Sistemas Operacionais

Gerenciamento de Memória Virtual (Segmentação)

Entrada e Saída

Norton Trevisan Roman
Marcelo Morandini
Jó Ueyama

Apostila baseada nos trabalhos de Kalinka Castelo Branco, Antônio Carlos Sementille, Luciana A. F. Martimiano e nas transparências fornecidas no site de compra do livro "Sistemas Operacionais Modernos"

Segmentação

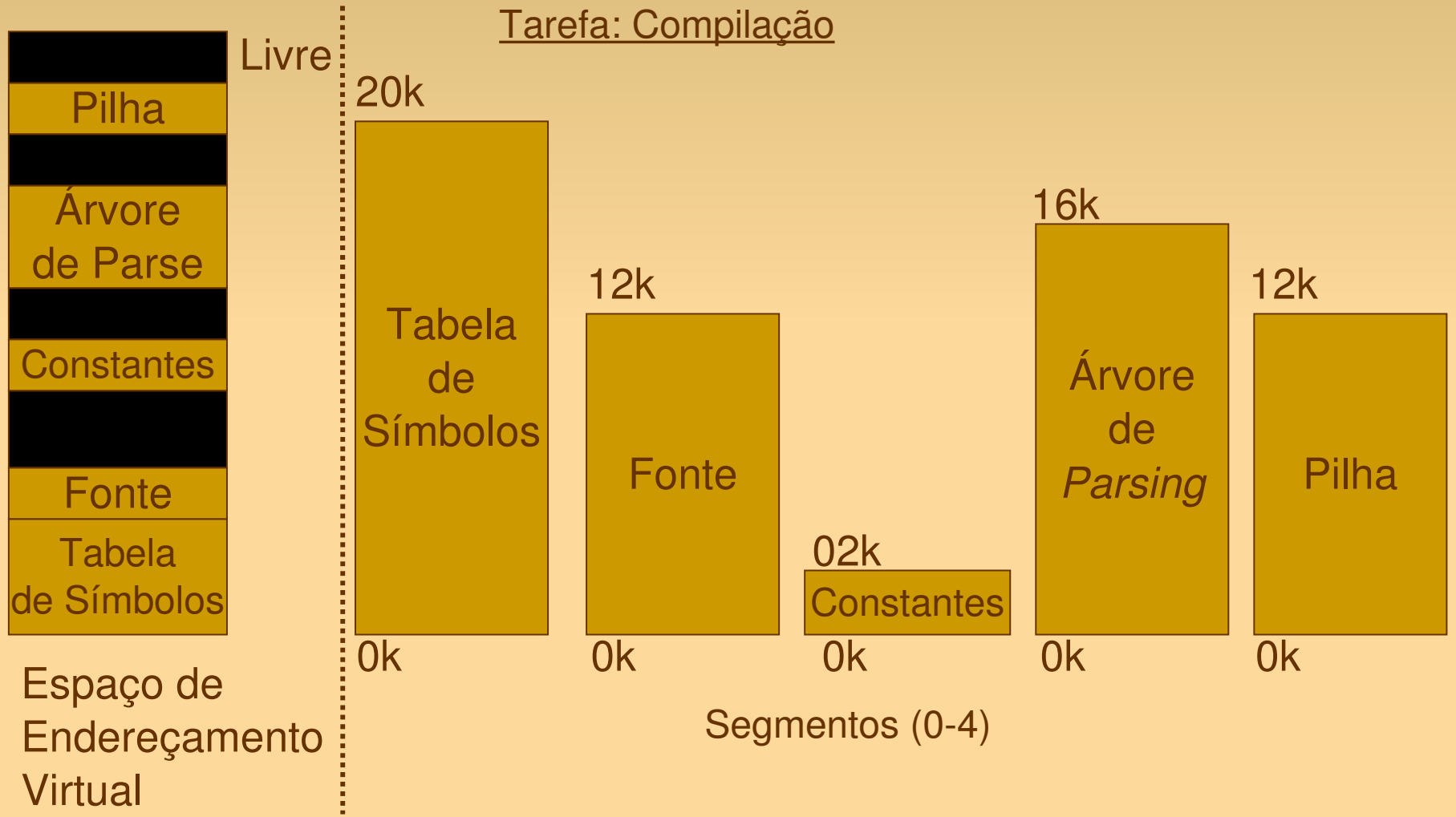
- A memória virtual é unidimensional
  - Endereços virtuais vão de 0 a um máximo
  - Considere um compilador:
    - Fonte
    - Tabela de símbolos
    - Constantes
    - Pilha de execução
    - Árvore de Parsing
    - Todas podem crescer... e podemos ficar “sem memória” para alguma, enquanto que outras possuem memória de sobra

Espaço de Endereçamento Virtual

Segmentação

- Solução
  - Segmentos: espaços de endereços completamente independentes
    - Sequência linear de endereços, de 0 a um máximo
    - Seu tamanho está entre 0 e esse máximo
    - Podem ter tamanhos diferentes, e o tamanho pode variar durante a execução
      - Ainda assim, até um máximo → pode “encher”
    - Endereçamento em 2 partes:
      - Número do segmento e o endereço dentro do segmento
    - Ex:
      - Um segmento para cada estrutura do compilador

