# A Arquitetura de Computadores: Uma Análise Detalhada do Modelo von Neumann e Seu Legado Duradouro

## Seção 1: Fundamentos da Arquitetura de Computadores

A compreensão da computação moderna exige um mergulho profundo nos seus alicerces. No cerne de todo dispositivo digital, do mais simples microcontrolador ao mais complexo supercomputador, reside um conjunto de princípios de design conhecido como arquitetura de computadores. Esta seção estabelece a base conceitual para a análise, definindo o termo e identificando os componentes estruturais que constituem um sistema computacional.

### 1.1. Definição de Arquitetura de Computadores

A arquitetura de computadores refere-se à organização conceitual e ao comportamento funcional de um sistema computacional, tal como é visível para um programador de software.1 É o conjunto de atributos e componentes que um programador deve compreender para desenvolver programas que executem corretamente, incluindo o conjunto de instruções, os modos de endereçamento de memória, os registradores e os formatos de dados.2 Essencialmente, a arquitetura define a interface fundamental entre o hardware e o software, ditando as regras segundo as quais o software pode interagir com a máquina.

É crucial distinguir *arquitetura* de *organização* (ou *microarquitetura*). A arquitetura é o "o quê" — o design lógico e abstrato. A organização é o "como" — a implementação física específica desses atributos arquitetônicos.1 Por exemplo, a instrução de máquina para uma adição faz parte da arquitetura. A forma como essa adição é fisicamente implementada com circuitos lógicos, a velocidade do clock e a tecnologia dos semicondutores pertencem à organização. Esta distinção explica como diferentes processadores podem partilhar a mesma arquitetura (e, portanto, executar o mesmo software) mas apresentar níveis de desempenho vastamente distintos. Em última análise, as decisões arquitetônicas determinam a qualidade, o desempenho e a aplicação para a qual um dispositivo é orientado.2

### 1.2. Componentes Estruturais de um Sistema Computacional

Apesar da enorme variedade de dispositivos, a maioria dos computadores modernos partilha um conjunto universal de blocos de construção.4 Estes componentes interagem de forma coordenada para executar programas e processar dados.

* **Unidade Central de Processamento (CPU):** Frequentemente descrita como o "cérebro" do computador, a CPU é o componente responsável por executar as instruções de um programa.5 Ela é composta por duas subunidades principais:
  + **Unidade de Controle (CU - Control Unit):** Atua como o maestro do sistema. Ela busca instruções da memória, decodifica-as e gera os sinais de controle que dirigem as outras unidades para executar a tarefa especificada.6
  + **Unidade Aritmética e Lógica (ALU - Arithmetic Logic Unit):** É o centro de cálculo da CPU. Realiza todas as operações matemáticas (como adição e subtração) e lógicas (como AND, OR e NOT) sobre os dados.6
* **Memória Principal (RAM - Random Access Memory):** É um dispositivo de armazenamento volátil que guarda temporariamente tanto as instruções dos programas em execução como os dados que estão a ser processados pela CPU.5 A sua característica de "acesso aleatório" significa que qualquer localização de memória pode ser acedida diretamente em aproximadamente o mesmo tempo.
* **Sistema de Entrada e Saída (E/S ou I/O):** Compreende o conjunto de dispositivos que permitem ao computador comunicar com o mundo exterior. Isto inclui dispositivos de entrada (teclados, ratos), dispositivos de saída (monitores, impressoras) e dispositivos de armazenamento secundário (discos rígidos - HDD, unidades de estado sólido - SSD), que fornecem armazenamento permanente e não volátil.4
* **Sistema de Interconexão (Barramento):** É o conjunto de vias de comunicação que conectam a CPU, a memória principal e o sistema de E/S. O barramento permite a transferência de dados, endereços e sinais de controle entre estes componentes essenciais, unificando-os num sistema funcional.8

A evolução tecnológica, desde os relés e válvulas até aos circuitos integrados de altíssima densidade (VLSI) 7, permitiu que as CPUs se tornassem exponencialmente mais rápidas. Contudo, a velocidade da memória principal (RAM) não acompanhou este ritmo vertiginoso. Esta crescente disparidade de desempenho criou um sério problema: a CPU, sendo muito mais rápida, passava grande parte do tempo ociosa, à espera que os dados e as instruções chegassem da memória mais lenta. Para resolver este descompasso, emergiu uma solução arquitetônica crucial: a

**hierarquia de memória**.9 Este conceito introduz níveis de memória mais pequenos, mais rápidos e mais caros (como a memória cache) entre a CPU e a RAM. A cache armazena os dados e instruções usados mais frequentemente, permitindo que a CPU os aceda quase instantaneamente. Portanto, a hierarquia de memória não é um mero complemento, mas uma consequência arquitetônica inevitável, nascida da necessidade de tornar os sistemas modernos viáveis ao mitigar o gargalo de desempenho criado pela própria evolução da tecnologia de semicondutores.

