Organização e Arquitetura de Computadores

Arquiteturas RISC x CISC

Prof. Msc. Francisco de Assis de Souza Rodrigues

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO

Objetivo

- Entender os conceitos de arquitetura RISC e CISC.
- Compreender as principais diferenças, vantagens e desvantagens de cada abordagem.
- Relacionar essas arquiteturas com processadores atuais (ARM, Intel, AMD).
- Estimular a análise crítica sobre escolhas de hardware na indústria.

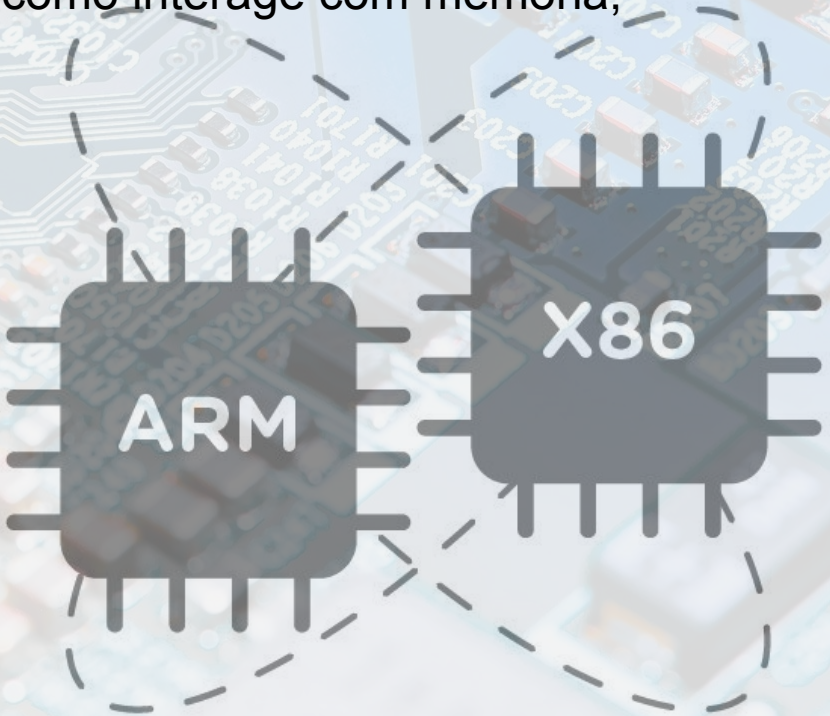


O que é arquitetura de processador?

A arquitetura de um processador é o conjunto de conceitos, especificações, padrões e organização interna que define como um processador funciona, como ele interpreta e executa instruções e como interage com memória, periféricos e outros componentes.

Ela engloba:

- Conjunto de Instruções
- Microarquitetura
- Fluxo de Dados e Controle
- Componentes internos



O que é arquitetura de processador?

🏗️ **Conjunto de Instruções (Instruction Set Architecture - ISA)**:

- Linguagem de máquina que o processador entende (ex.: x86, ARM, RISC-V).
- Define as operações que podem ser realizadas (somar, multiplicar, mover dados, comparar, etc.).

🔧 **Microarquitetura**:

- Implementação física da arquitetura.
- Dois processadores podem ter a mesma arquitetura (ex.: x86), mas microarquiteturas diferentes (ex.: Intel Core i7 e AMD Ryzen).

O que é arquitetura de processador?

🔄 **Fluxo de Dados e Controle**:

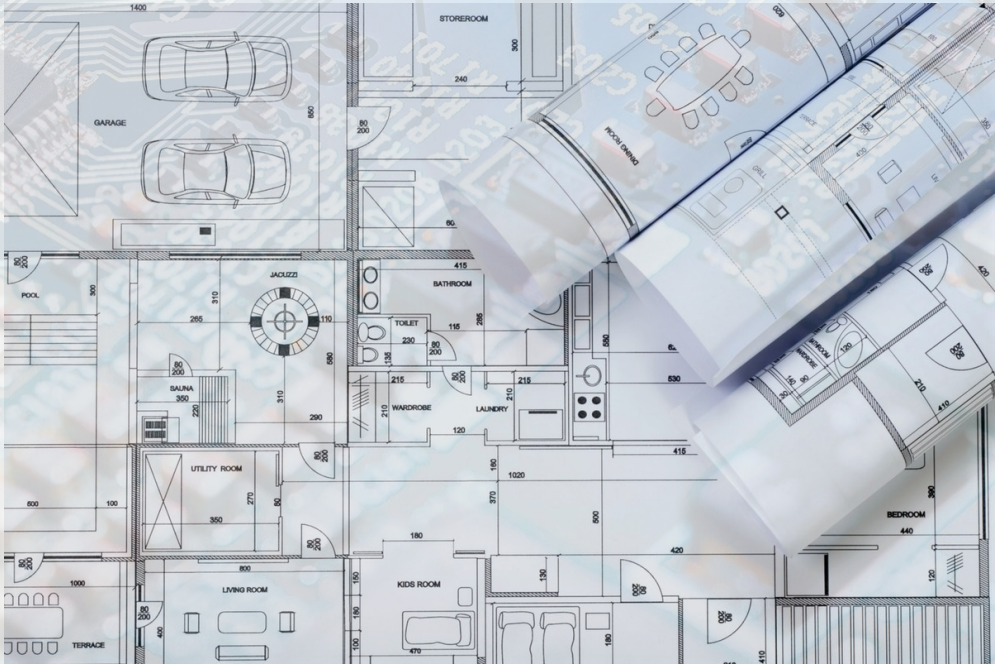
- Como dados fluem entre registradores, memória, ALU (Unidade Lógica e Aritmética) e unidade de controle.

