

SO

Sistemas Operacionais

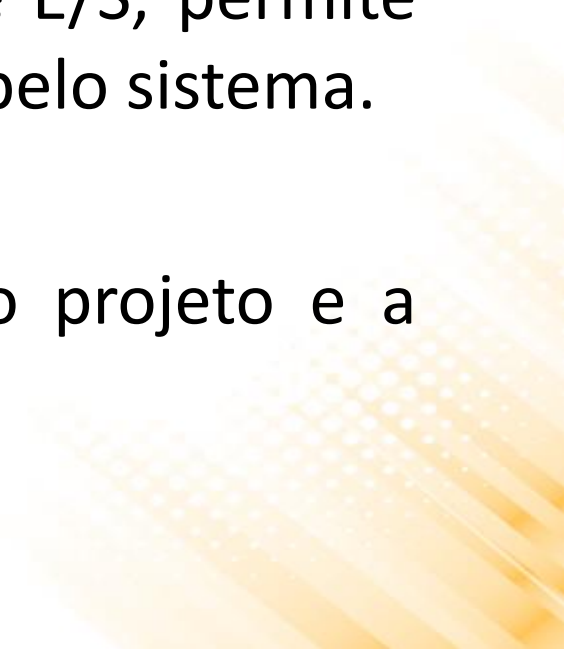
Prof. Rodrigo Martins
rodrigo.martins@francomontoro.com.br



Cronograma da Aula

- ▶ Sistemas Monoprogramáveis X Multiprogramáveis
- ▶ Interrupções e Exceções
- ▶ Operação de Entrada e Saída
- ▶ Buffering
- ▶ Spooling
- ▶ Reentrância

