Sistemas Operacionais

Gerência de Entrada e Saída (E/S)

Lesandro Ponciano

2024

Objetivos da Aula

- Examinar a estrutura do subsistema de E/S
- Discutir princípios e complexidade do hardware
- Explicar aspectos de desempenho do hardware e software relacionados a E/S

Entrada e Saída (E/S)

- Existem duas tarefas principais nos computadores
  - Processamento
  - Entrada e Saída (E/S)
- O papel do SO em relação a E/S é gerenciar e controlar operações dos dispositivos de E/S
- Aspectos de E/S já foram discutidos de forma tangencial nos diversos tópicos tratados na disciplina
  - Aqui apresenta-se um quadro completo desses aspectos

Subsistema de E/S do Kernel

- Dispositivos de E/S (tela, mouse, teclado, disco, etc) variam em termos de função e de velocidade

- Tal variação implica na necessidade do uso de diferentes métodos para controlá-los
  - O conjunto de métodos implementados pelo sistema operacional formam o subsistema de E/S do kernel

Questões de Mercado

- O subsistema de E/S precisa lidar com diversas questões de mercado
  - Uma questão favorável: tende-se a uma convergência de interface padrão de software e hardware
  - Uma questão desfavorável: surgem cada vez mais dispositivos e sendo eles cada vez mais diversos entre si

Subsistema de E/S

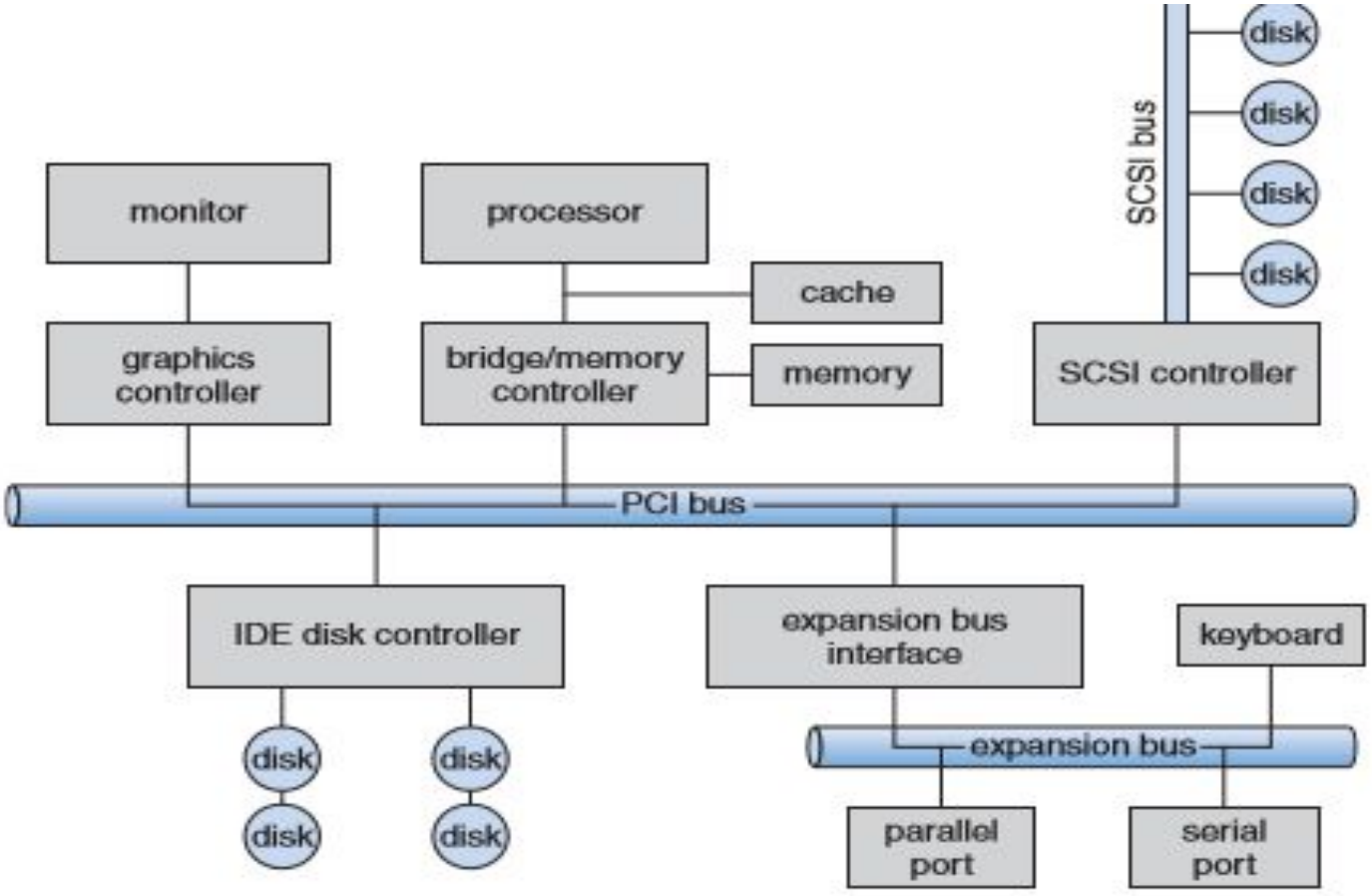
- O subsistema de E/S é responsável por realizar as funções comuns a todos os tipos de dispositivo
  - ficando o tratamento de aspectos específicos de cada dispositivo como responsabilidade dos *device drivers*