## Seção 2: A Revolução de von Neumann: O Conceito de Programa Armazenado

A transição de máquinas de calcular especializadas para os computadores programáveis de uso geral que conhecemos hoje foi catalisada por uma única e poderosa ideia. Esta seção explora o contexto histórico e o salto conceitual da arquitetura de von Neumann, que redefiniu fundamentalmente a natureza da computação.

### 2.1. Contexto Histórico: A Era Pré-von Neumann

Os primeiros computadores eletrônicos, como o ENIAC (Electronic Numerical Integrator and Computer), eram maravilhas da engenharia, mas fundamentalmente inflexíveis. Eram máquinas de "programa fixo".10 A sua função era determinada pela sua fiação física. Para "reprogramar" o ENIAC para uma tarefa diferente, os engenheiros tinham de passar dias ou semanas a reconfigurar manualmente milhares de cabos e interruptores. Este processo não só era extremamente demorado e propenso a erros, como também limitava drasticamente a utilidade da máquina a um conjunto restrito de problemas computacionais.

### 2.2. O Salto Conceitual: O Programa Armazenado

Em 1945, no seu influente artigo "First Draft of a Report on the EDVAC", o matemático John von Neumann formalizou uma ideia revolucionária que mudaria para sempre o curso da computação: o **conceito de programa armazenado** (*stored-program concept*).10 A proposta era radicalmente simples: as instruções que compõem um programa e os dados sobre os quais esse programa opera deveriam ser armazenados no mesmo espaço de memória.7

A implicação desta unificação foi profunda. Ao tratar as instruções como se fossem dados, a máquina ganhava a capacidade de ler, analisar e até mesmo modificar os seus próprios programas durante a execução.7 Esta capacidade de auto-referência é o que separa fundamentalmente um computador de uma calculadora. Deixou de ser necessário reconfigurar o hardware para cada nova tarefa; em vez disso, bastava carregar um novo programa na memória. Esta inovação deu origem aos "stored-program computers" 12 e tornou possível o desenvolvimento de ferramentas de software essenciais como compiladores (programas que traduzem código de alto nível para instruções de máquina), interpretadores e os próprios sistemas operativos. A máquina tornou-se universalmente programável.

Esta flexibilidade, no entanto, introduziu uma consequência não intencional que se tornou a base de muitas vulnerabilidades de segurança modernas. A unificação do espaço de memória para código e dados significa que não há uma distinção física inerente entre um byte que representa um número e um byte que representa uma instrução de CPU; a distinção é puramente contextual. Esta característica abriu a porta para ataques como o *buffer overflow*.10 O mecanismo de um ataque deste tipo explora diretamente o modelo de von Neumann:

1. Um programa vulnerável (por exemplo, que não verifica o tamanho da entrada de um utilizador) permite que um atacante escreva dados para além dos limites de um buffer de memória alocado.
2. O atacante cria uma entrada maliciosa que contém não apenas dados, mas também código de máquina (o *payload*).
3. Ao exceder o buffer, estes dados (incluindo o código malicioso) podem sobrescrever áreas adjacentes da memória, como o endereço de retorno de uma função na pilha de chamadas.
4. Quando a função termina, em vez de regressar ao ponto original do programa, o fluxo de execução é desviado para o endereço do código malicioso que o atacante inseriu na memória.
5. A CPU, sem distinguir a origem, começa a executar o *payload* do atacante como se fossem instruções legítimas.

Assim, a mesma flexibilidade arquitetônica que tornou os computadores programáveis também os tornou inerentemente vulneráveis. Grande parte da história da segurança informática tem sido um esforço contínuo para impor separações lógicas (como proteção de memória e o bit NX - *No-eXecute*) sobre uma arquitetura que, na sua essência, é unificada.

