EAD

EDUCAÇÃO A DISTÂNCIA

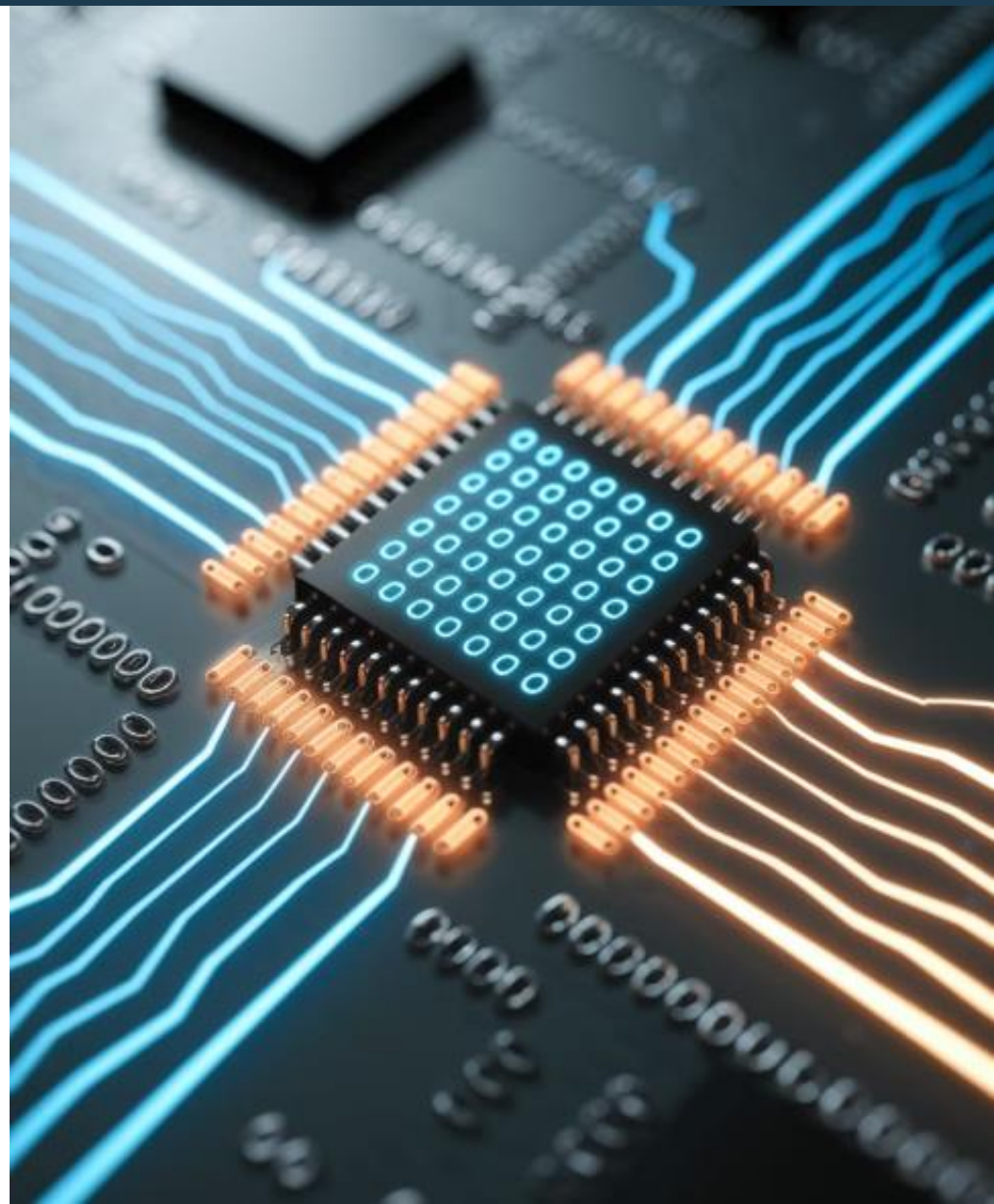
UNIDADE II

Infraestrutura Computacional

Profa. Sandra Bozolan

Introdução às portas lógicas

- Exploraremos os fundamentos das portas lógicas, os blocos fundamentais que sustentam toda a computação moderna.
- As portas lógicas são componentes eletrônicos que realizam operações baseadas na álgebra booleana, processando sinais binários (0 e 1) para criar sistemas digitais complexos. Desde os smartphones que usamos diariamente até os supercomputadores mais avançados, as portas lógicas são os elementos essenciais que tornam possível o processamento de informações no mundo digital.



Fonte: Flux Fast 1.1, 2025 – A imagem representa um computador como catalisador da criatividade, utllizado para automatizar processos básicos humanos.

O que são portas lógicas?

- **Portas lógicas**: São dispositivos eletrônicos que implementam funções booleanas, operando com um ou mais sinais de entrada para produzir uma única saída, baseada em regras lógicas específicas.

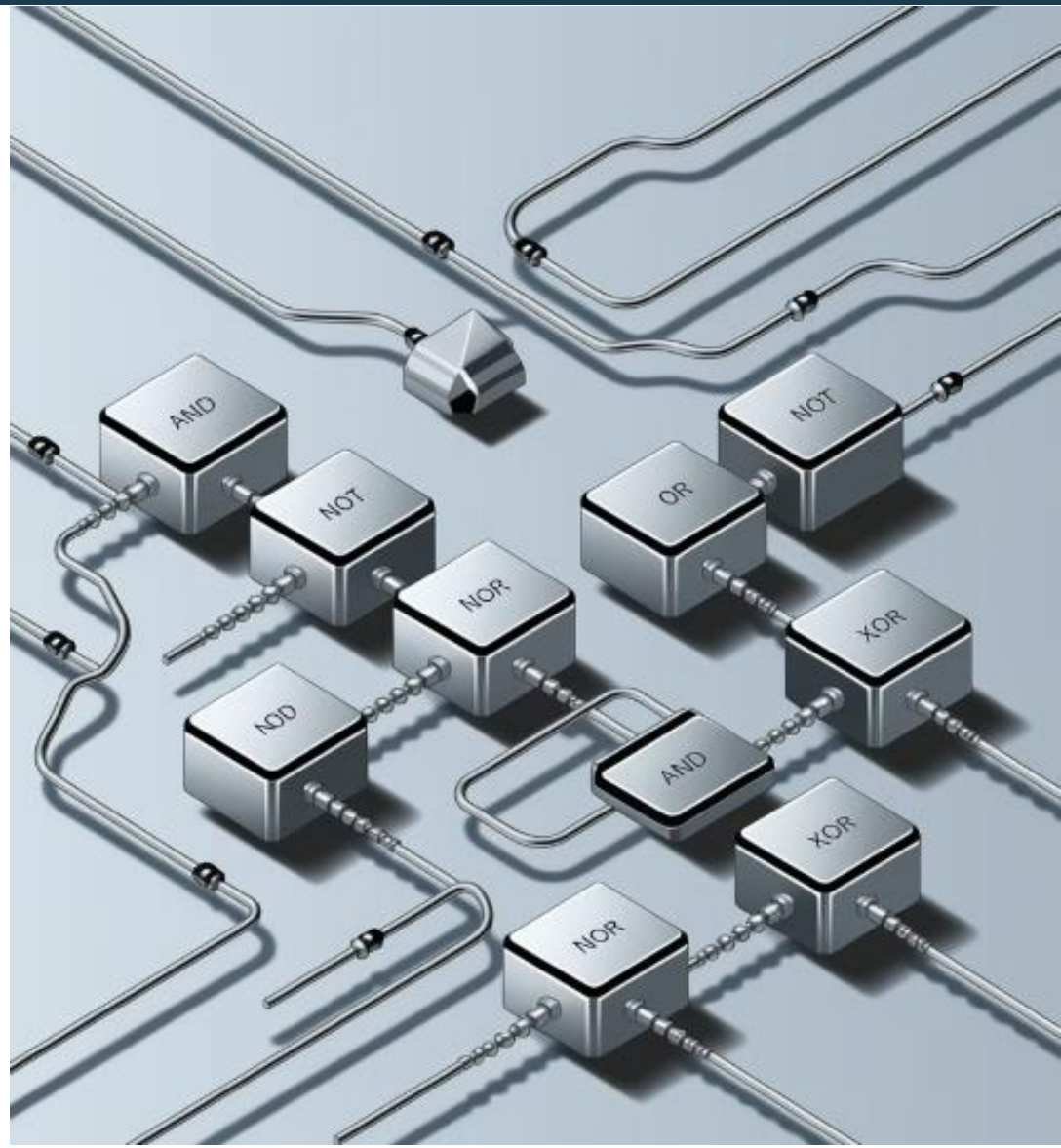
- **Sinais binários**: Trabalham exclusivamente com sinais binários, onde “0” geralmente representa baixa tensão (0V) e “1” representa alta tensão (5V ou 3.3V, dependendo da tecnologia).

- **Componentes fundamentais**: São os blocos construtores básicos de todos os circuitos digitais, desde calculadoras simples até processadores complexos de computadores e sistemas de controle industrial.

Álgebra Booleana: base teórica

Criada por George Boole

- A álgebra booleana, desenvolvida pelo matemático inglês George Boole no século XIX, é o fundamento matemático que permite descrever e analisar operações lógicas. Inicialmente concebida como um sistema algébrico para expressar operações lógicas, tornou-se a base teórica para o design de circuitos digitais.