Segmentação



Segmentação

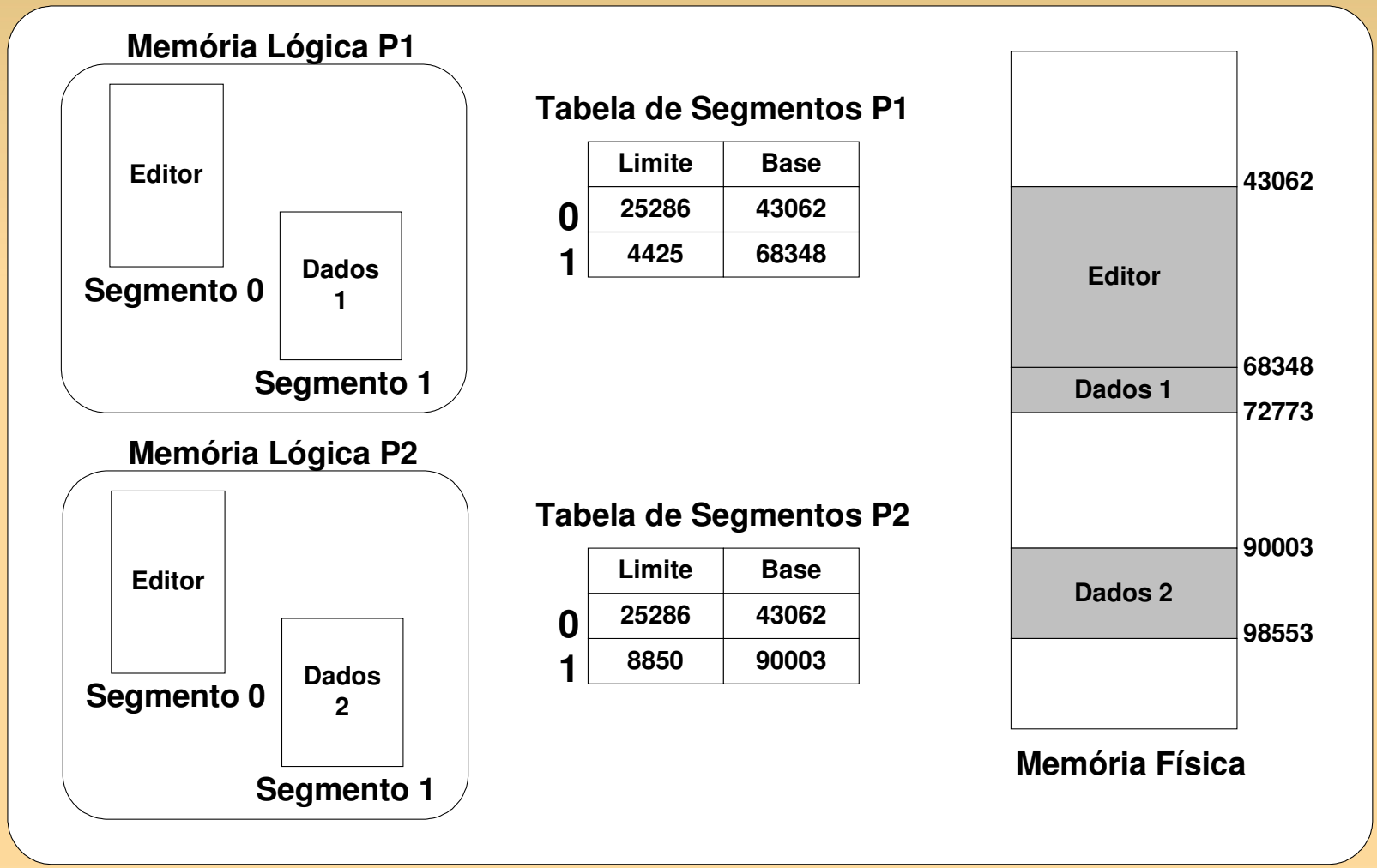
- Segmentos são unidades lógicas
  - O programador os vê e usa como entidades lógicas
    - Na forma de procedimentos, arranjos, coleções de variáveis, pilhas, etc
    - Normalmente não contém misturas de tipos
    - O usuário tem a ilusão de que todos os segmentos estão na memória principal, o tempo todo
- Implementação
  - Difere-se da paginação em que os segmentos não possuem tamanho fixo, enquanto que páginas possuem

Segmentação

- Implementação
  - Tabelas de segmentos com n linhas, cada qual apontando para um segmento de memória;
    - Vários espaços de endereçamento;
    - Endereço real -> base + deslocamento;
  - Alocação de segmentos segue os algoritmos já vistos
    - FIRST-FIT, BEST-FIT, NEXT-FIT, WORST-FIT, QUICK-FIT;
  - MMU também é utilizada para mapeamento entre os endereços lógicos e físicos;
    - Tabela de segmentos informa qual o endereço da memória física do segmento e seu tamanho;

Segmentação

- Implementação:



Segmentação

- Problemas encontrados
  - Embora haja espaço na memória, não há espaço contínuo:
    - Política de realocação: um ou mais segmentos são realocados para abrir espaço contínuo;
    - Política de compactação: todos os espaços são compactados;
    - Política de bloqueio: fila de espera;
    - Política de troca: substituição de segmentos;
  - Não possui fragmentação interna, embora sofra de fragmentação externa;

Segmentação

- Vantagens:
  - Facilita proteção dos dados;
  - Facilita compartilhamento de procedimentos e dados entre processos;
    - Ex: Suponha que cada procedimento ocupe um segmento
      - Cada procedimento terá o endereço (n,0) → (segmento, deslocamento
      - Se mudamos o procedimento em (5,0) e recompilamos, nada mais tem que ser mudado (não mudamos nenhum endereço)
        - Em um esquema unidimensional, ao aumentarmos uma, pode mudar o endereçamento da outra