### 2.3. Os Pilares do Modelo de von Neumann

A arquitetura proposta por von Neumann, também conhecida como Modelo de Princeton, é a base de quase todos os computadores de uso geral atuais.12 O modelo é definido por um conjunto de componentes interligados:

* Uma **Unidade Central de Processamento (CPU)**, que contém a Unidade de Controle (CU) e a Unidade Aritmética e Lógica (ALU).7
* Uma **única Memória Principal**, que armazena tanto dados como instruções, com cada localização a ser acessível através de um endereço único.4
* Um **Sistema de Entrada e Saída (E/S)**, para a comunicação com periféricos e o mundo exterior.8
* Um **Sistema de Interconexão (Barramento)**, que serve como caminho de comunicação partilhado para ligar todos os componentes acima mencionados.8

Esta estrutura elegante e funcional provou ser extraordinariamente duradoura, formando o esqueleto de praticamente todos os sistemas computacionais que usamos hoje.

## Seção 3: A Dinâmica Operacional: O Ciclo de Instrução

A execução de um programa, por mais complexo que pareça, resume-se a um processo fundamental e repetitivo realizado pela CPU a uma velocidade vertiginosa. Este processo, conhecido como o ciclo de instrução, é o motor que impulsiona toda a atividade computacional num sistema baseado na arquitetura de von Neumann.

### 3.1. O Motor da Computação

Um programa de computador é, na sua essência, uma longa sequência de instruções armazenadas na memória. A função da CPU é executar estas instruções uma a uma, de forma ordenada. O mecanismo pelo qual a CPU processa cada uma destas instruções é o **ciclo de instrução**.19 Este ciclo repete-se continuamente, desde que o computador está ligado, processando milhões ou milhares de milhões de instruções por segundo.

### 3.2. As Fases do Ciclo

O ciclo de instrução, também conhecido como ciclo de busca-decodificação-execução, pode ser dividido em três fases principais que ocorrem sequencialmente para cada instrução 13:

1. **Busca (Fetch):** O ciclo começa com a Unidade de Controle (CU) a buscar a próxima instrução a ser executada da memória principal. Para saber onde encontrar essa instrução, a CU consulta um registrador especial chamado **Contador de Programa** (PC - *Program Counter*), que armazena o endereço de memória da próxima instrução. A instrução é então transferida da memória para outro registrador dentro da CPU, o **Registrador de Instrução** (IR - *Instruction Register*). Após a busca, o PC é automaticamente incrementado para apontar para a instrução seguinte na sequência.19
2. **Decodificação (Decode):** Uma vez que a instrução está no IR, a CU analisa o seu código de operação (opcode). Esta fase de decodificação determina que tipo de operação a instrução representa (por exemplo, uma adição, uma leitura da memória, um desvio condicional) e identifica os operandos (os dados) necessários para a sua execução.19
3. **Execução (Execute):** A operação especificada é finalmente realizada. Esta fase pode envolver diferentes partes da CPU. Se for uma operação aritmética ou lógica, a ALU é ativada. Se a instrução exigir a leitura ou escrita de dados na memória, a CU coordena o acesso à memória. Se for uma instrução de desvio (*jump* ou *branch*), o valor do Contador de Programa (PC) é alterado para um novo endereço, quebrando a execução sequencial e fazendo com que o ciclo de busca seguinte comece a partir de um ponto diferente do programa.19

Este ciclo repete-se para a instrução seguinte, e assim sucessivamente, até ao final do programa.4 A natureza estritamente sequencial deste processo é a causa direta de uma das limitações mais famosas da arquitetura: o "Gargalo de von Neumann". O problema reside no facto de tanto a fase de

*Busca* (para obter a instrução) como a fase de *Execução* (para obter ou guardar os dados) poderem necessitar de aceder à memória principal. Como a arquitetura de von Neumann possui apenas um único barramento partilhado para ligar a CPU à memória 10, estas duas operações não podem ocorrer em simultâneo. A CPU não pode iniciar a busca da instrução

N+1 enquanto ainda está a aceder à memória para obter os dados necessários para a instrução N. Esta contenção pelo acesso ao único caminho de comunicação força a CPU a esperar, criando um "gargalo" que limita a taxa de transferência de informação e, consequentemente, o desempenho geral do sistema.