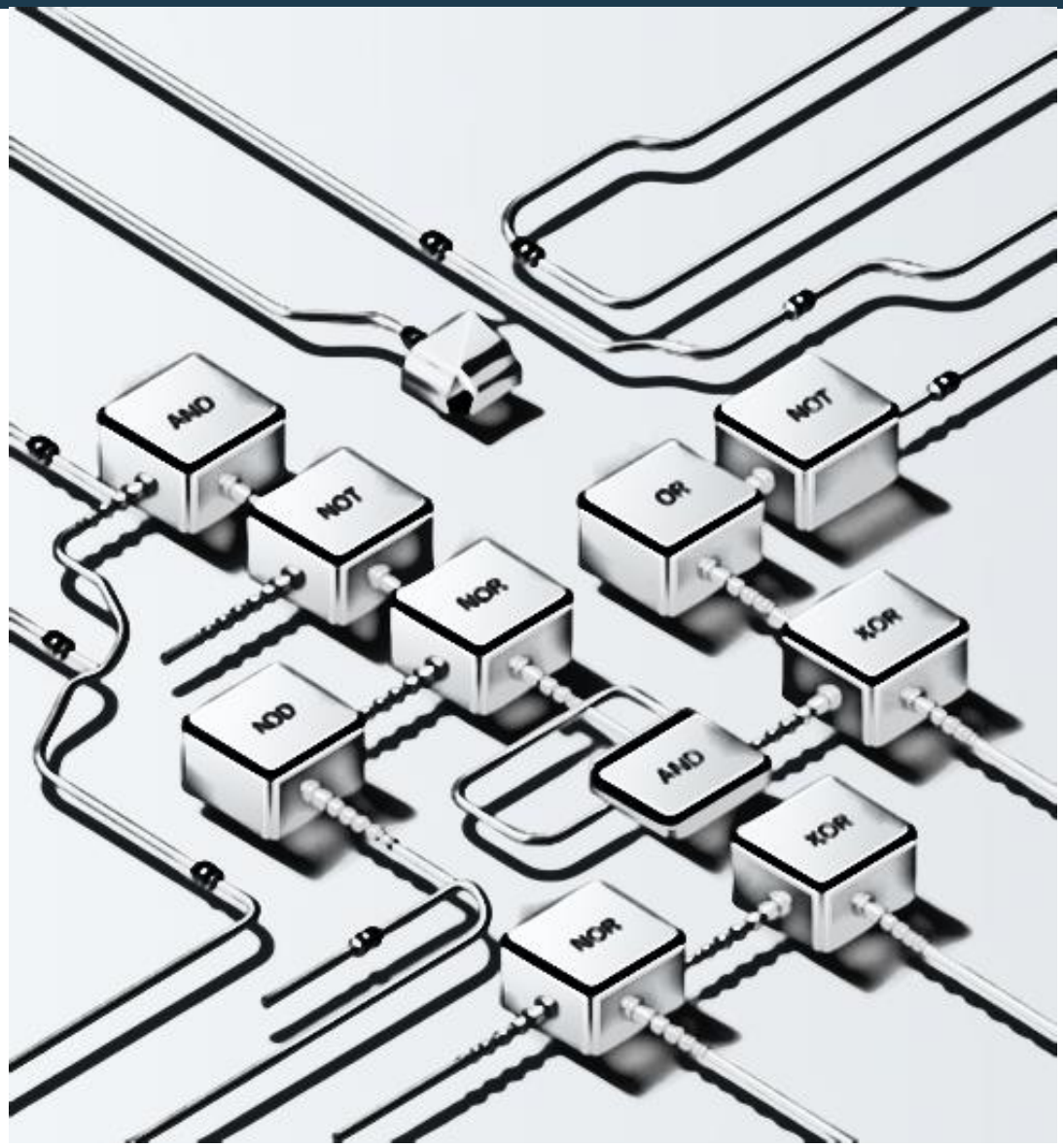


Fonte: Flux Fast 1.1, 2025 – Álgebra Booleana.

Álgebra Booleana: base teórica

Operações Básicas

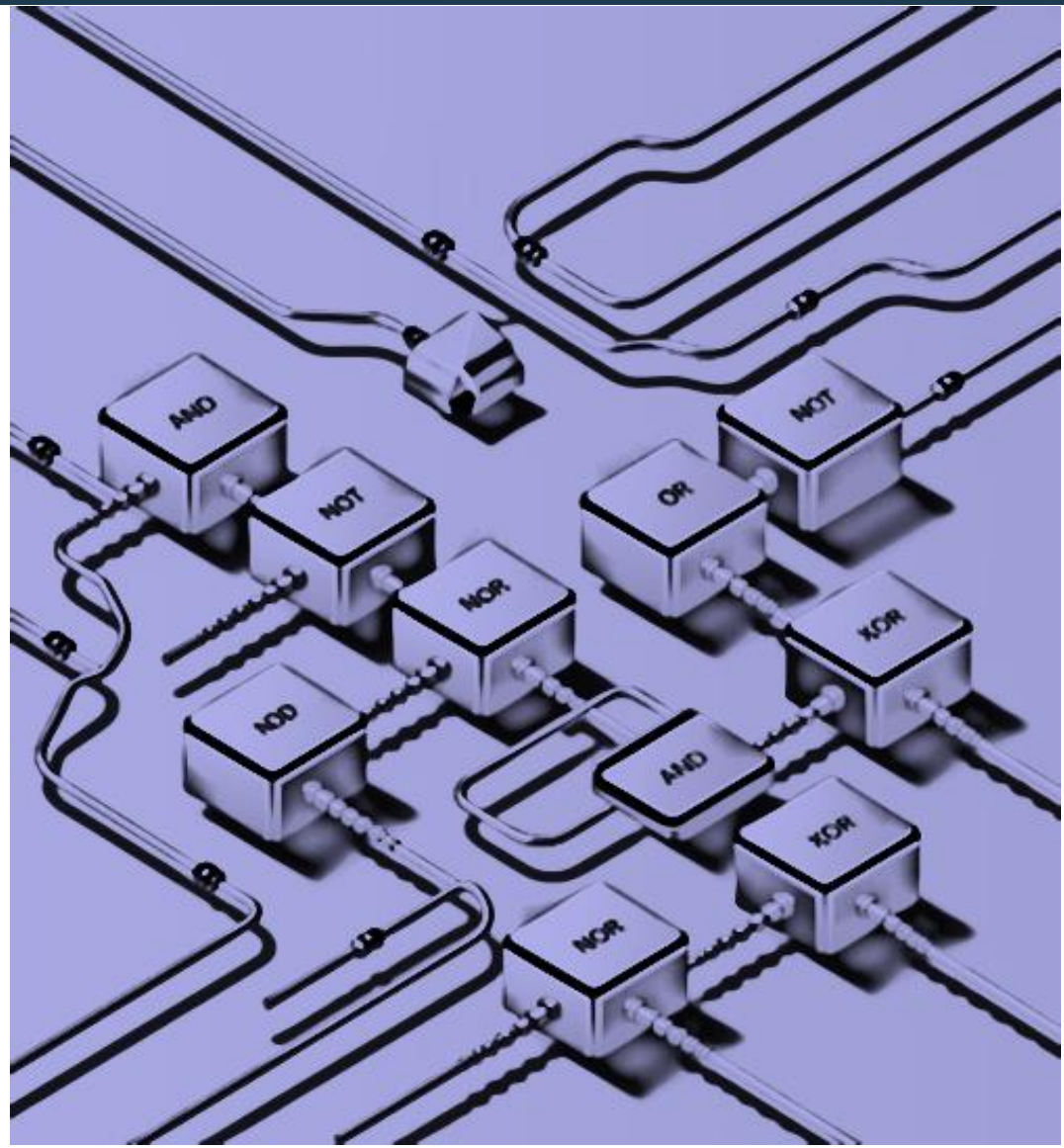
- As três operações fundamentais da álgebra booleana são AND (E), OR (OU) e NOT (NÃO), que correspondem diretamente às portas lógicas básicas. Estas operações podem ser combinadas para criar funções mais complexas, permitindo a implementação de qualquer operação lógica imaginável.



Fonte: Flux Fast 1.1, 2025 – Álgebra Booleana.

Álgebra Booleana: base teórica

- **Leis e Teoremas**
- A álgebra booleana possui leis como comutatividade, associatividade e distributividade, além de teoremas como os de De Morgan, que são essenciais para simplificar e otimizar circuitos lógicos, reduzindo o número de componentes necessários.



Fonte: Flux Fast 1.1, 2025 - Álgebra Booleana.

Porta lógica NOT (Inversora)

Funcionamento

A porta NOT, também conhecida como inversora, é a mais simples das portas lógicas. Ela possui apenas uma entrada e uma saída, invertendo o valor de entrada. Quando recebe “O”, produz “1”; quando recebe “1”, produz “O”.

Símbolo e Tabela-Verdade

É representada por um triângulo com um pequeno círculo na saída. Sua tabela-verdade mostra claramente a inversão: para entrada A = O, saída Y = 1; para entrada A = 1, saída Y = O.

Aplicações Práticas

É utilizada para inverter sinais, detectar ausência de condições, implementar sistemas de segurança (alarmes que disparam quando um circuito é interrompido) e como bloco básico para construir portas mais complexas.

Porta lógica AND

Princípio de Funcionamento

A porta AND implementa a operação lógica “E”, produzindo saída “1” apenas quando todas as suas entradas forem “1”. Com duas ou mais entradas, basta uma delas ser “O” para que a saída seja “O”. É como um interruptor em série: todos precisam estar fechados para que a corrente flua.

Representação e Tabela-Verdade

Simbolizada por uma forma semicircular ou retangular com uma extremidade arredondada, sua tabela-verdade para duas entradas (A e B) mostra que a saída Y só é 1 quando A = 1 E B = 1. Nos outros três casos possíveis, a saída é O.

Uso no Cotidiano

É amplamente utilizada em sistemas de segurança (onde múltiplas condições devem ser satisfeitas), verificação de senhas (todos os caracteres devem estar corretos) e em circuitos de controle industrial (várias condições devem ser verdadeiras para ativar um processo).

Porta lógica OR

Conceito Básico

A porta OR implementa a operação lógica “OU”, gerando saída “1” se pelo menos uma de suas entradas for “1”. Apenas quando todas as entradas forem “O”, a saída será “O”. Funciona como interruptores em paralelo: basta um estar fechado para que a corrente passe.

Simbologia e Tabela-Verdade

Representada por um símbolo similar a uma seta ou um “escudo” com entradas múltiplas, sua tabela-verdade para duas entradas (A e B) mostra que a saída Y é O apenas quando A = O e B = O. Nos outros três casos, quando pelo menos uma entrada é 1, a saída é 1.