- Entre as funções do subsistema de E/S do kernel, pode-se destacar
  - Criar uma unidade lógica de transferência independente do dispositivo
  - Tratamento de erros nas operações de E/S
  - Mecanismo de proteção de acesso aos dispositivos
  - *Bufferização*
  - Interface padronizada com os *device drivers*

Hardware de E/S

- Tipos de dispositivos
  - armazenamento (ex. discos, fitas),
  - transmissão (ex. placas de rede, modens)
  - interface humana (e.g. monitor, teclado, mouse)
- O dispositivo se comunica com o computador por um ponto de conexão, ou *porta*
  - e.g. uma porta serial
- Quando os dispositivos usam conjunto de fios comuns, a conexão é chamada *bus*
  - Um *bus* é um conjunto de fios e um protocolo que especifica o conjunto de mensagens que podem ser trafegadas nos fios

Uma Estrutura Típica de *Bus*



Buses e Controladores

- Um computador possui diversos *buses*
  - **Bus PCI**: conecta o subsistema processador-memória aos dispositivos
  - **Bus de expansão**: conecta dispositivos relativamente lentos
  - **Bus SCSI**: conecta discos SCSI

- Também há diversos controladores de dispositivos
  - Um controlador é um conjunto de componentes eletrônicos que pode operar uma porta, um *bus* ou um dispositivo

Registradores das Portas

- Uma porta geralmente é composta de 4 registradores
  1) Entrada de dados: lido pelo *host* para obtenção de informações
  2) Saída de dados: escrito pelo *host* para envio de informações
  3) Status: indica confirmações e/ou erros nas operações
  4) Controle: usado para iniciar comandos ou alterar modalidades