## Seção 4: Análise Crítica da Arquitetura de von Neumann

Nenhuma arquitetura é perfeita; cada design representa um conjunto de compromissos. O modelo de von Neumann, apesar do seu sucesso e longevidade, não é exceção. Esta seção oferece uma avaliação equilibrada das suas vantagens duradouras e das suas desvantagens intrínsecas, com um foco particular no seu mais notório ponto fraco.

### 4.1. Vantagens

A dominância da arquitetura de von Neumann ao longo de décadas pode ser atribuída a um conjunto de vantagens claras e pragmáticas:

* **Simplicidade e Flexibilidade:** O design de memória unificada é conceitualmente mais simples do que modelos com espaços de memória separados. Esta simplicidade facilita não só o projeto e a construção do hardware, mas também o desenvolvimento de software de sistema, como compiladores e sistemas operativos, que podem gerir um único e flexível pool de memória para alocar dinamicamente espaço para código ou dados conforme necessário.10
* **Eficiência de Custos:** A simplicidade do modelo e a consequente padronização dos componentes permitiram a produção em massa a uma escala sem precedentes. Isto reduziu drasticamente os custos de fabrico, tornando os computadores uma tecnologia acessível e omnipresente.10

### 4.2. Desvantagens e o "Gargalo de von Neumann"

Apesar dos seus pontos fortes, a arquitetura possui limitações significativas que os engenheiros de computadores têm tentado mitigar desde a sua concepção.

* **O Gargalo de von Neumann (The von Neumann Bottleneck):** Esta é a desvantagem mais fundamental. O termo, cunhado por John Backus em 1977, descreve a limitação na taxa de transferência de dados (largura de banda) entre a CPU e a memória principal. Esta limitação é uma consequência direta do uso de um único barramento partilhado para transportar tanto as instruções como os dados.10 A CPU, capaz de processar dados a uma velocidade muito superior à que o barramento consegue fornecer, é constantemente forçada a esperar, limitando a velocidade efetiva de processamento.
* **Processamento Sequencial:** O modelo original executa as instruções estritamente uma após a outra. Este processamento sequencial é inerentemente mais lento do que arquiteturas que permitem algum grau de paralelismo na execução de instruções.10
* **Vulnerabilidades de Segurança:** Como discutido anteriormente, o espaço de memória partilhado para código e dados cria vulnerabilidades intrínsecas, como o *buffer overflow*, que podem permitir a execução de código malicioso.10

O "Gargalo de von Neumann" não é um problema estático; é um desafio dinâmico que, paradoxalmente, se agrava com o avanço da tecnologia. A Lei de Moore previu com sucesso o crescimento exponencial da velocidade dos processadores (medida pelo número de transístores). A capacidade da memória também cresceu a um ritmo impressionante. No entanto, a largura de banda do barramento que os conecta não evoluiu na mesma proporção. Isto significa que a *diferença relativa* de velocidade entre a CPU e a sua capacidade de ser "alimentada" com dados e instruções aumenta a cada nova geração de hardware. Uma CPU mais rápida torna-se ainda mais "faminta" por dados, e um barramento comparativamente lento torna-se um constrangimento ainda maior. É por esta razão que a inovação em arquitetura de computadores nas últimas décadas se concentrou menos em simplesmente aumentar a velocidade de clock da CPU e mais em estratégias sofisticadas para *mitigar o gargalo*. Técnicas como caches de múltiplos níveis, predição de desvios (*branch prediction*), e computação paralela (múltiplos núcleos) são todas respostas diretas a este desafio crescente. A luta contra o gargalo de von Neumann tornou-se, de facto, o principal motor da evolução da microarquitetura moderna.

## Seção 5: Perspectivas Comparativas: von Neumann vs. Harvard

Para apreciar plenamente os compromissos de design inerentes ao modelo de von Neumann, é instrutivo compará-lo com a sua principal alternativa histórica e conceitual: a arquitetura de Harvard. Esta comparação ilumina diferentes filosofias de design e os seus respetivos domínios de aplicação.

