Universidade Federal da Bahia

Sistemas Operacionais

MATA58

Prof. Maycon Leone M. Peixoto

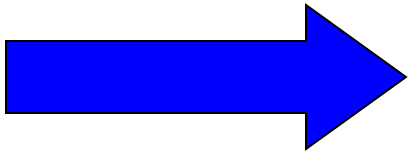
maycon.leone@ufba.br

Roteiro

- Por que é necessário um sistema operacional
- O que é um Sistema Operacional
- Histórico
- Conceitos Básicos

Por que?

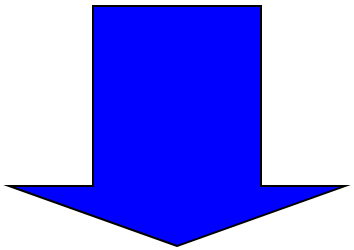
- Sistemas de computadores modernos são compostos por diversos dispositivos:
  - Processadores;
  - Memória;
  - Controladoras;
  - Monitor;
  - Teclado;
  - Mouse;
  - Impressoras;
  - Etc...



Alta Complexidade

Por que?

- Com tantos dispositivos, surge a necessidade de gerenciamento e manipulação desses diversos dispositivos
  - Tarefa difícil

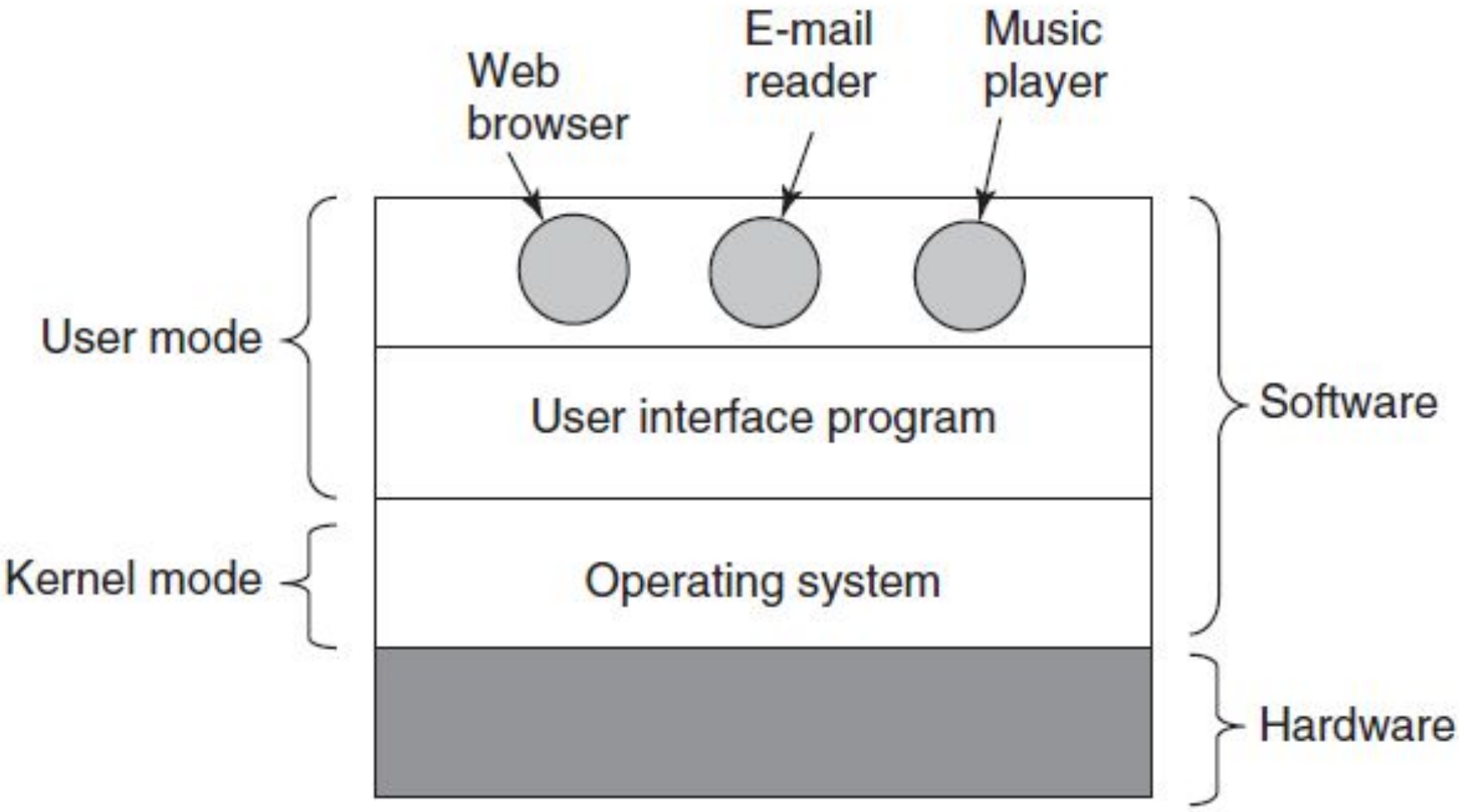


SISTEMAS OPERACIONAIS

O que é um SO?

- Software responsável por gerenciar dispositivos que compõem um sistema computacional e realizar a interação entre o usuário e esses dispositivos;
- Hardware
  - Processador;
  - Memória Principal;
  - Dispositivos de Entrada/Saída;
- Software
  - Programas de Aplicação;
  - Programas do Sistema;

Onde o SO está



Arquitetura do Sistema

Arquitetura do Sistema

- **Hardware**: Diversas camadas
  - Dispositivos físicos:
    - Circuitos (*chips*)
    - Cabos
    - Transistores
    - Capacitores
    - Memória
    - Disco rígido
    - etc...

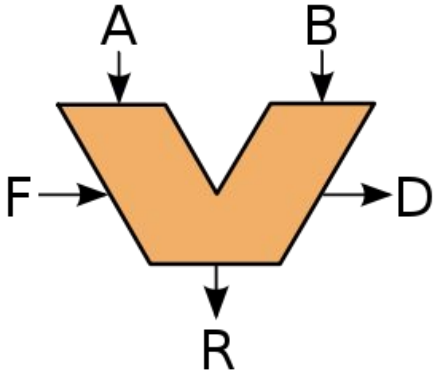
Arquitetura do Sistema

- **Micro Arquitetura**: dispositivos físicos são agrupados para formar unidades funcionais
  - CPU – processamento;
  - ULA (Unidade Lógica Aritmética) – operações aritméticas. Essas operações podem ser controladas por software (micro programas) ou por circuitos de hardware;

Unidade Lógica e Aritmética

A unidade lógica e aritmética pode realizar diversas operações, entre elas:

- Adição
- Subtração
- Operações lógicas (E, OU, XOR, INVERSÃO)
- Deslocamento (à esquerda e à direita)
- Comparação



As unidades aritméticas e lógicas mais modernas realizam também as operações de multiplicação e divisão.

Algumas operações da ULA

- NOT – Inversão
- AND – E lógico
- OR – OU lógico
- XOR – OU exclusivo
- Shift
- +, -, *, /
- =, <, >, etc

Arquitetura do Sistema

- **Linguagem de Máquina**: conjunto de instruções interpretadas pelos dispositivos que compõem a micro arquitetura;
  - Possui entre 50 e 300 instruções;
  - Realiza operações por meio de registradores;
  - Baixo nível de abstração;
  - Ex.: **Assembler**.

Assembly e Assembler

- Um programa assembly também fica acima da camada do sistema operacional, podendo fazer chamadas a ele para requisitar serviços, por exemplo de entrada e saída.
  - A linguagem Assembly é de baixo nível, porém ainda precisa ser transformada na linguagem que a máquina entende
- Um programa *montador* ou *assembler* faz a tradução da linguagem assembly para a linguagem de máquina (uma espécie de compilador, porém bastante restrito).

- Assembler é o compilador
- Assembly é a linguagem.

Processo geral de um executável

- Hello world

code/intro/hello.c

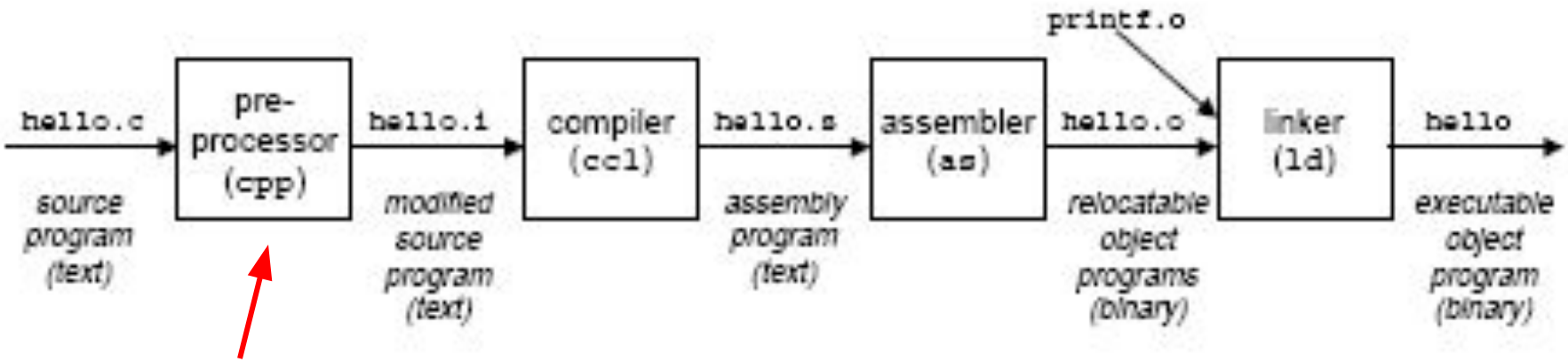
```
1 #include <stdio.h>
2
3 int main()
4 {
5     printf("hello, world\n");
6 }
```

code/intro/hello.c

Processo geral de um executável

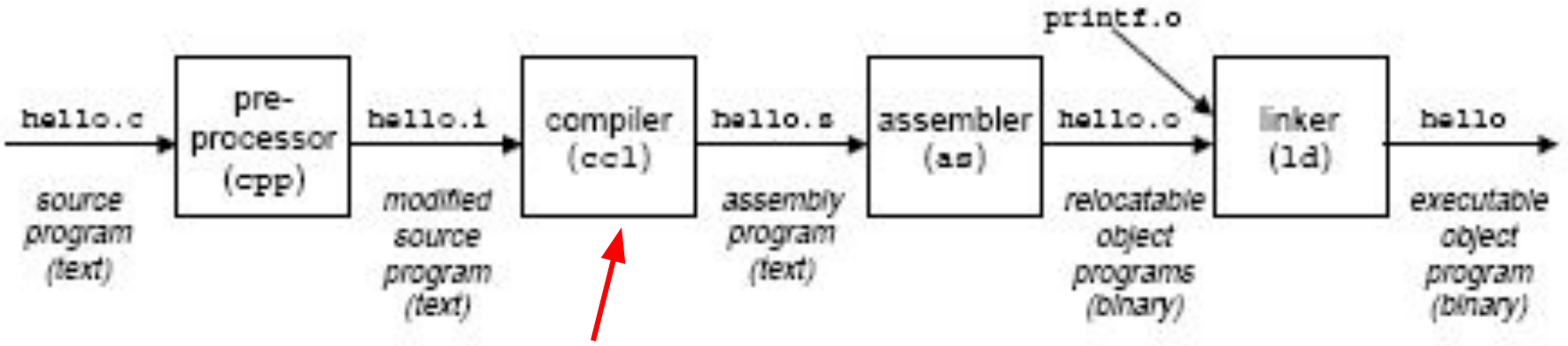
```
unix> gcc -o hello hello.c
```

(usaremos o compilador gcc do Linux no curso)