⚙️ **Componentes internos:**

- Registradores, ALU, unidade de controle, caches, barramentos, pipeline.

O que é arquitetura de processador?

💡 **Analogia:** A arquitetura de processador é como o projeto de uma casa (planta arquitetônica). Já a microarquitetura é como os materiais e técnicas usadas para construir aquela casa (madeira, tijolo, concreto, acabamento, etc.).



Evolução Histórica das Arquiteturas de CPUs

Décadas de 1940–1950: Primeiras Gerações

- Computadores como ENIAC e UNIVAC.
- Arquitetura baseada em válvulas.
- Programação feita diretamente com cabos ou cartões perfurados.
- Pouca ou nenhuma distinção entre hardware e software.

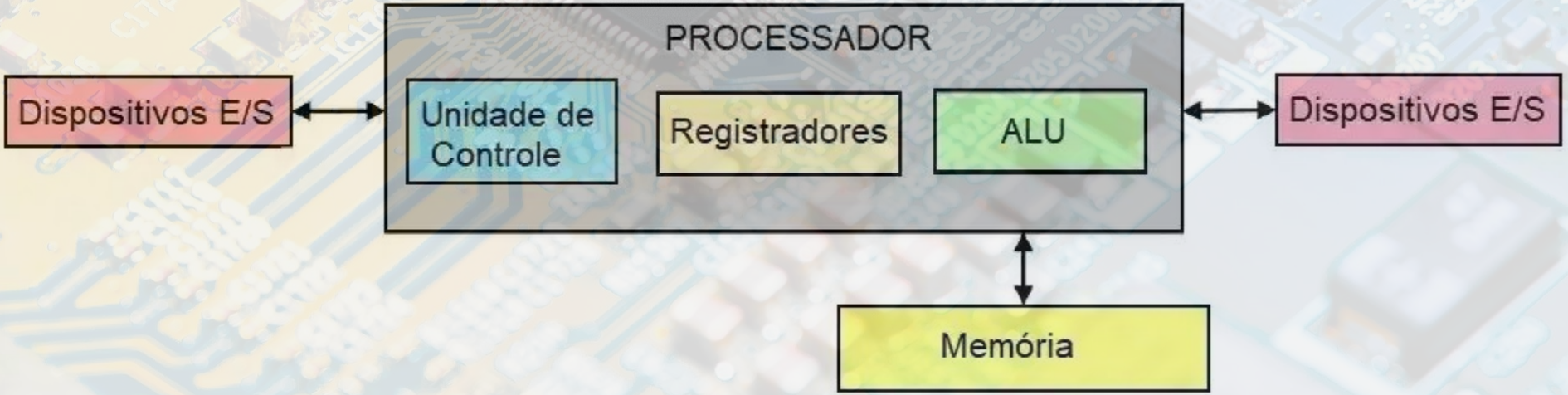




🕰️ Evolução Histórica das Arquiteturas de CPUs

🔣 1950–1960: Arquitetura de Von Neumann

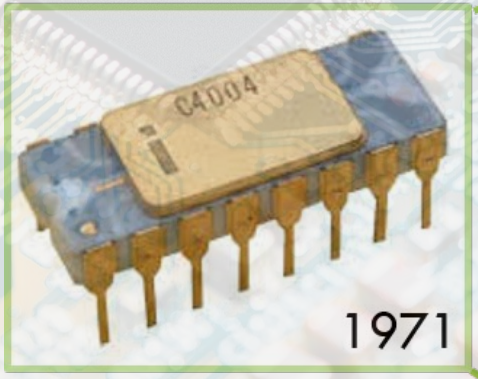
- Proposta por John von Neumann.
- Ideia de um **armazenamento comum para dados e instruções**.
- Pipeline básico: busca, decodificação e execução.