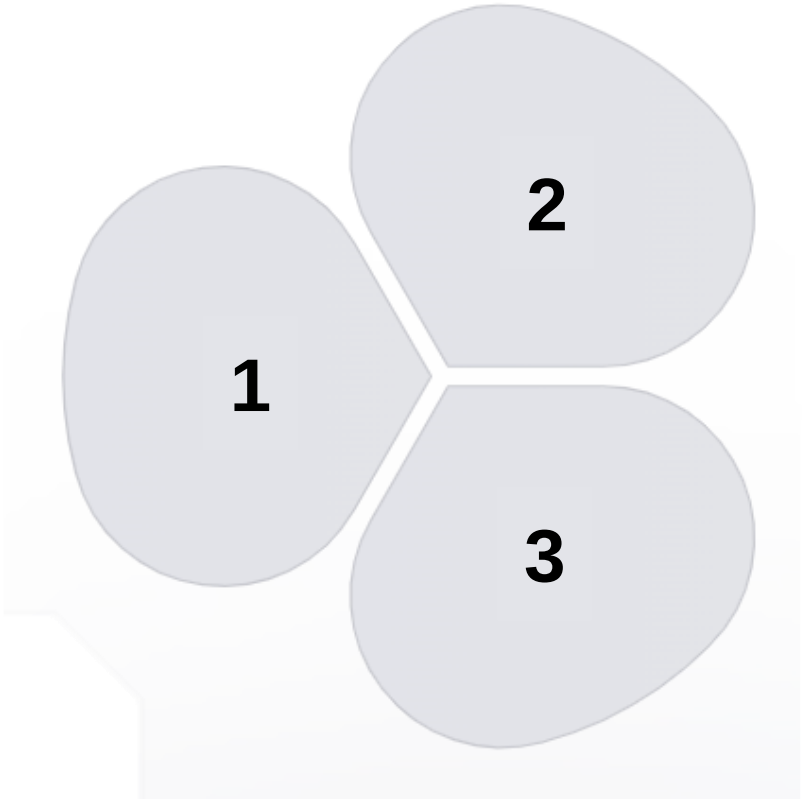
Aplicações

Utilizada em sistemas de alarme (acionado se qualquer sensor detectar movimento), controles de acesso (autorização por diferentes métodos) e em circuitos detectores de condições alternativas (qualquer uma das condições verdadeiras aciona o sistema).

Porta lógica XOR (OU Exclusivo)

Conceito e Funcionamento

A porta XOR (Exclusive OR ou OU Exclusivo) produz saída “1” quando um número ímpar de entradas é “1”. Para duas entradas, isso significa que a saída é “1” quando as entradas são diferentes entre si, e “O” quando as entradas são iguais.



Representação e Tabela-Verdade

Simbolizada por um símbolo similar ao OR, mas com uma linha adicional na entrada. Sua tabela-verdade para A e B mostra: A=0/B=0→ Y=O; A=0/B=1→ Y=1; A=1/B=O → Y=1; A=1/B=1→ Y=O. Observe o padrão: saída = 1 apenas quando as entradas são diferentes.

Aplicações Práticas

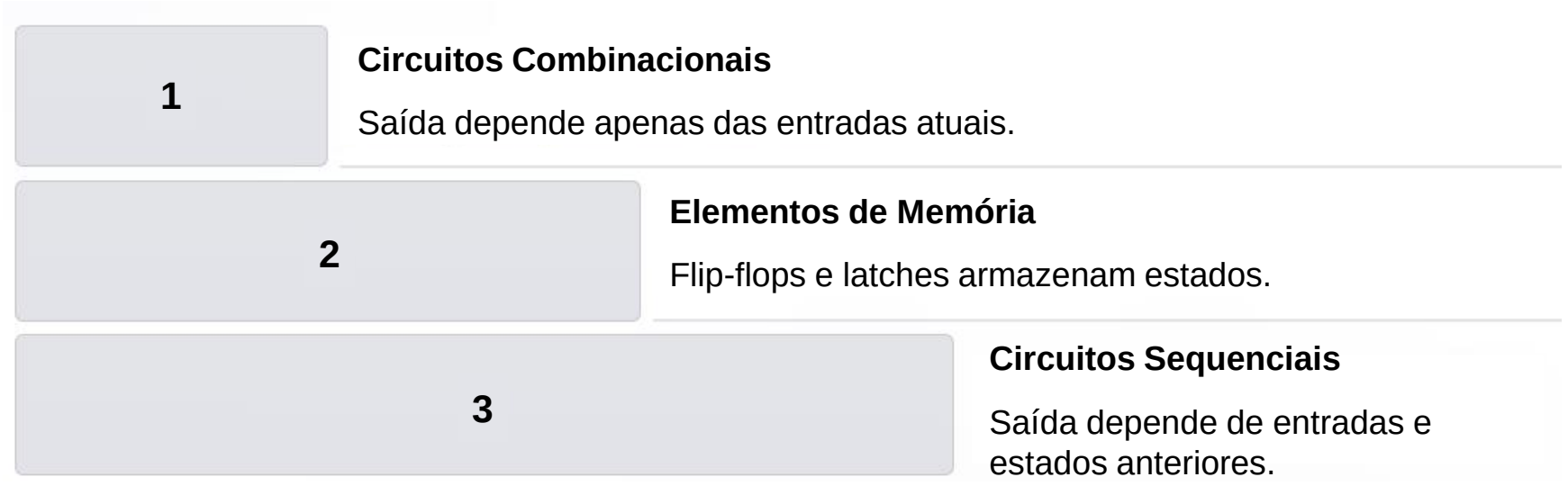
Utilizada em circuitos comparadores, detectores de paridade, somadores binários e em criptografia. É essencial em circuitos aritméticos, permitindo a implementação de operações matemáticas básicas em processadores.

Implementação física das portas lógicas

- As portas lógicas podem ser implementadas de várias formas físicas, evoluindo significativamente ao longo da história da computação. Inicialmente construídas com relés eletromecânicos e válvulas a vácuo, hoje são predominantemente baseadas em transistores integrados em chips.

Circuitos combinacionais *vs*. sequenciais

- Os circuitos digitais são classificados em dois grandes grupos: combinacionais e sequenciais. Os circuitos combinacionais, como decodificadores, multiplexadores e somadores, têm suas saídas determinadas exclusivamente pelos valores atuais nas entradas, sem memória de estados anteriores.
- Já os circuitos sequenciais, como contadores, registradores e máquinas de estado, utilizam elementos de memória (flip-flops) para armazenar informações de estados anteriores. Isso permite que a saída dependa não só das entradas atuais, mas também da sequência de eventos passados, tornando possível a implementação de sistemas mais complexos como processadores e controladores.



Interatividade

Considere a porta XOR (OU Exclusivo). Qual das alternativas abaixo descreve corretamente a regra de saída da XOR para duas entradas A e B?

a) A saída é 1 apenas se ambas as entradas forem 1.
b) A saída é 1 apenas se ambas as entradas forem 0.
c) A saída é 1 se pelo menos uma das entradas for 1.
d) A saída é 1 se exatamente uma das entradas for 1.
e) A saída é 1 se nenhuma das entradas for 1.

Resposta

Considere a porta XOR (OU Exclusivo). Qual das alternativas abaixo descreve corretamente a regra de saída da XOR para duas entradas A e B?

a) A saída é 1 apenas se ambas as entradas forem 1.
b) A saída é 1 apenas se ambas as entradas forem 0.
c) A saída é 1 se pelo menos uma das entradas for 1.
d) A saída é 1 se exatamente uma das entradas for 1.
e) A saída é 1 se nenhuma das entradas for 1.

O relógio do computador: entendendo o clock

- Um dos componentes mais fundamentais em sistemas computacionais: o clock. Vamos explorar como este elemento essencial funciona como o “coração” dos dispositivos eletrônicos, ditando o ritmo no qual todas as operações são executadas.
- Ao longo desta apresentação, vamos desmistificar o conceito de clock, entender seu papel na sincronização dos componentes e ver exemplos práticos de como ele impacta o desempenho de sistemas computacionais. Prepare-se para uma jornada fascinante ao universo da temporização eletrônica!

Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma abstrata o clock do computador.

O que é o clock?

- **O clock:** É um componente que gera pulsos elétricos regulares e repetitivos em um sistema computacional. Funciona como um contador de tempo, criando uma sequência contínua de sinais que ditam quando cada operação deve ocorrer.
- **O maestro digital:** Assim como um maestro coordena os músicos de uma orquestra, o clock sincroniza todos os componentes do sistema. Sem esta coordenação precisa, os diferentes elementos não conseguiriam trabalhar em harmonia.

Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma abstrata o clock do computador.

O que é o clock?

- **Medição em Hertz:** A frequência do clock é medida em Hertz (Hz), indicando quantos ciclos ocorrem por segundo. Um processador de 3 GHz, por exemplo, executa 3 bilhões de ciclos a cada segundo.

Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma abstrata o clock do computador.

Anatomia de um ciclo de clock

Nível Alto (1)

- Durante esta fase do ciclo, o sinal elétrico atinge seu valor máximo, representando o estado lógico “1”. Neste momento, determinadas operações são ativadas no sistema computacional.

Nível Baixo (0)

- Nesta fase, o sinal cai para seu valor mínimo, representando o estado lógico “0”. Outras operações dependem especificamente desta fase para serem executadas corretamente.

Transições

- As mudanças entre os níveis alto e baixo são chamadas de bordas de subida (quando passa de 0 para 1) e bordas de descida (quando passa de 1 para 0). Muitas operações são sincronizadas nestas transições.

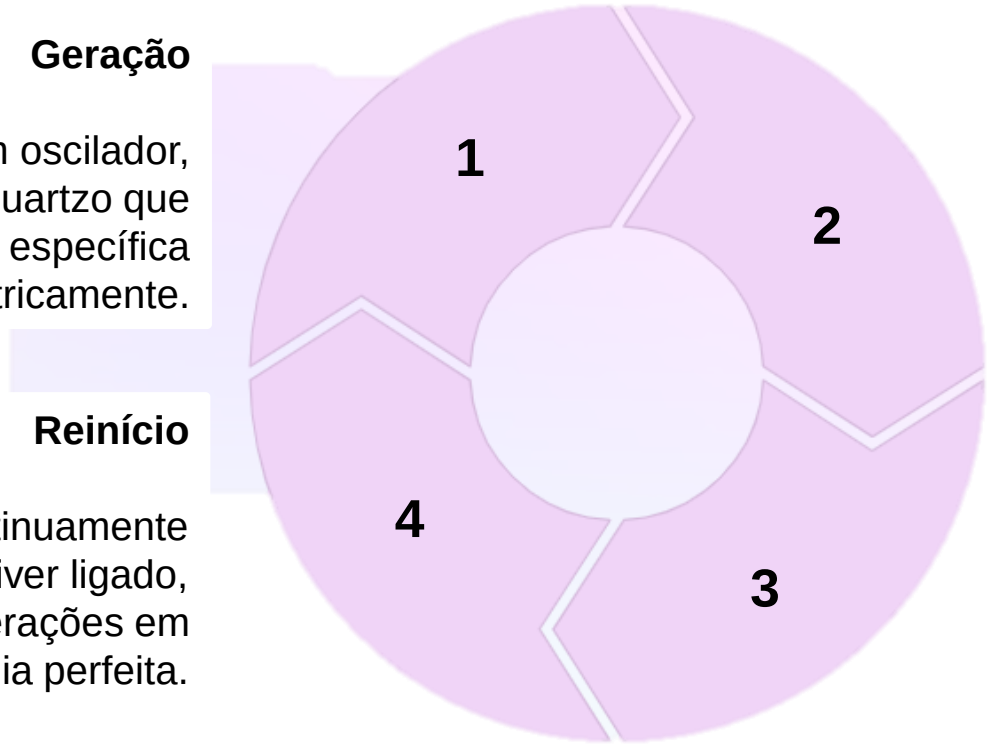
O clock como gerador de pulsos

Geração

O pulso é gerado por um oscilador, geralmente um cristal de quartzo que vibra em uma frequência específica quando estimulado eletricamente.