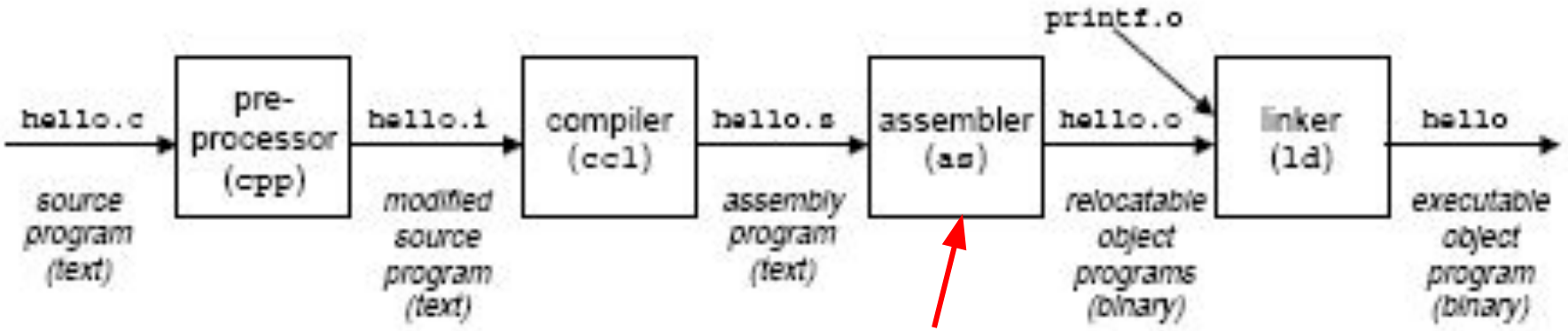
- Modifica o programa em C de acordo com diretivas começadas com #
  - Ex.: #include <stdio.h> diz ao pré-processador para ler o arquivo stdio.h e inseri-lo no programa fonte (o resultado é um programa expandido em C, normalmente com extensão .i, em Unix)

Processo geral de um executável



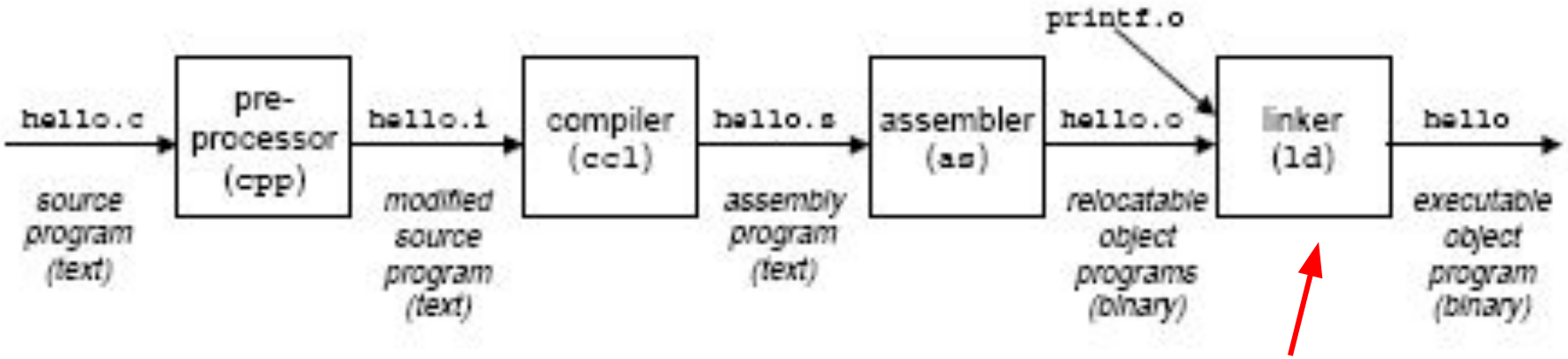
- Compilador traduz o programa .i em programa assembly. É o formato de saída comum para os compiladores nas várias linguagens de programação de alto nível
  - I.e., programas em C, Java, Fortran, etc vão ser traduzidos para a mesma linguagem Assembly.

Processo geral de um executável



- Transforma o programa assembly em um programa binário em linguagem de máquina (chamado programa objeto – extensão .o).

Processo geral de um executável



- O ligador (linker) gera o programa executável a partir do .o gerado pelo assembler. Porém, pode haver funções padrão da linguagem (por ex., printf) que não estão definidas no programa, mas em outro arquivo .o pré-compilado (no caso, printf.o).
  - O ligador faz a junção dos programas objetos necessários para gerar o executável.

Sistema Operacional

- Pode atuar de duas maneiras diferentes:
  - Como máquina estendida (*top-down*) – tornar uma tarefa de baixo nível mais fácil de ser realizada pelo usuário;
  - Como gerenciador de recursos (*bottom-up*) – gerenciar os dispositivos que compõem o computador;

Sistema Operacional como Máquina Estendida

- Ex.: como é feita a entrada/saída de um disco flexível – tarefa: Leitura e Escrita
  - SO: baixo nível de detalhes
    - Número de parâmetros;
    - Endereço de bloco a ser lido;
    - Número de setores por trilha;
    - Modo de gravação;
  - Usuário: alto nível – abstração simples
    - Visualização do arquivo a ser lido e escrito;
    - Arquivo é lido e escrito;
    - Arquivo é fechado.

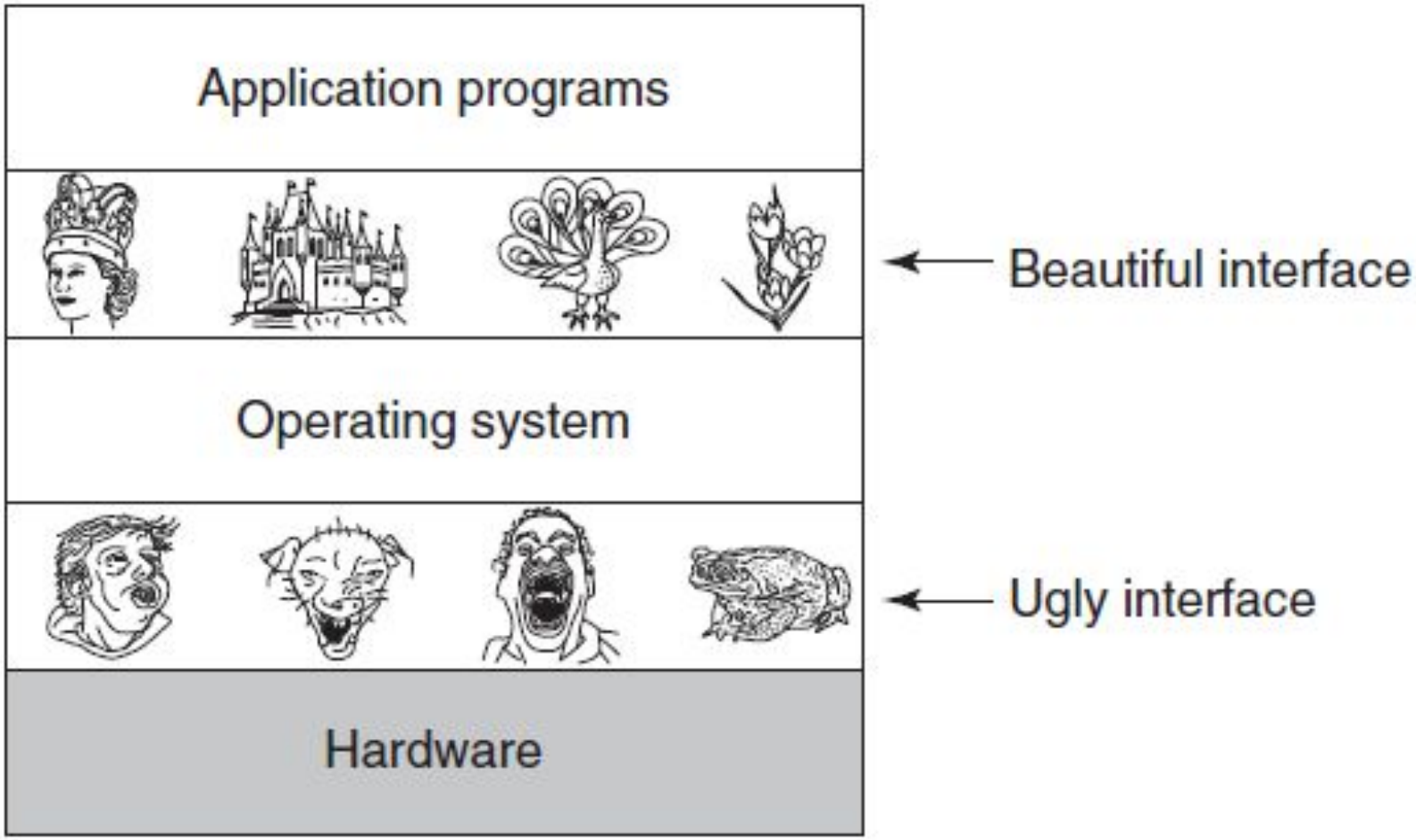
Sistema Operacional como Gerenciador de Recursos

- Gerenciar todos os dispositivos e recursos disponíveis no computador
  - Ex.: se dois processos querem acessar um mesmo recurso, por exemplo, uma impressora, o SO é responsável por estabelecer uma ordem para que ambos os processos possam realizar sua tarefa de utilizar a impressora.
  - Uso do HD;
  - Uso da memória;
- Coordena a alocação controlada e ordenada dos recursos;

Sistemas Operacionais

- Possuem muitas linhas de códigos.
- O núcleo do Windows/Linux possui 5M de linhas de códigos.
  - Em formato de livro: 50 linhas por página e 1000 páginas por volume = 100 volumes.
  - Com as bibliotecas essências: 70M (ainda sem incluir Windows Explorer, Windows Media Player, etc.)