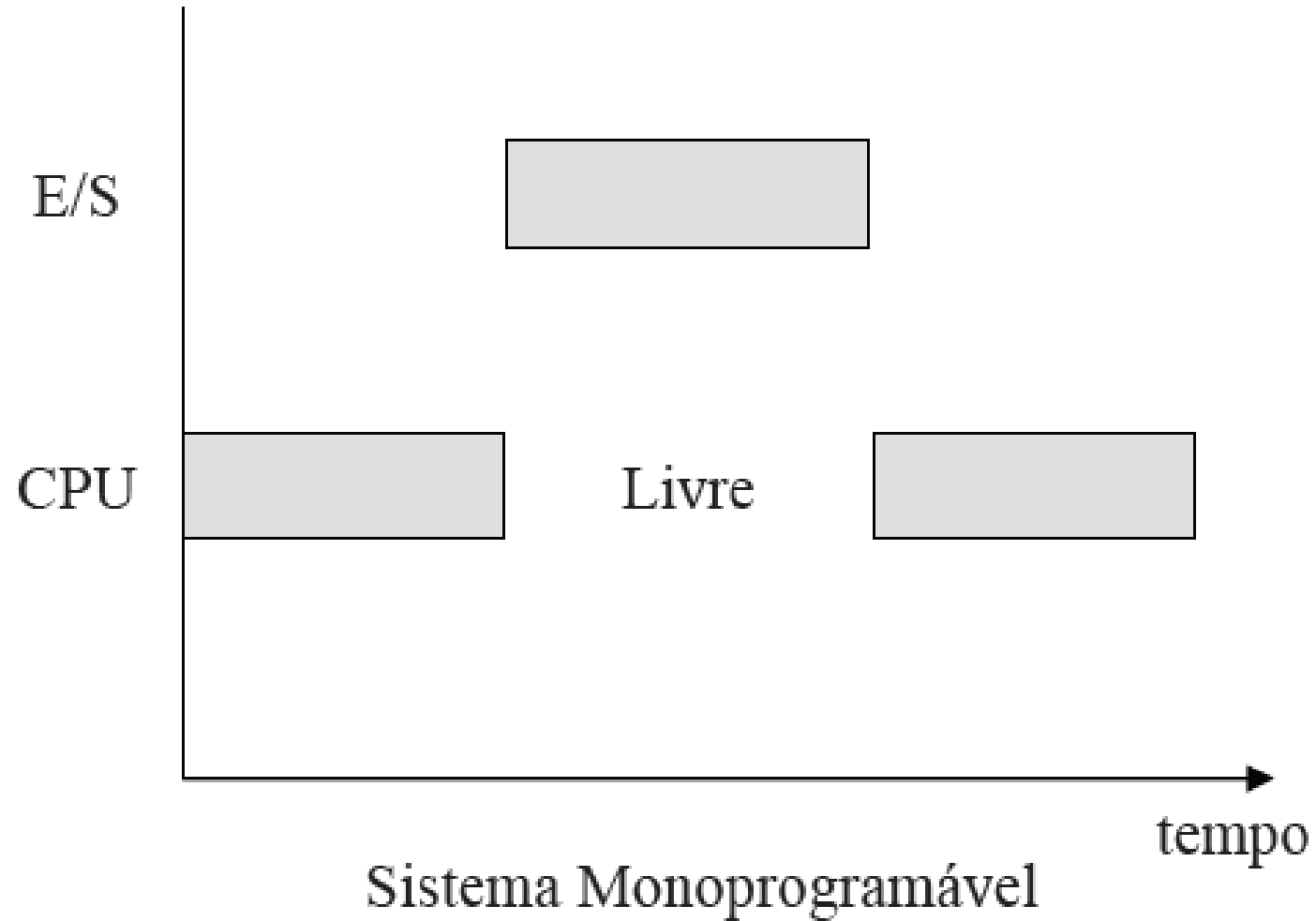
Introdução

- Sistemas operacionais podem ser vistos como um conjunto de rotinas executadas de forma concorrente e ordenada (Pinkert, 1990).
 - A possibilidade de o processador executar instruções ao mesmo tempo que outras operações, como, por exemplo, operações de E/S, permite que diversas tarefas sejam executadas concorrentemente pelo sistema.
 - O conceito de concorrência é o princípio básico para o projeto e a implementação dos sistemas multiprogramáveis.
- 

Monoprogramáveis/Monotarefa

- Sistemas Monoprogramáveis/Monotarefa
 - Voltados tipicamente para a execução de um único programa.
 - Qualquer outra aplicação, para ser executada, deveria aguardar o término do programa corrente.
 - Processador, a memória e os periféricos permanecem exclusivamente dedicados à execução de um único programa.
 - Exemplo: MS-DOS.

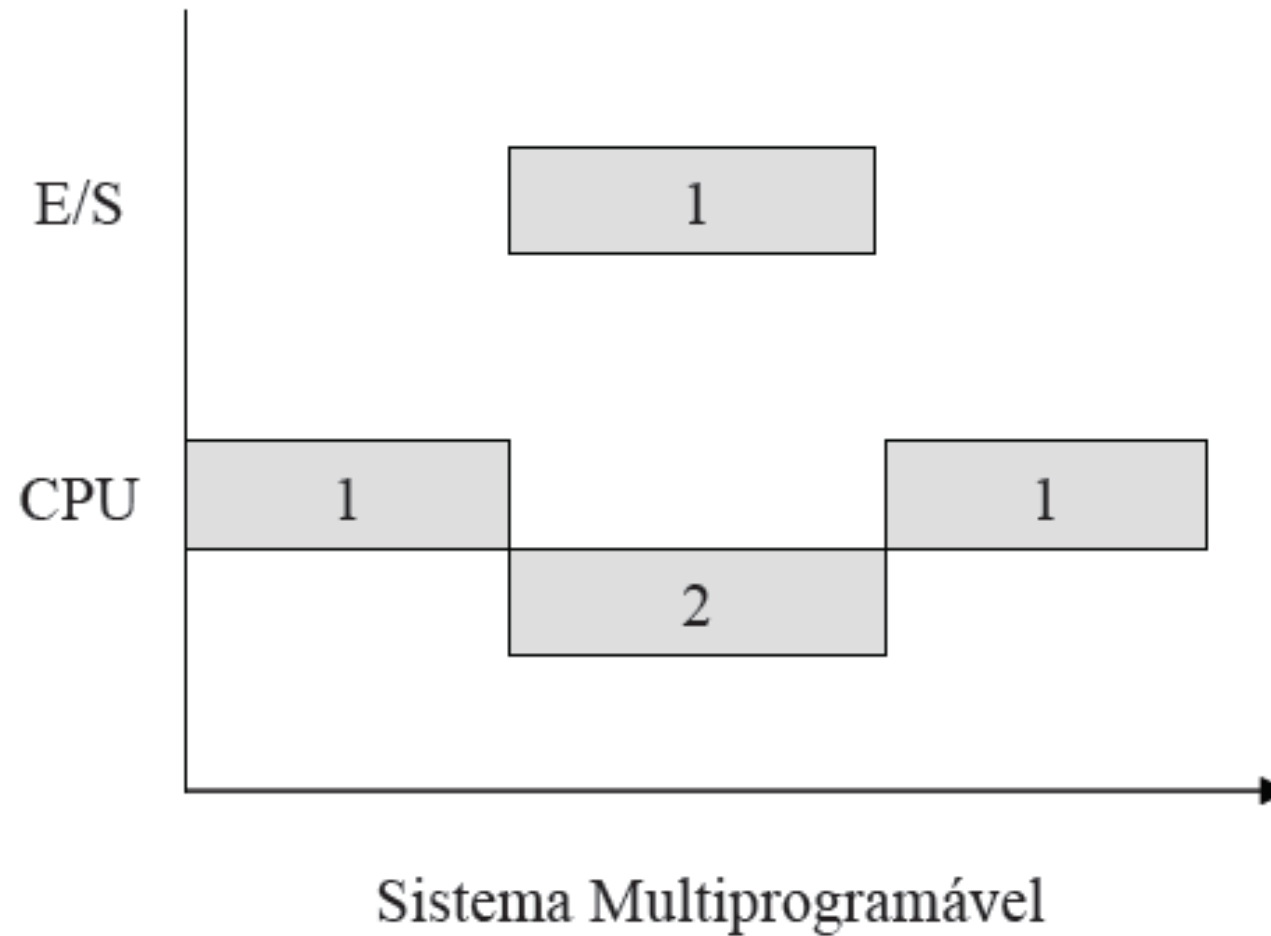
Tempo...