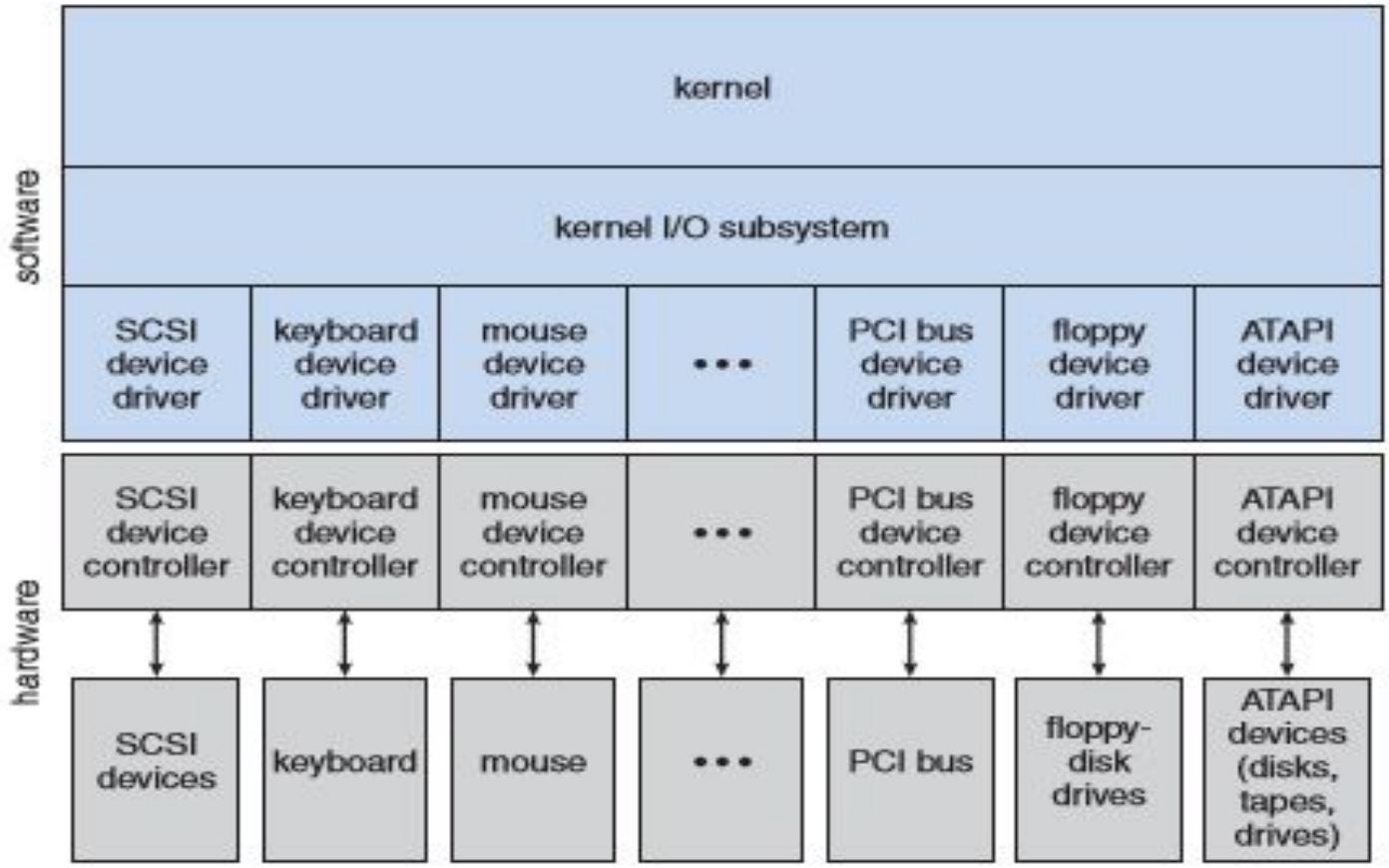
Tratamento de Requisições de E/S

- Formas de tratar requisições de dispositivos
  - ***Polling*** (ou sondagem) consiste na verificação contínua do dispositivo por meio de três instruções
    - Ler um registrador de dispositivo
    - Executar uma operação lógica para extrair o bit de estado
    - Ramificar uma dada ação, caso necessário
  - **Interrupções** consistem no controlador de dispositivos colocar uma informação na linha de interrupções do processador
- Quando o tratamento da requisição envolve a manipulação de muitos dados
  - Usa-se o controlador de acesso direto à memória (DMA)
  - O uso de DMA evita que se transfira poucos bytes de cada vez

