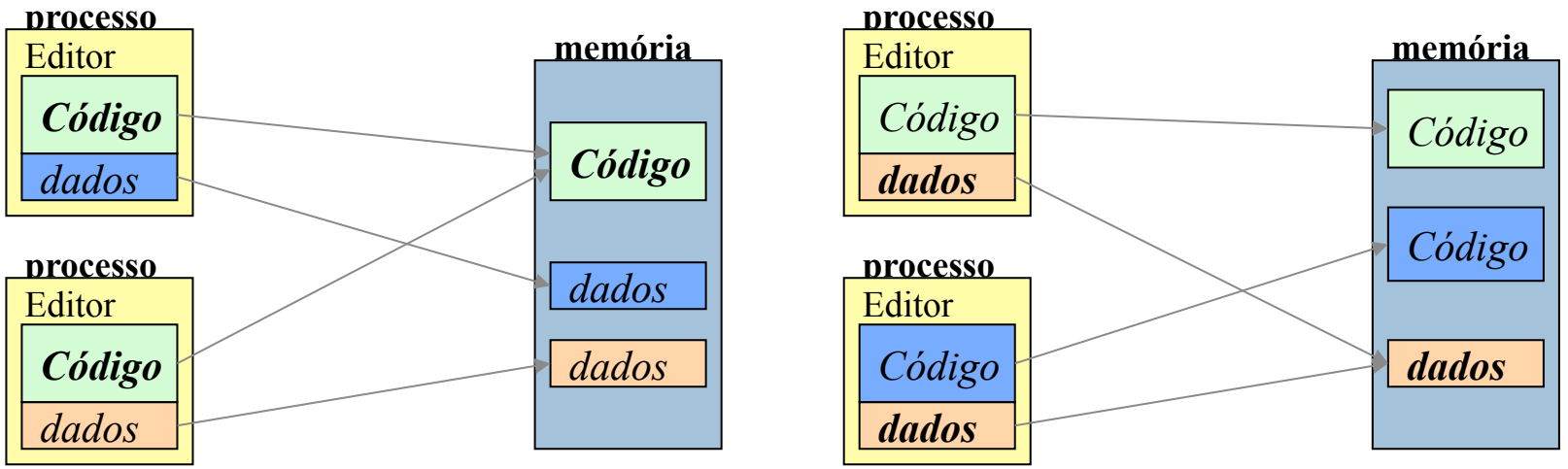


Memória – Compartilhamento

- Compartinhamento:
  - Existem grandes áreas dos programas (código e dados) que podem ser **compartilhados** por diferentes processos para reduzir os requisitos de memória e aumentar a multi-programação do sistema.
    - Deve-se permitir que vários processos acessem a mesma área de memória principal
    - Deve-se permitir o compartilhamento sem comprometer o requisito de proteção

Memória – Compartilhamento

- Proveitos:
  - Processos que são executados em um mesmo programa podem acessar a mesma cópia do programa, e
  - Processos que estão cooperando em uma mesma tarefa podem compartilhar acesso a uma mesma estrutura de dados



Memória – Organização

- A memória **física** de um sistema se organiza como um espaço de endereços linear e unidimencional definido por uma sequência de palavras.
- Entretanto, esta organização não se corresponde à utilizada pelos programas, os quais, em geral, se organizam em módulos (código, dados, pilha, etc.).
- O sistema de gerência de memória deve **suportar** a **organização lógica** requerida pelos programas e garantir a conversão até o **formato** organizacional **real** da memória.

Memória – Organização

- Organização:
  - Física
    - Organizada como uma hierarquia
    - Uso de memória
      - Cache (extremamente rápida, volátil, alto custo)
      - Principal (rápida, volátil, custo mediano)
      - Sencundária (lenta, armazenamento permanente e barata)
    - Gerenciamento deverá ser feito de forma transparente

Memória – Organização

- Organização:
  - Lógica
    - Programas são normalmente separados em módulos, que podem ser escritos e compilados separadamente.
      - referências de um módulo para o outro são resolvidas em tempo de execução;
    - Diferentes graus de proteção (*read only, execute only*) podem ser atribuídos aos diferentes módulos.
    - É possível introduzir mecanismo de compartilhamento entre módulos

Sistemas de Gestão de Memória

Sistemas de Gestão de Memória

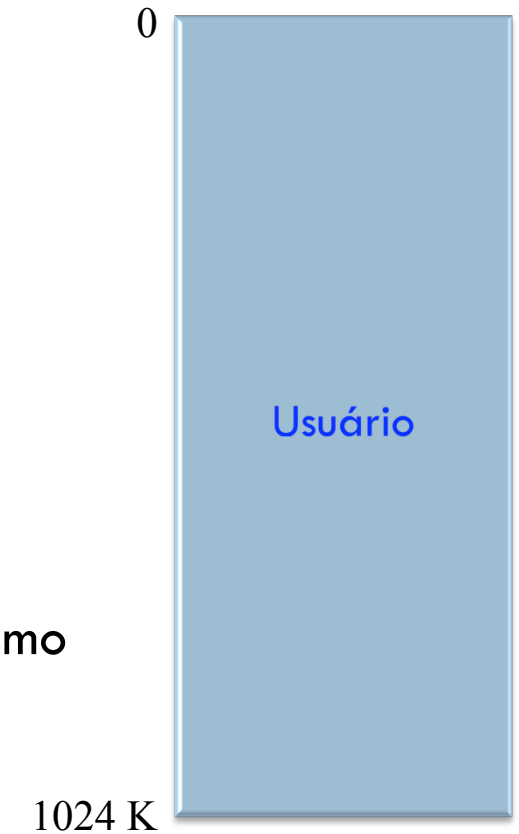
Alocação Contínua

Sistemas de Gestão de Memória

- **Máquina Simples** (Monoprogramação)
  - Não existe um sistema de gestão da memória definido.
  - O usuário tem o controle completo sobre todo o espaço de memória.
  - Espaço de endereçamento lógico é igual ao espaço de endereçamento físico