Sistemas Multiprogramáveis/Multitarefa

- Sistemas Multiprogramáveis/Multitarefa
 - Recursos compartilhados entre os diversas aplicações: enquanto um programa espera por um evento, outros programas podem estar processando neste mesmo intervalo de tempo.
 - Neste caso, podemos observar o compartilhamento da memória e do processador.
 - O sistema operacional se incumba de gerenciar o acesso concorrente aos seus diversos recursos, como processador, memória e periféricos, de forma ordenada e protegida, entre os diversos programas.
 - Exemplo: Windows.

Tempo...



Interrupções e Exceções

- Durante a execução de um programa podem ocorrer alguns eventos inesperados, ocasionando um desvio forçado no seu fluxo de execução.
 - Estes tipos de eventos são conhecidos por **interrupção ou exceção** e podem ser consequência da sinalização de algum dispositivo de hardware externo ao processador ou da execução de instruções do próprio programa.
- 

Interrupções e Exceções

- A **interrupção** é o mecanismo que tornou possível a implementação da concorrência nos computadores, sendo o fundamento básico dos **sistemas multiprogramáveis**.
- É em função desse mecanismo que o sistema operacional sincroniza a execução de todas as suas rotinas e dos programas dos usuários, além de controlar dispositivos.
- Um exemplo de interrupção ocorre quando um dispositivo avisa ao processador que alguma operação de E/S está completa. Nesse caso, o processador deve interromper o programa para tratar o término da operação.

Interrupções e Exceções



Fig. 3.2 Mecanismos de interrupção e exceção.


Interrupções e Exceções

- A Tabela descreve como é o mecanismo de interrupção. Este mecanismo é realizado tanto por hardware quanto por software.

Tabela 3.4 Mecanismo de interrupção

Via hardware	<ol style="list-style-type: none">1. Um sinal de interrupção é gerado para o processador.2. Após o término da execução da instrução corrente, o processador identifica o pedido de interrupção.3. Os conteúdos dos registradores PC e de status são salvos.4. O processador identifica qual a rotina de tratamento que será executada e carrega o PC com o endereço inicial desta rotina.
Via software	<ol style="list-style-type: none">5. A rotina de tratamento salva o conteúdo dos demais registradores do processador na pilha de controle do programa.6. A rotina de tratamento é executada.7. Após o término da execução da rotina de tratamento, os registradores de uso geral são restaurados, além do registrador de status e o PC, retornando à execução do programa interrompido.

