Introdução a Sistemas Operacionais

Professor Danilo

Introdução

- Um computador moderno consiste em um ou mais processadores, alguma memória principal, discos, impressoras, um teclado, um mouse, um monitor, interfaces de rede e vários outros dispositivos de entrada e saída. Como um todo, trata-se de um sistema complexo.
- Se todo programador de aplicativos tivesse de compreender como todas essas partes funcionam em detalhe, nenhum código jamais seria escrito.
- Além disso, gerenciar todos esses componentes e usá-los de maneira otimizada é um trabalho extremamente desafiador.

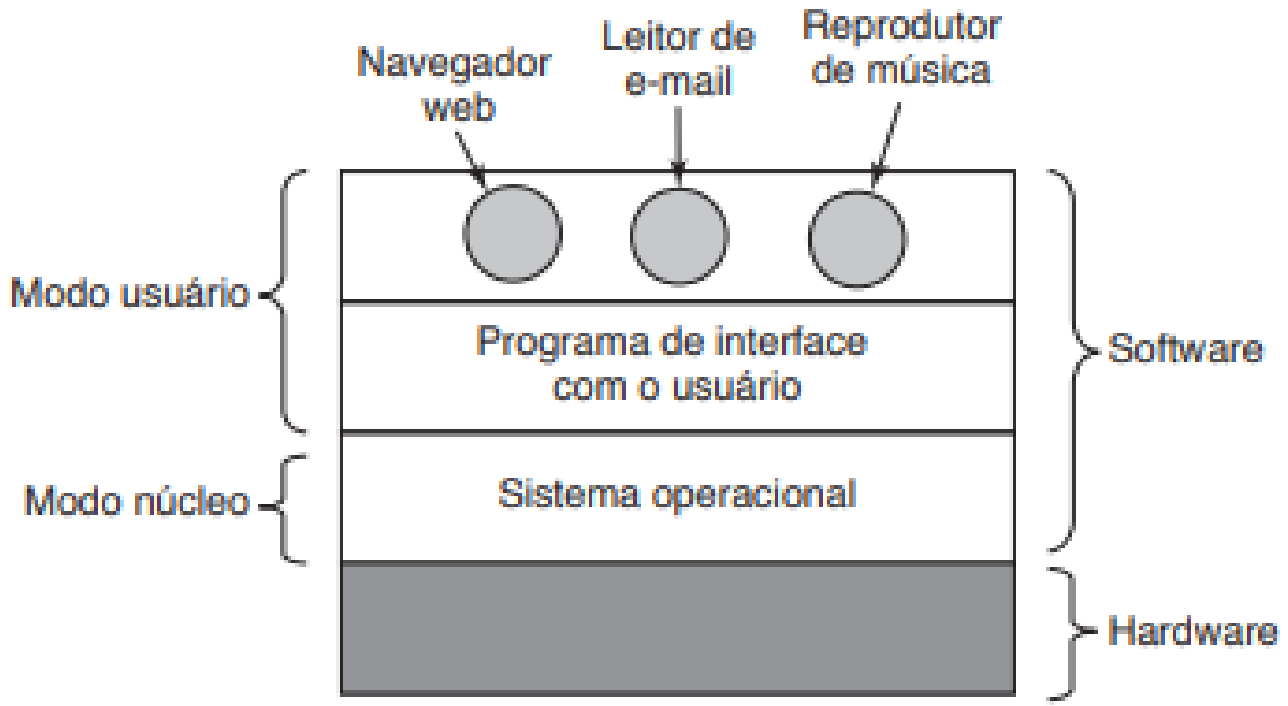
Introdução

- Por essa razão, computadores são equipados com um dispositivo de software chamado de sistema operacional, cuja função é **fornecer aos programas do usuário um modelo do computador melhor, mais simples e mais limpo, assim como lidar com o gerenciamento de todos os recursos mencionados**.

Introdução

- Quais Sistemas Operacionais você conhece?

Introdução



- Onde o sistema operacional se encaixa.

Introdução

- Os sistemas operacionais diferem de programas de usuário (isto é, de aplicativos) de outras maneiras além de onde estão localizados. Em particular, eles são enormes, complexos e têm vida longa.
- O código-fonte do coração de um sistema operacional como Linux ou Windows tem cerca de cinco milhões de linhas.
- Para entender o que isso significa, considere como seria imprimir cinco milhões de linhas em forma de livro, com 50 linhas por página e 1.000 páginas por volume. Seriam necessários 100 volumes para listar um sistema operacional desse tamanho — em essência, uma estante de livros inteira.

Introdução

O que é um sistema operacional?

- É difícil dizer com absoluta precisão o que é um sistema operacional
- Parte do problema é que os sistemas operacionais realizam **duas funções essencialmente não relacionadas**: **fornecer a programadores de aplicativos** (e programas aplicativos, claro) **um conjunto de recursos abstratos limpo em vez de recursos confusos de hardware, e gerenciar esses recursos de hardware.**

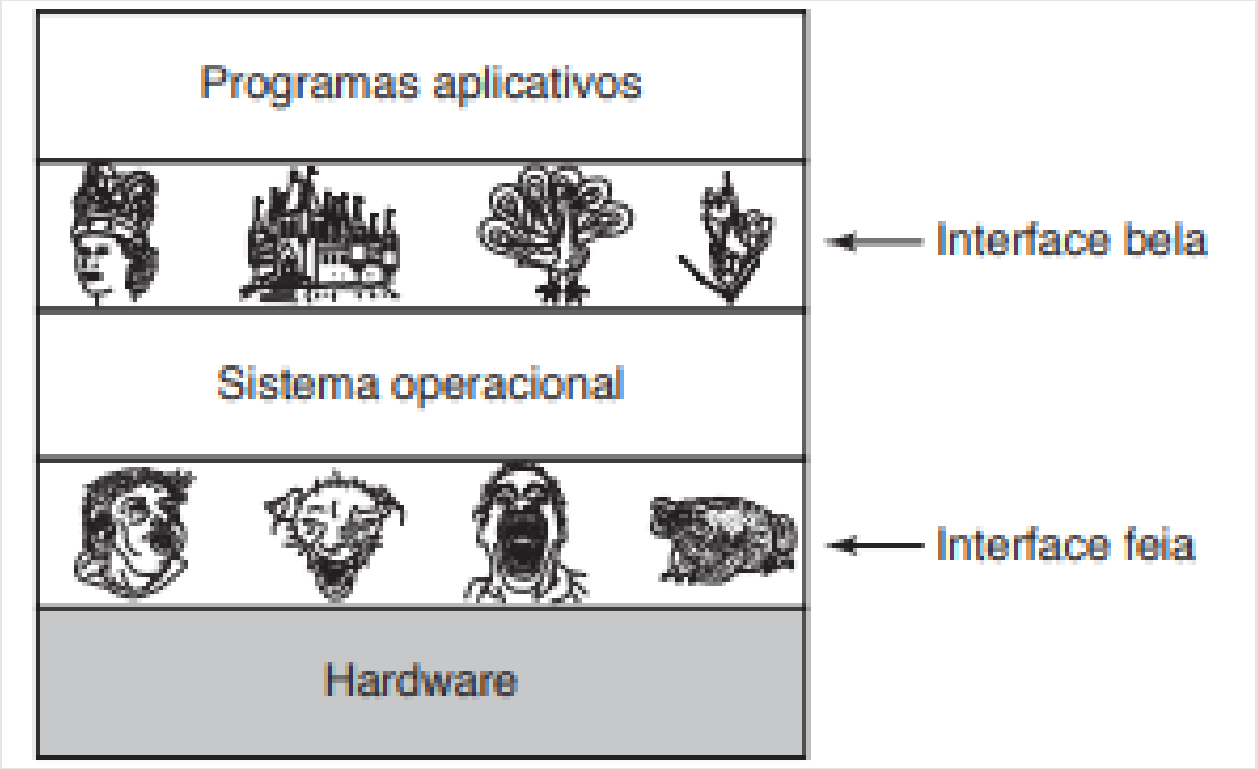
O sistema operacional como uma máquina estendida