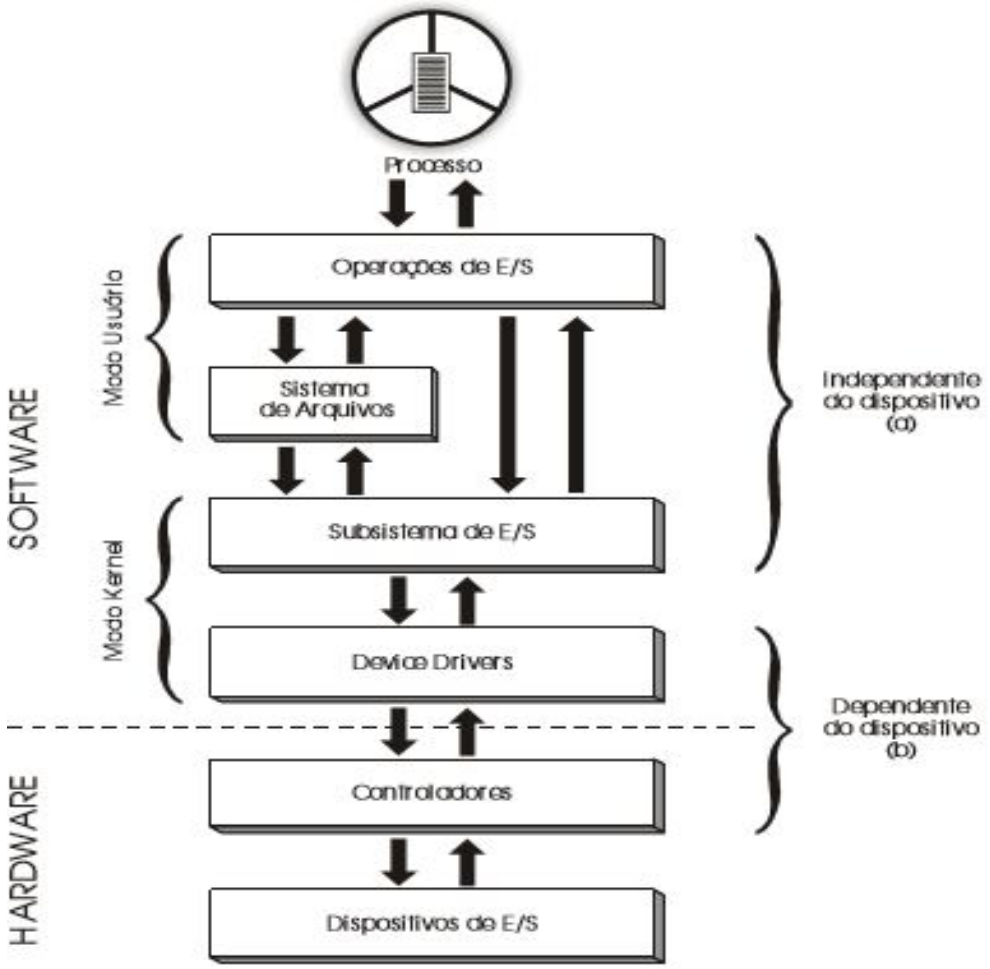
Interface de E/S da Aplicação

- O SO deve oferecer uma interface de chamada de sistemas que permita operar os dispositivos de E/S
- Busca-se técnicas de estruturação e interfaces do SO que permitam que dispositivos de E/S sejam tratados de forma padrão e uniforme
- As partes do *kernel* relacionadas a E/S são estruturadas em camadas de software

Estrutura de E/S do Kernel



Estrutura de E/S do Kernel



Interface de Chamadas de Sistemas

- Pode-se destacar quatro aspectos considerados na definição da interface de chamadas de sistemas relacionados a E/S
  1) Dispositivos de blocos e de caracteres
  2) Dispositivos de rede
  3) Relógios e *timers*
  4) E/S com e sem bloqueio

Transferência de Dados

- Um dispositivo de fluxo de caracteres transfere byte
  - Exemplo de dispositivos de caracteres: mouse, terminal
  - É esperado a existência de chamadas como *get()* a *put()*, que permitem capturar e inserir caracteres, respectivamente