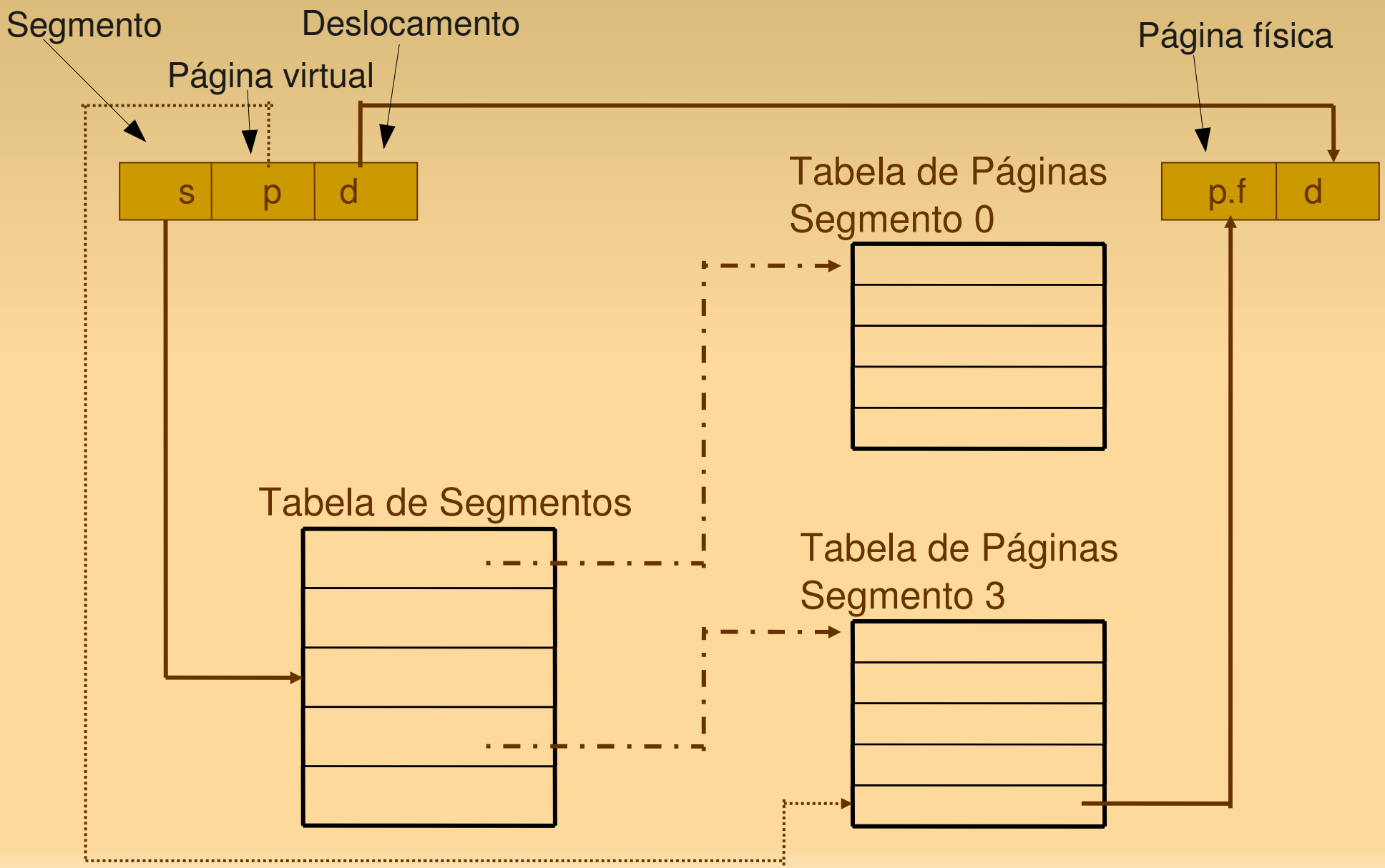
Segmentação Paginada

- Segmentos grandes podem não caber na memória
  - Devemos paginá-los (Ex: pentium)
- Espaço lógico é formado por segmentos
  - Cada segmento é dividido em páginas lógicas;
  - Cada segmento possui uma tabela de páginas
    - Usada para mapear o endereço de página lógica do segmento em endereço de página física;
  - No endereçamento, a tabela de segmentos indica, para cada segmento, onde sua respectiva tabela de páginas está;

Segmentação Paginada

- Ao referenciar um endereço de memória:
  - Verifica se o segmento buscado está na memória
    - Há bits específicos para isso na tabela
    - Senão estiver → page fault
  - Obtém, na tabela de segmentos, o endereço da tabela de páginas para aquele segmento.
    - Usa o endereço da página virtual como offset nessa tabela
  - Da tabela de páginas, obtém o endereço físico da moldura
    - Age como em paginação

Segmentação Paginada



Paginação × Segmentação

Consideração	Paginação	Segmentação
Programador deve saber da técnica?	Não	Sim
Espaços de endereçamento existentes	1	Vários
Espaço total de endereço pode exceder memória física?	Sim	Sim
É possível distinguir procedimento de dados e protegê-los?	Não	Sim

Paginação × Segmentação

Consideração	Paginação	Segmentação
Tabelas de tamanho variável podem ser acomodadas sem problemas?	Não	Sim
Compartilhamento de procedimentos entre usuário é facilitado?	Não	Sim
Por que?	Para obter espaço de endereçamento maior sem aumentar memória física	Para permitir que programas e dados possam ser divididos em espaços de endereçamento logicamente independentes; compartilhamento e proteção

Gerenciamento de Memória

- Thrashing (paginação excessiva)
  - Associado com o problema de definição do número de páginas/segmentos
    - Troca de páginas/segmentos é uma tarefa cara e lenta;
  - Se o processo tiver um número de páginas muito reduzido
    - Pode ficar muito tempo esperando pelo atendimento de uma falta de página
      - Muitos processos bloqueados;
  - Número de páginas grande
    - Trocas excessivas (thrashing)