- A arquitetura (conjunto de instruções, organização de memória, E/S e estrutura de barramento) da maioria dos computadores em nível de linguagem de máquina é primitiva e complicada de programar, especialmente para entrada/saída.
- Exemplo: Disco rígido SATA - Driver de disco
- Mais um nível de abstração para se utilizarem discos: Arquivos
- Usando essa abstração, os programas podem criar, escrever e ler arquivos, sem ter de lidar com os detalhes complexos de como o hardware realmente funciona.
- É muito mais fácil lidar com fotos, e-mails, músicas e páginas da web do que com detalhes de discos SATA (ou outros).

O sistema operacional como uma máquina estendida

- **A função dos sistemas operacionais é criar boas abstrações e então implementar e gerenciar os objetos abstratos criados desse modo.**

O sistema operacional como uma máquina estendida



- **Uma das principais tarefas dos sistemas operacionais é esconder o hardware e em vez disso apresentar programas (e seus programadores) com abstrações de qualidade, limpas, elegantes e consistentes com as quais trabalhar.**

O sistema operacional como uma máquina estendida

- Os clientes reais dos sistemas operacionais são os programas aplicativos. São eles que lidam diretamente com as abstrações fornecidas pela interface do usuário, seja uma linha de comandos (shell) ou uma interface gráfica.

O sistema operacional como um gerenciador de recursos

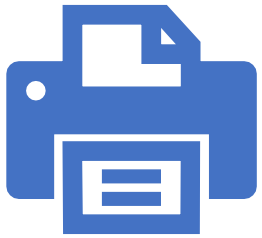
- **O conceito de um sistema operacional como fundamentalmente fornecendo abstrações para programas aplicativos é uma visão top-down (abstração de cima para baixo).**
- **Uma visão alternativa, bottom-up (abstração de baixo para cima), sustenta que o sistema operacional está ali para gerenciar todas as partes de um sistema complexo.**
- **Na visão bottom-up, a função do sistema operacional é fornecer uma alocação ordenada e controlada dos processadores, memórias e dispositivos de E/S entre os vários programas competindo por eles.**

O sistema operacional como um gerenciador de recursos

- Computadores modernos consistem de processadores, memórias, temporizadores, discos, dispositivos apontadores do tipo mouse, interfaces de rede, impressoras e uma ampla gama de outros dispositivos**.**
- **Na visão bottom-up, a função do sistema operacional é fornecer uma alocação ordenada e controlada dos processadores, memórias e dispositivos de E/S entre os vários programas competindo por eles.**

O sistema operacional como um gerenciador de recursos

Sistemas Operacionais modernos permitem que múltiplos programas estejam na memória e sejam executados ao mesmo tempo.



Exemplo: Impressora

Exemplo Redes (vários usuários)

O sistema operacional como um gerenciador de recursos

- **O gerenciamento de recursos inclui a multiplexação (compartilhamento) de recursos de duas maneiras diferentes: no tempo e no espaço.**

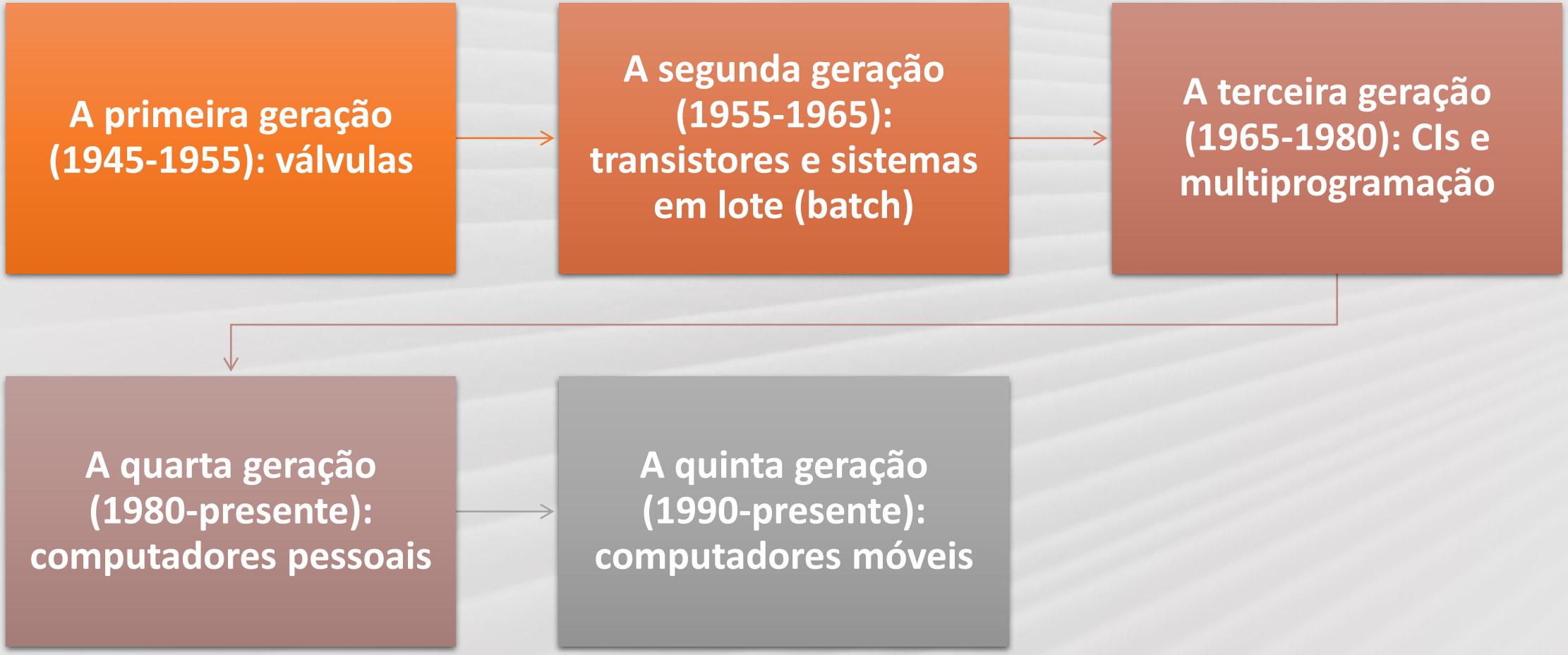
O sistema operacional como um gerenciador de recursos

- Quando um recurso é multiplexado no tempo, diferentes programas ou usuários se revezam usando-o. Primeiro, um deles usa o recurso, então outro e assim por diante.
- Por exemplo, com apenas uma CPU e múltiplos programas querendo ser executados nela, o sistema operacional primeiro aloca a CPU para um programa, então, após ele ter sido executado por tempo suficiente, outro programa passa a fazer uso da CPU, então outro, e finalmente o primeiro de novo. Determinar como o recurso é multiplexado no tempo — quem vai em seguida e por quanto tempo — é a tarefa do sistema operacional.
- Outro exemplo da multiplexação no tempo é o compartilhamento da impressora. Quando múltiplas saídas de impressão estão na fila para serem impressas em uma única impressora, uma decisão tem de ser tomada sobre qual deve ser impressa em seguida.