- Um dispositivo de bloco transfere blocos de byte
  - Exemplo de dispositivos de bloco: disco
  - É esperado a existência de chamadas como *read()*, *write()* e *seek()*

Dispositivos de Rede

- Para dispositivos de rede, uma interface comum é a de *socket* de rede
  - Que permite a realização e operação de conexões
- Para implementação de servidores, a interface de *sockets* também fornece uma chamada select(), que permite
  - Gerenciar um conjunto de *sockets*
  - Otimizar operações de rede

Relógios e *Timers*

- Interface de relógios e *timers* de hardware fornecem três funções básicas
  - Informam a hora corrente
  - Informam o tempo decorrido
  - Posicionam um *timer* para disparar a operação Z no momento T
- Essas funções são muito usadas pelo SO e por aplicativos de tempo crítico

E/S com e sem Bloqueio

- A interface também deve permitir a realização de E/S com bloqueio e E/S sem bloqueio
  - E/S com bloqueio: quando uma aplicação realiza uma chamada de sistemas de E/S com bloqueio ela fica bloqueada até a chamada ser concluída
  - E/S sem bloqueio: quando uma aplicação realiza uma chamada de sistemas de E/S sem bloqueio ela continua executando

- Tais modos de E/S são fundamentais, por exemplo, na implementação de árvores de processos

Supervisão pelo Subsistema de E/S

- O subsistema de E/S do *Kernel* supervisiona os seguintes procedimentos
  - Gerenciamento do espaço de nomes de arquivos e dispositivos
  - Controle de acesso a arquivos e dispositivos
  - Controle de operações que cada dispositivo é capaz de executar
  - Alocação do espaço do sistema de arquivos
  - Alocação de dispositivos
  - Armazenamento em *buffer*, cache e *spool*
  - Escalonamento de E/S
  - Monitorar status, manipular erros e recuperar de falhas dos dispositivos
  - Configuração e inicialização de *drivers* de dispositivos

Subsistema de E/S do *Kernel*

- Eficiência do computador
  - Escalonamento de E/S
  - Armazenamento em *buffer*
  - Armazenamento em cache
  - Armazenamento em *Spool* e reservas de dispositivos
- Manipulação de erros
- Proteção de E/S
- Estruturas de dados do *Kernel*

Escalonamento de E/S

- É preciso determinar qual solicitação terá acesso a um dispositivo em cada momento
- O escalonador de E/S visa determinar uma boa ordem para sua execução
  - Seguir a ordem em que as aplicações emitem as chamadas de sistemas raramente é a melhor forma de escalonamento
- O escalonador de E/S pode ser projetado para atender a diversos objetivos, tais como:
  - Melhorar o desempenho geral do sistema
  - Compartilhar o acesso a dispositivos de maneira justa
  - Reduzir o tempo médio de espera para acessar o dispositivo

Escalonamento de E/S

- Suponha que o braço do disco está próximo do começo do disco quando três aplicações (A, B e C) emitem chamadas com bloqueio para esse disco
  - A solicita um bloco perto do fim do disco
  - B solicita um bloco perto do começo
  - C solicita um bloco perto do meio

- Nesse caso, o escalonador de E/S pode reduzir a distância que o braço percorrerá atendendo as requisições na ordem B, C, A

Armazenamento em *Buffer*

- Um *buffer* é uma área de memória que armazena os dados que estão sendo transferidos
  - entre dois dispositivos
  - entre um dispositivo e uma aplicação
- Três razões para o uso de *buffer*
  1) Lidar com a discrepância de velocidade entre os dispositivos produtor e consumidor do conjunto de dados
     - Ex. Modem -> *buffer* -> HD
  2) Fornecer adaptações para dispositivos que tenham diferentes tamanhos e transferência de dados
     - Ex. Fragmentação e remontagem de pacotes de rede
  3) Dar suporte à semântica de cópia de E/S de aplicações
     - Ex. Versão dos dados existentes no *buffer*