Função do SO


Application programs
Operating system
Hardware
Beautiful interface
Ugly interface

Roteiro

- Por que é necessário um sistema operacional
- O que é um Sistema Operacional
- **Histórico**
- Conceitos Básicos

Histórico de Evolução (SO)

- Meados do século XIX: Charles Babbage (1792-1871), por volta de 1833, projetou o primeiro computador. No entanto, a pouca tecnologia da época não permitiu que o projeto tivesse sucesso.
  - Máquina analítica:
    - Não tinha um SO;
    - Percebeu que precisava de um software que possibilitasse seu uso;
    - Contratou Ada Lovelace, que se tornou a 1ª programadora

Histórico de Evolução
Primeira Geração

- Primeira Geração (1940-1955): Válvulas
  - Prof. John Atanasoff e seu aluno Clifford Berry construíram o que é agora considerado o primeiro computador digital funcional na Universidade de Iowa. Possuía 300 tubos de vácuos.
  - Z3 (Berlim) -> Relés eletromecânicas.
  - Colossus (Inglaterra) –> 1944 (Grupo incluindo Alan Turgin)
- Máquinas enormes que ocupavam salas imensas;
- Dezenas de milhares de válvulas – 20.000
- Não existiam ainda os conceitos de sistema operacional e linguagem de programação de alto nível;

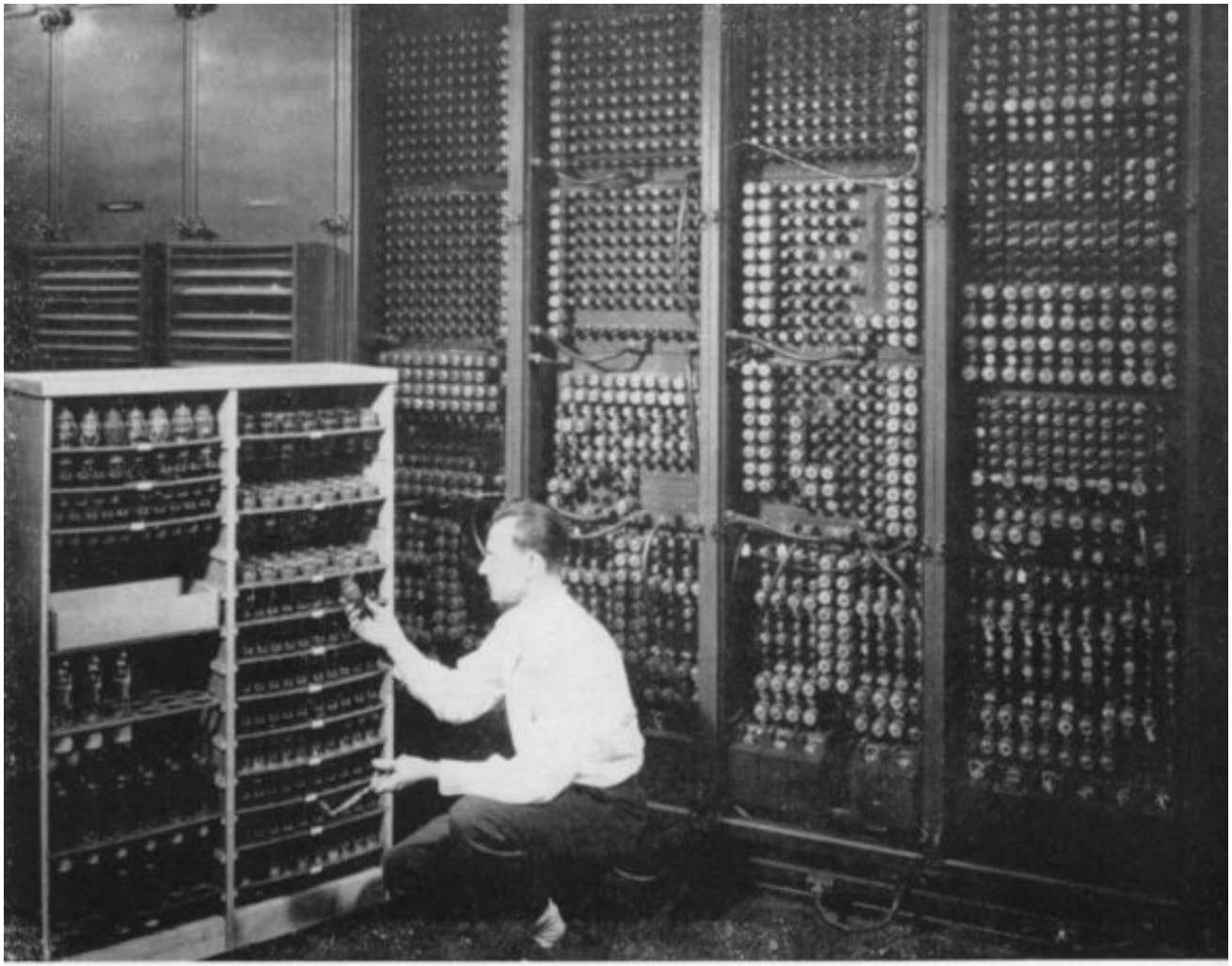
Histórico de Evolução
Primeira Geração

- Mesmo grupo de pessoas projetava, construía, programava, operava e fazia a manutenção de cada máquina;
- O acesso às máquinas era feito por meio de reserva de tempo: cada usuário fazia sua programação diretamente nos painéis das máquinas ;
- Máquinas realizavam cálculos numéricos;

Histórico de Evolução
Primeira Geração

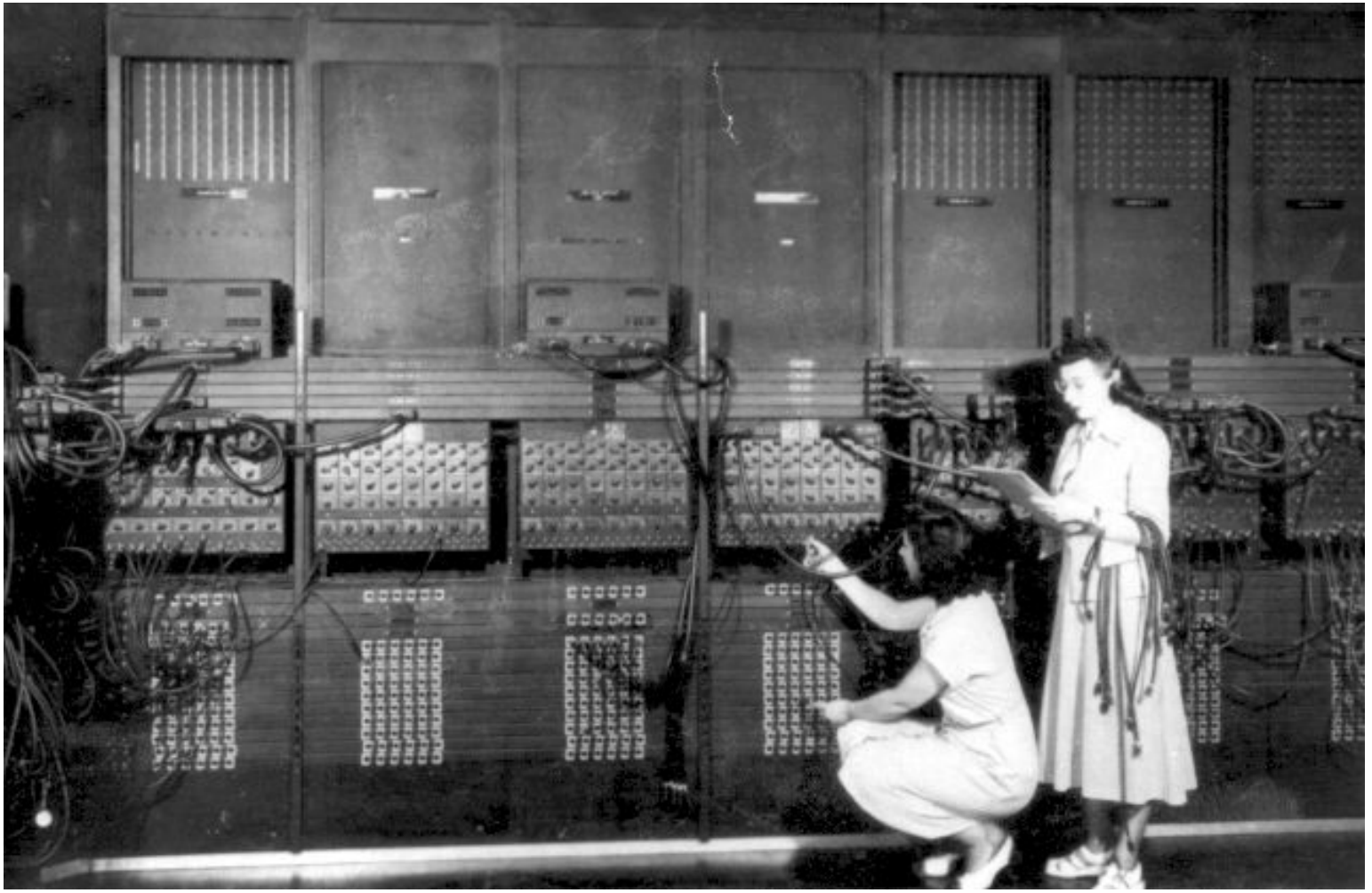
- Final dos anos 40: primeiro computador eletrônico ENIAC (***E****lectronic* ***N****umerical* ***I****ntegrator* ***A****nd* ***C****omputer*);
- 1950: surgem os cartões perfurados
  - Os programas eram codificados nos cartões e sua leitura era feita por máquina operadores de máquina
- Nasce o Assembler/Assembly;