O sistema operacional como um gerenciador de recursos

- O outro tipo é a multiplexação de espaço.
- Em vez de os clientes se revezarem, cada um tem direito a uma parte do recurso. Por exemplo, a memória principal é normalmente dividida entre vários programas sendo executados, de modo que cada um pode ser residente ao mesmo tempo (por exemplo, a fim de se revezar usando a CPU).
- Presumindo que há memória suficiente para manter múltiplos programas, é mais eficiente manter vários programas na memória ao mesmo tempo do que dar a um deles toda ela, especialmente se o programa precisa apenas de uma pequena fração do total. É claro, isso gera questões de justiça, proteção e assim por diante, e cabe ao sistema operacional solucioná-las.
- Outro recurso que é multiplexado no espaço é o disco. Em muitos sistemas um único disco pode conter arquivos de muitos usuários ao mesmo tempo. Alocar espaço de disco e controlar quem está usando quais blocos do disco é uma tarefa típica do sistema operacional.

História dos Sistemas Operacionais



A primeira geração (1945-1955): válvulas

- Os primeiros computadores digitais surgiram na II Guerra Mundial. Eles eram formados por milhares de válvulas e ocupavam áreas enormes, sendo de funcionamento lento e não confiável.
- O primeiro computador digital de propósito geral foi o ENIAC (Eletronic Numerical Integration and Computer). Sua estrutura possuía válvulas, capacitores, resistores, pesava 30 toneladas, e ele realizava 5 mil adições por segundo. Para trabalhar nessas máquinas, era necessário conhecer profundamente o funcionamento do hardware, pois a programação era feita em painéis, por meio de fios, e em linguagem de máquina.
- Existia um grupo de pessoas que projetava, construía, programava, operava e realiza a manutenção nestes computadores. Nesta fase não existia o conceito de Sistema Operacional e nem de linguagem de programação.

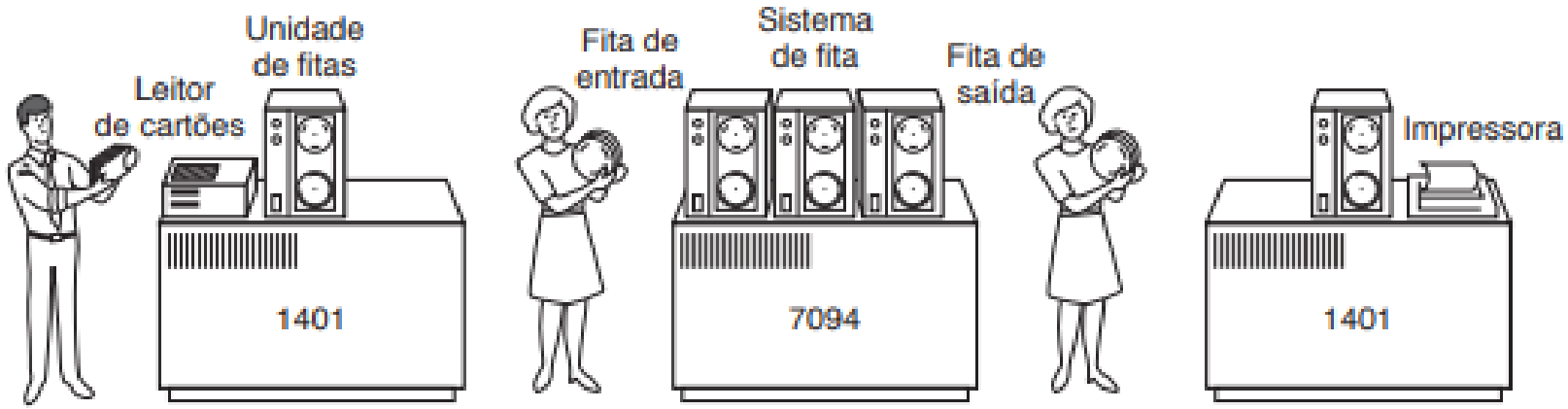
A segunda geração (1955-1965): transistores e sistemas em lote (batch)

- A criação do transistor e das memórias magnéticas contribui para o enorme avanço dos computadores da época.
- O transistor permitiu o aumento da velocidade e da confiabilidade do processamento, e as memórias magnéticas permitiram o acesso mais rápido aos dados, maior capacidade de armazenamento e computadores menores.
- Com o surgimento das primeiras linguagens de programação, como Assembly e Fortran, os programas deixaram de ser feitos diretamente no hardware, o que facilitou o processo de desenvolvimento de programas.
- Os primeiros Sistemas Operacionais surgiram, justamente, para tentar automatizar as tarefas manuais até então utilizadas.

A segunda geração (1955-1965): transistores e sistemas em lote (batch)

- Para otimizar a entrada de programas, surgiram as máquinas leitoras de cartão perfurado que aceleravam muito a entrada de dados.
- Os programadores deveriam então escrever seus programas e transcrevê-los em cartão perfurados.
- Cada programa e seus respectivos dados eram organizados em conjuntos denominados jobs que poderiam ser processados da seguinte forma.

A segunda geração (1955-1965): transistores e sistemas em lote (batch)



(a) Programadores levavam cartões para o 1401.
(b) O 1401 lia o lote de tarefas em uma fita.
(c) O operador levava a fita de entrada para o 7094.
(d) O 7094 executava o processamento.
(e) O operador levava a fita de saída para o 1401.
(f) O 1401 imprimia as saídas.