Interrupções e Exceções

- Uma **exceção** é semelhante a uma interrupção, sendo a principal diferença o motivo pelo qual o evento é gerado.
 - A **exceção** é resultado direto da execução de uma instrução do próprio programa, como a divisão de um número por zero ou a ocorrência de overflow em uma operação aritmética.
- 

Interrupções e Exceções

- A diferença fundamental entre exceção e interrupção é que a primeira é gerada por um evento síncrono, enquanto a segunda é gerada por eventos assíncronos.
- Um evento é síncrono quando é resultado direto da execução do programa corrente. Tais eventos são previsíveis e, por definição, só podem ocorrer um de cada vez.
- Se um programa que causa esse tipo de evento for reexecutado com a mesma entrada de dados, a exceção ocorrerá sempre na mesma instrução.

Interrupções e Exceções

- Da mesma forma que na interrupção, sempre que uma exceção é gerada o programa em execução é interrompido e o controle é desviado para uma rotina de tratamento de exceção.



Fig. 3.2 Mecanismos de interrupção e exceção.

Operações de Entrada/Saída

- Nos primeiros sistemas computacionais, a comunicação entre o processador e os periféricos era controlada por um conjunto de instruções especiais, denominadas **instruções de entrada/saída**, executadas pelo próprio processador.
- Essas instruções continham detalhes específicos de cada periférico, como em qual trilha e setor de um disco deveria ser lido ou gravado um determinado bloco de dados. Esse modelo criava uma forte dependência entre o processador e os dispositivos de E/S.

Operações de Entrada/Saída

- O surgimento do controlador ou interface permitiu ao processador agir de maneira independente dos dispositivos de E/S. Com esse novo elemento, o processador não se comunicava mais diretamente com os periféricos, mas sim através do controlador.

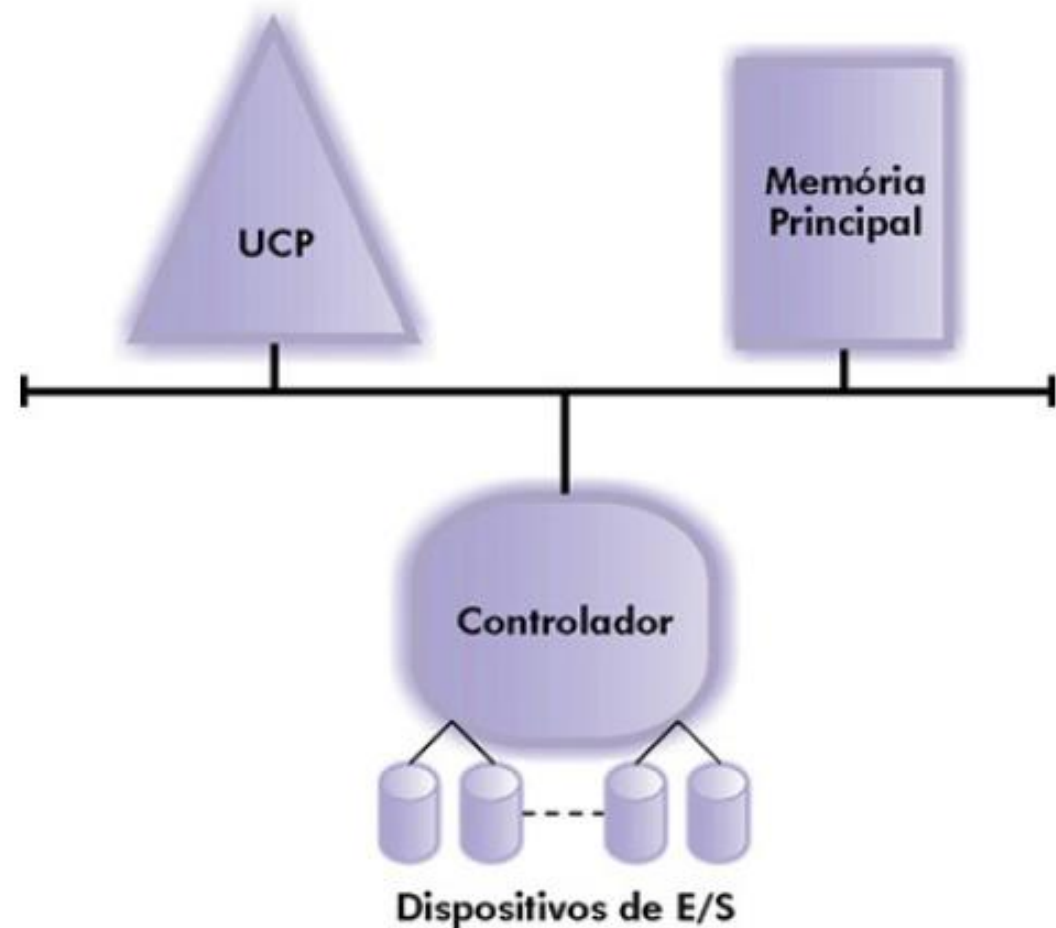



Fig. 3.3 Controlador.