ENIAC

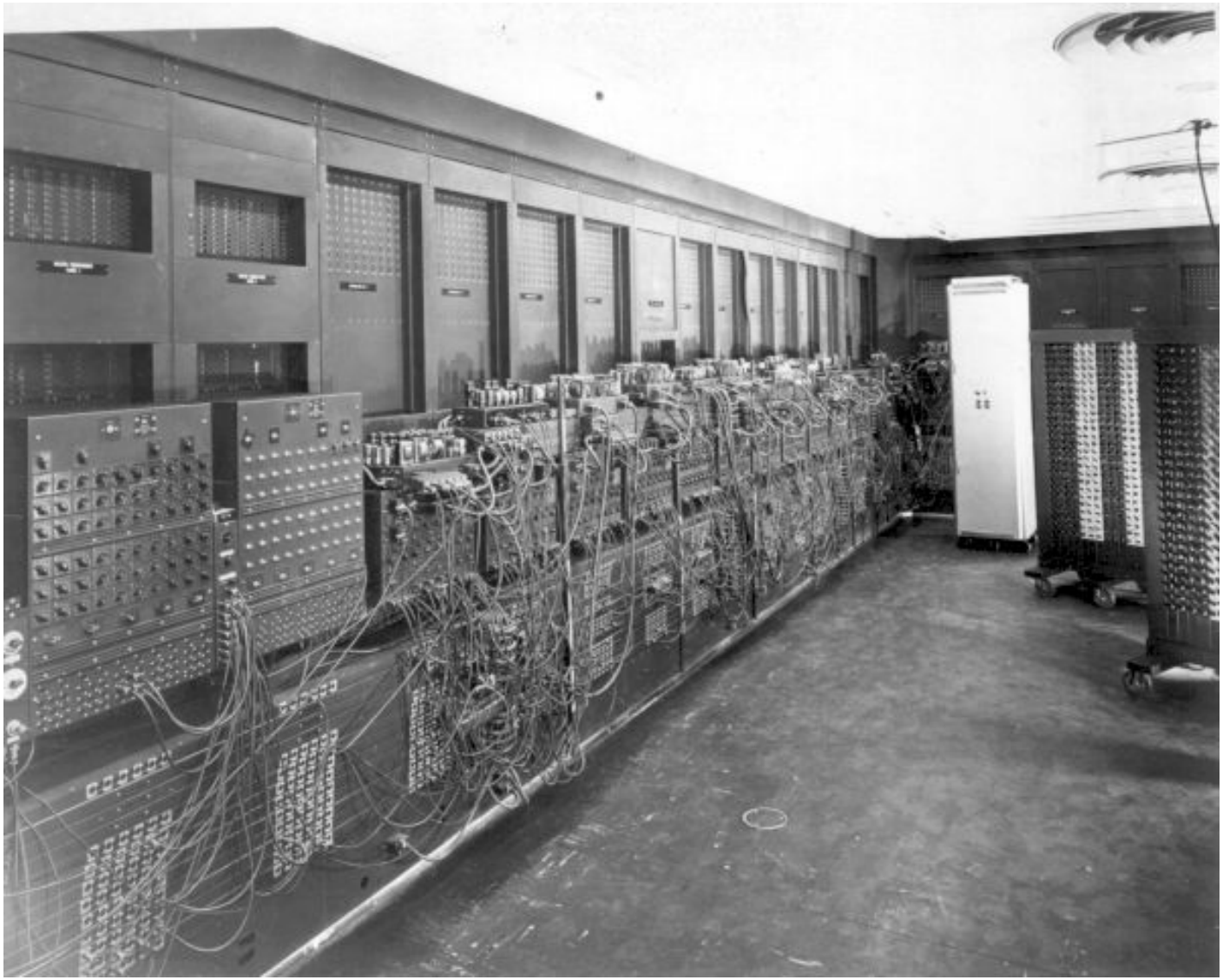


Replacing a bad tube meant checking among ENIAC's 19,000 possibilities.

ENIAC

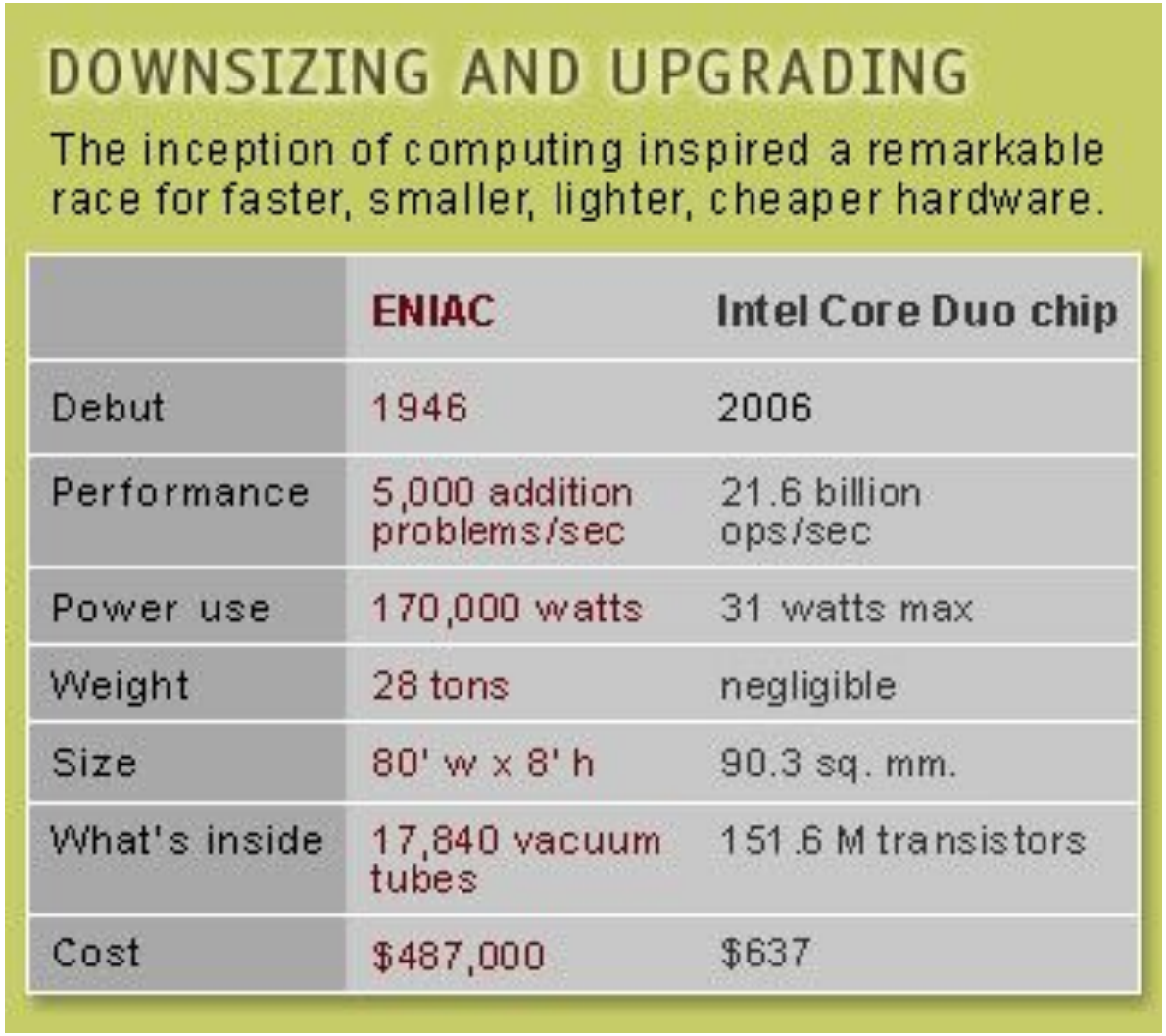


ENIAC



ENIAC

Comparação



DOWNSIZING AND UPGRADING

The inception of computing inspired a remarkable race for faster, smaller, lighter, cheaper hardware.

	ENIAC	Intel Core Duo chip
Debut	1946	2006
Performance	5,000 addition problems/sec	21.6 billion ops/sec
Power use	170,000 watts	31 watts max
Weight	28 tons	negligible
Size	80' w x 8' h	90.3 sq. mm.
What's inside	17,840 vacuum tubes	151.6 M transistors
Cost	$487,000	$637

Histórico de Evolução
Segunda Geração

- Segunda Geração (1955-1965) – Transistores e Sistemas em *Batch*
  - O desenvolvimento dos transistores tornou o computador mais confiável possibilitando sua comercialização - *Mainframes*;
  - Separação entre projetistas, fabricante, programadores e técnicos de manutenção;
  - No entanto, devido aos altos custos, poucos tinham acesso a essa tecnologia – somente grandes empresas, órgãos governamentais ou universidades;

Histórico de Evolução
Segunda Geração

- Surge a idéia de linguagem de programação de alto nível – Fortran (desenvolvida pela IBM – 1954-1957);
- Cartões perfurados ainda são utilizados
  - Operação: cada programa (*job*) ou conjunto de programas escrito e perfurado por um programador era entregue ao operador da máquina para que o mesmo fosse processado – alto custo
  - Sistemas em *Batch* (lote)
    - Consistia em coletar um conjunto de *jobs* e fazer a gravação desse conjunto para uma fita magnética

Histórico de Evolução
Segunda Geração

Sistema em Batch

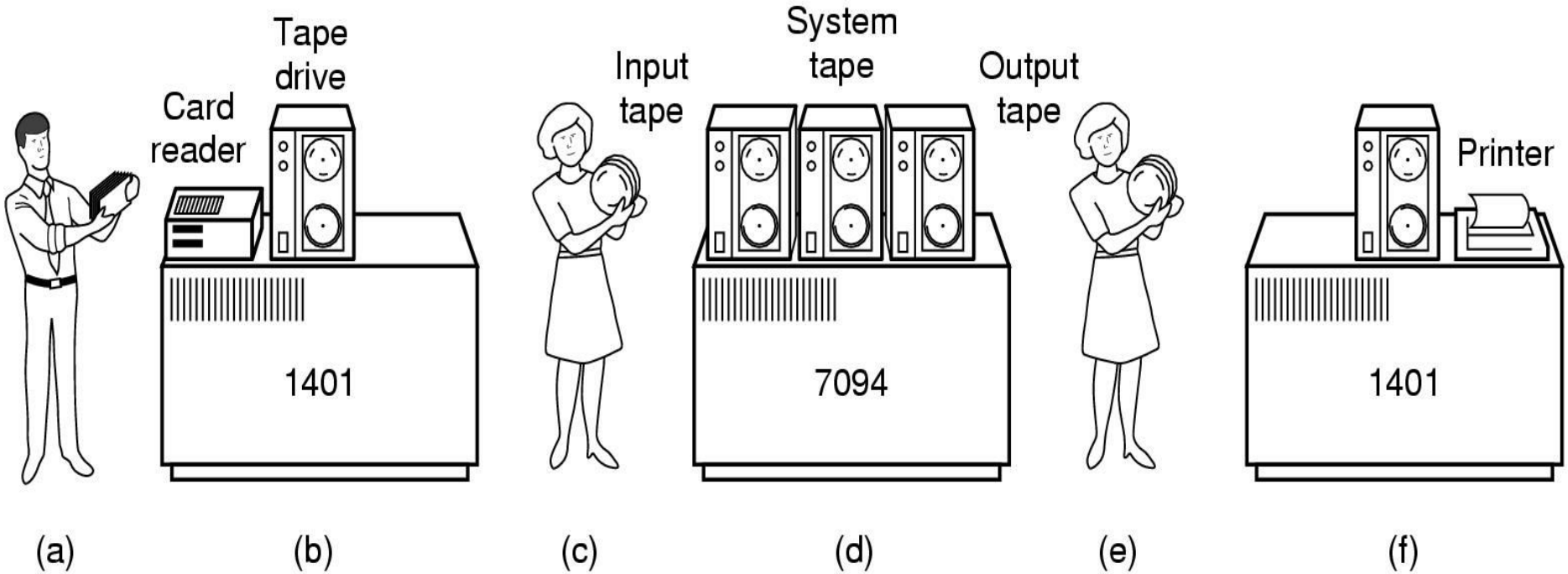
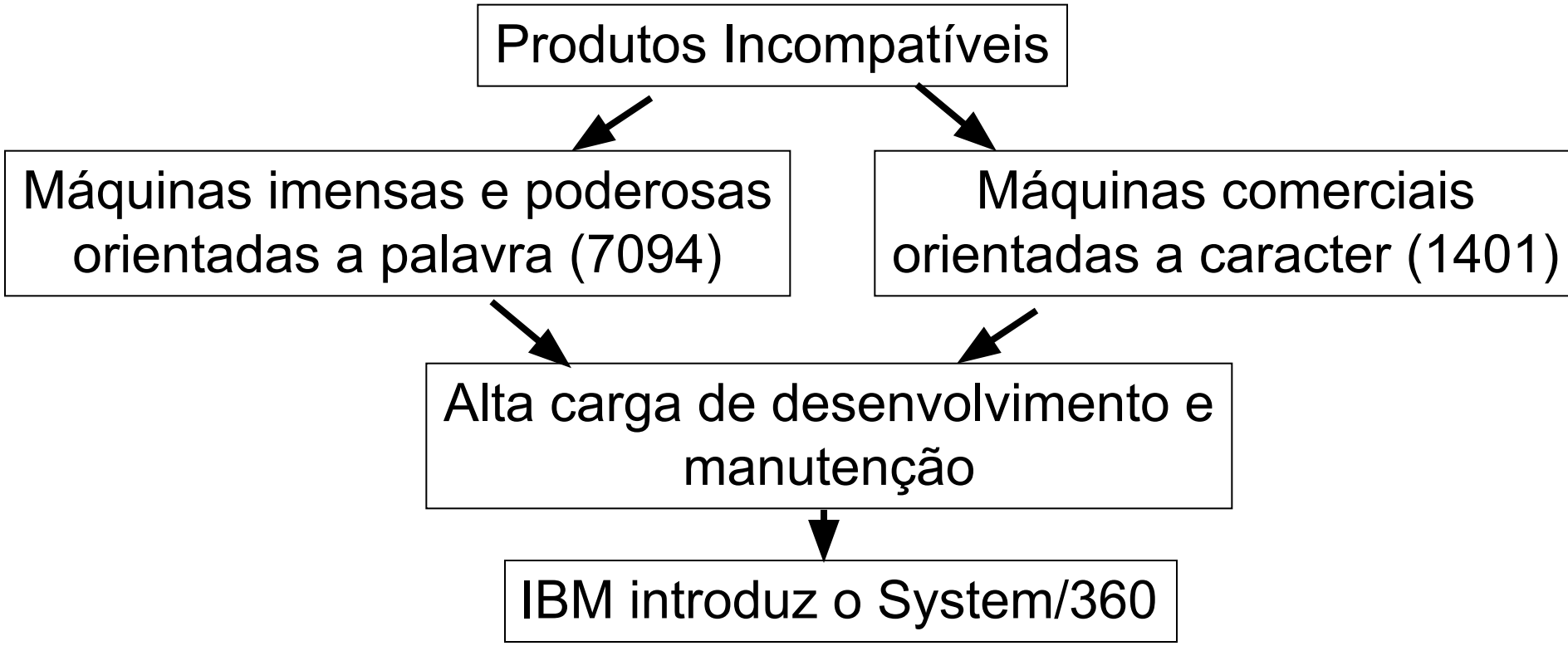