A segunda geração (1955-1965): transistores e sistemas em lote (batch)

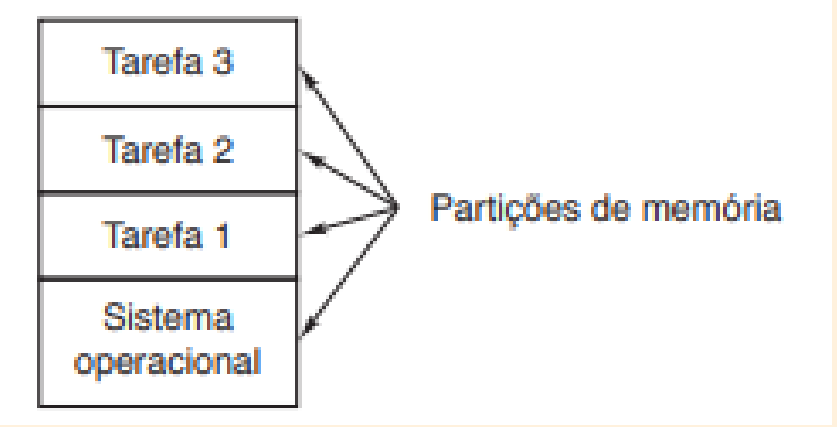
- Apesar da natureza sequencial do processamento, para os usuários era como se um lote de jobs fosse processado a cada vez, originando o termo Processamento em Lote (Batch Processing).
- Com o Processamento Batch, um grupo de programas era submetido de uma só vez, o que diminuía o tempo existente entre a execução dos programas, permitindo o melhor uso do processador.
- Os SOs passaram e ter seu próprio conjunto de rotinas para operações de Entrada/Saída (Input/Output Control System – IOCS), o que veio a facilitar bastante o processo de programação.
- O IOCS eliminou a necessidade de os programadores desenvolverem suas próprias rotinas de leitura/gravação para cada dispositivo periférico.

A terceira geração (1965-1980): CIs e multiprogramação

- Com o uso de circuitos integrados (CIs) e, posteriormente, dos microprocessadores, foi possível viabilizar e difundir o uso de sistemas computacionais por empresas. Um dos principais avanços desta fase foi a utilização da multiprogramação.
- Segundo Deitel, **"multiprogramação é quando vários jobs estão na memória principal simultaneamente, enquanto o processador é chaveado de um job para outro job, fazendo-os avançarem enquanto os dispositivos periféricos são mantidos em uso quase constante**".
- Enquanto o processamento chamado científico era muito bem atendido pelo Processamento em Lote comum, o mesmo não acontecia com o processamento dito comercial.
- No processamento científico, ocorre a execução de grande quantidade de cálculos com quantidade relativamente pequena de dados, mantendo o processador ocupado na maior parte do tempo.
- Já no processamento comercial, o processador permanece bastante ocioso dado que os cálculos são relativamente simples, em comparação com o processamento científico, e o uso de operações de Entrada/Saída é frequente, devido a quantidade de dados a ser processada.

A terceira geração (1965-1980): CIs e multiprogramação

- A multiprogramação permitiu uma solução para o problema da ociosidade do processador por meio da divisão da memória em partes, denominadas partições, onde em cada divisão um job poderia ser mantido.



A terceira geração (1965-1980): CIs e multiprogramação

- A utilização de fitas magnéticas obrigava o processamento a ser estritamente sequencial, ou seja, o primeiro programa a ser gravado na fita era o primeiro a ser processado. Assim, se um programa que levasse várias horas antecedesse pequenos programas seus tempos de resposta ficariam comprometidos.
- Com o surgimento de dispositivos de acesso direto, como os discos, foi possível escolher os programas a serem executados, realizando o escalonamento de tarefas. Isto permitia a alteração na ordem de execução das tarefas, tornando o Processamento em Lote mais eficiente.

A terceira geração (1965-1980): CIs e multiprogramação

- Outra técnica presente nesta geração foi a técnica de spooling (simultaneous peripheral operation on-line), isto é, a habilidade de certos SOs em ler novos jobs de cartões ou fitas armazenando-os em uma área temporária do disco rígido interno, para uso posterior quando uma partição de memória fosse liberada.
- Um exemplo dessa técnica nos dias atuais ocorre quando impressoras são utilizadas. No momento em que um comando de impressão é executado por um programa, as informações que serão impressas são gravadas em um arquivo em disco (arquivo de spool), para ser impresso posteriormente pelo sistema. Essa técnica permite um maior grau de compartilhamento na utilização de impressora.
- Apesar destas novas técnicas, os sistemas da época operavam basicamente em lote, com multiprogramação. Associem-se a isto a limitação que ainda existia na comunicação de usuários com a máquina e, também, o problema de monopólio de CPU por parte de programas que tomam muito tempo de processamento e realizam relativamente poucas operações de E/S como, por exemplo, os programas científicos.

A terceira geração (1965-1980): CIs e multiprogramação

- Tais problemas motivaram o surgimento de uma variação de multiprogramação chamada **sistemas de tempo compartilhado (time-sharing systems).**
- Os sistemas de tempo compartilhado permitiam a interação de vários usuários com o sistema, basicamente por meio de terminais de vídeo. Devido a essa característica, eles ficaram também conhecidos como sistemas on-line.
- O computador poderia, então, ser usado por vários usuários ao mesmo tempo, por meio de pseudoparalelismo. O **pseudoparalelismo** poderia ser obtido com o **chaveamento do processador entre vários processos** que poderiam atender aos usuários. A idéia central era dividir o poder computacional do computador entre seus diversos usuários, passando a impressão de que o computador estava totalmente disponível para cada usuário, embora isto não fosse verdade.

A terceira geração (1965-1980): CIs e multiprogramação

- Nos sistemas de tempo compartilhado, o controle da execução dos programas é feita interativamente e ocorre a eliminação do monopólio sobre a CPU da multiprogramação, realizando uma distribuição mais justa do tempo de uso do processador.
- Outro fato importante nessa fase foi o surgimento do sistema operacional UNIX (1969).
- Concebido inicialmente em um minicomputador PDP-7, baseado no sistema MULTICS (Multiplexed Information and Computing Service), o Unix foi depois escrito em uma linguagem de alto nível (Linguagem C). O Unix tornou-se conhecido por sua portabilidade, pois era mais fácil modificar o código do sistema Unix para implementá-lo em um novo computador do que escrever um novo Sistema Operacional.
- No final dessa fase, com a evolução dos microprocessadores, surgiram os primeiros microcomputadores, muito mais baratos que qualquer um dos computadores até então comercializados.

A quarta geração (1980-presente): computadores pessoais

- A integração em larga escala (Large Scale Integration – LSI) e integração em muito larga escala (Very Large Scale Integration – VLSI) levaram adiante o projeto de miniaturização e barateamento dos equipamentos.
- Os mini e superminicomputadores se firmaram no mercado e os microcomputadores ganharam um grande impulso.
- Nesse quadro, surgiram os microcomputadores (Personal Computer - PC) de 16 bits da IBM e o Sistema Operacional da Microsoft MS-DOS, criando a filosofia dos computadores pessoais.
- O termo tempo compartilhado (time-sharing) está associado aos Mainframes, computadores grandes, potentes e caros, que possuíam terminais de vídeo, com quase nenhum processamento, ligados a eles. A idéia era permitir que múltiplos usuários compartilhassem um único Mainframe.
- Atualmente, existe um conceito mais amplo que veio a substituir o conceito de Sistemas de Tempo Compartilhado, que é justamente **Sistemas Multitarefa (Método em que múltiplas tarefas podem ser executadas, aparentemente de forma simultânea, em um computador com uma única CPU).**

A quarta geração (1980-presente): computadores pessoais

- Pode-se dividir os Sistemas Multitarefas em dois tipos: Sistemas Preemptivos e Sistemas Cooperativos.
- Um Sistema Operacional Multitarefa Preemptivo contempla justamente as mesmas características dos chamados sistemas de tempo compartilhado. Exemplos de Sistemas Multitarefa são o Windows NT, Unix e Linux.
- Um Sistema Operacional Multitarefa Cooperativo se caracteriza pelo fato de que o processo que está atualmente de posse da CPU, ou seja, o que está sendo executado, é quem passa o controle da CPU para outros processos. Um exemplo deste tipo de Sistema Operacional é o Windows 3.1.
- As redes distribuídas (Wide Area Networks – WANs) se difundiram por todo o mundo, permitindo o acesso a outros sistemas de computação, em locais distantes. Também foram desenvolvidos inúmeros protocolos de rede como o DECnet da Digital E.C., SNA (System Network Architecture) da IBM, e outros de domínio público, como o TCP/IP e o X25.
- Surgem as primeiras redes locais (Local Area Networks – LANs) interligando pequenas áreas. Os softwares de rede passaram a estar intimamente relacionados ao Sistema Operacional e surgem os Sistemas Operacionais de Rede.