Operações de Entrada/Saída

- A evolução do modelo anterior foi permitir que, após o início da transferência dos dados, o processador permanecesse livre para realizar outras tarefas.
 - Em determinados intervalos de tempo o sistema operacional deveria testar cada dispositivo para saber do término da operação de E/S (polling).
- 

Operações de Entrada/Saída

- Isso permitiu o surgimento dos primeiros sistemas multiprogramáveis, em que vários programas poderiam ser executados concorrentemente, já que o tempo de uma operação de E/S é relativamente grande.
- O problema dessa implementação é que, no caso de existir um grande número de periféricos, o processamento é interrompido frequentemente para testar os diversos periféricos, já que é difícil determinar o momento do término das operações de E/S em andamento.

Operações de Entrada/Saída

- Com a implementação do mecanismo de interrupção, as operações de E/S puderam ser realizadas de uma forma mais eficiente.
- Em vez de o sistema periodicamente verificar o estado de uma operação pendente, o próprio controlador interrompia o processador para avisar do término da operação.
- Com esse mecanismo, denominado **E/S controlada por interrupção**, o processador, após a execução de um comando de leitura ou gravação, permanece livre para o processamento de outras tarefas.

Operações de Entrada/Saída

- A **operação de E/S controlada por interrupção** é muito mais eficiente que a controlada por programa, já que elimina a necessidade de o processador esperar pelo término da operação, além de permitir que várias operações de E/S sejam executadas simultaneamente.
- Apesar disso, a transferência de grande volume de dados exige muitas intervenções do processador, reduzindo sua eficiência.
- A solução para esse problema foi a implementação de uma técnica de transferência de dados denominada **DMA (Direct Memory Access)**.

Operações de Entrada/Saída

- A técnica de DMA permite que um bloco de dados seja transferido entre a memória principal e dispositivos de E/S sem a intervenção do processador, exceto no início e no final da transferência.
- Quando o sistema deseja ler ou gravar um bloco de dados, o processador informa ao controlador sua localização, o dispositivo de E/S, a posição inicial da memória de onde os dados serão lidos ou gravados e o tamanho do bloco.
- Com estas informações, o controlador realiza a transferência entre o periférico e a memória principal, e o processador é somente interrompido no final da operação. A área de memória utilizada pelo controlador na técnica de DMA é chamada de **buffer de entrada e saída**.

Operações de Entrada/Saída

- O canal de E/S realiza a transferência e, ao final, gera uma interrupção, avisando do término da operação. Um canal de E/S pode controlar múltiplos dispositivos através de diversos controladores.