Figure 1-3. An early batch system. (a) Programmers bring cards to 1401. (b) 1401 reads batch of jobs onto tape. (c) Operator carries input tape to 7094. (d) 7094 does computing. (e) Operator carries output tape to 1401. (f) 1401 prints output.

Histórico de Evolução
Terceira Geração

- Terceira Geração (1965-1980) – Circuitos integrados e Multiprogramação



Histórico de Evolução
Terceira Geração

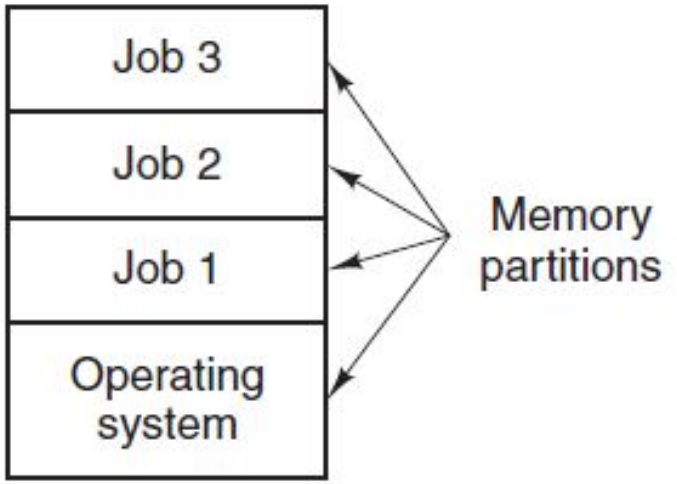
- System/360
  - Série de máquinas com software compatível;
  - Essas máquinas diferiam apenas no preço e desempenho, variando da 1401 até a 7094;
  - Foi a primeira a usar circuito integrado em pequena escala, ao invés de transistores;
  - O sistema operacional era o OS/360
    - Sua maior vantagem era também sua maior fraqueza: SO enorme e muito complexo, pois precisava realizar as funções de todas as máquinas

Histórico de Evolução
Terceira Geração

- Aplicações que eram CPU-bound não tinham problema com relação ao tempo que se precisava esperar para realizar E/S
- Aplicações que eram IO-bound gastavam de 80 a 90% do tempo realizando E/S
  - Enquanto isso, a CPU ficava parada
  - Solução: Multiprogramação

Histórico de Evolução
Terceira Geração

- **Multiprogramação**:
  - Dividir a memória em diversas partes e alocar a cada uma dessas partes um *job*.
  - Manter na memória simultaneamente uma quantidade de *jobs* suficientes para ocupar 100% do tempo do processador, diminuindo a ociosidade.
  - Importante: o hardware é que protegia cada um dos *jobs* contra acesso indevidos de outros *jobs*.

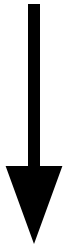


Histórico de Evolução
Terceira Geração

- ***Spooling*** (*Simultaneous Peripheral Operation On Line*):
  - Possibilitar que a leitura de cartões de *jobs* fosse feita direta do disco;
  - Assim que um *job* terminava, o sistema operacional já alocava o novo *job* à uma partição livre da memória direto do disco;
  - Eliminação de máquinas como as 1401 e a necessidade de se ficar andando entre as máquinas

Histórico de Evolução
Terceira Geração

- Mesmo com o surgimento de novas tecnologias, o tempo de processamento ainda era algo crítico. Para corrigir um erro de programação, por exemplo, o programador poderia levar horas pois cada job era tratado dentro de um lote



TimeSharing

Histórico de Evolução
Terceira Geração

- ***TimeSharing***: cada usuário tinha um terminal *on-line* à disposição;
  - Primeiro sistema *TimeSharing*: CTSS (*Compatible Time Sharing System*) – 7094 modificado.
  - Ex.: se 20 usuários estão ativos e 17 estão ausentes, o processador é alocado a cada um dos 3 *jobs* que estão sendo executados;

Histórico de Evolução
Quarta Geração

- Quarta Geração (1980-Atual) – Computadores Pessoais
  - Com a tecnologia de circuitos integrados de larga escala (LSI) surgem *chips* com milhares de transistores encapsulados em um centímetro quadrado de silício
    - Intel – 8080 (1974)
    - IBM – PC (início dos anos 80)
    - Apple - Macintosh

Histórico de Evolução
Quinta Geração - (1990-hoje): Computadores Móveis

- Primeiro telefone móvel real apareceu em 1946 e pesava 40 kilos.
  - "You could take it wherever you went as long as you had a car in which to carry it"
- O primeiro telefone verdade de mão apareceu nos anos 70 com aproximadamente 1 kilo.
  - Ficou conhecido como "the brick".
- Hoje, 90% da população mundial utiliza um telefone móvel.
- Sistemas Operacionais:
  - Google Android em primeiro e Apple iOS em segundo
  - Mas isso pode mudar, de forma semelhante o que ocorreu com o Symbiam. (Samsung, Sony Ericsson, Motorola, e principalmente Nokia)
  - Vantagens do Android: Open Source, adaptação de hardware, grande comunidade

Tipos de Sistemas Operacionais

- Propósito Geral: Windows XP, MacOS X e Linux.
- Sistemas Operacionais de Tempo Real
    - Gerenciamento de Tempo;
        - soft real-time systems, nos quais a perda de prazos implica na degradação do serviço prestado. Um exemplo seria o suporte à gravação de CDs ou à reprodução de músicas.
        - hard real-time systems a perda de prazos pelo sistema pode perturbar o objeto controlado, com graves consequências humanas, econômicas ou ambientais. Exemplos: controle de funcionamento de uma turbina de avião a jato ou de uma caldeira industrial. QNX, RT-Linux e VxWorks.
            - » Gerenciamento de processos críticos (aviões, caldeiras);
- Sistemas Operacionais Embarcados: telefones, aparelhos eletrodomésticos; PDAs;

Roteiro

- Por que é necessário um sistema operacional
- O que é um Sistema Operacional
- Histórico
- **Conceitos Básicos**

Conceitos Básicos de Sistemas Operacionais

- Principais conceitos:
  - Processo;
  - Memória;
  - Chamadas de Sistema;

Processos

- Ex.: processo bloqueado (suspenso)

Quando o SO suspende um processo *P1* temporariamente para executar um processo *P2*, o processo *P1* deve ser reiniciado exatamente no mesmo estado no qual estava ao ser suspenso. Para tanto, todas as informações a respeito do processo *P1* são armazenadas em uma **tabela de processos** (*process table*). Essa tabela é um vetor ou uma lista encadeada de estruturas.