### 5.1. A Arquitetura de Harvard

Desenvolvida aproximadamente na mesma época que o modelo de von Neumann, a arquitetura de Harvard foi implementada no computador Harvard Mark I. A sua característica definidora, e o seu principal ponto de divergência em relação ao modelo de von Neumann, é a **separação física da memória e dos barramentos para instruções e dados**.17 Num sistema Harvard, existem dois sistemas de memória completamente independentes, cada um com o seu próprio barramento de acesso: um dedicado a armazenar o programa (instruções) e outro a armazenar os dados.

A consequência imediata desta separação é a capacidade de a CPU aceder a instruções e dados em simultâneo. Enquanto a Unidade de Controle busca a próxima instrução da memória de programa, a Unidade Aritmética e Lógica pode estar a ler ou a escrever dados na memória de dados, eliminando a contenção pelo barramento que cria o gargalo de von Neumann.22

### 5.2. Análise Comparativa

As diferenças fundamentais entre as duas arquiteturas levam a vantagens e desvantagens distintas, tornando cada uma mais adequada para diferentes tipos de aplicações.

* **Desempenho:** A arquitetura de Harvard oferece um potencial de desempenho superior. A capacidade de sobrepor a busca de uma instrução com a execução da instrução anterior (uma técnica conhecida como *pipelining*) é muito mais eficiente num sistema Harvard, pois os acessos à memória de instrução e de dados não interferem um com o outro. Isto resulta num processamento mais rápido e num maior rendimento (*throughput*).22
* **Complexidade e Custo:** O preço a pagar pelo maior desempenho é uma maior complexidade de hardware. A necessidade de gerir dois sistemas de memória e dois barramentos separados torna o design mais complexo e, historicamente, mais caro de implementar.
* **Casos de Uso:** A flexibilidade e o menor custo da arquitetura de von Neumann tornaram-na a escolha dominante para computadores de uso geral, como PCs e servidores, onde os programas e os dados variam muito em tamanho e a gestão dinâmica da memória é crucial.24 Por outro lado, a arquitetura de Harvard é prevalente em domínios onde o desempenho é crítico e os padrões de acesso são mais previsíveis. É comum em Processadores de Sinal Digital (DSPs), usados em processamento de áudio e vídeo, e em muitos microcontroladores (como a família PIC da Microchip), onde a execução rápida e determinística de um programa fixo é a principal prioridade.22

A tabela seguinte resume as principais diferenças estruturais e funcionais entre as duas arquiteturas.

**Tabela 1: Comparativo Estrutural: Arquitetura von Neumann vs. Arquitetura Harvard**

| Característica | Arquitetura de von Neumann | Arquitetura de Harvard |
| --- | --- | --- |
| **Memória** | Unificada para dados e instruções.7 | Separada para dados e instruções.17 |
| **Barramento** | Único para acesso a dados e instruções.10 | Barramentos distintos para dados e instruções.22 |
| **Acesso à Memória** | Sequencial (conflito de acesso).24 | Potencialmente simultâneo.22 |
| **Ciclo de Instrução** | Mais lento; não pode buscar a próxima instrução durante o acesso a dados da atual.24 | Mais rápido; permite *pipelining* (busca da próxima instrução durante a execução).22 |
| **Complexidade do Hardware** | Mais simples e de menor custo.10 | Mais complexo e potencialmente mais caro. |
| **Aplicações Típicas** | Computadores de uso geral (PCs, servidores).24 | Sistemas embarcados, microcontroladores (PIC), DSPs.22 |
| **Principal Vantagem** | Flexibilidade e simplicidade de design.10 | Alta performance e velocidade de processamento.22 |
| **Principal Desvantagem** | Gargalo de von Neumann.10 | Maior complexidade de design e menor flexibilidade de memória. |

## Seção 6: O Legado de von Neumann na Computação Moderna

Apesar das suas limitações e do surgimento de arquiteturas alternativas, o modelo de von Neumann não é uma mera relíquia histórica. Os seus princípios fundamentais não só persistem, como também evoluíram, formando a base sobre a qual os complexos sistemas híbridos da atualidade são construídos.

### 6.1. A Prevalência do Modelo Conceitual

A nível conceitual, a arquitetura de programa armazenado de von Neumann continua a ser o paradigma dominante para todos os computadores de uso geral.11 A razão para esta longevidade é a sua flexibilidade inigualável. A capacidade de ter um único e vasto espaço de memória que pode ser dinamicamente alocado para programas complexos, bibliotecas partilhadas e grandes conjuntos de dados é indispensável para os sistemas operativos multitarefa e as aplicações ricas em funcionalidades que definem a computação moderna. Um sistema com memórias de código e dados estritamente separadas (como um modelo Harvard puro) seria extremamente ineficiente e impraticável para este tipo de ambiente.