Thrashing

Thrashing

- Evitar o problema (paginação):
  - Taxa máxima aceitável de troca de páginas;
    - Suspender alguns processos, liberando páginas físicas (swapping);
    - Risco de aumentar o tempo de resposta dos processos;
  - Determinar periodicamente o número de processos em execução e alocar para cada um mesmo número de páginas;
    - Problema: processos grandes teriam o mesmo número de páginas de processos pequenos, causando paginação excessiva;

Thrashing

- Possível solução:
  - Número inicial de páginas proporcional ao tamanho do processo
    - Alocação atualizada dinamicamente durante a execução dos processos;
    - PFF (Page Fault Frequency):
      - Algoritmo informa quando aumentar ou diminuir a alocação de páginas de um processo, controlando o tamanho do conjunto de alocação;
      - Não diz nada sobre que página trocar, em um page fault
      - Controla apenas o conjunto de alocação

Entrada e Saída

Dispositivos de Entrada e Saída

Entrada e Saída

- SO pode atuar de duas maneiras diferentes:
  - Como máquina estendida
    - Tornar uma tarefa de baixo nível mais fácil de ser realizada pelo usuário;
  - Como gerenciador de recursos
    - Gerenciar os dispositivos que compõem o computador;
    - E/S é parte disso

Entrada e Saída

- Funções específicas:
  - Atender interrupções;
  - Gerenciar comandos aceitos e funcionalidades (serviços prestados);
  - Tratar possíveis erros;
  - Prover interface entre os dispositivos e o sistema;
  - Prover uma interface única e padronizada
- Geralmente, o código para tratamento da entrada e saída representa uma fração significativa do sistema operacional total

Entrada e Saída

- Tipos de E/S
  - Os dispositivos de E/S podem ser classificados de forma ampla, sendo que as mais utilizadas são quanto ao:
    - Tipo de conexão
    - Tipo de transferência de dados
    - Tipo de compartilhamento de conexões
  - Quanto ao tipo de conexão:
    - Leva em consideração a natureza da conexão entre o módulo de E/S e o periférico
    - Do ponto de vista dos dados, as conexões são projetadas para operação serial ou paralela

Entrada e Saída – Classificação

- Quanto ao tipo de conexão:
  - Conexão serial:
    - Uma única linha de sinal é utilizada para o estabelecimento de toda a conexão, protocolo e transferência de dados, entre o módulo de E/S e o periférico
    - Características principais:
      - Mais barata que a paralela
      - Mais lenta que a paralela
      - Relativamente confiáveis
      - Usada em dispositivos mais baratos e lentos, como impressoras e terminais

Entrada e Saída – Classificação

- Quanto ao tipo de conexão:
  - Conexão paralela:
    - Várias linhas de sinais são usadas, de forma que vários bits de dados possam ser transferidos em paralelo
    - É comum que existam linhas independentes para tráfego de sinais de controle
    - Características principais:
      - Mais complexa que a serial
      - Mais cara
      - Mais rápida
      - Altamente confiável
      - Usada em dispositivos mais velozes, como unidades de disco, fita ou impressoras rápidas

Entrada e Saída – Classificação

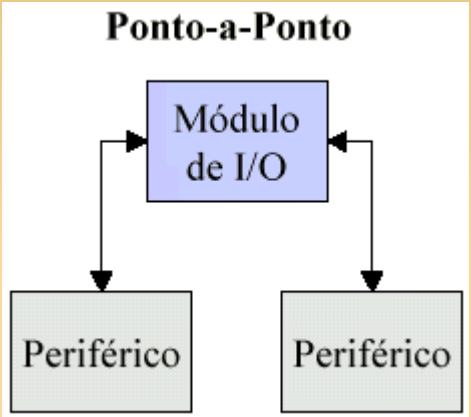
- Quanto ao tipo de transferência de E/S:
  - Dispositivos de bloco (Block devices):
    - Armazenam informação em blocos de tamanho fixo, cada um com seus próprios endereços
      - Os tamanhos dos blocos geralmente variam de 128 à 1024 bytes
    - Transferências são feitas com um ou mais blocos consecutivos
    - Lê ou escreve em cada bloco de maneira independente
    - Ex: HD, CD-ROM, Bastão USB

Entrada e Saída – Classificação

- Quanto ao tipo de transferência de E/S:
  - Dispositivos de caractere (Character devices)
    - Acessam um fluxo de caracteres, sem considerar qualquer estrutura de bloco
    - Não são endereçáveis e não possuem qualquer operação de acesso aleatório (“seek operation”)
    - Ex: impressoras, interfaces de rede, mouses
  - Este esquema de classificação não é perfeito, porém é genérico o suficiente