Processos

- Um processo pode resultar na execução de outros processos, chamados de processos-filhos:
  - Características para a hierarquia de processos:
    - Comunicação (Interação) e Sincronização;
    - Segurança e proteção;
    - Uma árvore de no máximo três níveis;
- Escalonadores de processos – processo que escolhe qual será o próximo processo a ser executado;
  - Diversas técnicas para escalonamento de processos;

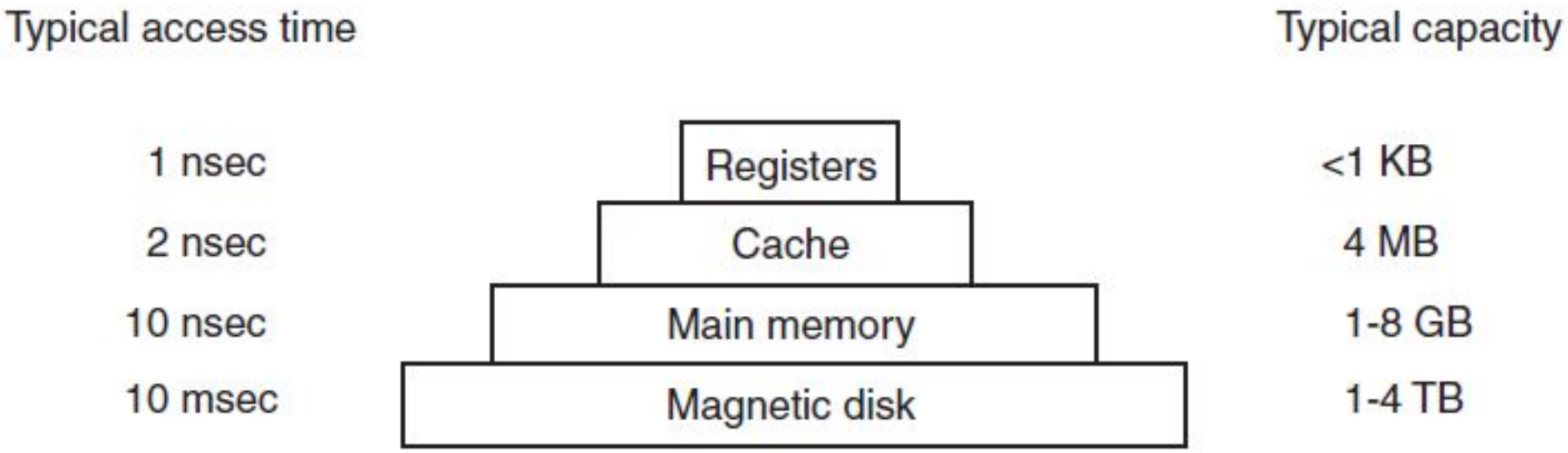
Processos

- Comunicação e sincronismo entre processos – solução:
  - Semáforos;
  - Monitores;
  - Instruções especiais em hardware;
  - Troca de mensagens;

Gerenciamento de Memória

- Gerenciamento elementar (década de 60)
  - Sistema monoprogramado;
  - Sem paginação:
    - Apenas um processo na memória;
    - Acesso a toda a memória;
- Gerenciamento mais avançado (atualidade)
  - Sistema multiprogramado;
  - Mais de um processo na memória;
  - Chaveamento de processos: por entrada/saída ou por limite de tempo (sistema de tempo compartilhado);

Típica Hierarquia de Memória



Memórias Digitais

Existem dois tipos básicos de memória :

Memória ROM - **R**ead **O**nly **M**emory (memória apenas de leitura)

Mémória RAM - **R**andom **A**ccess **M**emory (memória de acesso aleatório)

Memória ROM

- Serve somente como leitura;
- Uma vez seus dados gravados, não pode mais ser alterado, apenas através de procedimentos especiais;
- Suas informações são lidas pelo computador;
- São memórias não-voláteis, ou seja, quando o computador for desligado ela não perde suas informações.

Tipos de ROM

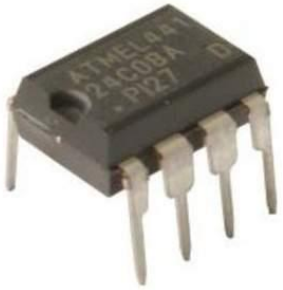
- Gravada durante a fabricação:
  - Muito cara para pequenas quantidades.
- Programável (uma vez):
  - PROM.
  - Precisa de equipamento especial para programar.
- Lida “na maioria das vezes”:
  - Erasable Programmable (EPROM).
    - Apagada por UV.
  - Electrically Erasable (EEPROM):
    - Leva muito mais tempo para escrever que para ler.
  - Memória flash:
    - Apaga memória inteira eletricamente.

Principais Tipos de Memória ROM

- PROM – (**P**rogrammable **R**ead **O**nly **M**emory): a gravação nesses dispositivos é feito por meio de reação física através de equipamentos que impulsionam elementos elétricos;
  - mediante o rompimento de "fusíveis", que são queimados de forma a produzir o registro de sinais digitais
- Uma vez acontecido isso, suas informações não podem ser alteradas;

Principais Tipos de Memória ROM

- EPROM – (**P**rogrammable **R**ead **O**nly **M**emory): as memórias EPROM têm como principal característica a capacidade de permitir que dados sejam regravados no dispositivo.
- Para isso, tem um auxílio de um componente que emite luz ultravioleta.
- Nesse processo, os dados gravados precisam ser apagados por completo. Somente depois disso é que uma nova gravação pode ser feita;

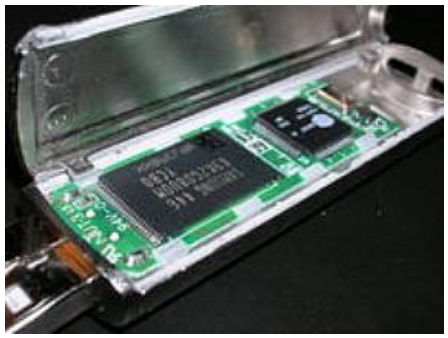


Principais Tipos de Memória ROM

- EEPROM (Electrically-Erasable Programmable Read-Only Memory - Memória Somente de Leitura Apagável-Programável Eletricamente): também permite a regravação de dados;
- Esse processo é feito eletricamente, fazendo com que não seja necessário retirar o dispositivo para um equipamento específico.

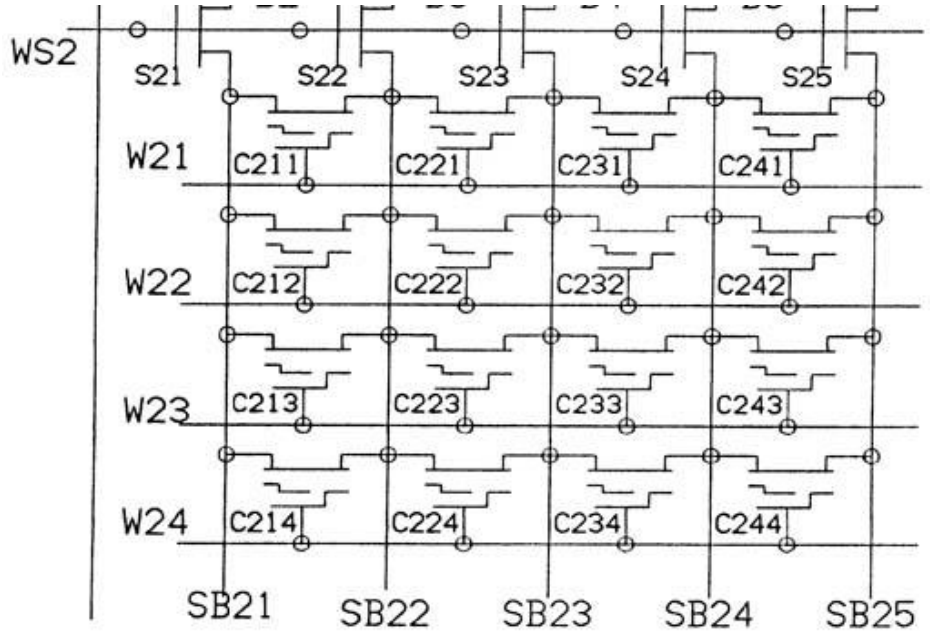
Principais Tipos de Memória ROM

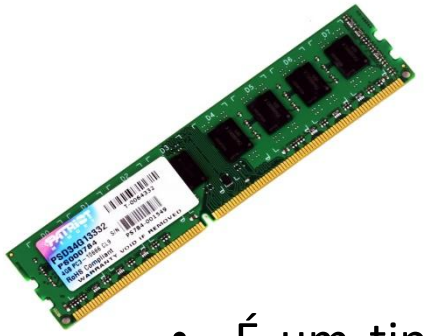
- Memória FLASH: também podem ser vistas como um tipo de EEPROM;
- O processo de gravação (e regravação) é muito mais rápido do que a EEPROM;
- É mais durável e comporta um volume maior de dados.



Flash

- **GRAVAR** -- Quando habilitamos a linha W22, por exemplo, os 5 bits aplicados nas entradas de SB21 a SB25 são gravados nos transistores de C212 a C242.
- **LER** -- Para ler, basta habilitar as mesmas linhas com tensão apropriada e sensoriar os níveis lógicos nas mesmas saídas.
- Evidentemente, este é um exemplo simplificado já que existem bilhões de células semelhantes a estas num único pen drive.





Memória RAM

- É um tipo de memória essencial para o computador, sendo usada para guardar dados e instruções de um programa.
- Tem como características fundamentais, a volatilidade, ou seja, o seu conteúdo é perdido quando o computador é desligado;
  - Quando não há energia elétrica.
- O processo de gravação é destrutivo e a leitura é não destrutiva.
- Memória de acesso aleatório;
- Umas das principais partes dos computadores;
- É nessa memória onde fica armazenado os dados que o processador quer trabalhar, ou está trabalhando;
- Sua forma de armazenamento de dados é bastante rápida em comparação a memória ROM;

SRAM - Estática

- Usam flip-flops.
- Não requer atualização dos dados.
- Consome mais energia (o que gera mais calor) comparando-se com a memória dinâmica.
- Possui capacidade de armazenamento bem menor que a memória dinâmica.
- Esse tipo é muito mais rápido que as memórias DRAM;
- Armazena menos dados e possui preço elevado se considerarmos o custo por megabyte.
- Bits armazenados como chaves ligado/desligado.
- Sem carga para vazar.
- Não precisa de *refresh* quando alimentada.
- Construção mais complexa.
- Maior por bit.
- Mais cara.
- Não precisa de circuitos de *refresh.*
- Mais rápida.
- Cache.
- Digital: flip-flops.

DRAM - dinâmica

- Memória baseada na tecnologia de capacitores.
  - Bits armazenados com carga em capacitores.
- Requer a atualização periódica do conteúdo de cada célula do chip, consumindo, assim, pequenas quantidades de energia.
- Possui acesso lento aos dados.
- Uma importante vantagem é a grande capacidade de armazenamento oferecida por este tipo de tecnologia.
- As cargas vazam.
  - Precisa de renovação mesmo se alimentada.
  - Precisa de circuitos de *refresh.*
- Construção mais simples.
- Menor por bit.
- Mais barata.
- Mais lenta.
- Memória principal.

Memória RAM

	Vantagens	**Desvantagens**
RAM Dinâmica	-Barata -Baixo Consumo -Alta Densidade	-Necessita de Atualização -Lenta
RAM Estática	-Rápida -Não necessita de atualização	-Mais cara -Consome Mais Energia -Baixa Densidade

SRAM *versus* DRAM

- Ambas voláteis.
  - É preciso energia para preservar os dados.
- Célula dinâmica:
  - Mais simples de construir, menor.
  - Mais densa.
  - Mais barata.
  - Precisa de *refresh.*
  - Maiores unidades de memória.
- Estática:
  - Mais rápida.
  - Cache.

RAMs e ROMs

- **RAMs** podem ser de duas variedades, estáticas e dinâmicas.
- Nas estáticas (**Static RAMs** – **SRAMs**), a construção interna usa circuitos similares ao nosso *flip-flop* D básico.
- RAMS dinâmicas (**Dynamic RAMs** – **DRAMs**), ao contrário, não usam flip-flops.
- Em vez disso, uma RAM dinâmica é um arranjo de células, cada uma contendo um transistor e um pequenino capacitor.