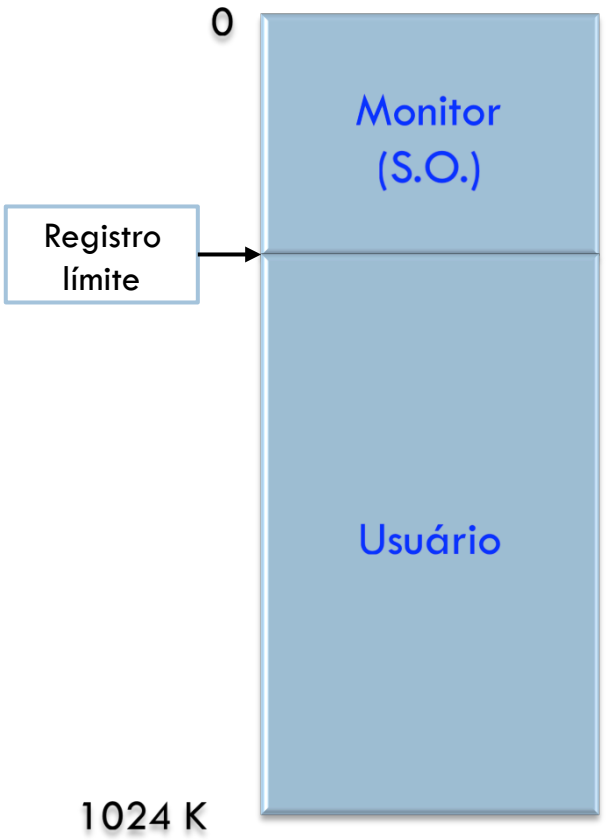
Sistemas de Gestão de Memória

- **Máquina Simples** (Monoprogramação)
- Características:
  - Vantagens:
    - Maior flexibilidade
    - Simplicidade
    - Não requer hardware / software específico
  - Desvantagens:
    - processos e o kernel do S.O. compartilham o mesmo espaço de endereçamento
    - Não existe proteção
    - Pouco eficiente