Reinício

O ciclo se repete continuamente enquanto o sistema estiver ligado, mantendo todas as operações em sincronia perfeita.



Distribuição

O sinal é distribuído por todo o sistema através de circuitos especializados, garantindo que chegue a todos os componentes que precisam dele.

Sincronização

Os componentes recebem o sinal e usam-no para coordenar suas operações internas e interações com outros componentes.

Frequência e período do clock

Frequência

- A frequência do clock indica quantos ciclos completos ocorrem em um segundo. É medida em Hertz (Hz) e seus múltiplos: MHz (milhões de ciclos por segundo) e GHz (bilhões de ciclos por segundo).

Período

- O período é o tempo necessário para completar um ciclo inteiro de clock. É o inverso da frequência (T = 1/f) e geralmente é medido em nanossegundos (ns) ou picossegundos (ps) para sistemas modernos.

Relação fundamental

- Um processador com frequência mais alta (por exemplo, 4 GHz) tem um período menor (0,25 ns) e, portanto, pode executar mais operações por segundo que um processador mais lento.

Como o clock é gerado

1. **Cristal de Quartzo**

   O componente mais comum para geração de clock é o cristal de quartzo, que vibra em uma frequência muito precisa quando uma voltagem é aplicada, devido ao efeito piezoelétrico.

2. **Circuito Oscilador**

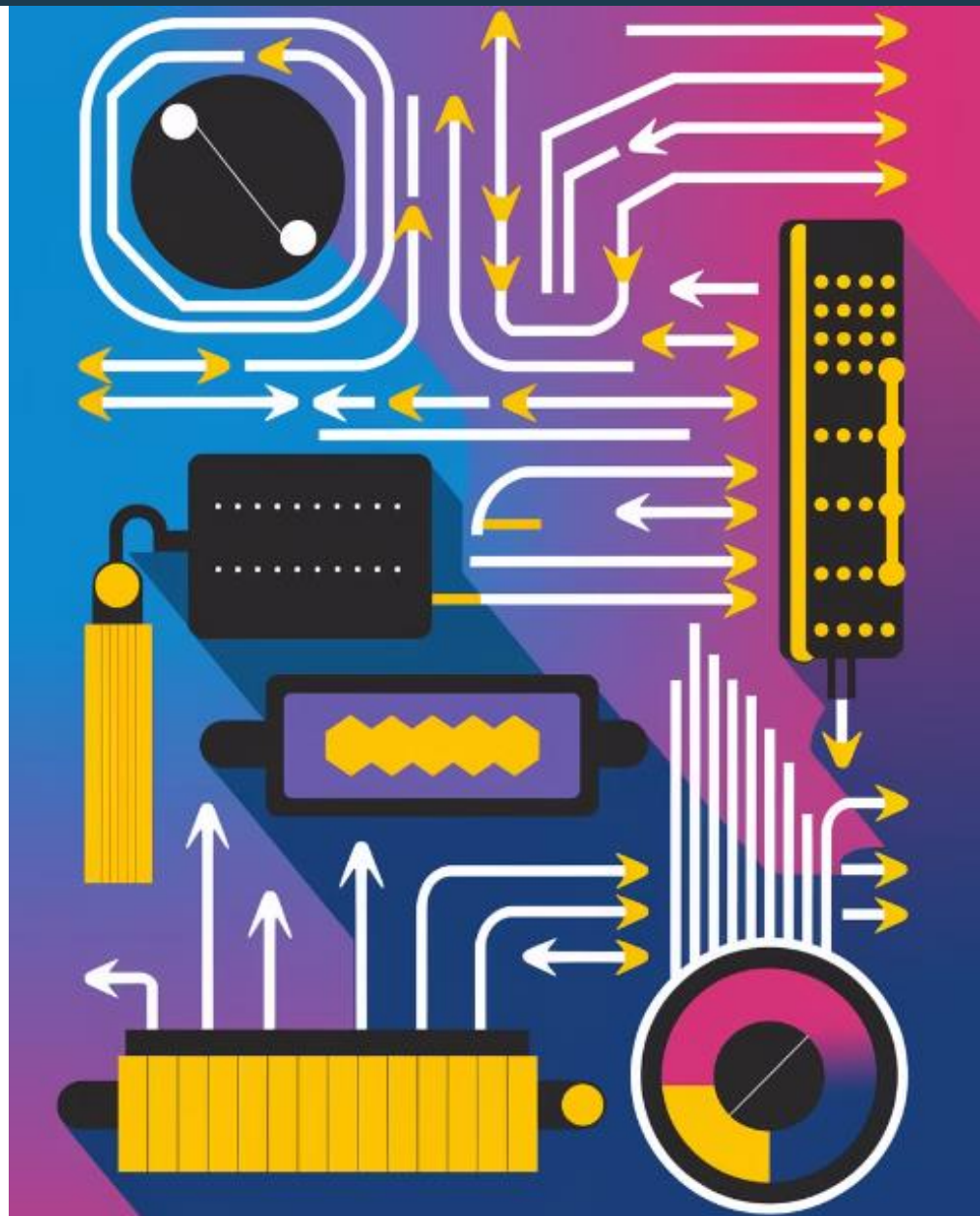
   O cristal faz parte de um circuito oscilador que converte as vibrações mecânicas em um sinal elétrico estável e de frequência constante, adequado para uso digital.

3. **Multiplicação e Divisão**

   Muitas vezes, a frequência básica gerada pelo cristal é multiplicada ou dividida por circuitos PLLs (Phase-Locked Loops) para criar diferentes frequências necessárias no sistema.

4. **Bufferização e Distribuição**

   O sinal de clock precisa ser amplificado e distribuído para todos os componentes do sistema, mantendo sua integridade e timing precisos.



Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma abstrata o clock do computador.

O clock e as operações do processador

Busca de instrução

- No primeiro estágio, o processador usa um pulso de clock para buscar a próxima instrução a ser executada da memória. Este processo é sincronizado precisamente com o sinal de clock para garantir que os dados corretos sejam acessados.

Decodificação

- A instrução obtida é decodificada em um formato que o processador possa entender. Esta operação também é sincronizada com o clock, ocorrendo em um ciclo específico ou subciclo de clock.

O clock e as operações do processador

Execução

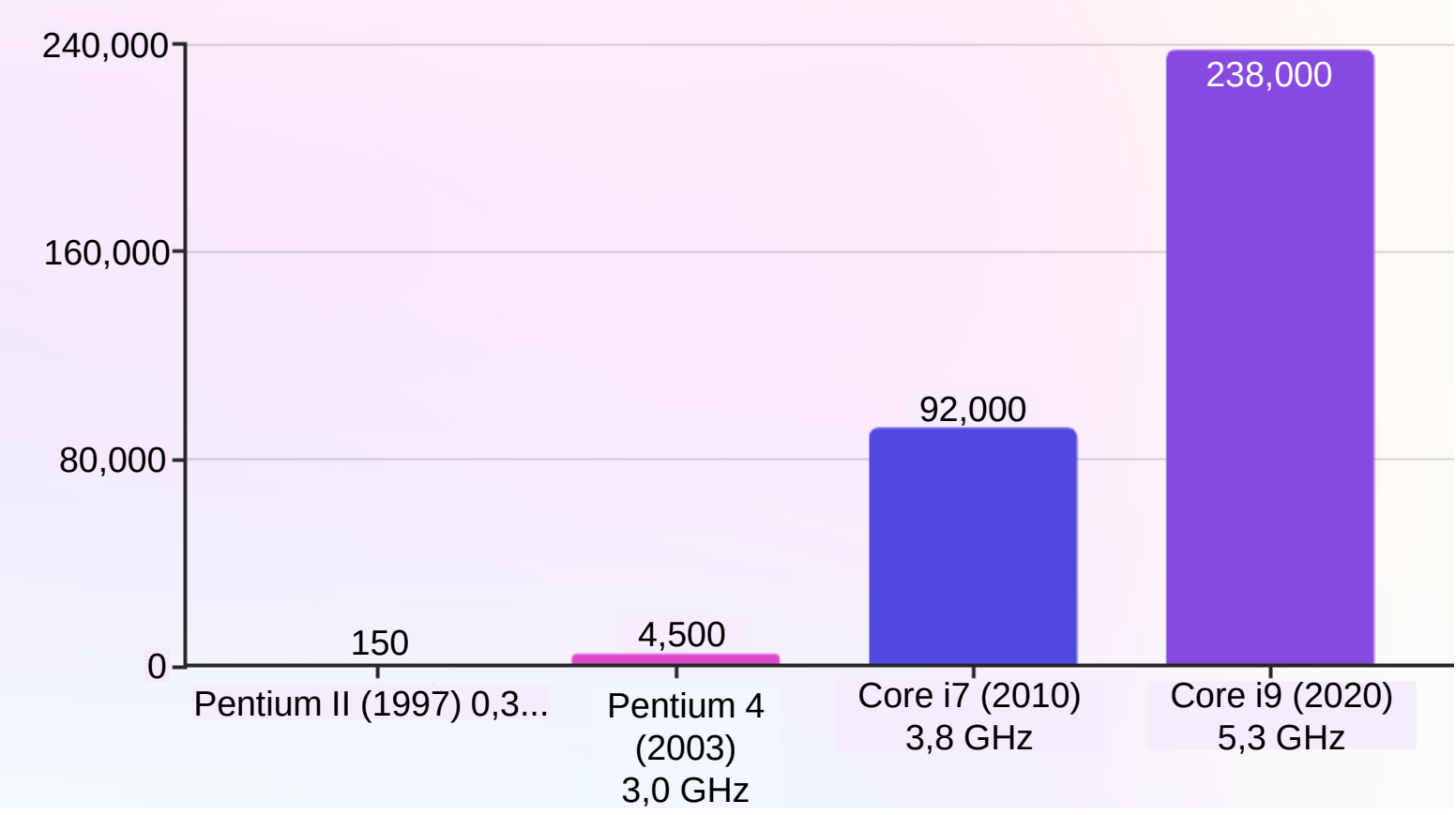
- Os circuitos aritméticos e lógicos realizam a operação solicitada, como uma adição ou comparação. A complexidade da instrução determina quantos ciclos de clock serão necessários para completá-la.