RAMs e ROMs

- Em muitas aplicações, como brinquedos, eletrodomésticos e carros, o programa e alguns dos dados devem permanecer armazenados mesmo quando o fornecimento de energia for interrompido.

- Uma vez instalados, nem o programa nem os dados são alterados.

- Esses requisitos levaram ao desenvolvimento de **ROMs** (**Read-Only Memories** – **memórias somente de leitura**), que não podem ser alteradas nem apagadas, seja intencionalmente ou não.

Módulos de Memória RAM

- Entendemos como módulo ou, ainda, pente, uma pequena placa onde são instalados os encapsulamentos de memória.
- Essa placa é encaixada na placa-mãe por meio de encaixes (slots)

Módulo SIPP

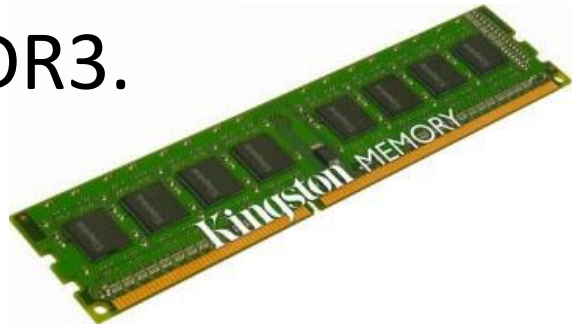
- É um dos primeiros tipos de módulos que chegaram ao mercado.
- É formato por chips com encapsulamento DIP. Em geral, esses módulos eram soldados na placa-mãe;

Módulo SIMM

- Módulos deste tipo não eram soldados, mas encaixados na placa-mãe.
- A primeira versão continha 30 terminais de contato (SIMM de 30 vias) e era formada por um conjunto de 8 chips.
- Posteriormente surgiu uma versão com 72 pinos;
- 30 vias equivaliam a 1 até 16 MB
- 72 vias equivaliam a 4 até 64 MB

Módulo DIMM (Double In-Line Memory Module)

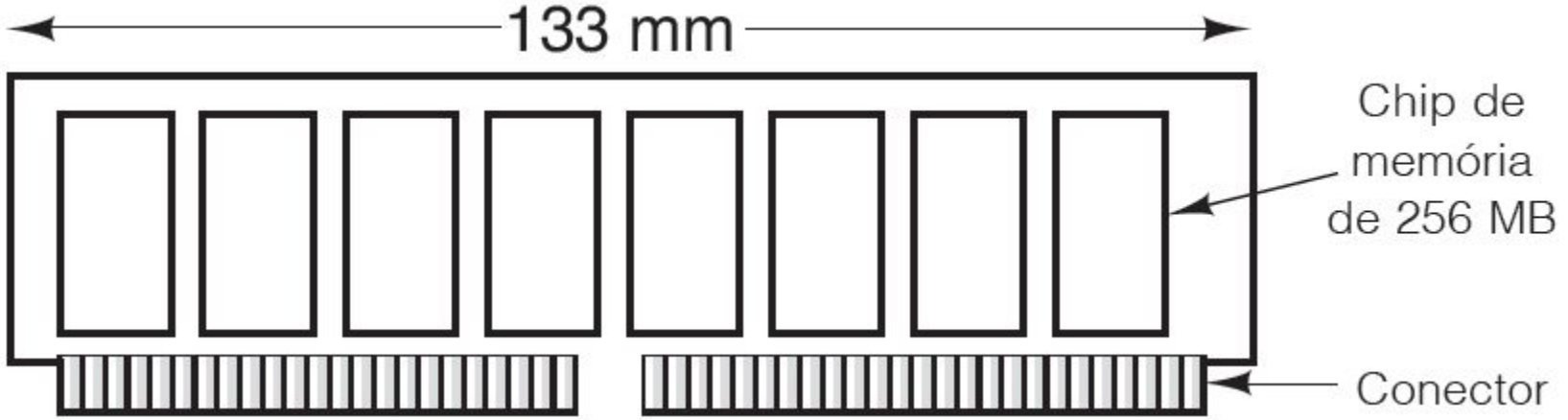
- Levam esse nome por terem terminais de contatos em ambos os lados do pente;
- São capazes de transmitir 64 bits por vez. A primeira versão aplicada em memória SDR SDRAM - tinha 168 pinos. Em seguida, foram lançados módulos de 184 vias, utilizados em
- memórias DDR, e módulos de 240 vias, utilizados em módulos DDR2 e DDR3.



Empacotamento e tipos de memória

- Uma configuração típica de DIMM poderia ter oito chips de dados com 256 MB cada. Então, o módulo inteiro conteria 2 GB.

- Muitos computadores têm espaço para quatro módulos, o que dá uma capacidade total de 8 GB se usarem módulos de 2 GB e mais, se usarem módulos maiores.

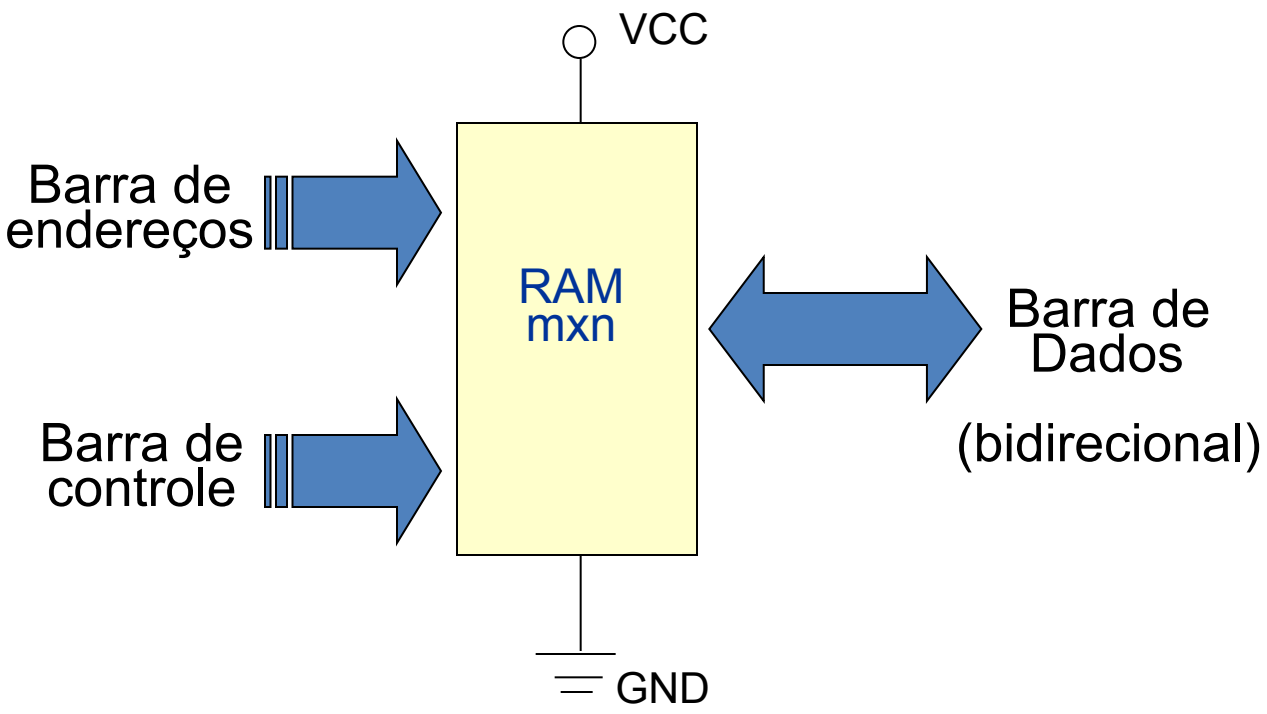


Módulos DIMM

- Existe um padrão DIMM conhecido como SODIMM, são menores e são utilizados em notebooks.

Memórias RAM

Estrutura de Acesso de uma RAM :



RAM mxn: “m” endereços de “n” bits

Características de Memória

- Localização:
  - Processador: registradores;
  - Interna: memória principal;
  - Externa: dispositivos de armazenamento periféricos (controladores de E/S);

Características de Memória

- Capacidade:
  - Palavras:
    - Unidade “natural”de organização do computador
    - 8 bits, 16 bits e 32 bits;
  - Bytes: normalmente utilizado para memórias externas;

Características de Memória

- Unidade de transferência:
  - Linha de dados do módulo de memória:
    - Palavra: tamanho de bits usado na representação de números inteiros e instruções (Geralmente);
    - Unidade endereçável: normalmente é a palavra, porém existem sistemas onde podemos endereçar bytes;
    - Unidade de transferência: número de bits que podem ser lidos ou escritos de cada vez (blocos).
      - Interna: Normalmente controlada pela largura do barramento.
      - Externa: Normalmente um bloco que é muito maior que uma palavra.

Características de Memória

- Método de acesso:
  - Acesso sequencial: acesso de registros feito sequencialmente (tempo de acesso variável);
    - Ex: Unidade de fita
  - Acesso direto: acesso de registros através de sua vizinhança (tempo de acesso variável);
    - Unidades de disco
  - Acesso aleatório: acesso de uma posição de memória através de um endereço único;
    - Memória principal
  - Associativo: palavra buscada com base no seu conteúdo;
    - Memória cache

Características de Memória

- Desempenho:
  - Tempo de acesso:
    - Aleatório: tempo para realizar operação de escrita ou de leitura
    - Não aleatório: tempo para posicionar o mecanismo de leitura-escrita;
  - Tempo de ciclo de memória: (acesso aleatório)
    - tempo entre duas operações de acesso;
      - Barramento do sistema
  - Taxa de transferência: taxa de transferência de dados (geralmente bits/segundo).

Características de Memória

- Tecnologias:
  - Memórias de semicondutores;
  - Memórias de superfície magnéticas;
  - Memórias ópticas;
  - Memórias magneto-ópticas.

Características de Memórias

- Características físicas:
  - Volátil;
  - Não-volátil;
  - Apagável;
  - Não-apagável.

Hierarquia de Memórias

- Pontos importantes de projeto:
  - Capacidade;
  - Velocidade;
  - Custo.

- Relações existentes:
  - Tempo de acesso mais rápido, custo por bit maior;
  - Capacidade maior, custo por bit menor;
  - Capacidade maior, mais lenta.

Gerenciamento de Memória

Hierarquia de Memória:

- Cache – vários sub-níveis
- Memória Principal
- Disco

Gerenciamento de Memória

- Registrador: O registrador de uma CPU é uma unidade de memória capaz de armazenar n bits.
- Os registradores estão no topo da hierarquia de memória, sendo assim, são o meio mais rápido e caro de se armazenar um dado.
- São circuitos digitais capazes de armazenar e deslocar informações binárias, e são tipicamente usados como um dispositivo de armazenamento temporário.