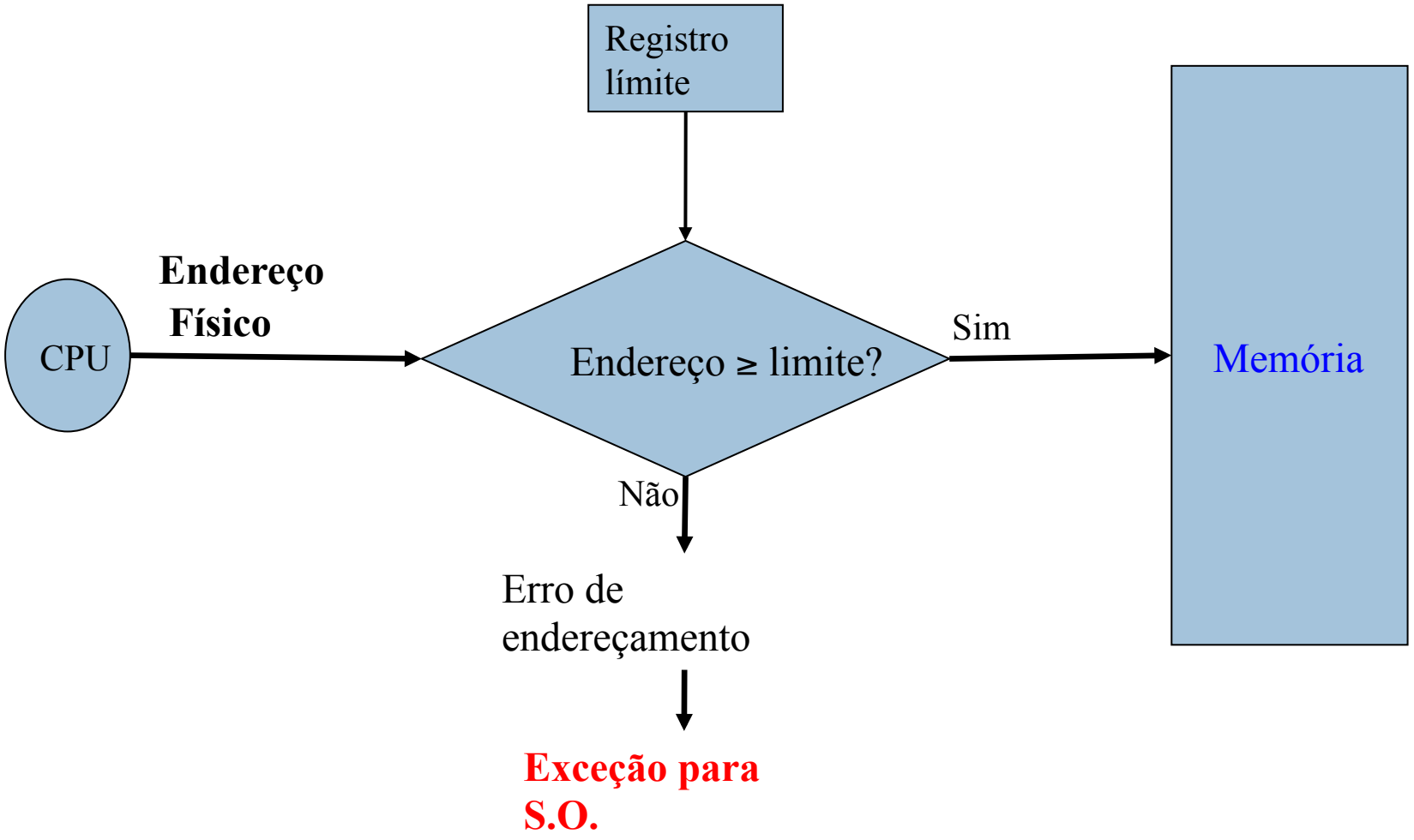


Sistemas de Gestão de Memória

- **Monitor Residente**
  - Divide a memória em duas seções diferentes:
    - Uma área de memória específica para o monitor residente (núcleo) do S.O.
      - Pode estar situado indefinidamente na parte alta ou baixa de memória.
    - Uma zona para o usuário.
      - Apenas um único processo de um usuário
      - O processo de um usuário só pode utilizar endereços de memória que não são referentes ao monitor.



Sistemas de Gestão de Memória

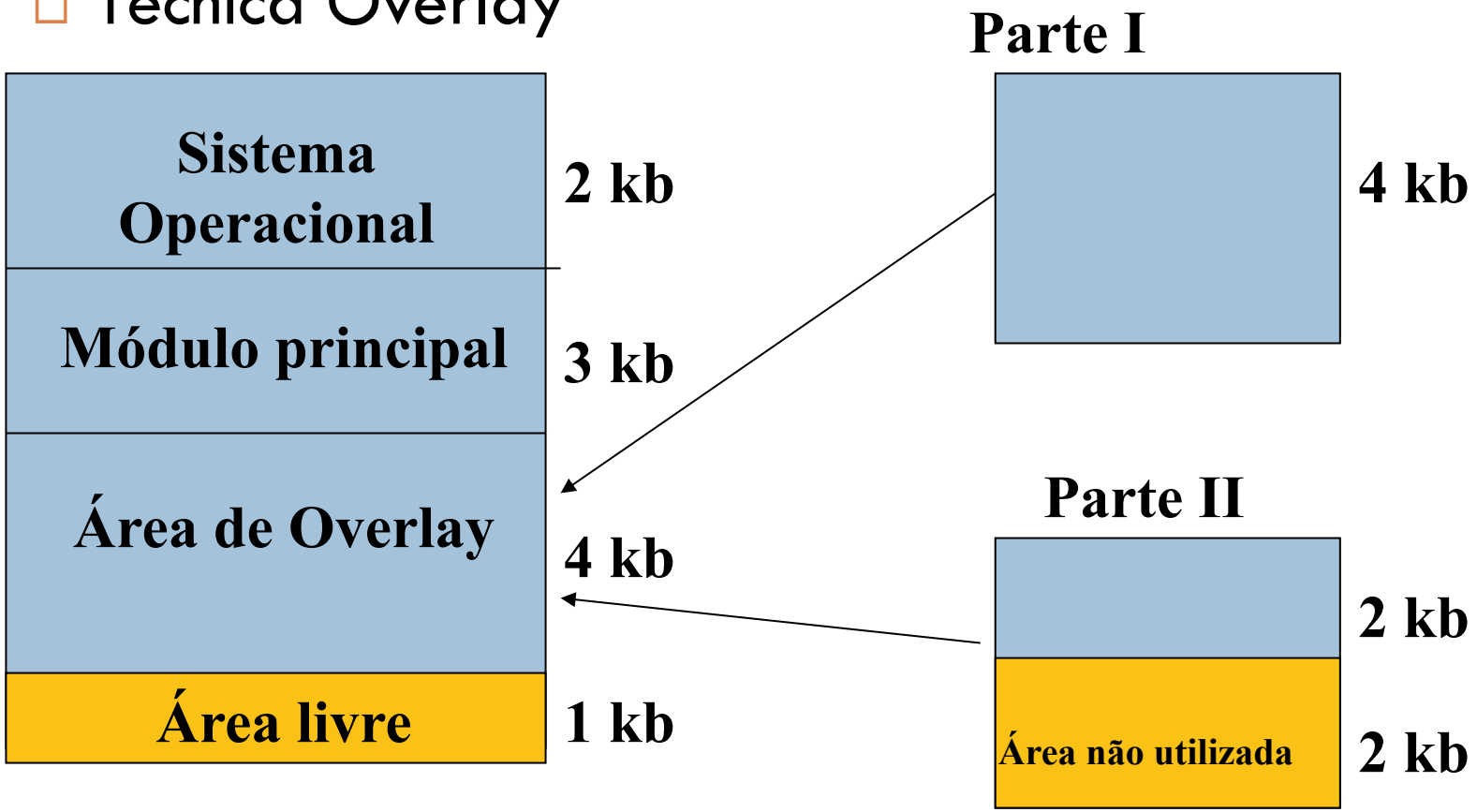


Sistemas de Gestão de Memória

- E se o tamanho do programa for maior do que a memória disponível?
  - Neste caso utiliza-se a técnica conhecida como **overlay**:
    - O programa é dividido em módulos que são executados independentemente na mesma área de memória