A quarta geração (1980-presente): computadores pessoais

- Os grandes avanços em termos de hardware, software e telecomunicações no final de século passado, são consequência da evolução das aplicações, que necessitam cada vez mais de capacidade de processamento e armazenamento de dados.
- Sistemas especialistas, sistemas multimídia, banco de dados distribuídos, inteligência artificial e redes neurais são apenas alguns exemplos da necessidade cada vez maior de capacidade de processamento.
- A evolução da microeletrônica permitirá o desenvolvimento de processadores e memórias cada vez mais velozes e baratos, além de dispositivos menores, mais rápidos e com maior capacidade de armazenamento. Os componentes baseados em tecnologia VLSI evoluem rapidamente para o ULSI (Ultra Large scale Integration).

A quarta geração (1980-presente): computadores pessoais

Tópicos importantes:

- Modificações profundas nas disciplinas de programação no uso de arquiteturas paralelas;
- Processamento distribuído explorado nos Sistemas Operacionais, espalhando funções por vários processadores (redes);
- Arquitetura cliente-servidor em redes locais passa a ser oferecida em redes distribuídas, permitindo o acesso a todo tipo de informação, independentemente de onde esteja armazenada. - Consolidação dos SOs baseados em interfaces gráficas. Novas interfaces homem-máquina serão utilizadas, como linguagem naturais, sons e imagens.
- Conceitos e implementações só vistos em sistemas considerados de grande porte estão sendo introduzidos na maioria dos sistemas desktop (Windows, Unix, Linux, etc.).
- Durante o ano de 1991, as primeiras versões (releases) do Sistema Operacional Linux começaram a ser desenvolvidas por Linus Torvalds, com ajuda de outros desenvolvedores. Desenvolvido para os clones AT 386, 486, o Linux era (é) um sistema tipo Unix (Unixtype) para computadores pessoais e tinha como grande atrativo, além do fato de ser parecido com o Unix, o fato de ser um Sistema Operacional gratuito. Estima-se hoje que existam 18 milhões de usuários no Mundo usando Linux.

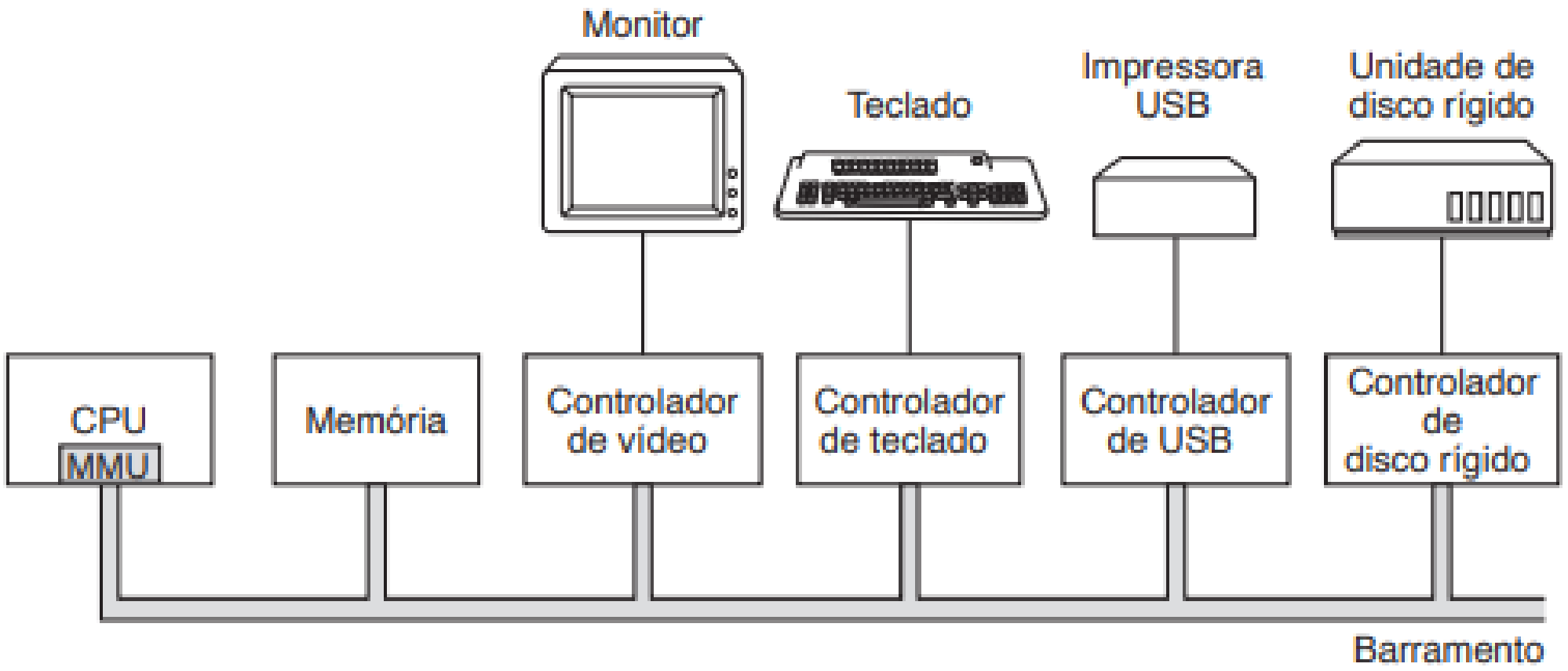
A quinta geração (1990-presente): computadores móveis

- Embora a ideia de combinar a telefonia e a computação em um dispositivo semelhante a um telefone exista desde a década de 1970 também, o primeiro smartphone de verdade não foi inventado até meados de 1990, quando a Nokia lançou o N9000, que literalmente combinava dois dispositivos mormente separados: um telefone e um PDA (Personal Digital Assistant — assistente digital pessoal). Em 1997, a Ericsson cunhou o termo smartphone para o seu “Penelope” GS88.
- Agora que os smartphones tornaram-se onipresentes, a competição entre os vários sistemas operacionais tornou-se feroz e o desfecho é mais incerto ainda que no mundo dos PCs. A maioria dos smartphones na primeira década após sua criação era executada em Symbian OS.
- Não levou muito tempo para o Android, um sistema operacional baseado no Linux lançado pelo Google em 2008, dominar os seus rivais. Para os fabricantes de telefone, o Android tinha a vantagem de ser um sistema aberto e disponível sob uma licença permissiva. Como resultado, podiam mexer nele e adaptá-lo a seu hardware com facilidade. Ele também tem uma enorme comunidade de desenvolvedores escrevendo aplicativos, a maior parte na popular linguagem de programação Java.