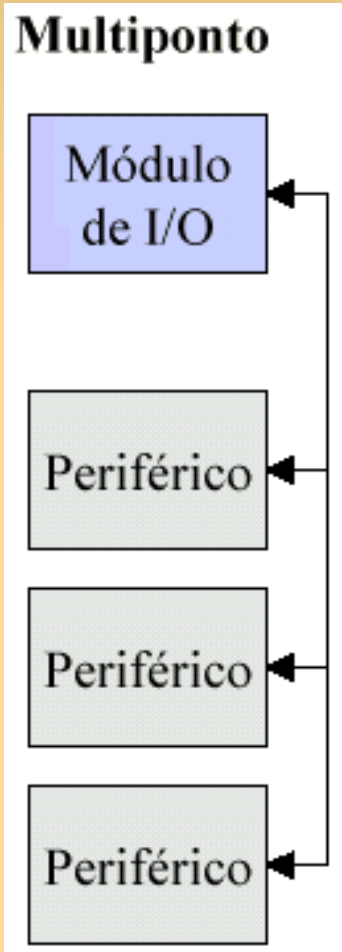
Entrada e Saída – Classificação

- Quanto ao compartilhamento de conexões
  - Podem ser divididos em 2 categorias:
    - Ponto-a-Ponto
      - É a conexão mais simples, onde existe um conjunto de linhas dedicadas para a ligação entre o módulo de E/S e cada periférico.
      - Oferecem maior confiabilidade
      - Permitem a operação simultânea de diversos dispositivos
      - Usadas em dispositivos mais simples, tais como modens, teclado e impressora



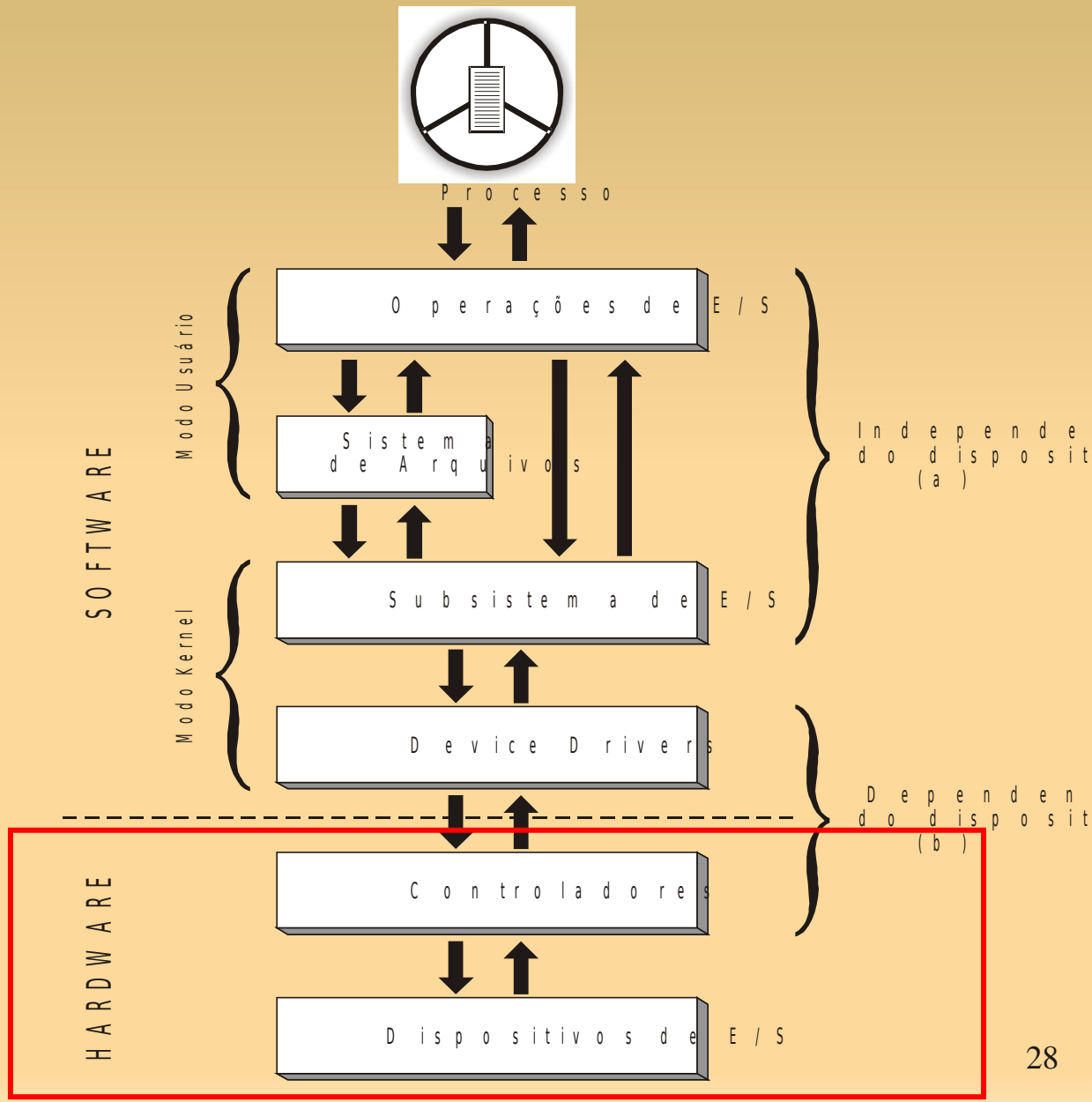
Entrada e Saída – Classificação

- Quanto ao compartilhamento de conexões
  - Podem ser divididos em 2 categorias:
    - Multiponto
      - Um módulo de E/S compartilha um conjunto de linhas de sinais entre diversos periféricos.
      - Algumas dessas linhas devem servir para realizar o endereçamento do dispositivo
      - Maior escalabilidade que a ponto a ponto
      - Não permite operação simultânea dos periféricos conectados
      - Usada para armazenamento:
      - Padrões:
        - IDE, SCSI, USB etc



Entrada e Saída

- Princípios a serem obedecidos:
  - Hardware;
  - Software;