Sistemas de Gestão de Memória

- Técnica Overlay



Sistemas de Gestão de Memória

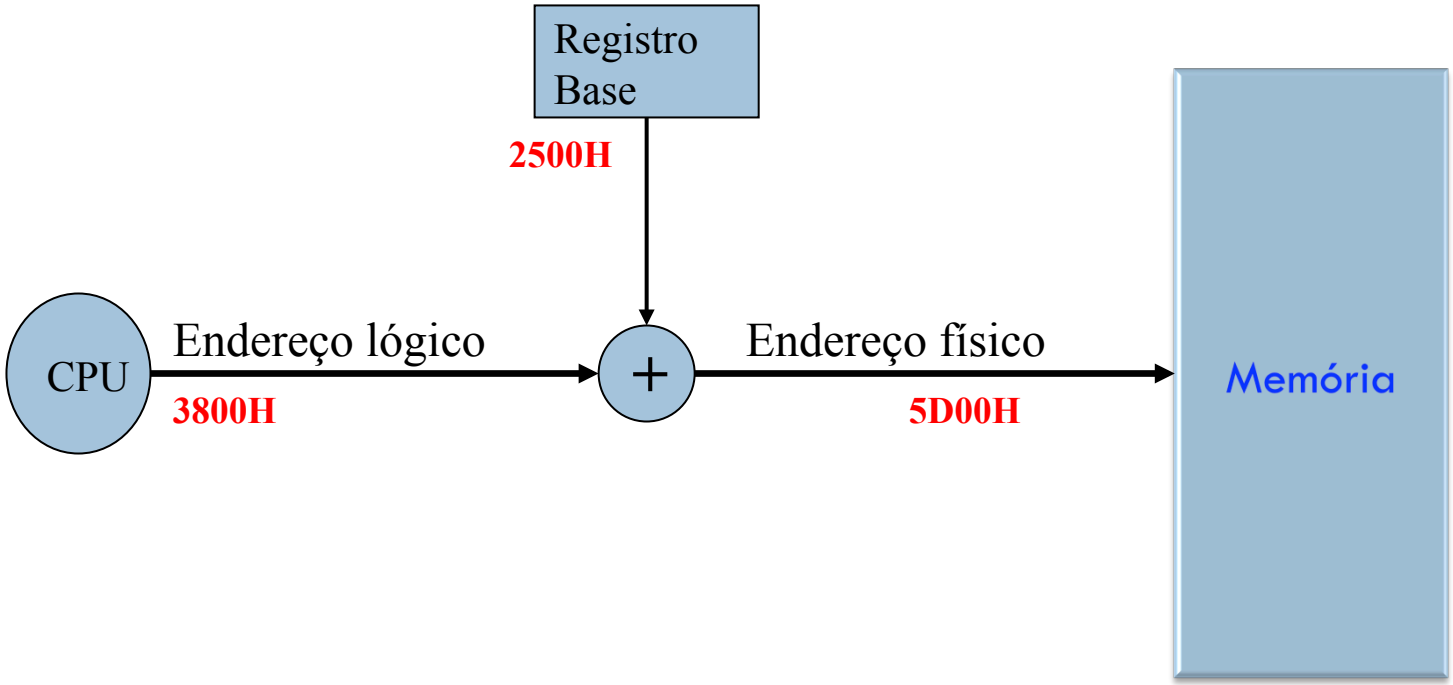
Inconvenientes Monitor Residente

Apesar da utilização das técnicas de swapping para suportar a multi-tarefa este sistema tem dois inconvenientes principais:

- Multiprogramação limitada:
  - Somente é permitido um processo em memória de forma simultânea.
- Ineficiente utilização da memória:
  - Grande parte da memória permanece mal aproveitada, quando os processos de usuário são pequenos.

Sistemas de Gestão de Memória

- Monitor Residente (Relocação)

