Estudantes: Barbara Letícia da Silva e Marcelo de Alcantra Janotti.

Prontuários: CJ3029921 e CJ3019675.

1° Semestre.

Professor: Josivan Pereira da Silva.

Atividade de Pesquisa — Arquitetura de Computadores

Tópicos e questões para pesquisa

1. Arquitetura *Harvard*

O que caracteriza a Arquitetura *Harvard*?

R: A Arquitetura *Harvard* é um modelo de computação que separa fisicamente a memória de instruções da memória de dados, cada uma com seu barramento próprio. Isso possibilita que o processador acesse ambas simultaneamente, evitando conflitos e aumentando a velocidade. Por sua eficiência, é comum em sistemas que exigem alta performance, como *DSPs* e microcontroladores avançados. Sua principal vantagem é a execução paralela de buscas e operações, otimizando o fluxo de processamento.

Quais são as principais diferenças entre *Harvard* e *Von Neumann*?

R: A principal diferença entre as arquiteturas *Harvard* e *Von Neumann* está na organização da memória e dos barramentos.

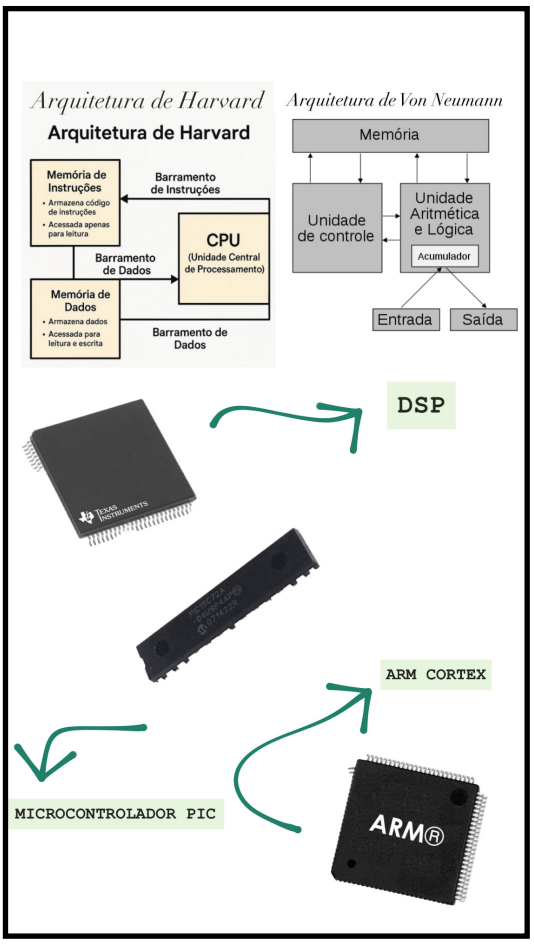
Von Neumann: Usa uma única memória para instruções e dados, com um único barramento. Isso simplifica o sistema, mas cria um gargalo, impedindo o acesso simultâneo. Já a arquitetura de *Harvard*, possui memórias separadas, cada uma com seu barramento próprio. Isso permite acesso simultâneo, eliminando gargalos e melhorando o desempenho, mas com maior complexidade de implementação.

Cite 3 exemplos de dispositivos ou processadores que utilizam a arquitetura *Harvard*.

R: A arquitetura de *Harvard* é comum em sistemas embarcados, microcontroladores e *DSPs*, onde desempenho e eficiência são essenciais. Sua preferência deve-se ao acesso simultâneo a instruções e dados, o que melhora a eficiência. Exemplos incluem: ***ARM Cortex-M3/M4/M7*:** Usam barramento *Harvard* interno para maior velocidade. **Microcontroladores *PIC* (*PIC*10/12/16/18, *PIC24*, *dsPIC*):** Adotam versões modificadas dessa arquitetura. ***DSPs* (*TMS320* da *Texas Instruments*, *SHARC* da *Analog Devices*):** Utilizam *Harvard* para processamento rápido de sinais.

Quais são as vantagens e desvantagens dessa arquitetura?

R: A arquitetura apresenta **vantagens,** como maior velocidade devido ao acesso paralelo e maior segurança pela separação entre dados e código. Por outro lado, também possui **desvantagens,** incluindo maior complexidade de *hardware* e menor flexibilidade, já que não permite a auto-modificação de código com facilidade.