### 6.2. A Evolução para Arquiteturas Híbridas

A realidade, no entanto, é mais matizada. Os processadores modernos não são implementações "puras" da arquitetura de von Neumann.26 Para alcançar os níveis de desempenho exigidos atualmente, eles tiveram de adotar, na sua

*microarquitetura* interna, características da arquitetura de Harvard como forma de mitigar o inevitável gargalo de von Neumann.

O exemplo mais proeminente desta evolução é a implementação da **memória cache**. A maioria das CPUs modernas possui caches de Nível 1 (L1) que são separadas para instruções (L1i) e dados (L1d).7 Esta estrutura cria um pequeno subsistema de Harvard no coração de um sistema von Neumann maior. Ao interagir com a cache L1, a CPU pode buscar uma instrução da L1i e, simultaneamente, ler ou escrever um dado na L1d, contornando eficazmente o gargalo ao nível mais próximo do processamento. No entanto, quando a CPU precisa de aceder aos níveis de cache superiores (L2, L3) ou à memória RAM principal, a comunicação volta a seguir o modelo de von Neumann, através de um caminho unificado.

Esta abordagem híbrida representa uma síntese pragmática e engenhosa das duas filosofias. A computação moderna não representa uma vitória de uma arquitetura sobre a outra, mas sim a sua integração inteligente.

1. O problema a ser resolvido era como obter a flexibilidade da memória unificada de von Neumann e, ao mesmo tempo, a velocidade do acesso paralelo de Harvard.
2. A solução não foi escolher um modelo, mas combiná-los hierarquicamente.
3. No nível mais interno e crítico para a velocidade, o da cache L1, implementou-se uma arquitetura de Harvard para maximizar o rendimento do processador.
4. Nos níveis mais externos do sistema de memória, onde a capacidade e a flexibilidade são mais importantes do que a latência mínima, manteve-se a arquitetura de von Neumann.

Em suma, o computador moderno pode ser descrito como um sistema von Neumann por fora, visível ao sistema operativo e ao programador, e um sistema Harvard por dentro, na sua microarquitetura otimizada para o desempenho. Este legado híbrido demonstra a genialidade duradoura do modelo de von Neumann, não como um projeto rígido, mas como uma fundação conceitual robusta o suficiente para absorver e integrar inovações de paradigmas concorrentes, adaptando-se continuamente aos desafios da computação de alta performance.