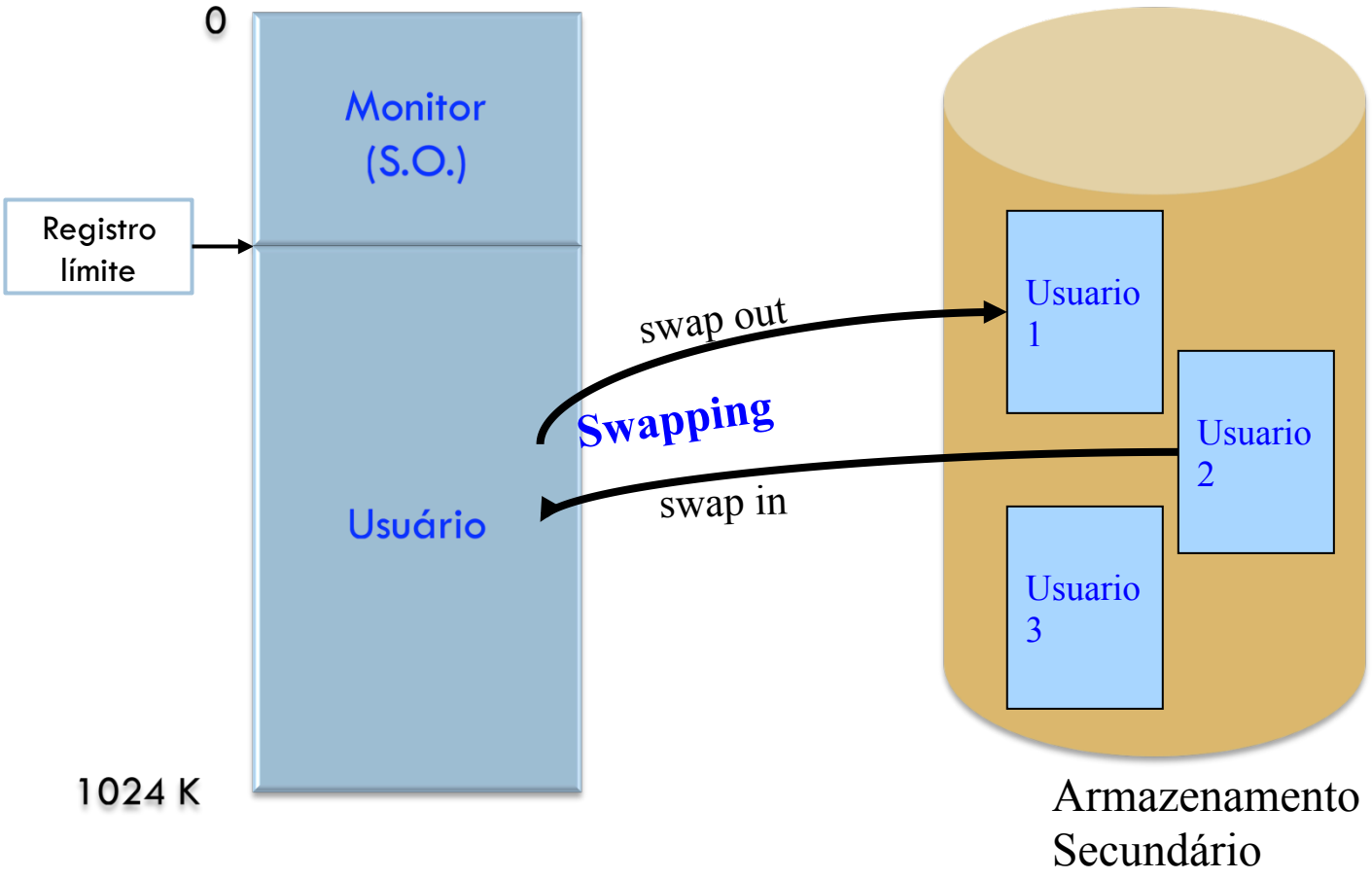


Memória – Swapping

- **Intercambio (Swapping)**
  - Consiste em transferir o código e os dados de um processo completo da memória para o sistema de armazenamento, para dar lugar a outro processo previamente armazenado.
  - Neste mecanismo, cada vez que um processo é retirado da CPU e é gerada Swapping:
    - O processo que está em memória e deixa a CPU é enviado ao disco.
    - O processo selecionado para tomar a CPU e que atualmente se encontra no disco é copiado na memória principal.