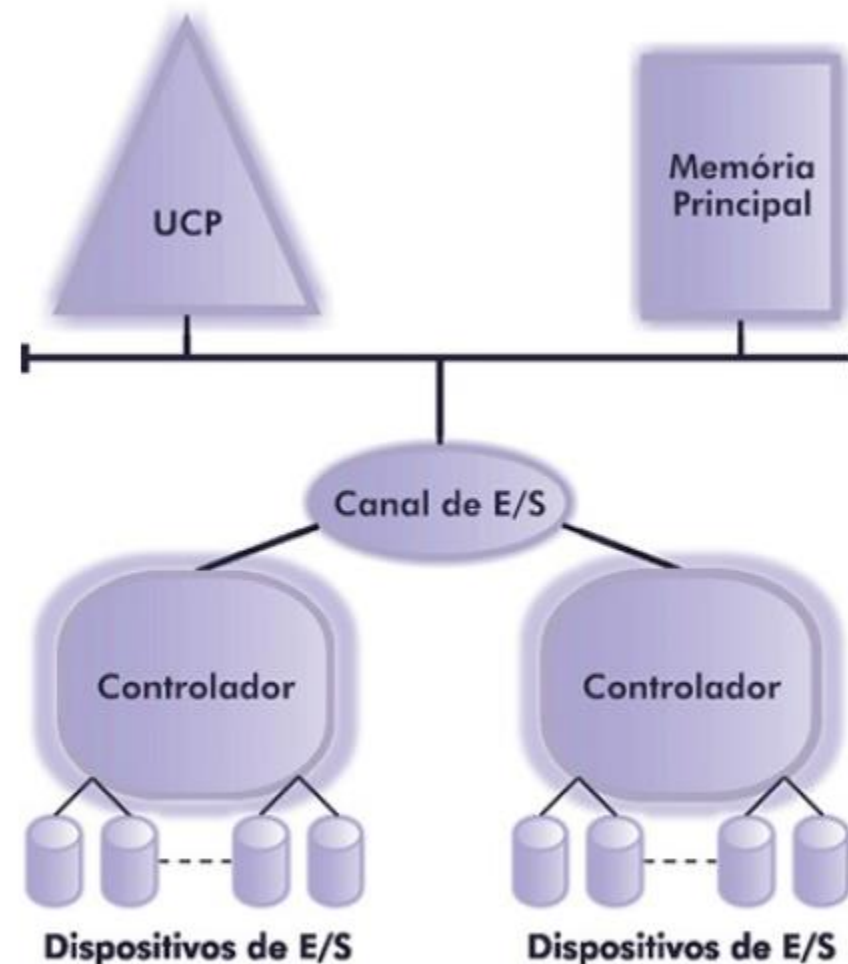



Fig. 3.4 Canal de E/S.

Buffering

- A **técnica de buffering** consiste na utilização de uma área na memória principal, denominada buffer, para a transferência de dados entre os dispositivos de E/S e a memória.
 - Esta técnica permite que em uma operação de leitura o dado seja transferido primeiramente para o buffer, liberando imediatamente o dispositivo de entrada para realizar uma nova leitura.
- 

Buffering

- Enquanto o processador manipula o dado localizado no buffer, o dispositivo realiza outra operação de leitura no mesmo instante.

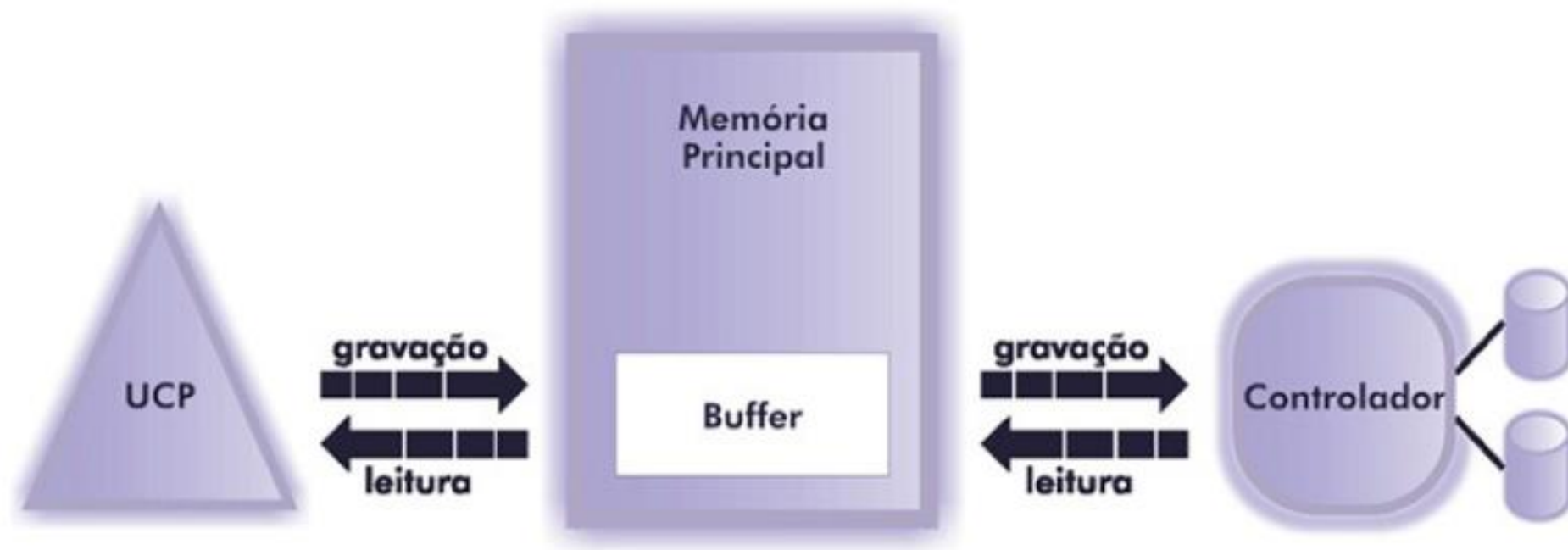


Fig. 3.5 Operações de E/S utilizando buffer.