Armazenamento de resultado

- O resultado da operação é armazenado em registradores ou na memória, completando o ciclo de instrução. Novamente, o timing preciso do clock garante que os dados sejam gravados corretamente.

Impacto da frequência de clock no desempenho

- O gráfico mostra como a frequência de clock evoluiu ao longo do tempo em processadores populares, e como isso influenciou o número de operações por segundo que eles podem realizar. Note que o aumento de desempenho não é apenas proporcional ao aumento da frequência, pois outros fatores como eficiência da arquitetura, número de núcleos e tamanho do cache também contribuem significativamente.



Overclocking: acelerando além dos limites

- **Definição e propósito:** Overclocking é a prática de aumentar a frequência do clock além das especificações definidas pelo fabricante, buscando maior desempenho.
- **Desafios térmicos:** O aumento da frequência gera mais calor, exigindo sistemas de refrigeração avançados como water cooling ou até mesmo nitrogênio líquido.
- **Riscos e limitações:** O overclocking pode reduzir a vida útil dos componentes, causar instabilidade do sistema e anular garantias.

Fonte: Flux Fast 1.1, 2025 – A imagem: o overclocking de frequência do computador.

Interatividade

Em relação à frequência de clock de um processador, é correto afirmar que:

a) Frequências de clock mais altas sempre resultam em melhor desempenho, independentemente de outros fatores.
b) A frequência de clock corresponde diretamente ao número de instruções que o processador executa por segundo, sem variações.
c) Uma frequência de clock alta permite um maior número de ciclos por segundo, possibilitando a execução de mais operações em um determinado intervalo de tempo.
d) A frequência de clock não tem relação com o consumo de energia do processador.
e) O clock não é mais usado nos processadores modernos, tendo sido substituído por sistemas baseados em eventos assíncronos.

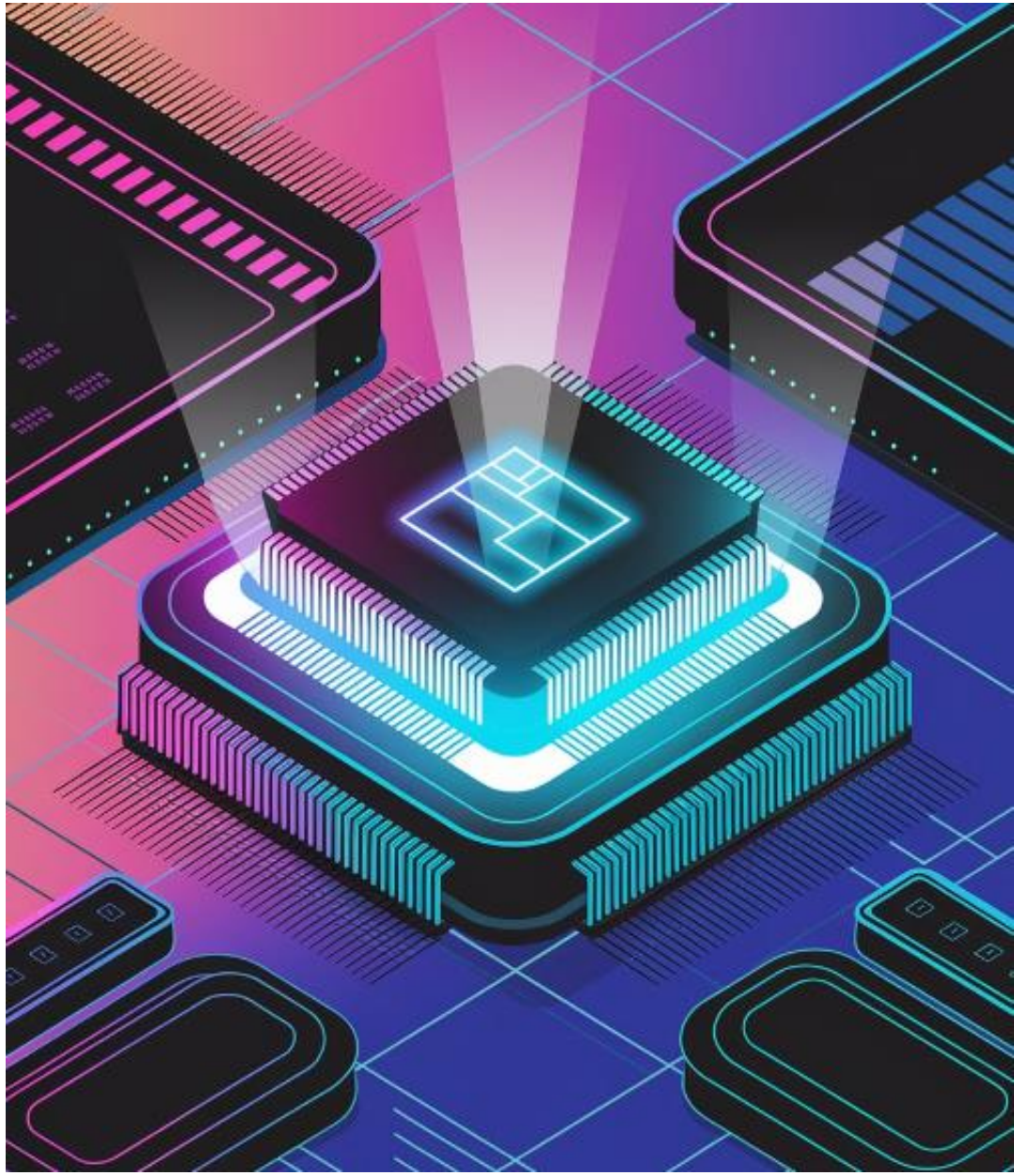
Resposta

Em relação à frequência de clock de um processador, é correto afirmar que:

a) Frequências de clock mais altas sempre resultam em melhor desempenho, independentemente de outros fatores.
b) A frequência de clock corresponde diretamente ao número de instruções que o processador executa por segundo, sem variações.
c) Uma frequência de clock alta permite um maior número de ciclos por segundo, possibilitando a execução de mais operações em um determinado intervalo de tempo.
d) A frequência de clock não tem relação com o consumo de energia do processador.
e) O clock não é mais usado nos processadores modernos, tendo sido substituído por sistemas baseados em eventos assíncronos.

Arquiteturas CISC e RISC: uma análise detalhada

- Arquitetura de Von Neumann traz os fundamentos mais importantes da computação moderna. Exploraremos como esse modelo revolucionário, proposto pelo matemático John von Neumann em 1945, estabeleceu as bases para o funcionamento dos computadores que utilizamos até hoje.
- Entenderemos os quatro componentes essenciais dessa arquitetura: a memória principal, a unidade lógica e aritmética (ULA), a unidade de controle e os dispositivos de entrada e saída. Abordaremos cada um deles com exemplos práticos, imagens explicativas e discussões sobre como influenciam.



Fonte: Flux Fast 1.1, 2025 –
A imagem representa uma abstração das arquiteturas CISC e RISc

Visão geral da arquitetura

- **Armazenamento:** Memória principal onde dados e instruções são armazenados de forma binária, permitindo acesso rápido pelo processador.
- **Processamento:** Unidade Central de Processamento (CPU) composta pela Unidade Lógica e Aritmética (ULA) e pela Unidade de Controle, responsáveis por executar cálculos e gerenciar operações.
- **Entrada/Saída:** Dispositivos que permitem a comunicação entre o computador e o mundo exterior, desde teclados e mouses até monitores e impressoras.
- **Interconexão:** Sistema de barramentos que permite a comunicação entre todos os componentes, transportando dados, endereços e sinais de controle.

Memória principal: armazenamento unificado

- **Estrutura Unificada:** Armazena tanto dados quanto instruções de programa no mesmo espaço de memória, seguindo o conceito de “programa armazenado” que distingue a arquitetura de Von Neumann.
- **Células de Memória:** Organizadas em células endereçáveis sequencialmente, cada uma armazenando um padrão binário que pode representar um dado ou uma instrução, dependendo de como é interpretado pela CPU.
- **Acesso Aleatório:** Possibilita acessar qualquer célula diretamente, independentemente de sua posição física, tornando o processamento mais eficiente em comparação com sistemas sequenciais.
    - **Hierarquia de Memória:** Embora Von Neumann propusesse uma memória única, os computadores modernos implementam uma hierarquia que inclui registradores, cache, RAM e armazenamento secundário para otimizar o desempenho.

Gargalo de Von Neumann

- **O Problema:** O compartilhamento do mesmo barramento para instruções e dados limita a velocidade de processamento, criando um “gargalo” entre a CPU e a memória.
- **Impacto no Desempenho:** A CPU frequentemente fica ociosa esperando que instruções ou dados sejam buscados na memória, reduzindo a eficiência geral do sistema independentemente da velocidade do processador.
- **Soluções Modernas:** Memórias cache multicamadas, arquiteturas multicore, barramentos dedicados e técnicas de pipelining foram desenvolvidas para mitigar os efeitos do gargalo, embora a limitação fundamental permaneça presente.

Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa o Gargalo de Von Neumann

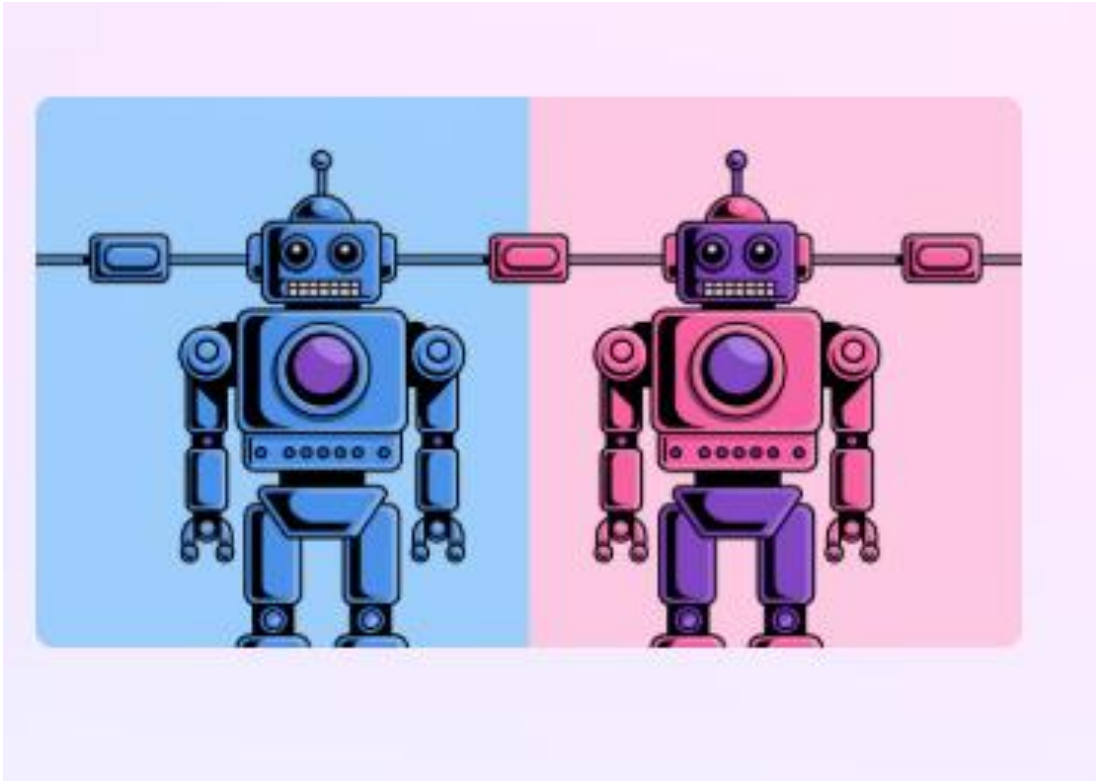
Ciclo de instrução detalhado

1. **Fetch (Busca):** A unidade de controle utiliza o contador de programa para determinar o endereço da próxima instrução a ser executada. A instrução é buscada na memória principal e armazenada no Registrador de Instrução.
2. **Decode (Decodificação):** A instrução é analisada para determinar qual operação deve ser realizada e quais dados (operandos) serão necessários. Sinais de controle são gerados para ativar os componentes corretos do sistema.
3. **Execute (Execução):** A operação propriamente dita é realizada, seja um cálculo na ULA, uma operação de entrada/saída, ou uma transferência de dados entre registradores. Os resultados são armazenados no local especificado.
4. **Store (Armazenamento):** Os resultados da operação são armazenados em registradores ou na memória principal. O contador de programa é atualizado para apontar para a próxima instrução, completando o ciclo.

Arquitetura de Von Neumann *vs.* Harvard

Arquitetura de Von Neumann

- Memória única para dados e instruções.
- Barramento compartilhado.
- Flexibilidade para modificar programas.
- Sujeita ao “gargalo” de Von Neumann.
- Mais simples de implementar.
- Domina computadores de uso geral.

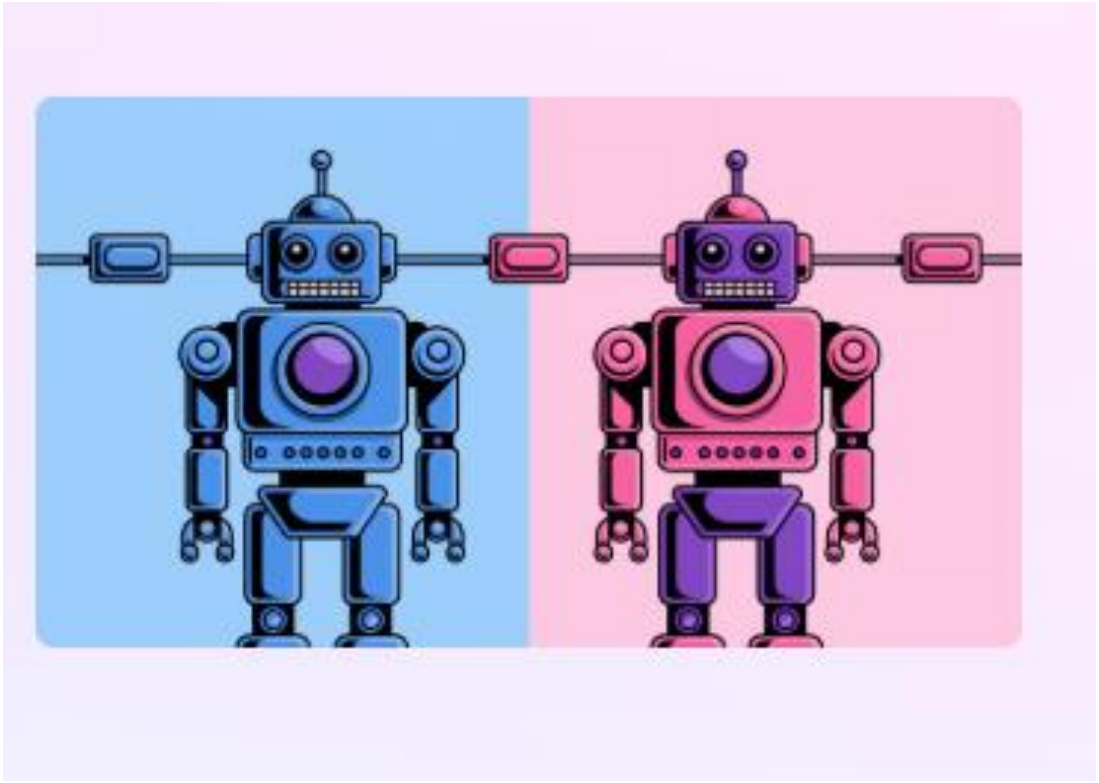


Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa a arquitetura de Von Neumann vs. Harvard

Arquitetura de Von Neumann vs. Harvard

Arquitetura Harvard

- Memórias separadas para dados e instruções.
- Barramentos independentes.
- Maior paralelismo de acesso.
- Melhor desempenho para tarefas específicas.
- Mais complexa e custosa.
- Comum em microcontroladores e DSPs.



Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa a arquitetura de Von Neumann vs. Harvard

Arquitetura de Von Neumann *vs.* Harvard

- A arquitetura Harvard, nomeada após os computadores Mark I e Mark II, desenvolvidos na Universidade de Harvard, representa uma alternativa importante ao modelo de Von Neumann. Muitos sistemas modernos implementam arquiteturas híbridas, utilizando aspectos de ambos os modelos para otimizar o desempenho em diferentes cenários.
- Por exemplo, a maioria dos microcontroladores e processadores de sinais digitais (DSPs) utiliza a arquitetura Harvard pura ou modificada, enquanto os processadores de computadores pessoais geralmente seguem o modelo de Von Neumann com elementos Harvard nas camadas de cache.

Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa a arquitetura de Von Neumann vs. Harvard

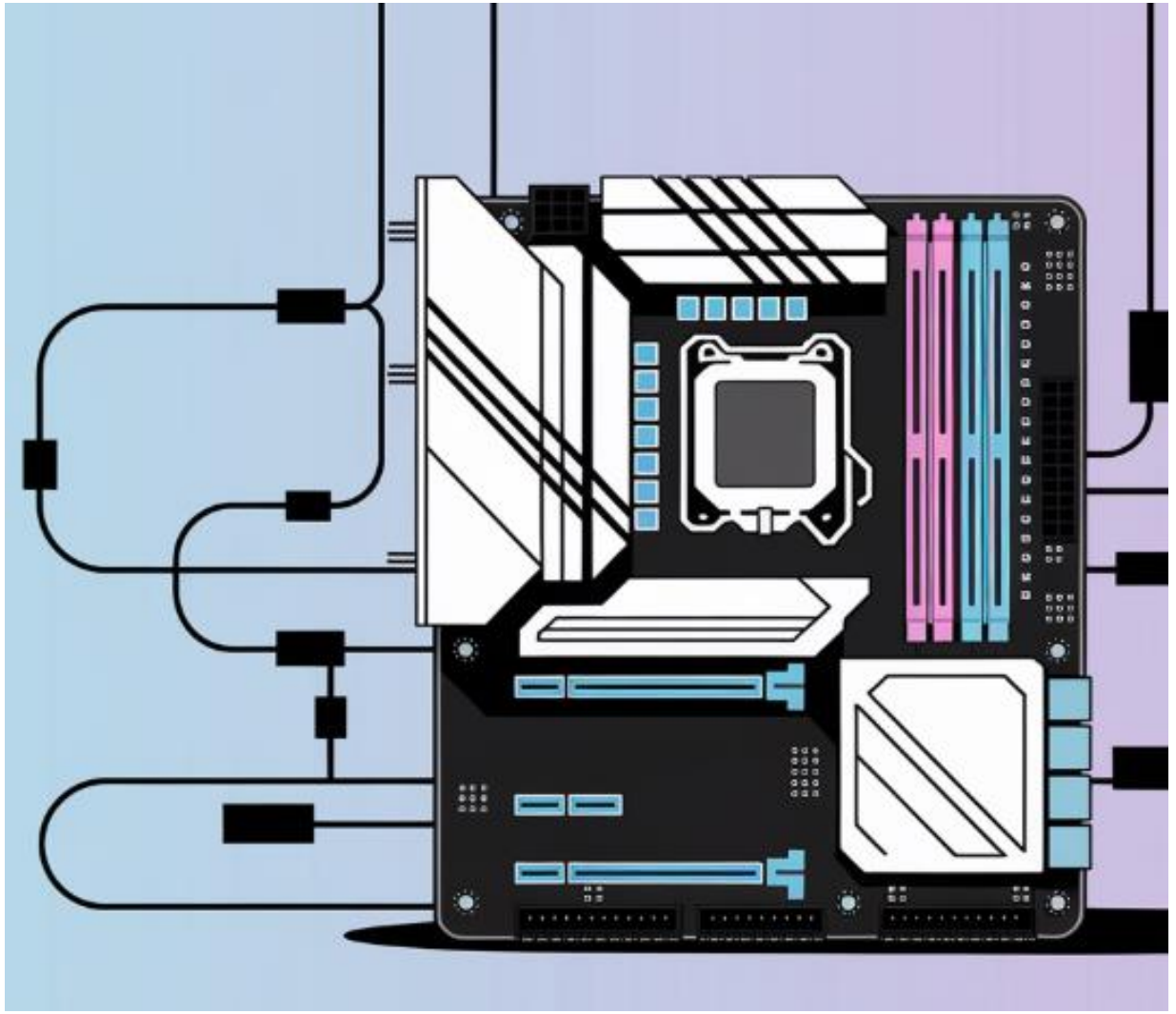
Arquitetura de Von Neumann *vs*. Harvard

- A arquitetura de Von Neumann estabeleceu os fundamentos da computação moderna, definindo um modelo que se mostrou extraordinariamente flexível e adaptável por mais de sete décadas. Do ENIAC aos smartphones, dos mainframes às nuvens computacionais, esta arquitetura evoluiu mantendo seus princípios fundamentais intactos.

Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa a arquitetura de Von Neumann vs. Harvard

Arquitetura de Von Neumann *vs.* Harvard

- Embora enfrente limitações crescentes com as demandas computacionais atuais, o modelo continua sendo a base da maioria dos sistemas. As inovações futuras provavelmente representarão evoluções ou complementos, em vez de substituições completas deste paradigma fundamental.



Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa a arquitetura de Von Neumann vs. Harvard

Arquitetura de Von Neumann *vs*. Harvard

- O legado mais importante de Von Neumann talvez seja o conceito do programa armazenado, que tornou os computadores verdadeiramente programáveis e versáteis, transformando-os de calculadoras especializadas em máquinas universais capazes de resolver praticamente qualquer problema computacional.

Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa a arquitetura de Von Neumann vs. Harvard

Interatividade

Em uma arquitetura de Von Neumann e em uma arquitetura Harvard, qual é a principal diferença no que diz respeito ao armazenamento de dados e instruções?

a) Na arquitetura Von Neumann, dados e instruções são armazenados em memórias físicas separadas, enquanto na arquitetura Harvard eles compartilham a mesma memória.

b) Na arquitetura Von Neumann, dados e instruções compartilham o mesmo barramento, enquanto na arquitetura Harvard há barramentos separados para cada um.

c) Em ambas as arquiteturas, não há diferença quanto ao modo de armazenamento de dados e instruções.

d) Na arquitetura Von Neumann, dados e instruções são sempre armazenados no mesmo espaço de endereçamento físico, enquanto na arquitetura Harvard não há endereçamento físico.

e) Na arquitetura Harvard, dados e instruções são guardados em dispositivos periféricos externos, enquanto na arquitetura Von Neumann são armazenados na memória principal do sistema.

Resposta

Em uma arquitetura de Von Neumann e em uma arquitetura Harvard, qual é a principal diferença no que diz respeito ao armazenamento de dados e instruções?

a) Na arquitetura Von Neumann, dados e instruções são armazenados em memórias físicas separadas, enquanto na arquitetura Harvard eles compartilham a mesma memória.

b) Na arquitetura Von Neumann, dados e instruções compartilham o mesmo barramento, enquanto na arquitetura Harvard há barramentos separados para cada um.

c) Em ambas as arquiteturas, não há diferença quanto ao modo de armazenamento de dados e instruções.

d) Na arquitetura Von Neumann, dados e instruções são sempre armazenados no mesmo espaço de endereçamento físico, enquanto na arquitetura Harvard não há endereçamento físico.

e) Na arquitetura Harvard, dados e instruções são guardados em dispositivos periféricos externos, enquanto na arquitetura Von Neumann são armazenados na memória principal do sistema.

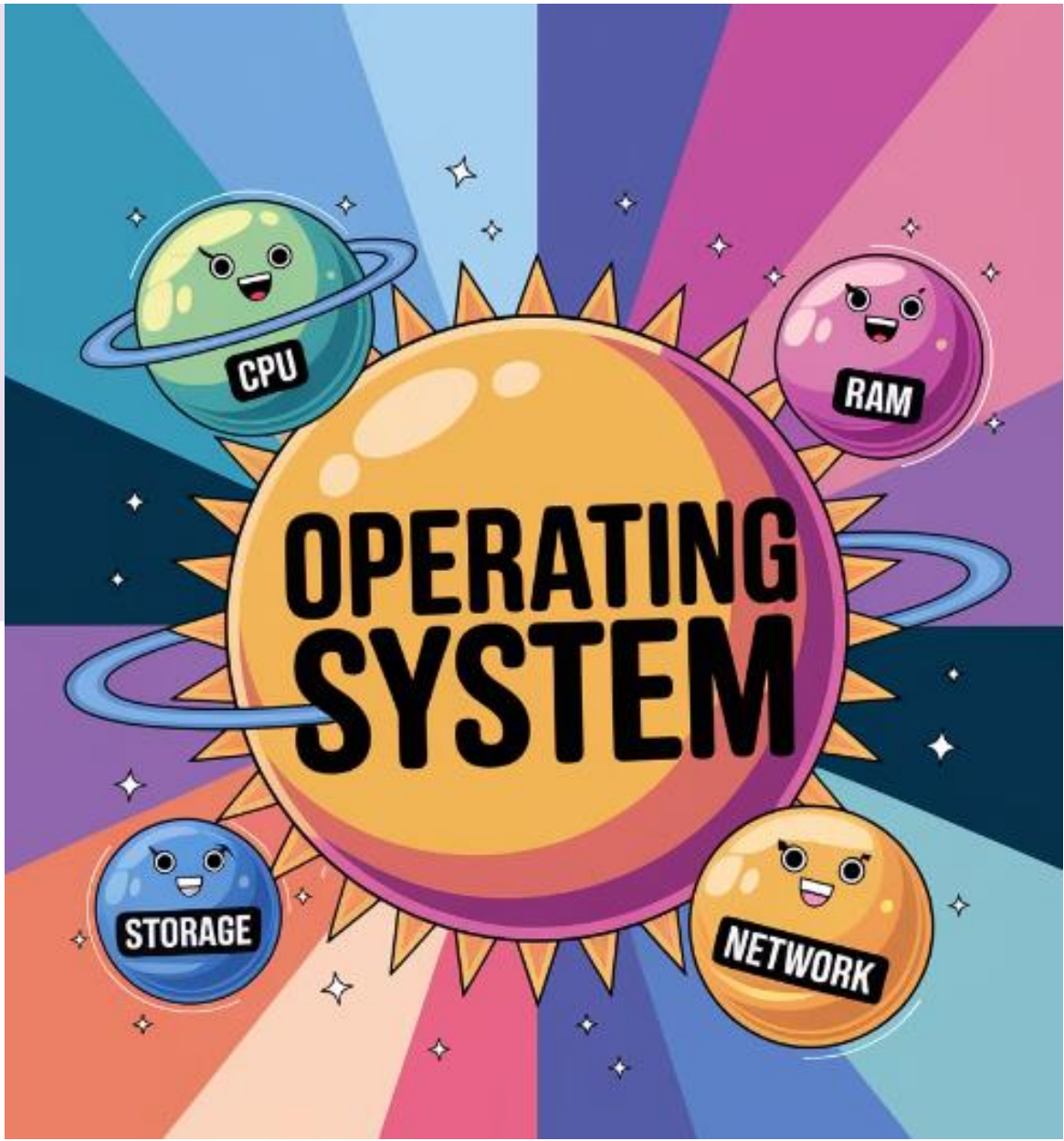
Sistema operacional

- Podemos considerar um sistema operacional como um alocador de recursos, uma vez que um sistema de computação possui muitos recursos que podem ser necessários à resolução de um problema (por exemplo: tempo de CPU, espaço de memória, espaço de armazenamento em arquivo, dispositivos de I/O etc.).
- SO atua como o gerenciador deles. Ao lidar com solicitações de recursos numerosas e possivelmente conflitantes, o sistema operacional precisa decidir como alocá-los a programas e usuários específicos, de modo a poder operar o sistema de computação de maneira eficiente e justa.

Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa o Sistema operacional

Recursos gerenciados pelo SO

- **Tempo de CPU:** Gerencia qual processo usa o processador e por quanto tempo.
- **Memória:** Controla a alocação e liberação do espaço de memória RAM.
- **Armazenamento**: Administra o espaço em dispositivos de armazenamento.
- **Dispositivos de I/O:** Gerencia o acesso a periféricos, como teclado e impressoras.



Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa o gerenciamento de recursos pelo Sistema operacional

O desafio da concorrência

- **Múltiplas Solicitações:** Vários programas e usuários competem pelos mesmos recursos limitados.
- **Conflitos Potenciais:** Dois programas podem precisar do mesmo recurso simultaneamente.
- **Papel do SO:** Resolver esses conflitos de forma transparente e eficiente para o usuário.

Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa o desafio de concorrência pelo Sistema operacional

Critérios de alocação de recursos

1. **Eficiência:** Maximizar o uso produtivo dos recursos disponíveis.
2. **Justiça:** Distribuir recursos equitativamente entre usuários e processos.
3. **Previsibilidade:** Garantir comportamento consistente do sistema.
4. **Desempenho:** Minimizar tempos de resposta e maximizar throughput.

Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa os critérios de alocação de recursos do Sistema operacional

Tipos de sistemas operacionais

Desktop: Interfaces amigáveis e multitarefa para computadores pessoais.

Servidor: Otimizados para estabilidade e gerenciamento de múltiplas conexões remotas.

Móvel: Projetados para eficiência energética e operação com toque em tela.