Memória – Swapping

- **Swapping:**

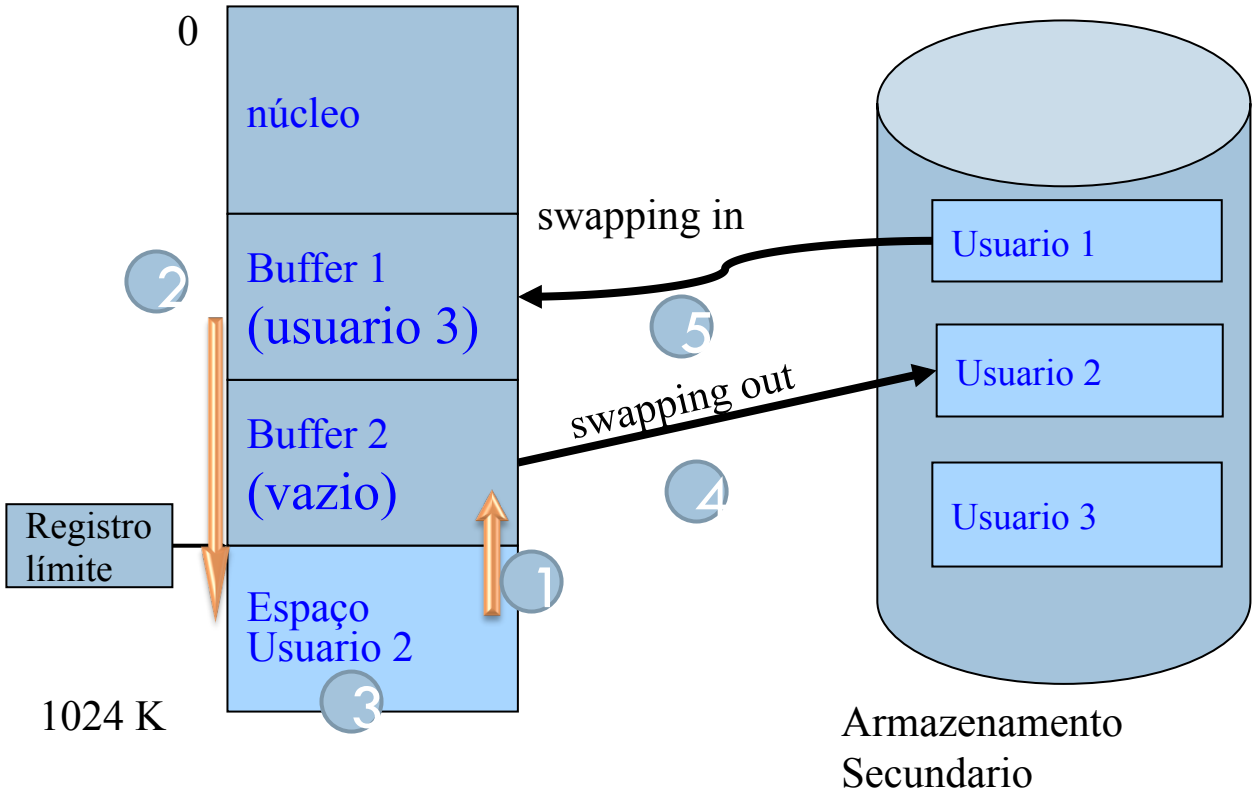


Memória – Swapping

- **Tempo de swapping**
  - O principal inconveniente associado ao sistema de swapping é o aumento do tempo requerido para a troca de contexto dos processos.
  - Para reduzir o tempo de swapping foram propostos:
    - Levar ao disco somente a parte de memória do usuário que não está sendo utilizada.
    - Swapping sobreposto. Dispõe-se dos buffers dentro do espaço de memória do núcleo do SO. Sobrepõe-se o swapping de um processo com a execução de outro, conseguindo que a CPU não fique inativa enquanto se realiza o swapping.

Memória – Swapping

- **Tempo de swapping**



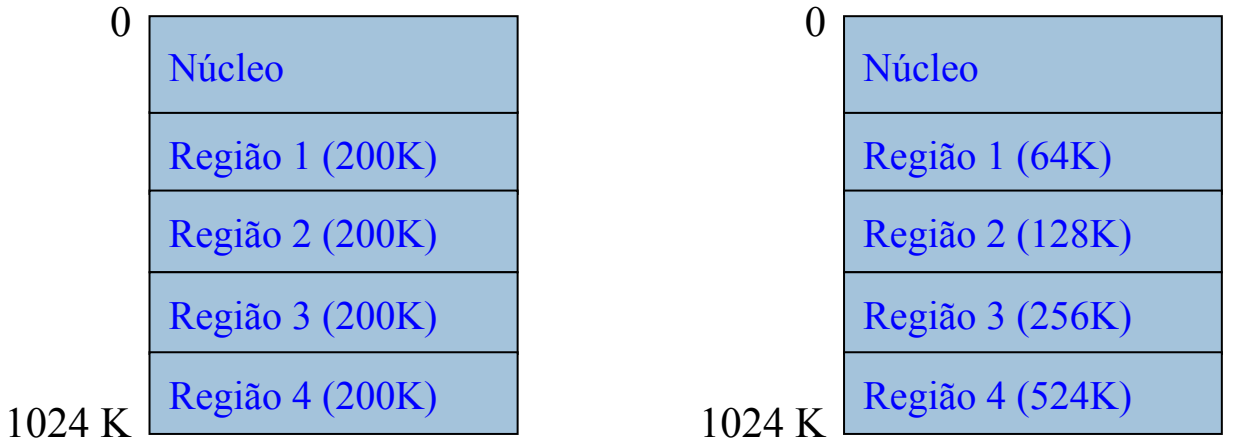
1. A área de memória do usuário 2, que deixa a CPU, é salva no buffer 2
2. A área de memória do usuário 3 é copiada do buffer 1 p/ a área de usuario
3. O processo do usuário 3 contínua com sua execução
4. A área do usuário 2 é envíado ao disco
5. É carregada em memória (buffer1) o seguinte usuário que tomará a CPU

Memória – Múltiplas Partições

- O sistema de gestão de memória é baseado na divisão da memória em um certo número de regiões ou partições.
  - Cada uma das partições pode conter um processo em execução.
  - Quando um processo finaliza sua execução libera sua partição, a qual pode ser utilizada por outro processo da fila de trabalhos.
  - Existem duas vertentes deste sistema:
    - Múltiplas partições de tamanho fixo
    - Múltiplas partições de tamanho variável