Gerenciamento de Memória

Hierarquia de Memória:

- Cache
  - Pequena quantidade – k/M bytes
  - Alto custo por byte
  - Muito rápida
  - Volátil
- Memória Principal
- Disco

Cache

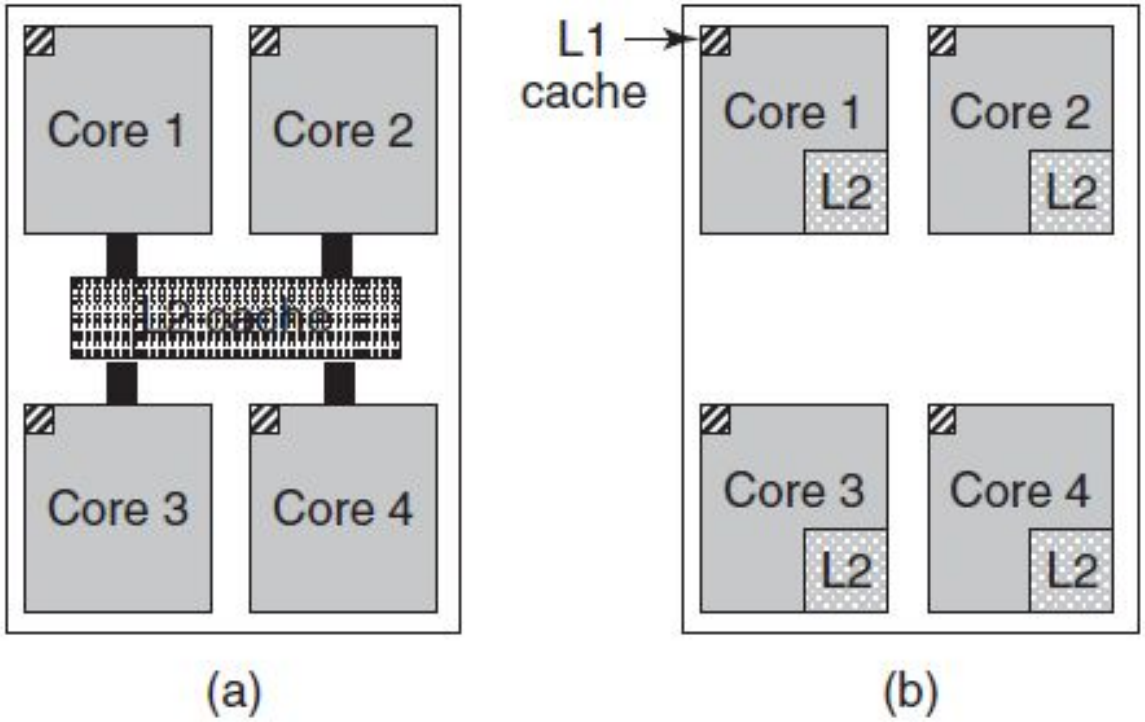


Figure 1-8. (a) A quad-core chip with a shared L2 cache. (b) A quad-core chip with separate L2 caches.

Gerenciamento de Memória

Hierarquia de Memória:

- Cache
- Memória Principal
  - Quantidade intermediária – M/G bytes
  - Custo médio por byte
  - Velocidade média
  - Volátil
- Disco

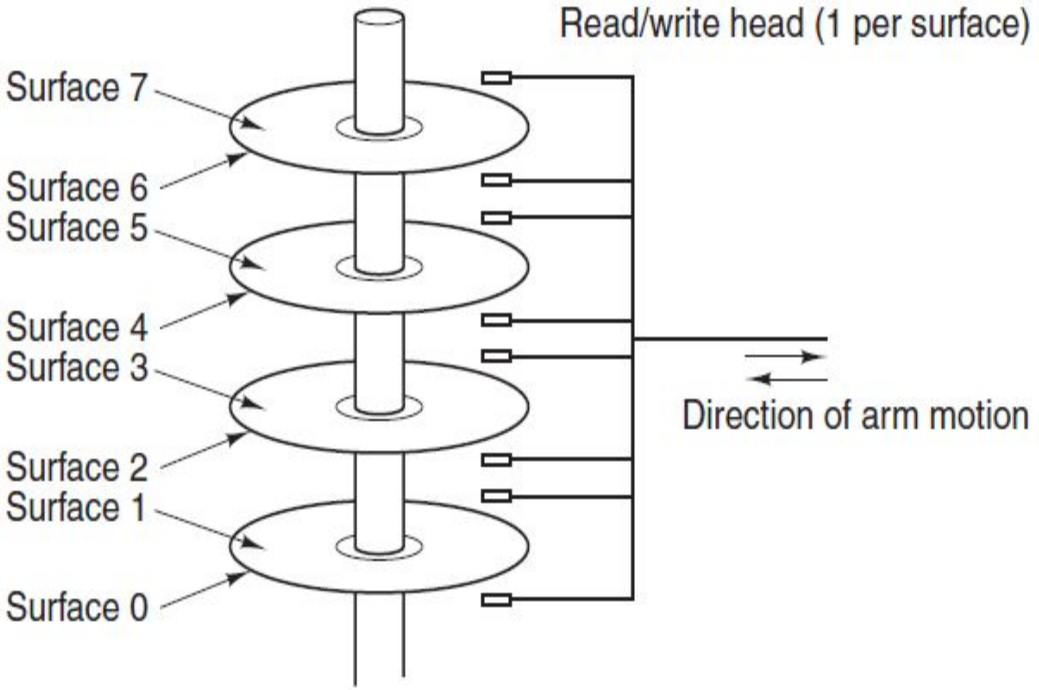
Gerenciamento de Memória

Hierarquia de Memória:

- Cache
- Memória Principal
- Disco
  - Grande quantidade – G/T bytes
  - Baixo custo por byte
  - Lenta
  - Não volátil

Gerenciamento de Memória

- Discos:
  - 5400, 7200, 10800 RPM
  - Possui alguns cabeçotes de leitura/escrita
  - O cabeçote pode ler de um região chamada track/trilha
  - Juntas, todas as trilhas, de uma determinada posição do braço, formam um cilindro.
  - Cada trilha é dividida em setores (512 bytes/setor)



SSD - solid-state drive ou unidade de estado sólido

Vantagens

- Tempo de acesso reduzido. O tempo de acesso à memória é muito menor do que o tempo de acesso a meios magnéticos ou ópticos.
- Eliminação de partes móveis eletromecânicas, reduzindo vibrações, tornando-os completamente silenciosos;
- Por não possuírem partes móveis, são muito mais resistentes que os HDDs comuns contra choques físicos;
- Menor peso em relação aos discos rígidos;
- Consumo reduzido de energia;
- Possibilidade de trabalhar em temperaturas maiores que os HDDs comuns - cerca de 70° C;
- Largura de banda muito superior aos demais dispositivos, apresentando até 250 MB/s na gravação e até 700 MB/s nas operações de leitura.

Desvantagens

- Custo mais elevado;
- Capacidade de armazenamento inferior

aos discos rígidos IDE e SATA.

Compartilhamento de Memória

- Partições Fixas
  - Cada processo é alocado em uma dada partição da memória (pré-definida);
  - Partições são liberadas quando o processo termina;
- Partições Variáveis
  - Memória é alocada de acordo com o tamanho e número de processos;
  - Otimiza o uso da memória;

System Calls – Chamadas de Sistema

- Interface entre o Sistema Operacional e os programas do usuário;
- As chamadas se diferem de SO para SO, no entanto, os conceitos relacionados às chamadas são similares independentemente do SO;
- Apenas uma chamada de sistema pode ser realizada em um instante de tempo (ciclo de relógio) pela CPU;

Roteiro

- Por que é necessário um sistema operacional
- O que é um Sistema Operacional
- Histórico
- Conceitos Básicos
  - Processo;
  - Memória;
  - Chamadas de Sistema;
- System Calls
- Estrutura de Sistemas Operacionais

Interfaces de um Sistema Operacional

- Usuário – SO:
  - **Shell ou Interpretador de comandos**
- Programas – SO:
  - **Chamadas ao Sistema**

Conceitos Básicos
Chamadas de Sistema

- **Modos de Acesso**:
  - Modo usuário;
  - Modo *kernel* ou Supervisor ou Núcleo;
  - São determinados por um conjunto de bits localizados no registrador de status do processador: PSW (*program status word*);
    - Por meio desse registrador, o hardware verifica se a instrução pode ou não ser executada pela aplicação;
  - Protege o próprio *kernel* do Sistema Operacional na RAM contra acessos indevidos;

Conceitos Básicos
Chamadas de Sistema

- Modo usuário:
  - Aplicações não têm acesso direto aos recursos da máquina, ou seja, ao hardware;
  - Quando o processador trabalha no modo usuário, a aplicação só pode executar **instruções sem privilégios**, **com um acesso reduzido de instruções**;
  - Por que? Para garantir a **segurança** e a **integridade do sistema**;

Conceitos Básicos
Chamadas de Sistema

- Modo *Kernel:*
  - Aplicações têm acesso direto aos recursos da máquina, ou seja, ao hardware;
  - **Operações com privilégios**;
  - Quando o processador trabalha no modo *kernel*, a aplicação tem **acesso ao conjunto total de instruções**;
  - Apenas o SO tem acesso às instruções privilegiadas;

Conceitos Básicos
Chamadas de Sistema

- Se uma aplicação precisa realizar alguma instrução privilegiada, ela realiza uma **chamada de sistema *(system call*)**, que altera do modo usuário para o modo *kernel*;
- Chamadas de sistemas são a **porta de entrada** para o modo *Kernel*;
  - São a interface entre os programas do usuário no modo usuário e o Sistema Operacional no modo kernel;
  - As chamadas diferem de SO para SO, no entanto, os conceitos relacionados às chamadas são similares independentemente do SO;

Conceitos Básicos
Chamadas de Sistema

- Exemplos de chamadas de sistema:
  - Chamadas para gerenciamento de processos:
    - `Fork (CreateProcess – WIN32)` – cria um processo;
  - Chamadas para gerenciamento de diretórios:
    - `Mount` – monta um diretório;
  - Chamadas para gerenciamento de arquivos:
    - `Close (CloseHandle – WIN32)` – fechar um arquivo;
  - Outros tipos de chamadas:
    - `Chmod`: modifica permissões;

Fork

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <time.h>

int main(int argc, char **argv)
{
   printf("--beginning of program\n");
   int counter = 0;
   pid_t pid = fork();
   if (pid == 0)
   {  // child process
      int i = 0;
      for (; i < 100; ++i)
      {
      printf("child process: counter=%d
pid=%d\n", ++counter, getpid());
      sleep(1);
      }
   }
```

```
   else if (pid > 0)
   {  // parent process
      int j = 0;
      for (; j < 100; ++j)
      {
         printf("parent process: counter=%d
pid=%d\n", ++counter, getpid());
         sleep(1);
      }
   }
   else
   {
      // fork failed
      printf("fork() failed!\n");
      return 1;
   }
   printf("--end of program--\n");
   return 0;
}
```