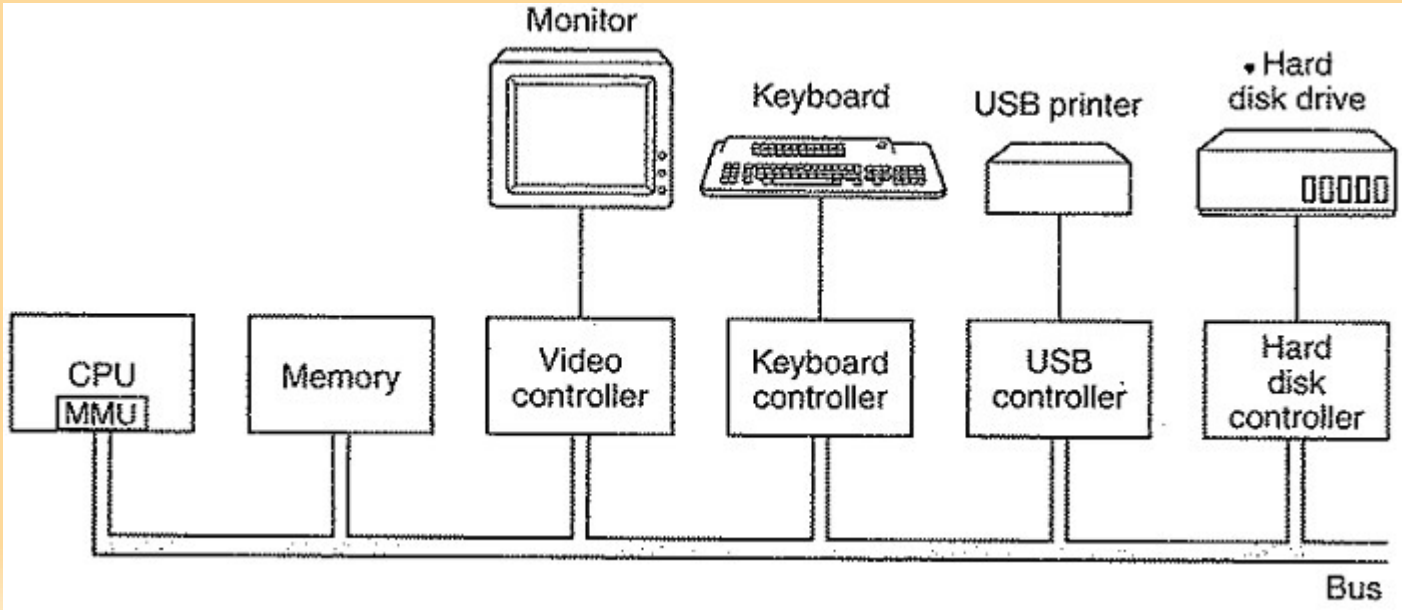


Princípios de Hardware

- As Unidades de E/S são geralmente compostas de dois componentes principais:
  - Componente Mecânico
    - O dispositivo em si
  - Controlador (ou adaptador) de dispositivo:
    - Componente eletrônico do dispositivo
    - Parte programável (Nos PCs é normalmente uma placa de circuito impresso ou chip);
    - Muitos controladores podem controlar vários dispositivos idênticos
    - Órgãos de padronização: IEEE, ISO, ANSI, etc.

Princípios de Hardware

- O S.O. sempre trata com o controlador, não com os dispositivos
  - A Comunicação entre UCP e controladores é feita através de barramentos comuns (interface de alto nível)
  - Interface entre controlador e dispositivo: baixo nível
  - Mainframes: múltiplos barramentos e processadores especializados em E/S (canais de E/S).



Princípios de Hardware

- Ex: Controlador de disco
  - Recebe fluxo de bits, com:
    - Preâmbulo
    - Os bits do setor/cilindro
    - Checksum (Error-Correcting Code – ECC)
  - Converte o fluxo serial de bits (vindo do dispositivo) em um bloco de bytes, executando qualquer correção necessária.
    - Monta o bloco de bytes em um buffer interno
    - Após verificar a checksum, copia o bloco na memória

Princípios de Hardware

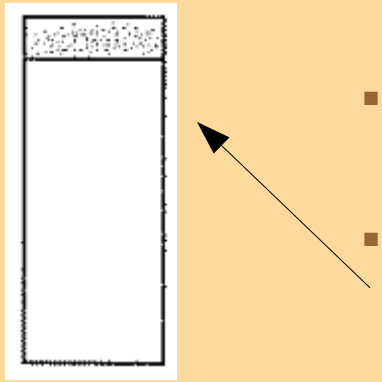
- Controladores
  - Cada controlador possui registradores, usados para comunicação com a CPU
    - Ao escrever nesses registradores, o SO pode comandar o dispositivo
      - executa E/S escrevendo comandos (e seus parâmetros, se existirem) nesses registradores
      - Quando um comando é aceito, a UCP pode deixar que o controlador trabalhe sozinho, indo executar outra tarefa.
      - Quando o dispositivo termina, avisa a UCP através de uma interrupção.
    - Ao ler desses registradores, o SO pode saber o estado do dispositivo, etc.