Revisão sobre hardware de computadores

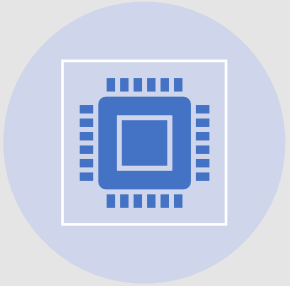
- Um sistema operacional está intimamente ligado ao hardware do computador no qual ele é executado. Ele estende o conjunto de instruções do computador e gerencia seus recursos. Para funcionar, ele deve conhecer profundamente o hardware, pelo menos como aparece para o programador.
- Conceitualmente, um computador pessoal simples pode ser abstraído em um modelo. A CPU, memória e dispositivos de E/S estão todos conectados por um sistema de barramento e comunicam-se uns com os outros sobre ele.
- Computadores pessoais modernos têm uma estrutura mais complicada, Monitor Teclado Impressora USB Unidade de disco rígido Controlador de disco rígido Controlador de USB Controlador de teclado Controlador de vídeo CPU Memória Barramento MMU envolvendo múltiplos barramentos.

Revisão sobre hardware de computadores



Processadores

O “cérebro” do computador é a CPU. Ela busca instruções da memória e as executa.



O ciclo básico de toda CPU é buscar a primeira instrução da memória, decodificá-la para determinar o seu tipo e operandos, executá- -la, e então buscar, decodificar e executar as instruções subsequentes. O ciclo é repetido até o programa terminar. É dessa maneira que os programas são executados.

Cada CPU tem um conjunto específico de instruções que ela consegue executar. Desse modo, um processador x86 não pode executar programas ARM e um processador ARM não consegue executar programas x86.

Como o tempo para acessar a memória para buscar uma instrução ou palavra dos operandos é muito maior do que o tempo para executar uma instrução, todas as CPUs têm alguns registradores internos para armazenamento de variáveis e resultados temporários.

Processadores

- Um desses registradores é o **contador de programa**, que contém o endereço de memória da próxima instrução a ser buscada. Após essa instrução ter sido buscada, o contador de programa é atualizado para apontar a próxima instrução.
- Outro registrador é o **ponteiro de pilha**, que aponta para o topo da pilha atual na memória. A pilha contém uma estrutura para cada rotina que foi chamada, mas ainda não encerrada. Uma estrutura de pilha de rotina armazena aqueles parâmetros de entrada, variáveis locais e variáveis temporárias que não são mantidas em registradores.
- Outro registrador ainda é o **PSW** (Program Status Word — palavra de estado do programa). Esse registrador contém os bits do código de condições, que são estabelecidos por instruções de comparação, a prioridade da CPU, o modo de execução (usuário ou núcleo) e vários outros bits de controle.

Processadores

Para melhorar o desempenho, os projetistas de CPU há muito tempo abandonaram o modelo simples de buscar, decodificar e executar uma instrução de cada vez. Muitas CPUs modernas têm recursos para executar mais de uma instrução ao mesmo tempo.

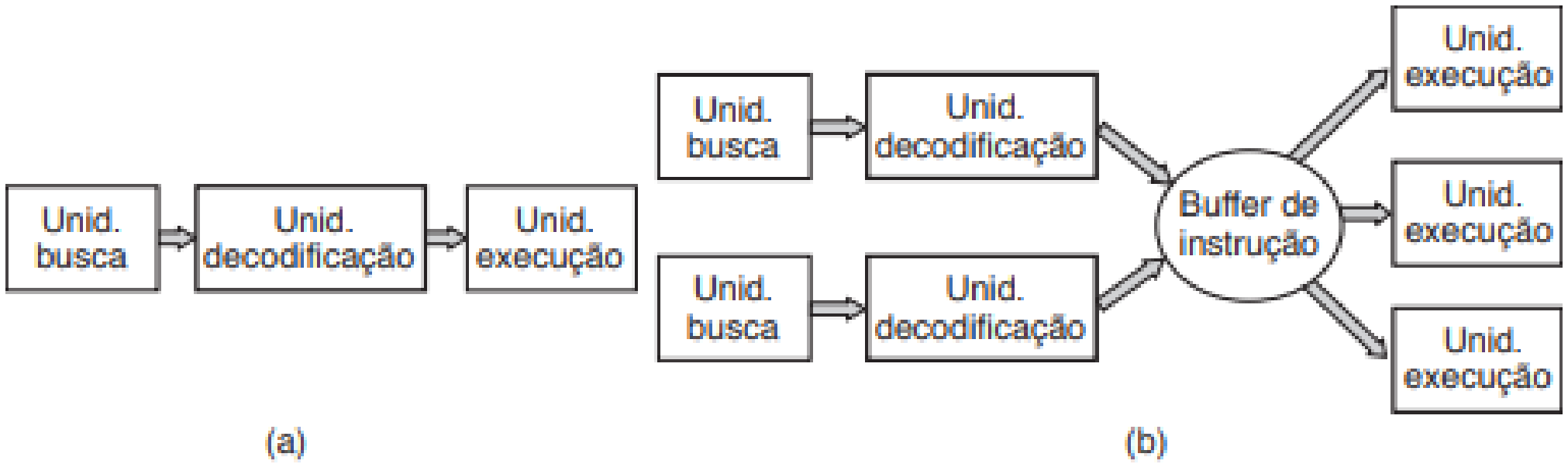
Uma organização com essas características é chamada de pipeline.

Processadores

- Ainda mais avançada que um projeto de pipeline é uma CPU superescalar.
- Nesse projeto, unidades múltiplas de execução estão presentes. Uma unidade para aritmética de números inteiros, por exemplo, uma unidade para aritmética de ponto flutuante e uma para operações booleanas. Duas ou mais instruções são buscadas ao mesmo tempo, decodificadas e jogadas em um buffer de instrução até que possam ser executadas.
- Tão logo uma unidade de execução fica disponível, ela procura no buffer de instrução para ver se há uma instrução que ela pode executar e, se assim for, ela remove a instrução do buffer e a executa.
- Uma implicação desse projeto é que as instruções do programa são muitas vezes executadas fora de ordem.
- Em geral, cabe ao hardware certificar-se de que o resultado produzido é o mesmo que uma implementação sequencial conseguiria, mas como veremos adiante, uma quantidade incômoda de tarefas complexas é empurrada para o sistema operacional.

Processadores

(a) Um pipeline com três estágios. (b) Uma CPU superescalar.