Fonte: Flux Fast 1.1, 2025 –
As imagens representam de forma figurativa tipos de sistemas operacionais

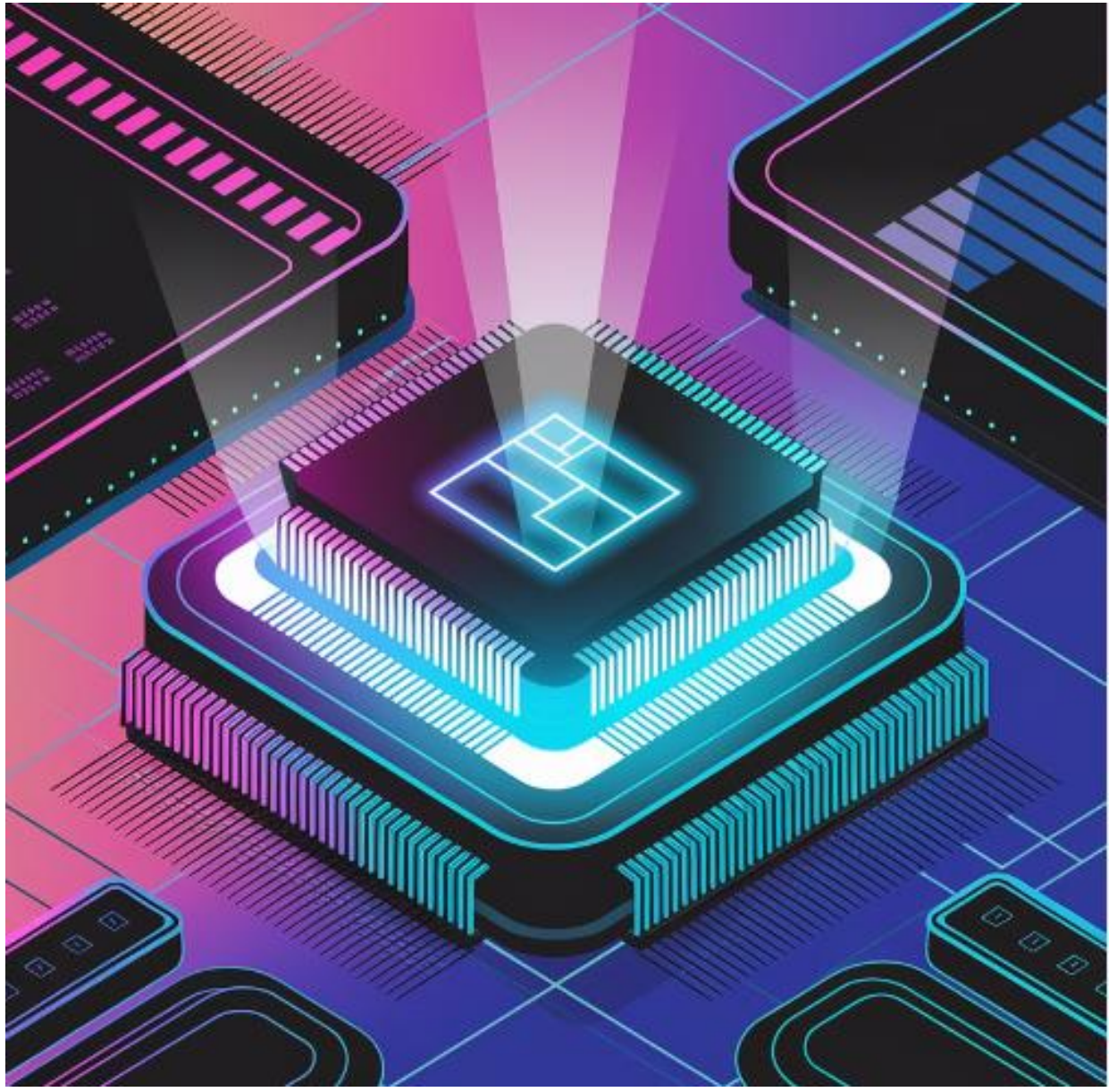
Gerenciamento de Processos

1. **Criação:** O SO inicia um novo processo alocando recursos necessários.
2. **Escalonamento:** Decide qual processo executará no processador e por quanto tempo.
3. **Sincronização:** Coordena processos que compartilham dados ou recursos.
4. **Finalização:** Encerra o processo e libera recursos para outros usos.

Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa gerenciamento de processos

Gerenciamento de Memória

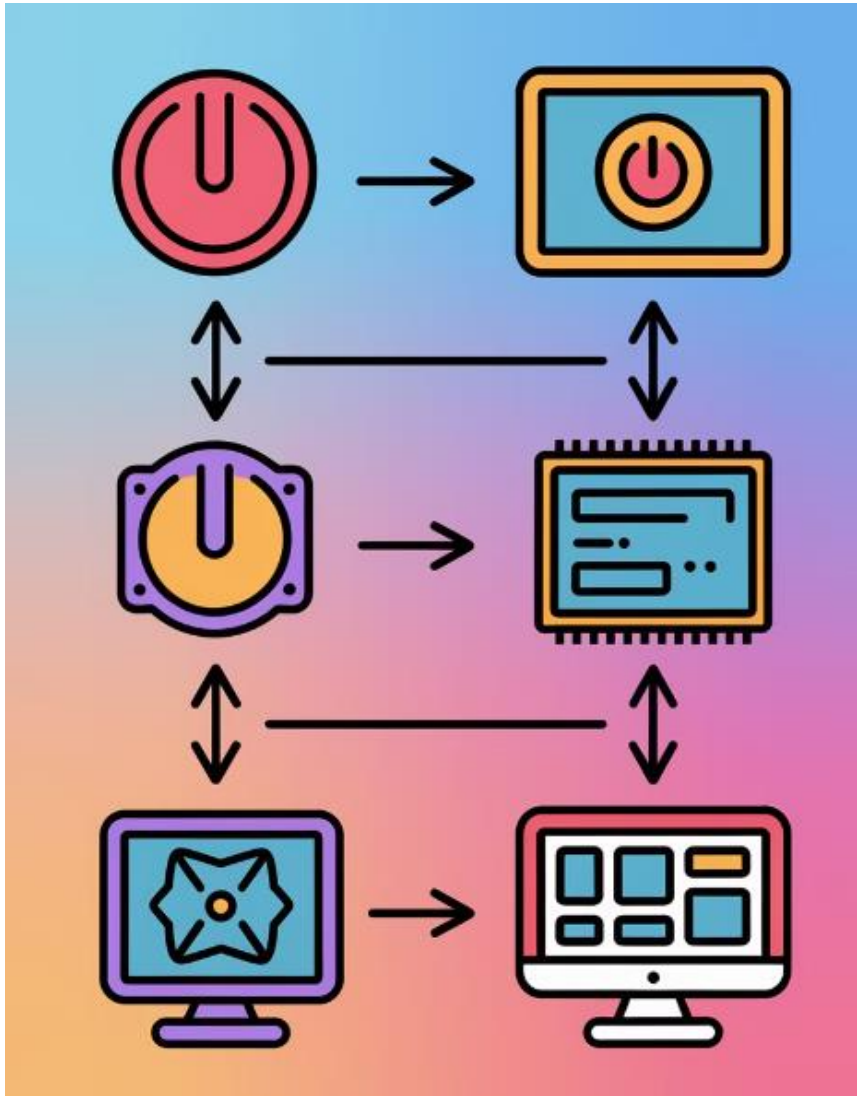
1. **Memória Virtual:** Espaço de endereçamento expandido usando disco.
2. **Paginação:** Divisão da memória em blocos de tamanho fixo.
3. **Segmentação:** Divisão lógica baseada na estrutura do programa.
4. **Memória Física:** Hardware RAM disponível no sistema.



Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa o gerenciamento de memória

Sistema de Arquivos

- **Interface do Usuário:** Manipulação de arquivos por nome e estrutura de diretórios.
- **Gerenciamento Lógico:** Organização de arquivos, permissões e metadados.
- **Alocação de Espaço:** Controle de blocos livres e ocupados no disco.
- **Drivers de Dispositivo:** Comunicação direta com hardware de armazenamento.



Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa os sistemas de arquivos

Gerenciamento de dispositivos de I/O

Fonte: Flux Fast 1.1, 2025 – As imagens representam de forma figurativa o gerenciamento de dispositivos de I/O

- O sistema operacional gerencia todos os dispositivos de entrada e saída através de drivers. Estes controlam o hardware e oferecem interfaces padronizadas para programas.

Segurança e Proteção

- **Modos de Operação:** Modo usuário executa aplicações com recursos limitados, enquanto o modo kernel permite operações privilegiadas no núcleo do sistema.
- **Mecanismos de Proteção:** Técnicas como isolamento de memória, controle de acesso e virtualização que impedem que processos interfiram ou comprometam a segurança de outros.
- **Níveis de Privilégio:** Hierarquia de permissões que restringe acesso ao hardware, protegendo operações críticas do sistema contra intervenções não autorizadas.

Fonte: Flux Fast 1.1, 2025 –
A imagem representa de forma figurativa a segurança e proteção

Desafios e futuro dos sistemas operacionais

- **Multicore e Paralelismo:** Otimizar o uso de processadores com múltiplos núcleos.
- **Virtualização:** Executar múltiplos sistemas operacionais em um único hardware.
- **Eficiência Energética:** Reduzir consumo em data centers e dispositivos móveis.
- **Segurança Avançada:** Proteger contra ameaças cada vez mais sofisticadas.

Fonte: Flux Fast 1.1, 2025 – A imagem representa de forma figurativa os desafios e future dos sistemas operacionais

Interatividade

Os Sistemas Operacionais utilizam diversas estruturas de dados para gerenciar recursos. Qual das alternativas descreve corretamente uma estrutura de dados utilizada para representar processos?

a) Tabela de Arquivos: mantém apenas as permissões de acesso a arquivos de cada usuário.
b) PCB (Process Control Block): armazena informações como registradores, contador de programa, estado e prioridade.
c) Tabela de I/O: cuida exclusivamente dos dispositivos de entrada, ignorando os de saída.
d) Vetor de Threads: um único vetor global que inclui todas as informações de processos e threads do sistema.
e) Fila de Mensagens: é usada apenas para comunicação entre processos, não armazenando informações do processo em si.

Resposta

Os Sistemas Operacionais utilizam diversas estruturas de dados para gerenciar recursos. Qual das alternativas descreve corretamente uma estrutura de dados utilizada para representar processos?

a) Tabela de Arquivos: mantém apenas as permissões de acesso a arquivos de cada usuário.

b) PCB (Process Control Block): armazena informações como registradores, contador de programa, estado e prioridade.

c) Tabela de I/O: cuida exclusivamente dos dispositivos de entrada, ignorando os de saída.

d) Vetor de Threads: um único vetor global que inclui todas as informações de processos e threads do sistema.

e) Fila de Mensagens: é usada apenas para comunicação entre processos, não armazenando informações do processo em si.

ATÉ A PRÓXIMA!