Armazenamento em *Cache*

- Um cache é uma região da memória rápida que mantém cópias de dados
  - O acesso à cópia existente no cache é mais eficiente do que o acesso ao dado original

- Cache *versus Buffer*
  - *Buffer* pode manter a única cópia existente de um dado enquanto o cache mantém uma cópia, em memória mais rápida, de um dado que reside em outro local

Armazenamento em *Spool*

- Um *spool* é um tipo de *buffer* que mantém saída para um dispositivo que não pode aceitar fluxos de dados intercalados

- Exemplo: impressora
  - Só imprime um arquivo de cada vez, não pode lidar com outro arquivo enquanto imprime
  - Novas impressões que forem chegando são colocadas em um *spool*

Manipulação de Erros

- Os diversos dispositivos de E/S podem falhar e o SO deve garantir que tal falha não comprometa o desempenho do sistema
- Falhas podem ser temporárias ou permanentes
  - Falhas temporárias podem ocorrer em decorrência do estado de uso do dispositivo, por exemplo, está sobrecarregado
  - Falhas permanentes podem ocorrer quando um dispositivo apresenta um defeito
- SOs geralmente conseguem se recuperar de falhas temporárias, mas não das permanentes

Proteção de E/S

- Erros são intimamente relacionados à questão da proteção
- O principal foco é evitar que um processo do usuário possa, intencionalmente ou acidentalmente, paralisar a operação normal do sistema
- Exemplos de proteções
  - Para evitar que um usuário realize operações inválidas, todas as instruções de E/S são definidas como privilegiadas
  - Locações que se mapeiam para memória e portas de E/S são protegidas pela própria memória

Estruturas de Dados do *Kernel*

- O *kernel* tem que manter informações de estado sobre o uso de componentes de E/S
- Isso é realizado com o uso de diversas estruturas de dados
- Exemplo de estruturas de dados usadas pelo *kernel*
  - Tabela de arquivos abertos por um processo
  - Tabela de arquivos abertos em todo o sistema
  - Tabela de *inodes* ativos
  - Tabela de informações de rede

Questões de Desempenho

- E/S é um fator importante no desempenho do sistema
  - Ele impõe pesadas demandas sobre a CPU para
    - Execução de códigos de *drivers* de dispositivos
    - Escalonamento de processos aos dispositivos
  - As mudanças de contexto para realização de E/S sobrecarregam a CPU e os caches
  - A troca de dados entre os controladores dos dispositivos e a memória física sobrecarrega o *bus* da memória durante sua realização
  - E/S dirigido a interrupções causa uma grande quantidade de trocas de contexto entre processos na CPU
  - Tráfego de rede também causa uma alta taxa de troca de contexto

Melhoria de Desempenho

- Pode-se empregar vários princípios para melhorar a eficiência de E/S
  - Reduzir o número de trocas de contexto
  - Reduzir o número de vezes que os dados devem ser copiados na memória enquanto passam entre dispositivos e a aplicação
  - Reduzir a frequência de interrupções usando transferências grandes, controladores inteligentes e sondagens
  - Liberar a CPU da cópia simples de dados usando canais com DMA
  - Permitir que algumas primitivas de processamento possam ser realizadas nos controladores dos dispositivos (assim o *bus* e a CPU podem desempenhar outras tarefas simultaneamente)
  - Balancear o uso da CPU, memórias, *bus* e E/S, por que uma sobrecarga em um deles causará ociosidade nos outros

Referências

SILBERSCHATZ, Abraham; GALVIN, Peter B.; GAGNE, Greg. Fundamentos de sistemas operacionais: princípios básicos. Rio de Janeiro, RJ: LTC, 2013. xvi, 432 p. ISBN 9788521622055

TANENBAUM, Andrew S. Sistemas operacionais modernos. 3. ed. São Paulo: Pearson Prentice Hall, 2009. xvi, 653 p. ISBN 9788576052371

Sistemas Operacionais

Prof. Dr. Lesandro Ponciano

https://orcid.org/0000-0002-5724-0094