Processadores

- A maioria das CPUs, exceto aquelas muito simples usadas em sistemas embarcados, tem dois modos, núcleo e usuário.
- Quando operando em modo núcleo, a CPU pode executar todas as instruções em seu conjunto de instruções e usar todos os recursos do hardware.
- Em computadores de mesa e servidores, o sistema operacional normalmente opera em modo núcleo, dando a ele acesso a todo o hardware.
- Na maioria dos sistemas embarcados, uma parte pequena opera em modo núcleo, com o resto do sistema operacional operando em modo usuário.
- Programas de usuários sempre são executados em modo usuário, o que permite que apenas um subconjunto das instruções possa ser executado e um subconjunto dos recursos possa ser acessado. Geralmente, todas as instruções envolvendo E/S e proteção de memória são inacessíveis no modo usuário.
- Para obter serviços do sistema operacional, um programa de usuário deve fazer uma chamada de sistema, que, por meio de uma instrução TRAP, chaveia do modo usuário para o modo núcleo e passa o controle para o sistema operacional. Quando o trabalho é finalizado, o controle retorna para o programa do usuário na instrução posterior à chamada de sistema.

Memória

- O segundo principal componente em qualquer computador é a memória. Uma memória deve ser rápida ao extremo (mais rápida do que executar uma instrução, de maneira que a CPU não seja atrasada pela memória), abundantemente grande e muito barata.
- O sistema de memória é construído como uma hierarquia de camadas. As camadas superiores têm uma velocidade mais alta, capacidade menor e um custo maior por bit do que as inferiores.
- A camada superior consiste em registradores internos à CPU. Eles são feitos do mesmo material que a CPU e são, desse modo, tão rápidos quanto ela. Em consequência, não há um atraso ao acessá-los.
- Em seguida, vem a memória cache, que é controlada principalmente pelo hardware. Sempre que um recurso pode ser dividido em partes, algumas das quais são usadas com muito mais frequência que as outras, o caching é muitas vezes utilizado para melhorar o desempenho. Sistemas operacionais o utilizam seguidamente.

Memória

- A memória principal trata-se da locomotiva do sistema de memória. A memória principal é normalmente chamada de RAM (Random Access Memory — memória de acesso aleatório). Todas as requisições da CPU que não podem ser atendidas pela cache vão para a memória principal.
- Além da memória principal, muitos computadores têm uma pequena memória de acesso aleatório não volátil. Diferentemente da RAM, a memória não volátil não perde o seu conteúdo quando a energia é desligada. A ROM (Read Only Memory — memória somente de leitura) é programada na fábrica e não pode ser modificada depois. Ela é rápida e barata. Em alguns computadores, o carregador (bootstrap loader) usado parainicializar o computador está contido na ROM.
- A EEPROM (Electrically Erasable PROM — ROM eletricamente apagável) e a memória flash também são não voláteis, mas, diferentemente da ROM, podem ser apagadas e reescritas. No entanto, escrevê-las leva muito mais tempo do que escrever em RAM, então elas são usadas da mesma maneira que a ROM, apenas com a característica adicional de que é possível agora corrigir erros nos programas que elas armazenam mediante sua regravação.
- A memória flash também é bastante usada como um meio de armazenamento em dispositivos eletrônicos portáteis. A memória flash é intermediária em velocidade entre a RAM e o disco. Também, diferentemente da memória de disco, ela se desgasta quando apagada muitas vezes.
- Outro tipo ainda de memória é a CMOS, que é volátil. Muitos computadores usam a memória CMOS para armazenar a hora e a data atualizadas. A memória CMOS e o circuito de relógio que incrementa o tempo registrado nela são alimentados por uma bateria pequena, então a hora é atualizada corretamente, mesmo quando o computador estiver desligado. A memória CMOS também pode conter os parâmetros de configuração, como de qual disco deve se carregar o sistema.

Discos

- Em seguida na hierarquia está o disco magnético (disco rígido).
- Um disco consiste em um ou mais pratos metálicos que rodam a 5.400, 7.200, 10.800 RPM, ou mais. Um braço mecânico move-se sobre esses pratos a partir da lateral, como o braço de toca-discos de um velho fonógrafo de 33 RPM para tocar discos de vinil. A informação é escrita no disco em uma série de círculos concêntricos. Em qualquer posição do braço, cada uma das cabeças pode ler uma região circular chamada de trilha. Juntas, todas as trilhas de uma dada posição do braço formam um cilindro.
- Cada trilha é dividida em um determinado número de setores, com tipicamente 512 bytes por setor. Em discos modernos, os cilindros externos contêm mais setores do que os internos.

Discos

- Às vezes você ouvirá as pessoas falando sobre discos que não são discos de maneira alguma, como os SSDs (Solid State Disks — discos em estado sólido). SSDs não têm partes móveis, não contêm placas na forma de discos e armazenam dados na memória (flash). A única maneira pela qual lembram discos é que eles também armazenam uma quantidade grande de dados que não é perdida quando a energia é desligada.
- Muitos computadores dão suporte a um esquema conhecido como memória virtual. Esse esquema torna possível executar programas maiores que a memória física colocando-os no disco e usando a memória principal como um tipo de cache para as partes mais intensivamente executadas.

Dispositivos de E/S

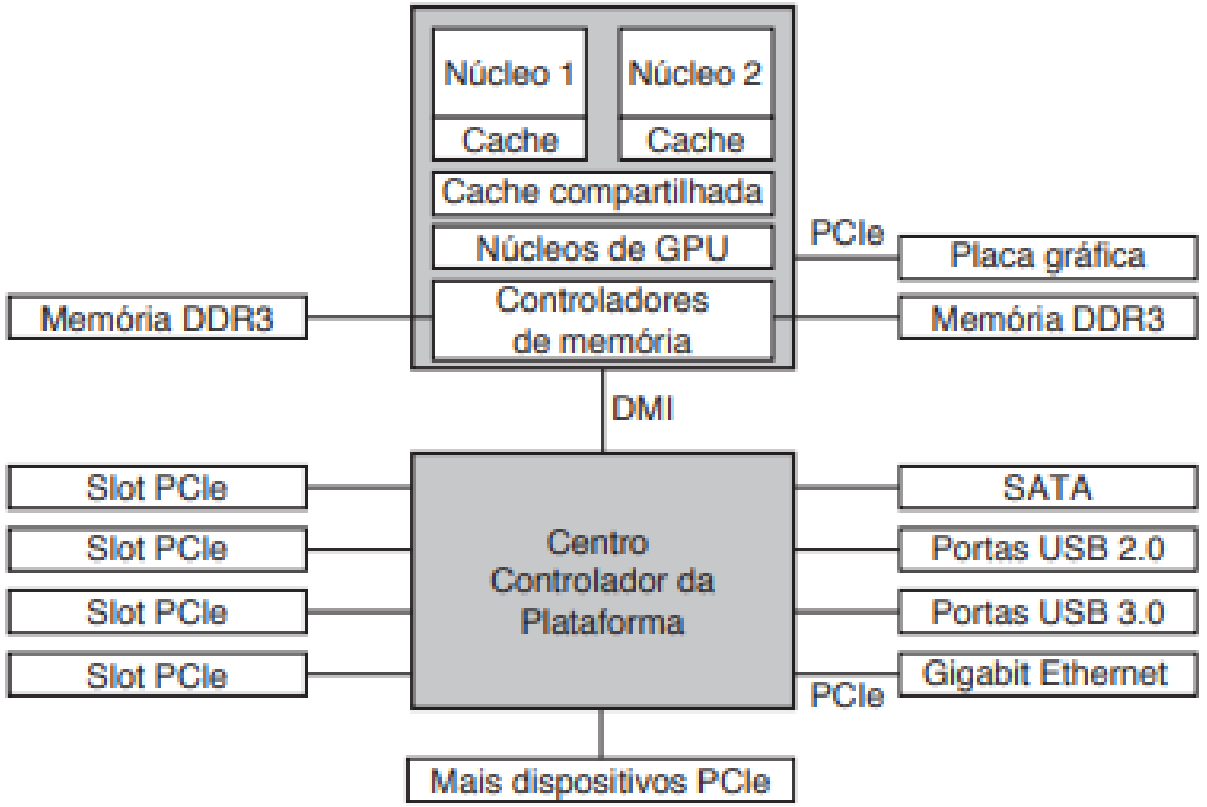
- Dispositivos de E/S também interagem intensamente com o sistema operacional. Os dispositivos de E/S consistem em geral em duas partes: um controlador e o dispositivo em si.
- O controlador é um chip ou um conjunto de chips que controla fisicamente o dispositivo. Ele aceita comandos do sistema operacional, por exemplo, para ler dados do dispositivo, e os executa.