Princípios de Hardware

- Controladores
  - Cada controlador possui registradores, usados para comunicação com a CPU
    - Em alguns computadores estes registradores podem fazer parte do espaço de endereçamento da memória principal.
  - Possuem também um buffer (em geral), que o SO pode ler ou escrever (e.g. placa de rede)
  - Comunicação CPU – controladores
    - Como é possível a um módulo de E/S controlar mais de um dispositivo de E/S?
      - Necessidade de associação de endereços a estes dispositivos

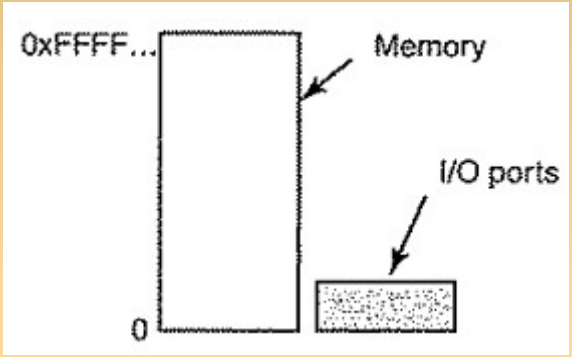
Princípios de Hardware

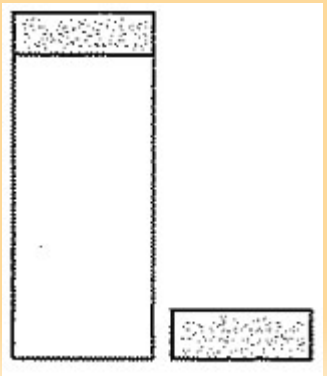
- Controladores
  - Comunicação CPU – controladores
    - Existem 2 formas de interpretação de endereços, pelo módulo de E/S, quando o barramento é compartilhado:
      - Mapeado em Memória:
        - O módulo de E/S opera dentro do espaço de endereçamento de memória, usando um conjunto de endereços reservados (registradores são tratados como posições de memórias);
        - Mapeia todos os registradores no espaço de memória
        - Cada registrador de controle recebe um endereço de memória único
          - Geralmente o topo do espaço de endereços



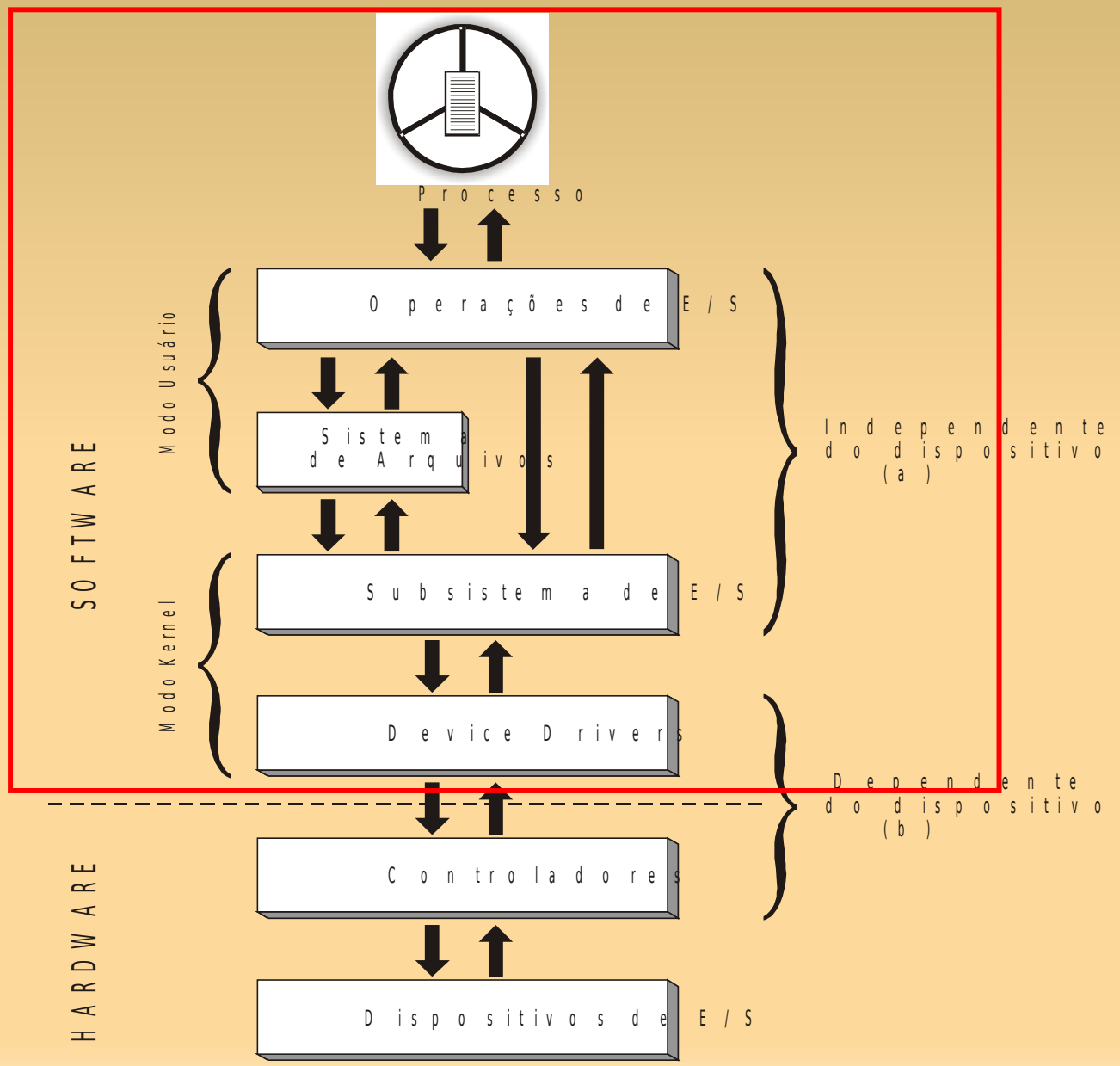
Princípios de Hardware

- Controladores
  - Comunicação CPU – controladores
    - Formas de interpretação de endereços:
      - Mapeado em E/S ou E/S isolada (I/O Ports):
        - Cada controlador recebe um número de porto de E/S
        - Acessado via instruções especiais, usadas apenas pelo SO
          - IN REG, PORT
          - OUT PORT,REG
        - Espaços de endereços diferentes para memória e E/S
          - Existe um espaço de endereçamento independente para os dispositivos de E/S.
      - Sistemas híbridos
        - Ex: Pentium