A **Figura 1** ao lado apresentará de forma gráfica como ocorre o funcionamento da Arquitetura de *Harvard* e de *Von Neumann*. Os respectivos exemplos que utilizam da arquitetura de *Harvard* também são apresentados. Todas as imagens foram retiradas da internet, já a montagem foi feita pela Estudante Barbara.

Figura 1

2. Compilador *vs.* Interpretador

O que é um compilador? Como ele funciona?

R: O compilador converte todo o código-fonte em código de máquina de uma vez, gerando um executável. Seu funcionamento ocorre dessa forma: Análise léxica, sintática e geração de código otimizado.

O que é um interpretador? Como ele funciona?

R: O interpretador executa o código linha por linha, sem gerar um executável prévio. Seu funcionamento ocorre durante a tradução em tempo real ao longo da execução.

1. **Principais diferenças entre compilador e interpretador.**
2. **Exemplos de linguagens compiladas (pelo menos 3).**
3. **Exemplos de linguagens interpretadas (pelo menos 3).**
4. **Existe alguma linguagem que possa ser tanto compilada quanto interpretada? Dê um exemplo.**

Característica	Compilador	Interpretador	Híbrido (Compilado + Interpretado)
Processamento	Conversão total do código em executável	Execução linha a linha	Compilação para *bytecode* + interpretação
Velocidade	Rápida (execução direta)	Mais lenta (análise em tempo real)	Velocidade intermediária
Portabilidade	Executável específico para cada SO	Altamente portátil (código-fonte universal)	Portável (*bytecode* roda em *VM*)
Exemplos de Linguagens	**Compiladas:** • C • *C++* • *Rust* • Go	**Interpretadas:** • *Python* • *JavaScript* • *Ruby* • *PHP*	

A **Figura 2** foi pensada e criada com o objetivo de simplificar as demandas das perguntas de 1 à 4 descritas acima.

Sendo assim, a **Figura 2** por sua vez é uma tabela na qual apresenta todas às respostas, além de seu design ser inspirado na logo do IFSP, realizado pela Estudante Barbara.

Figura 2

3. Arquiteturas *RISC*, *CISC* e *ARM*

O que é arquitetura *RISC*? Onde ela é mais utilizada?

R: *RISC* usa instruções simples para eficiência energética em *chips ARM*, *IoT* e sistemas embarcados.

O que é arquitetura *CISC*? Onde ela é mais utilizada?

R: *CISC* usa instruções complexas para multitarefa por ciclo. Dominante em *Intel Core*, *AMD Ryzen*(*x*86) para alta performance.

O que é arquitetura *ARM*? Onde é mais utilizada atualmente?

R: *ARM* é uma arquitetura *RISC* eficiente, usada em *chips* como *Snapdragon* e *Apple M*1 para dispositivos móveis e *IoT*.

Cite exemplos de processadores para cada arquitetura.

R: *RISC*: *ARM Cortex* (*Snapdragon*), *MIPS*, *RISC-V*; ***CISC*:** *Intel Core i7, AMD Ryzen*; ***ARM*:** *Apple M*1/*M*2, *Qualcomm Snapdragon*, *NVIDIA Tegra*.

Qual é a relação entre *ARM* e *RISC*?

R: Baseada em *RISC*, a arquitetura *ARM* combina eficiência energética e alto desempenho com técnicas como *Thumb* e *pipelines* avançados. É padrão em dispositivos móveis, *IoT* e computadores como *Apple Silicon*.

Figura 3

A **Figura 3** apresenta de maneira simplificada as principais características das arquiteturas *RISC* e *CISC*. Imagens da internet; montagem feita pela estudante Barbara.

Referências Bibliográficas

Arquitetura de *Von Neumann vs*. Arquitetura *Harvard*. Disponível em: *https://tiparafiscos.com.br/arquitetura/von-neumann-harvard.html* .

Acesso em 20 de Agosto de 2025.

Arquitetura de Computadores/CJOARQ *Von Neumann*, *Harvard* e Unidades de Controle - Prof. Dr Josivan. Disponível em : *https://suap.ifsp.edu.br/media/private-media/edu/material_aula/154016ffb78d-ebe3d66f99fa4f30b3921b3cf8e3d366.pdf?st=PMtKgUGtmkU2caNKOxxxXQ&e=1755775997* .

Acesso em 20 de Agosto de 2025.