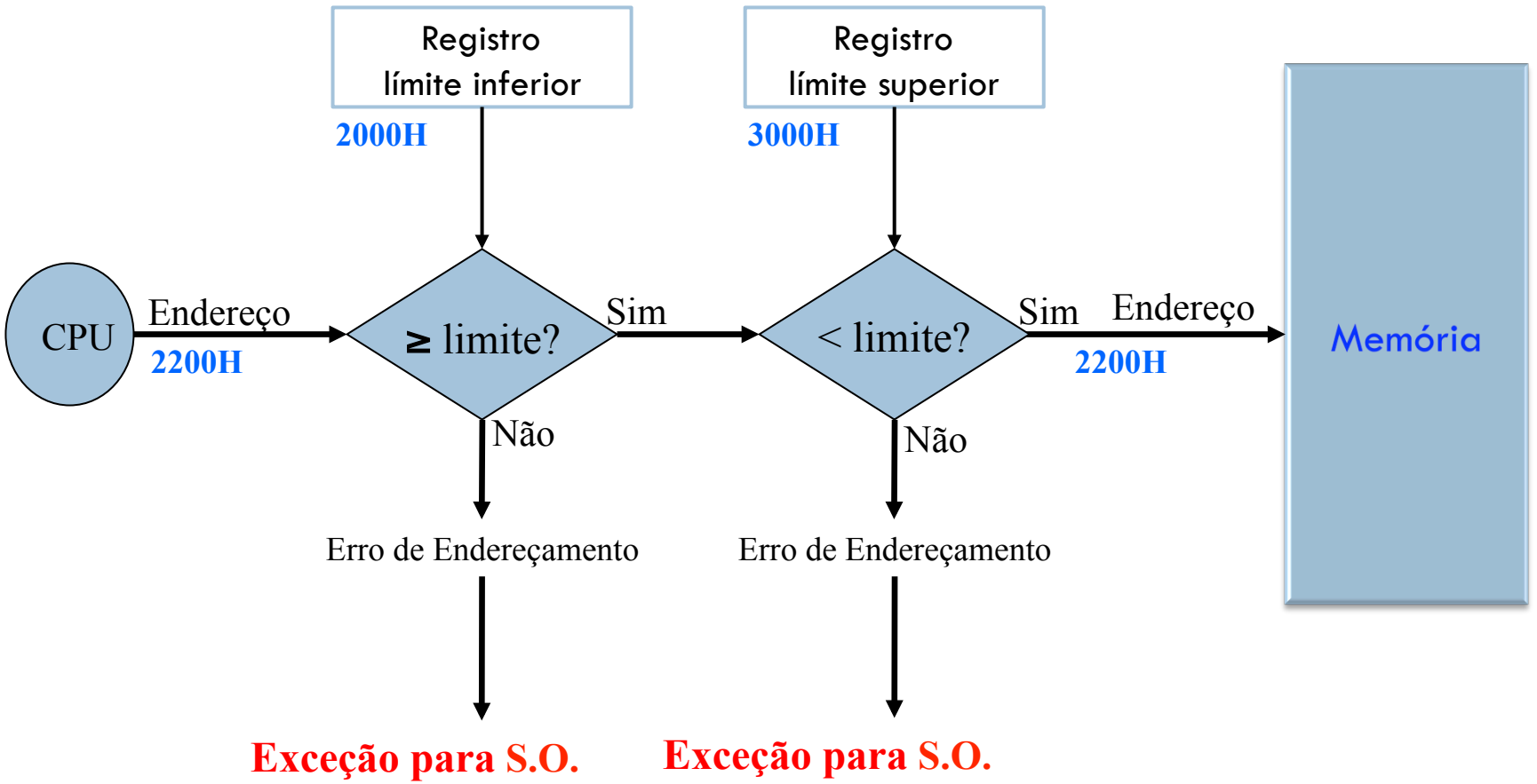
Memória – Múltiplas Partições

- **Partições de tamanho fixo**
  - A área do usuário é dividida em diversas regiões onde o tamanho não varia durante a execução.
  - Pode-se utilizar **tamanhos iguais** para todas as regiões ou regiões com **tamanhos diferentes** que permitam um melhor ajuste aos diferentes requisitos dos processos.



Memória – Múltiplas Partições

- **Hardware de Proteção**



Memória – Múltiplas Partições

- Escalonamento (tamanho fixo =)
- Quando um processo se inicia o escalonador de longo prazo deve decidir qual das partições será utilizada.

- Existem duas políticas básicas para escalonar os processos entre as distintas partições:
  - Sistema baseado em diferentes filas.
  - Sistema baseado em uma única fila.