Spooling

- A **técnica de spooling**, semelhante à **técnica de buffering**, utiliza uma área em disco como se fosse um grande buffer.
- Neste caso, dados podem ser lidos ou gravados em disco, enquanto programas são executados concorrentemente.
- Atualmente essa técnica está presente na maioria dos sistemas operacionais, sendo utilizada no gerenciamento de impressão. No momento em que um comando de impressão é executado, as informações que serão impressas são gravadas antes em um arquivo em disco, conhecido como arquivo de spool, liberando imediatamente o programa para outras atividades.

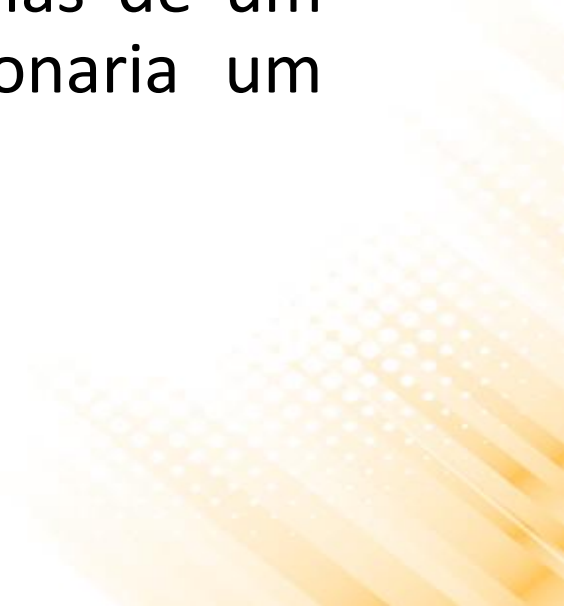
Spooling

- O uso do spooling permite desvincular o programa do dispositivo de impressão, impedindo que um programa reserve a impressora para uso exclusivo. O sistema operacional é o responsável por gerenciar a sequência de impressões solicitadas pelos programas, seguindo critérios que garantam a segurança e o uso eficiente das impressoras.



Fig. 3.6 Técnica de spooling.

Reentrância

- É comum, em sistemas multiprogramáveis, vários usuários utilizarem os mesmos aplicativos simultaneamente, como editores de textos e compiladores.
 - Se cada usuário que utilizasse um desses aplicativos trouxesse o código executável para a memória, haveria diversas cópias de um mesmo programa na memória principal, o que ocasionaria um desperdício de espaço.
- 

Reentrância

- **Reentrância** é a capacidade de um código executável (código reentrante) ser compartilhado por diversos usuários, exigindo que apenas uma cópia do programa esteja na memória.
- A **reentrância** permite que cada usuário possa estar em um ponto diferente do código reentrante, manipulando dados próprios, exclusivos de cada usuário.