Dispositivos de E/S

- A outra parte é o dispositivo real em si. Os dispositivos possuem interfaces relativamente simples, tanto porque eles não podem fazer muito, como para padronizá-los. A padronização é necessária para que qualquer controlador de disco SATA possa controlar qualquer disco SATA, por exemplo. SATA é a sigla para Serial ATA, e ATA por sua vez é a sigla para AT Attachment.
- Para ser usado, o driver tem de ser colocado dentro do sistema operacional de maneira que ele possa ser executado em modo núcleo. Na realidade, drivers podem ser executados fora do núcleo, e sistemas operacionais como Linux e Windows hoje em dia oferecem algum suporte para isso. A vasta maioria dos drivers ainda opera abaixo do nível do núcleo. Apenas muito poucos sistemas atuais, como o MINIX 3, operam todos os drivers em espaço do usuário. Drivers no espaço do usuário precisam ter permissão de acesso ao dispositivo de uma maneira controlada, o que não é algo trivial.
- Há três maneiras pelas quais o driver pode ser colocado no núcleo. A primeira é religar o núcleo com o novo driver e então reinicializar o sistema. Muitos sistemas UNIX mais antigos funcionam assim.
- A segunda maneira é adicionar uma entrada em um arquivo do sistema operacional dizendo-lhe que ele precisa do driver e então reinicializar o sistema. No momento da inicialização, o sistema operacional vai e encontra os drivers que ele precisa e os carrega. O Windows funciona dessa maneira.
- A terceira maneira é capacitar o sistema operacional a aceitar novos drivers enquanto estiver sendo executado e instalá-los rapidamente sem a necessidade da reinicialização. Essa maneira costumava ser rara, mas está se tornando muito mais comum hoje. Dispositivos do tipo hot-pluggable (acoplados a quente), como dispositivos USB e IEEE 1394 (discutidos a seguir), sempre precisam de drivers carregados dinamicamente.

Barramentos

- À medida que os processadores e as memórias foram ficando mais rápidos, a capacidade de um único barramento de lidar com todo o tráfego foi exigida até o limite.
- Como resultado, barramentos adicionais foram acrescentados, tanto para dispositivos de E/S mais rápidos quanto para o tráfego CPU para memória.
- Um sistema x86 grande tem muitos barramentos (por exemplo, cache, memória, PCIe, PCI, USB, SATA e DMI), cada um com uma taxa de transferência e função diferentes.
- O sistema operacional precisa ter ciência de todos eles para configuração e gerenciamento. O barramento principal é o PCIe (Peripheral Component Interconnect Express — interconexão expressa de componentes periféricos).



Barramentos

- O USB (Universal Serial Bus — barramento serial universal) foi inventado para conectar todos os dispositivos de E/S lentos, como o teclado e o mouse, ao computador.
- O USB é um barramento centralizado no qual um dispositivo-raiz interroga todos os dispositivos de E/S a cada 1 ms para ver se eles têm algum tráfego. O USB 1.0 pode lidar com uma carga agregada de 12 Mbps, o USB 2.0 aumentou a velocidade para 480 Mbps e o USB 3.0 chega a não menos que 5 Gbps.
- Qualquer dispositivo USB pode ser conectado a um computador e ele funcionará imediatamente, sem exigir uma reinicialização, algo que os dispositivos pré-USB exigiam para a consternação de uma geração de usuários frustrados.
- O barramento SCSI (Small Computer System Interface — interface pequena de sistema computacional) é um barramento de alto desempenho voltado para discos rápidos, digitalizadores de imagens e outros dispositivos que precisam de uma considerável largura de banda. Hoje em dia, eles são encontrados na maior parte das vezes em servidores e estações de trabalho, e podem operar a até 640 MB/s.

Barramentos

- O plug and play faz o sistema coletar automaticamente informações sobre os dispositivos de E/S, atribuir centralmente níveis de interrupção e endereços desses dispositivos e, então, informar a cada placa quais são os seus números.

Inicializando o computador

- De modo bem resumido, o processo de inicialização funciona da seguinte maneira: todo PC contém uma parentboard (placa-pais) (que era chamada de placa-mãe antes da onda politicamente correta atingir a indústria de computadores).
- Na placa-pais há um programa chamado de sistema BIOS (Basic Input Output System — sistema básico de entrada e saída). O BIOS conta com rotinas de E/S de baixo nível, incluindo procedimentos para ler o teclado, escrever na tela e realizar a E/S no disco, entre outras coisas. Hoje, ele fica em um flash RAM, que é não volátil, mas que pode ser atualizado pelo sistema operacional quando erros são encontrados no BIOS.
- Quando o computador é inicializado, o BIOS começa a executar. Primeiro ele confere para ver quanta RAM está instalada e se o teclado e os outros dispositivos básicos estão instalados e respondendo corretamente. Ele segue varrendo os barramentos PCIe e PCI para detectar todos os dispositivos ligados a ele. Se os dispositivos presentes forem diferentes de quando o sistema foi inicializado pela última vez, os novos dispositivos são configurados.

Inicializando o computador

- O BIOS então determina o dispositivo de inicialização tentando uma lista de dispositivos armazenados na memória CMOS.
- O primeiro setor do dispositivo de inicialização é lido na memória e executado. Ele contém um programa que normalmente examina a tabela de partições no final do setor de inicialização para determinar qual partição está ativa. Então um carregador de inicialização secundário é lido daquela partição. Esse carregador lê o sistema operacional da partição ativa e, então, o inicia.
- O sistema operacional consulta então o BIOS para conseguir as informações de configuração. Para cada dispositivo, ele confere para ver se possui o driver do dispositivo. Se não possuir, pede para o usuário inserir um CD-ROM contendo o driver (fornecido pelo fabricante do dispositivo) ou para baixá-lo da internet.
- Assim que todos os drivers dos dispositivos estiverem disponíveis, o sistema operacional os carrega no núcleo. Então ele inicializa suas tabelas, cria os processos de segundo plano necessários e inicia um programa de identificação (login) ou uma interface gráfica GUI.