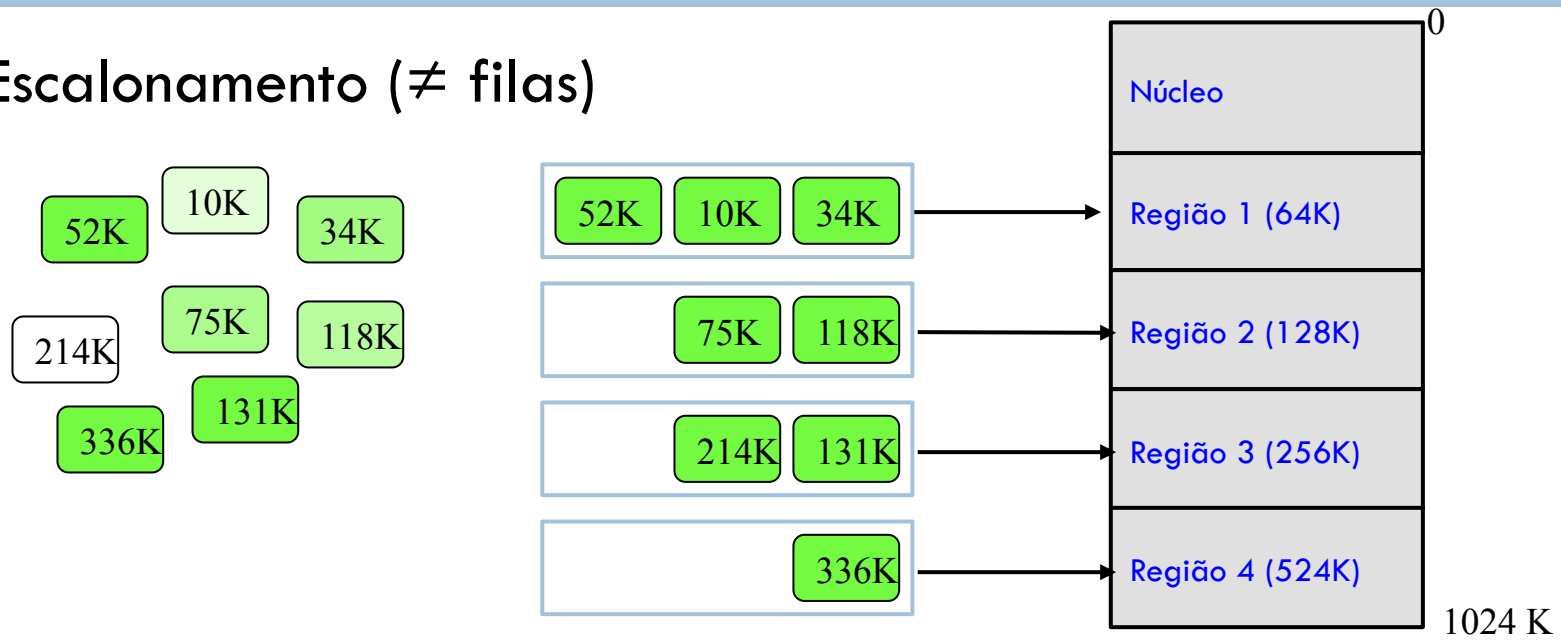
Memória – Múltiplas Partições

- Sistema baseado em diferentes filas.
  - **Cada partição** tem **uma fila** de trabalho associada, os quais não podem utilizar outra partição.
  - A localização dos processos nas filas é feita **em função dos seus requisitos** de memória.

- Sistema baseado em uma única fila.
  - Existe **uma única fila de trabalhos** para todas as partições.
  - Os processos **não estão restritos** a uma única partição.
  - É necessária uma **política** para selecionar a partição.

Memória – Múltiplas Partições

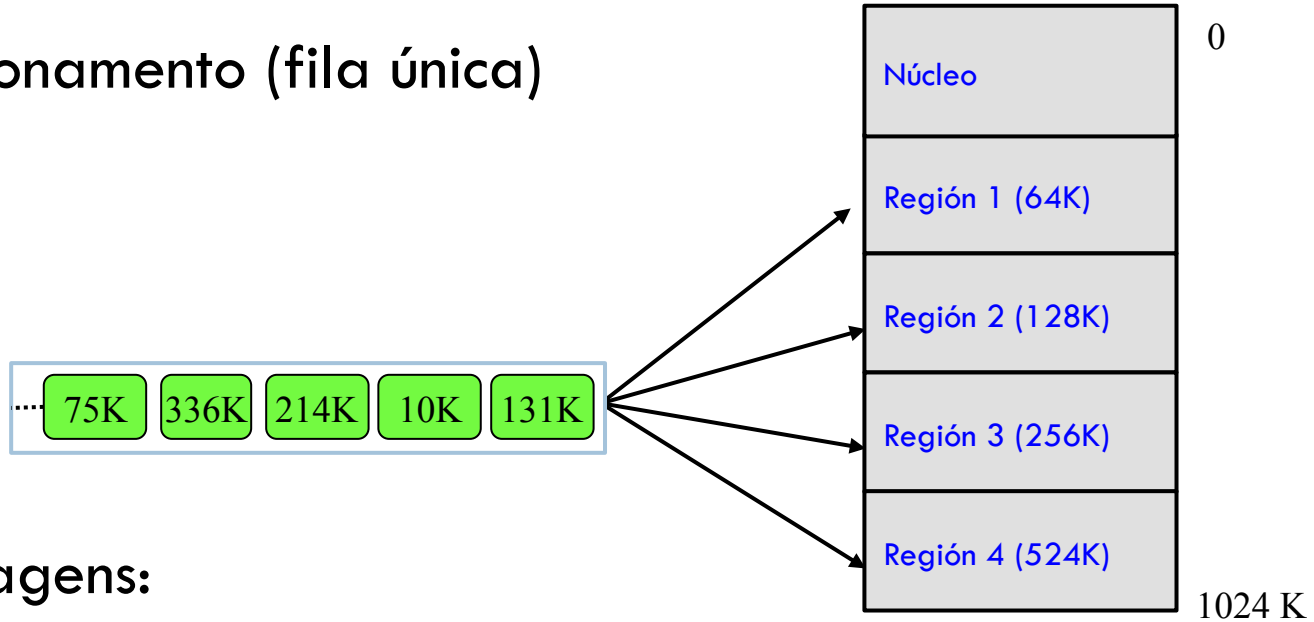
- Escalonamento (≠ filas)



- Desvantagens:
  - Trabalho não balanceado: Determinadas partiçoes podem ficar sem tarefas, enquanto outras tarefas ficam esperando em fila.
- Vantagens:
  - Mais eficiente: Melhor utilização dos recursos de memória
  - Hardware **mais simples:** Só necessita relocação estática em tempo de compilação.

Memória – Múltiplas Partições

- Escalonamento (fila única)



- Vantagens:
  - Maior grau de multi-programação: Permite executar um maior número de processos de forma simultânea
  - Melhor balanceamento de carga.
- Desvantagens:
  - Incrementa a quantidade de memória desaproveitada.
  - Novos Requerimentos: Relocação estática em tempo de execução e políticas de seleção de partição.

Memória – Múltiplas Partições

- **Políticas de seleção de partição:**
  - Primeira partição livre (**First-Fit**).

    É escolhida a primeira partição disponível com espaço suficiente para o processo.
  - Somente a que melhor se adapta (**Best-fit-only**)

    Em função do tamanho do processo é decidido qual a partição que permite minimizar o espaço mal aproveitado (fragmentação). Se está ocupada, espera que fique livre.
  - A partição livre que melhor se adapta (**Best-available-fit**).

    De todas as partições livres, é escolhida a que ocupa menos espaço de memória mal aproveitado.