Administração de Dispositivos

Aula 12 – Técnicas

Canais e Unidades de Controle

- **Canal** é um processador de uso especializado, que executa apenas **programas de canal** (sequências de instruções chamadas "***Channel Control Words***" que instruem o canal acerca das ações a serem executadas, visando efetuar as operações de entrada/saída desejadas).

- Normalmente, há mais dispositivos que canais: tipicamente oito controladoras ("***control-units***") para cada canal e oito dispositivos para cada "*control-unit*“.
- Assim, os equipamentos menos numerosos deverão ser partilhados (multiplexados no espaço e/ou no tempo) entre os mais numerosos.

- Durante o processamento de um programa, tipicamente poucos dispositivos permanecem ativos simultaneamente.
- Por isso, razões econômicas recomendam separar a parte eletrônica da parte mecânica desses dispositivos, usando-se uma **unidade de controle** ("*control unit*"), para multiplexar a parte comum (mais cara) dos circuitos entre os diversos dispositivos.

- Os administradores de dispositivos destinam-se a **acompanhar o estado** dos canais, das unidades de controle e dos dispositivos, afim de que seja possível detectar e resolver eventuais conflitos que possam surgir na ocasião da utilização dos mesmos.

- Algumas **técnicas de hardware**, que passamos a descrever, são utilizadas para eliminar o gargalo causado pela escassez de canais e de controladores, frente ao número mais elevado de dispositivos.

Independência na operação dos dispositivos

- Os dispositivos podem possuir recursos próprios de hardware para executar certas tarefas sem a intervenção do canal, e até sem a intervenção do controlador.
- Por exemplo, uma operação de "***seek***" no disco, uma operação de "***rewind***" na fita magnética.
- Naturalmente, tais operações devem provocar, ao seu final, uma **interrupção específica**, informando ao solicitante que ela já foi terminada.
- Assim, dependendo do caso, ocorrem interrupções de "***Device end***", "***Control Unit end***", e "***Channel end***", com códigos associados, para efeito de identificação.

Utilização de técnica de "*buffering*"

- Através da inclusão de memórias de alta velocidade nos controladores ou nos dispositivos, descongestiona-se a via de acesso à memória principal, pois desta maneira esta via só é utilizada quando conveniente, e não quando o caráter síncrono de alguns dispositivos o exigir.
- Assim, economicamente, a via de **acesso à memória** pode ser **compartilhada** pelos vários canais, pelo controlador, e pelos dispositivos, aumentando a eficiência do sistema.
- No seu modo mais simples, o mecanismo de "*buffering*" restringe-se à **incorporação de memórias rápidas** no hardware ligado aos dispositivos (leitoras, impressoras), desvinculando-os, no tempo, da via de acesso à memória.

Uso de caminhos alternativos para dados

- Isto se consegue promovendo-se, no hardware, a existência de **caminhos alternativos** entre os módulos que compõem o sistema de E/S
- **Não vincula** um dispositivo a nenhum controlador, nem um controlador a qualquer canal.

- Havendo uma solicitação de E/S para um dispositivo, **o sistema operacional escolhe**, para a realização da transferência, um dos múltiplos caminhos disponíveis, conforme a conveniência do sistema, na rara situação em que existam, fisicamente presentes e livres, muitos possíveis caminhos que possam estabelecer as conexões desejadas entre os módulos.

Redundância de rotas

- Um número suficiente de alternativas deve estar disponível para que, embora alguns dos caminhos eventualmente venham a ter seu funcionamento prejudicado por alguma falha, o sistema continue sendo capaz de efetuar corretamente a transferência de dados, de maneira **transparente ao programador**, fazendo uso para isso apenas das rotas alternativas restantes.

Técnica da multiplexação de blocos

- Tipicamente, em um sistema com "***block multiplexor channel***" até oito **programas de canal** podem ser **simultaneamente executados**.
- Esta solução implementa, em hardware, uma versão da técnica de **multiprogramação,** que se emprega no gerenciamento de processos.

- Como já foi citado anteriormente, o administrador de dispositivos pode ser funcionalmente decomposto em:
  - ***I/O scheduler***
  - ***I/O traffic controller***
  - ***I/O device handler***
- Detalha-se a seguir cada um desses módulos.

I/O traffic controller

- **Acompanha o estado** dos dispositivos, mantendo tabelas de status dos mesmos.
- Decide se o dispositivo requisitado **pode ser alocado** a um processo num certo instante ou não.
- Devido ao número reduzido de canais e de controladores, e das múltiplas opções de interligação entre os módulos de entrada/saída, a operação e a construção deste programa podem tornar-se bastante complexos.

- Para tanto, o ***I/O traffic controller*** deve responder basicamente a três perguntas:
  - **Há algum caminho disponível**, para que a solicitação de Entrada/Saída recebida possa ser atendida?
  - Há **mais de um** caminho nessas condições?
  - Se não há caminhos disponíveis, **quando ou em que condições** o pedido poderá ser atendido?
- Para isto, é preciso que seja mantida uma estrutura de informações que reflita o status dos dispositivos e as conexões entre os diversos módulos do sistema.