Entrada e Saída



Princípios de Software

- Objetivos
  - Independência de dispositivo
    - O programador não deve se preocupar com o tipo de dispositivo (CD, HD etc)
    - Cabe ao SO cuidar das particularidades
  - Nomenclatura uniforme
    - O nome de um dispositivo deve ser um string ou inteiro, independente do dispositivo
  - Tratamento de erro
    - Erros deveriam ser tratados o mais próximo do hardware possível, sem que o usuário tome conhecimento
      - Ex: repetindo a operação

Princípios de Software

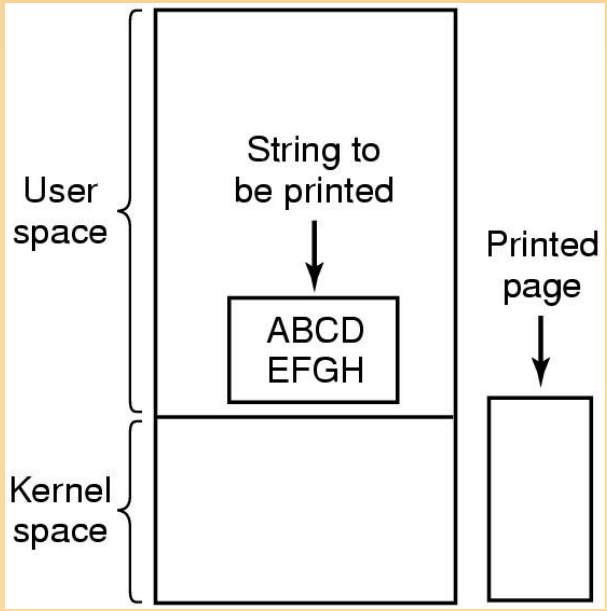
- Modos de execução de E/S:
  - Os módulos de E/S podem operar de 3 maneiras básicas:
    - E/S programada
    - E/S via Interrupções
    - E/S via Acesso Direto à Memória
  - O que distingue as três formas é a participação da UCP e a utilização das interrupções

Princípios de Software

- E/S programada
  - Forma mais simples de E/S → tudo é feito pela CPU
  - Os dados são trocados entre a CPU e o Módulo de E/S
  - A CPU executa um programa que:
    - Verifica o estado do módulo de E/S, preparando-o para a operação;
    - Se necessário, envia o comando que deve ser executado; aguardando o resultado do comando, para então, efetuar a transferência entre o módulo de E/S e algum registrador da CPU.

Princípios de Software

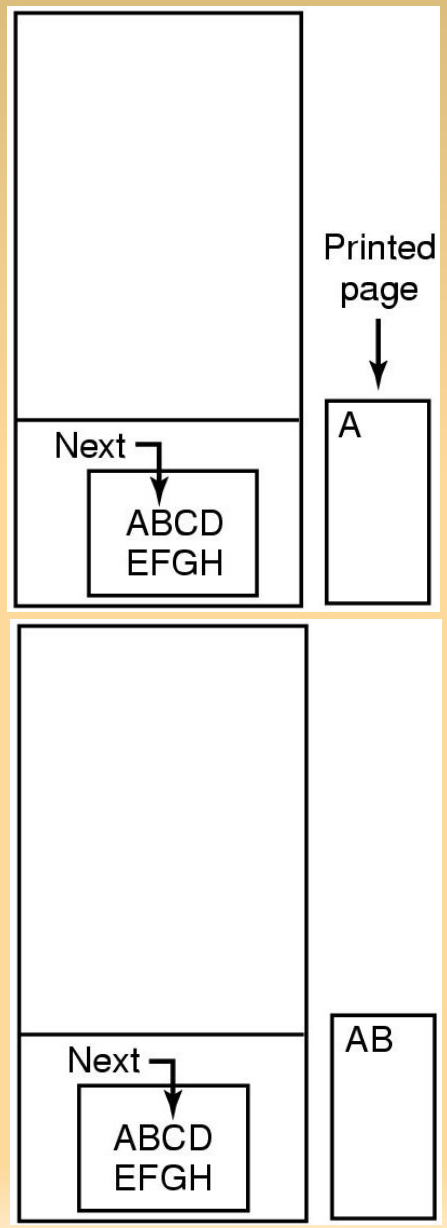
- E/S programada
  - Ex: imprimir “ABCDEFGH” na impressora
    - Monta o string em um buffer, no espaço do usuário
    - O processo então detém a impressora (via chamada de sistema – open)
    - Uma vez obtida a impressora, o processo diz ao SO para imprimir (chamada ao sistema)
    - O SO copia o string a um array no espaço do kernel



    - Assim que a impressora é liberada, o SO copia o primeiro caractere ao registrador de dados da impressora

Princípios de Software

- E/S programada
  - Ex: imprimir “ABCDEFGH” na impressora
    - O primeiro caractere é impresso
    - O sistema marca o próximo caractere, verifica novamente se a impressora está pronta para receber outro
    - Se pronta, o novo caractere é enviado
    - Continua até imprimir todo o string
    - Só então o controle volta ao processo



Princípios de Software

- E/S programada
  - Desvantagem:
    - CPU é ocupada o tempo todo até que a E/S seja feita;
    - CPU continuamente verifica se o dispositivo está pronto para aceitar outro caracter → espera ocupada
      - Também chamado de polling
  - Outro exemplo: Leitura de E/S
    - Escrita é semelhante