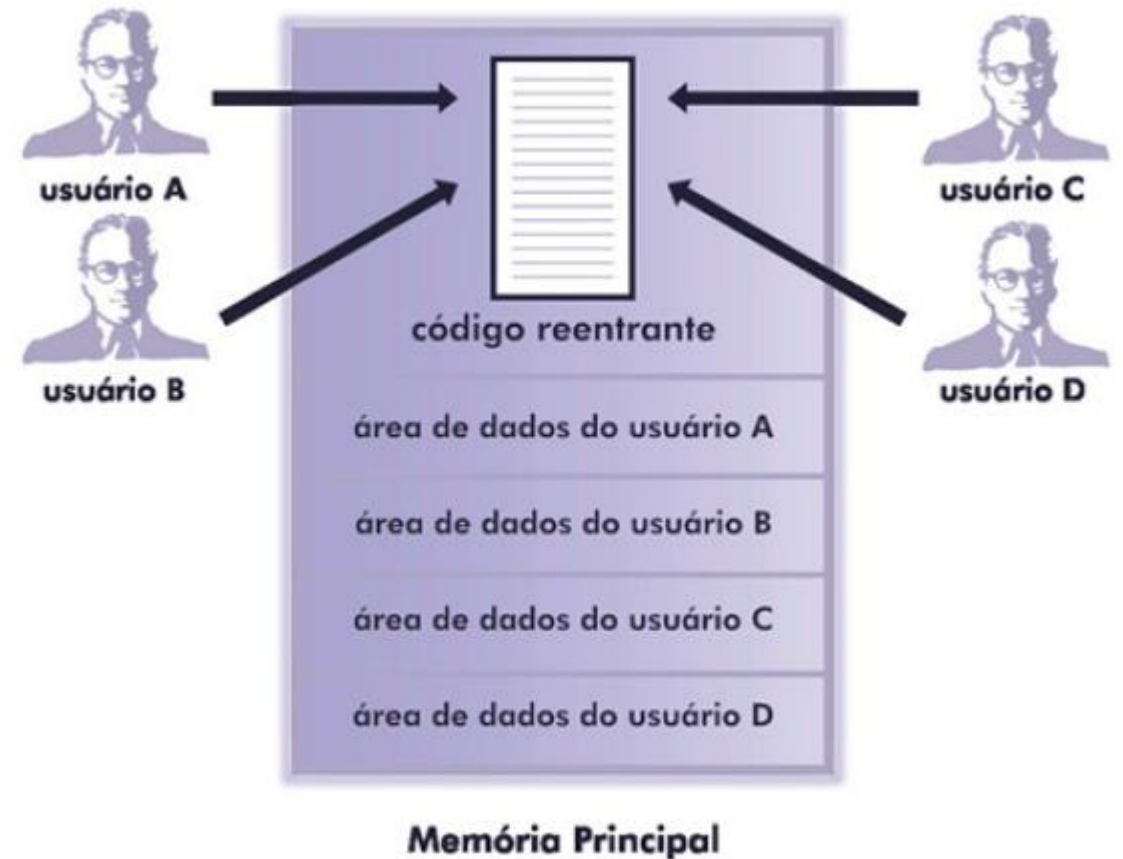
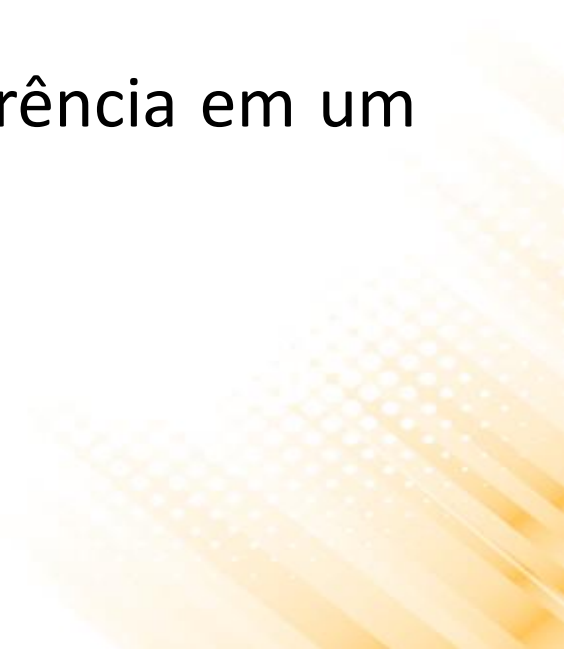


Fig. 3.7 Reentrância.

Exercícios

- 1) O que é concorrência e como este conceito está presente nos sistemas operacionais multiprogramáveis?
 - 2) Por que o mecanismo de interrupção é fundamental para a implementação da multiprogramação?
 - 3) O que é DMA e qual a vantagem desta técnica?
 - 4) Como a técnica de buffering permite aumentar a concorrência em um sistema computacional?
 - 5) Explique o mecanismo de spooling de impressão.
- 

Referências desta Aula

- TANENBAUM, A. S. Sistemas Operacionais Modernos, Prentice-Hall do Brasil, 4ª edição, 2016.
- MACHADO F. B., MAIA L. P. Arquitetura de Sistemas Operacionais, LTC, 5ª edição, 2014.

Fim

Obrigado

Rodrigo