UCB (*unit control blocks*)

- Sua função básica é a de proporcionar meios para a modelagem dos **dispositivos**.
- Cada **dispositivo** é representado por um **UCB**.
- Cada UCB apresenta os seguintes dados:
  - **identificação** do dispositivo.
  - ***status*** do dispositivo.
  - lista de **unidades de controle ligadas** ao dispositivo.
  - lista de **processos aguardando** o dispositivo.

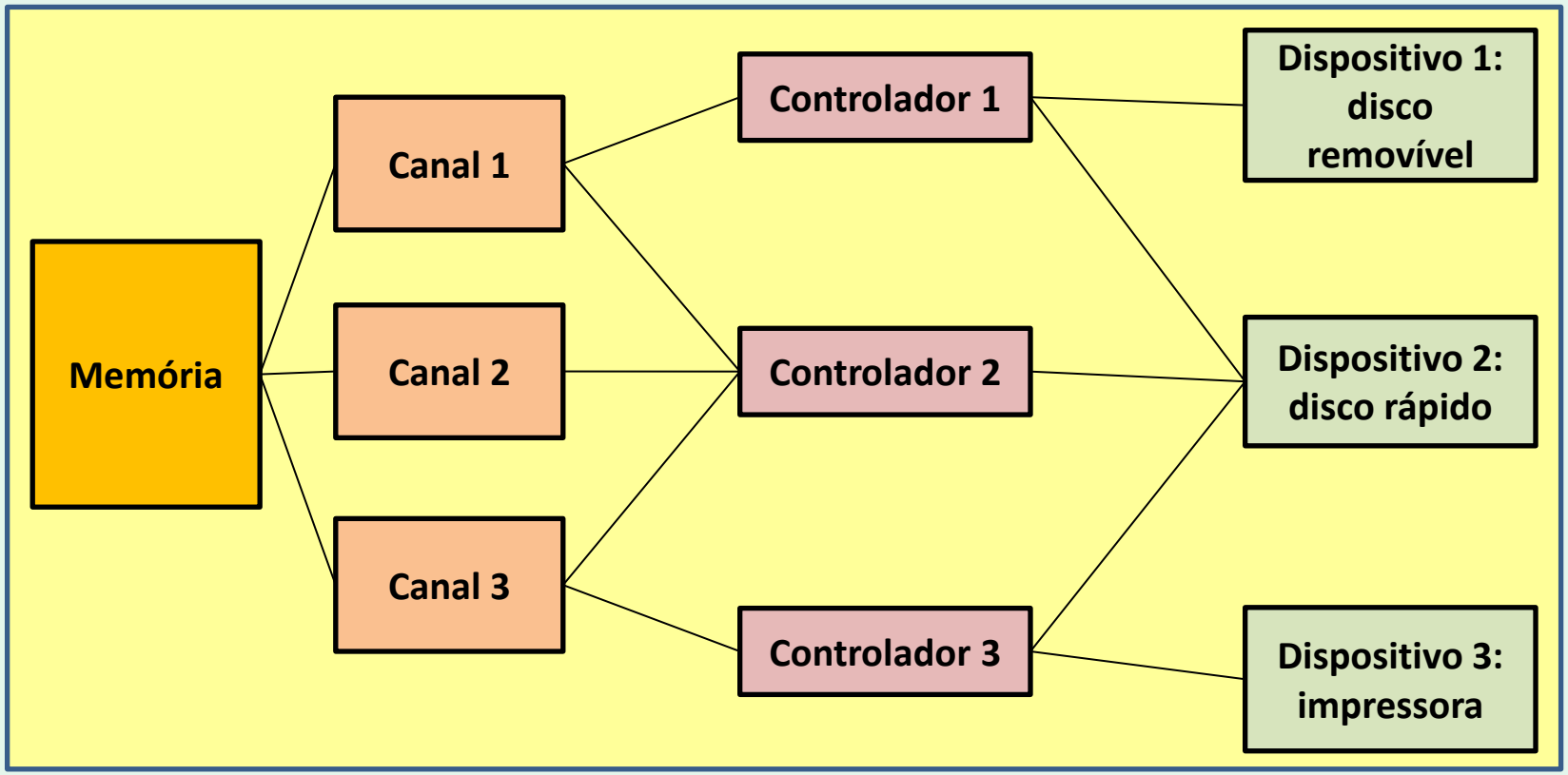
CCB (*channel control blocks*)

- Sua função é a de modelar os **canais**.
- Há um **CCB** para cada **canal**.
- Cada CCB contém:
  - **identificação** do canal.
  - ***status*** do canal.
  - lista das **unidades de controle ligadas** ao canal.
  - lista de **processos aguardando** este canal.