#### Referências citadas

1. Aula 01 – Arquitetura de Computadores - Site Professor Carlos Fernandes, acessado em setembro 12, 2025, <http://cecead.com/assuntos/disciplinas/arquitetura-de-computadores/aula-01-arquitetura-de-computadores/>
2. Arquitetura de computadores – Wikipédia, a enciclopédia livre, acessado em setembro 12, 2025, <https://pt.wikipedia.org/wiki/Arquitetura_de_computadores>
3. a:arquitetura\_de\_computadores [Wiki Computação], acessado em setembro 12, 2025, <https://wiki.inf.ufpr.br/computacao/doku.php?id=a:arquitetura_de_computadores>
4. Arquitetura de Computadores - UFSM, acessado em setembro 12, 2025, <https://www.ufsm.br/app/uploads/sites/413/2018/12/02_arquitetura_computadores.pdf>
5. Arquitetura Interna de um Computador - EbookMaker.ai, acessado em setembro 12, 2025, <https://ebookmaker.ai/pt-BR/arquitetura-interna-de-um-computador-j43yl>
6. Componentes do Computador e noções de Arquitetura de Computadores - dca.ufrn, acessado em setembro 12, 2025, <https://www.dca.ufrn.br/~affonso/DCA800/pdf/componentes_e_arquitetura_slides.pdf>
7. Arquitetura de von Neumann – Wikipédia, a enciclopédia livre, acessado em setembro 12, 2025, <https://pt.wikipedia.org/wiki/Arquitetura_de_von_Neumann>
8. 3 –– arquitetura de von neumann e desempenho de computadores - facom/ufu, acessado em setembro 12, 2025, <https://www.facom.ufu.br/~gustavo/OC1/Apresentacoes/Arquitetura%20von%20Newmann.pdf>
9. Arquitetura de Computadores - Universidade Estadual do Ceará, acessado em setembro 12, 2025, <https://www.uece.br/cct/wp-content/uploads/sites/28/2021/07/Arquitetura-de-Computadores.pdf>
10. Vantagens e Desvantagens da Arquitetura de Von Neumann | PDF - Scribd, acessado em setembro 12, 2025, <https://pt.scribd.com/document/905074249/Vantagens-e-Desvantagens-da-Arquitetura-de-Von-Neumann>
11. A Arquitetura que Revolucionou o Mundo: Uma breve introdução ao ..., acessado em setembro 12, 2025, <https://dev.to/gabrieldantasds/a-arquitetura-que-revolucionou-o-mundo-uma-breve-introducao-ao-trabalho-de-john-von-neumann-1j0m>
12. O que é a Arquitetura de von Neumann? - Canaltech, acessado em setembro 12, 2025, <https://canaltech.com.br/hardware/o-que-e-arquitetura-de-von-neumann/>
13. Compreendendo os enigmas da arquitetura de Von Neumann: o núcleo da tecnologia da computação - Dadao, acessado em setembro 12, 2025, <https://dadaoenergy.com/pt/blog/neumann-architecture/>
14. Arquitetura de Von Neumann e Harvard | PDF | Unidade central de processamento - Scribd, acessado em setembro 12, 2025, <https://fr.scribd.com/document/325355803/Arquitetura-de-Von-Neumann-e-Harvard>
15. O Que é Princípio De Neumann? - Chapecali, acessado em setembro 12, 2025, <https://chapecali.com.br/glossario/o-que-e-principio-de-neumann/>
16. Von Neumann: suas contribuições à Computação - SciELO, acessado em setembro 12, 2025, <https://www.scielo.br/j/ea/a/DsFyHhVJ6krf5vjzVjx3dRR/>
17. Arquitetura de von Neumann e arquitetura de Harvard | Distribuidor de componentes eletrónicos. Loja online: Transfer Multisort Elektronik Portugal - TME.eu., acessado em setembro 12, 2025, <https://www.tme.eu/pt/news/library-articles/page/56104/arquitetura-de-von-neumann-e-arquitetura-de-harvard/>
18. Arquitetura Von Neumann - Sistema Operacional | PDF - Scribd, acessado em setembro 12, 2025, <https://pt.scribd.com/document/730541612/Arquitetura-Von-Neumann-Sistema-Operacional>
19. O Que é a Arquitetura de Von Neumann e Sua Importância, acessado em setembro 12, 2025, <https://conceitos.tech/fundamentos-da-tecnologia/principios-de-computacao/arquitetura-de-von-neumann/>
20. A máquina de Von Neumann - Bruno Rodrigues, acessado em setembro 12, 2025, <https://brunorodriguesti.files.wordpress.com/2011/02/aula2.pdf>
21. Confira: Arquitetura de John Von Neumann - Embarcados, acessado em setembro 12, 2025, <https://embarcados.com.br/arquitetura-de-john-von-neumann/>
22. Arquitetura Harvard – Wikipédia, a enciclopédia livre, acessado em setembro 12, 2025, <https://pt.wikipedia.org/wiki/Arquitetura_Harvard>
23. Arquitetura Harvard vs. Von Neumann: Entenda as Diferenças e Aplicações, acessado em setembro 12, 2025, <https://conceitos.tech/hardware-e-arquitetura/arquitetura-de-processadores/arquitetura-harvard-vs-arquitetura-von-neumann/>
24. Arquitetura: Von Neumann Vs Harvard - Diego Macêdo, acessado em setembro 12, 2025, <https://www.diegomacedo.com.br/arquitetura-von-neumann-vs-harvard/>
25. Arquitetura de von Neumann: A Revolução dos Computadores Modernos - YouTube, acessado em setembro 12, 2025, <https://m.youtube.com/watch?v=eJlgHmHCNfY>
26. A arquitetura de Von-Neumann é usada em GPUs? : r/compsci - Reddit, acessado em setembro 12, 2025, <https://www.reddit.com/r/compsci/comments/z8myj8/is_the_vonneumann_architecture_used_on_gpus/?tl=pt-br>