CUCB (*control unit control block*)

- Modela as **unidades de controle**.
- Logicamente, o CUCB efetua a **ligação entre os UCB's e os CCB's.**
- Há um **CUCB** para cada **unidade de controle**
- Cada CUCB apresenta as seguintes informações:
  - **identificação** da unidade de controle
  - ***status*** da unidade de controle
  - lista dos **dispositivos conectados** à unidade de controle
  - lista dos **canais conectados** à unidade de controle
  - lista dos **processos que estão aguardando** esta unidade

Em última análise, estas informações descrevem uma ligação do tipo:



I/O Scheduler

- Decide qual processo deve receber **o caminho entre a memória e o dispositivo**, no caso de haver mais requisições que caminhos disponíveis.
- Os métodos utilizados no *scheduling* de processos podem ser aplicados em *I/O Schedulers*, naturalmente com as devidas adaptações dos conceitos.

- Por exemplo, em *I/O schedulers* não existe o conceito de *time-slice*.
- Os demais conceitos podem ser aplicados quase diretamente.
- O ***I/O Scheduler*** **ordena os pedidos** de Entrada/Saída
- O ***I/O traffic controller*** **decide quais** dos pedidos na fila **podem ser satisfeitos**.

- Os *I/O device handlers* efetuam mais um nível de *scheduling* e de otimização no uso dos periféricos, através da exploração das particularidades dos dispositivos (isto é, tirando vantagem de características individuais dos dispositivos, para otimizar sua utilização).

"*I/O device handler*"

- Executa os programas de Entrada e Saída propriamente ditos, que promovem **fisicamente** as operações de **transferência de dados.**
- Manuseiam **condições de erro** de operação e de transferência nos dispositivos.
- Atendem e tratam **interrupções de Entrada e Saída**.

Técnicas p/ otimizar o uso de periféricos

- As operações de entrada e saída podem ser otimizadas através da utilização de algumas **técnicas**, entre as quais se destacam:
  - ordenação cíclica
  - endereços alternativos
  - ordenação do "*seek*"
  - virtualização de dispositivos
- A seguir, detalham-se estas técnicas.

Ordenação Cíclica dos Acessos

- **Ordenando adequadamente os pedidos** de entrada e saída em disco ou tambor é possível reduzir o tempo total necessário para o atendimento dos mesmos.
- Por exemplo, se desejarmos fazer acesso aos setores 2, 8, 3, 1 e 5 de um disco, se soubermos que a cabeça de leitura/gravação está posicionada no setor 4, poderemos reordenar os acessos de forma tal que não seja necessário esperar uma revolução completa do disco para efetuar o primeiro acesso.

- Uma boa reordenação é, por exemplo: 5, 8, 1, 2, 3. Se a informação da posição atual não for disponível, uma reordenação razoável pode ser: 1, 2, 3, 5, 8.
- Neste caso é necessário esperar em média meia revolução antes do acesso ao setor 1.
- Com esta reordenação, há uma redução dos tempos de acesso para, em média, meia revolução do tambor, mais o mínimo dos tempos necessários para os acessos aos demais setores, consideradas todas as combinações.

- Quando os acessos são mutuamente exclusivos, outras técnicas podem ser utilizadas: o "***SLOTTING***" e a ordenação dos pedidos de **acesso por setor**.
- Pedidos de Entrada/Saída em **diversas trilhas**, mas em **setores de mesmo número**, devem **competir pelo uso do "*Slot*"**.

- Isto causa um pequeno congestionamento, que em alguns sistemas é eliminado ou reduzido por meio da técnica (em hardware) de efetuar o **acesso a várias trilhas simultaneamente.**
- Isto infelizmente **encarece** apreciavelmente o sistema.

- Alguns dispositivos sofisticados, como o disco IBM 305 de cabeça fixa, efetua esta **ordenação por hardware.**
- Sua unidade de controle aceita também **programas simultâneos de canal** (até 16), por meio de um canal "*block multiplexor*".

Endereços alternativos para os dados

- Pode-se diminuir o tempo de acesso a uma informação gravada, se se dispuser, no disco, de **várias cópias** da mesma informação, estrategicamente distribuídas.
- Assim, os mesmos dados têm **vários endereços**, devendo-se sempre escolher no instante do acesso o endereço mais adequado.

- Esta técnica é denominada "***Folding***".
- Ela, naturalmente, **reduz a capacidade** efetiva do dispositivo, em termos do volume de informações armazenadas.

Ordenação do "*seek*"

- Consta em minimizar o tempo de acesso à informação ordenando os acessos de acordo com os cilindros, se o tempo de "*seek*" for grande.
- Se não for, ordenam-se os acessos apenas segundo o setor.
- Deve-se, neste caso, **minimizar o "*thrashing*"** do braço de leitura/gravação impondo que, enquanto não terminar o movimento em um sentido para todos os acessos pendentes, este sentido não deve ser invertido.