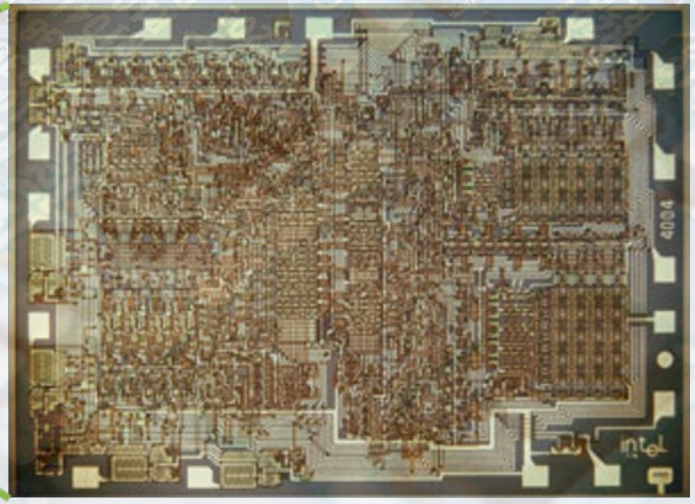


🕰️ Evolução Histórica das Arquiteturas de CPUs

🏗️ 1970: Explosão dos Microprocessadores

- Lançamento do Intel 4004 (1971), primeiro microprocessador comercial.
- Processadores começam a integrar milhares de transistores em um único chip.
- Arquiteturas começam a adotar conjuntos de instruções mais robustos (CISC).





2300 transistores

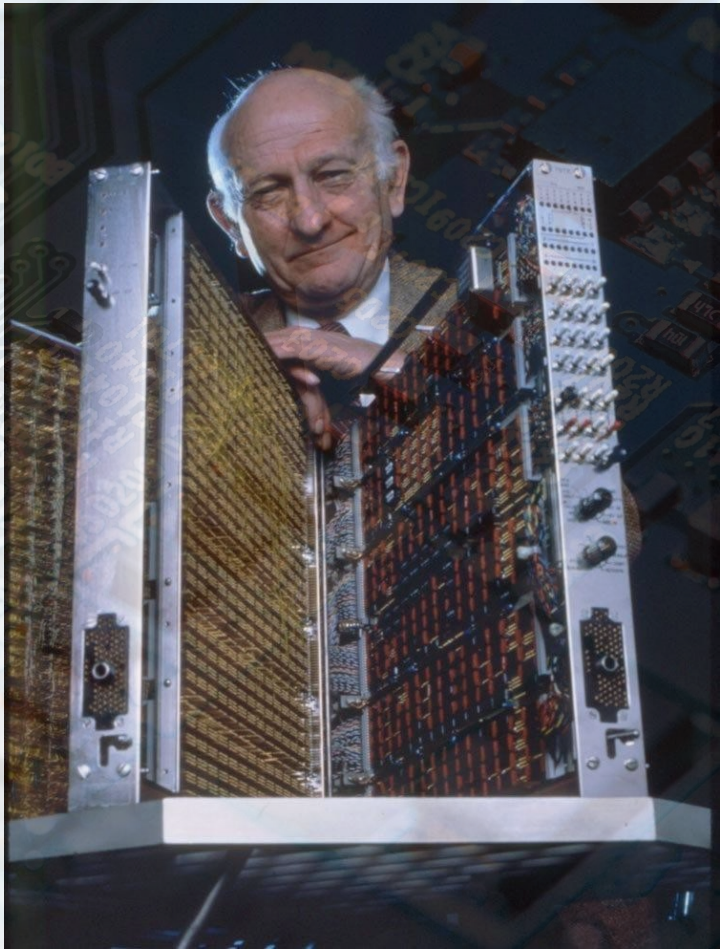
🕰️ Evolução Histórica das Arquiteturas de CPUs

⚙️ 1980: Surgimento do RISC

- Observação acadêmica: a maioria dos softwares usa apenas um subconjunto das instruções CISC.

- Pesquisadores como John Hennessy (Stanford) e David Patterson (Berkeley) propõem o conceito RISC:
  - Menos instruções, mais rápidas, mais eficientes.

- Exemplos: MIPS, SPARC, IBM POWER.



🕰️ Evolução Histórica das Arquiteturas de CPUs

📈 1990–2000: Consolidação e Mistura

- Intel domina mercado desktop com x86 (CISC com otimizações internas baseadas em RISC).
- ARM começa a crescer em dispositivos móveis e embarcados, priorizando eficiência energética.

🕰️ Evolução Histórica das Arquiteturas de CPUs

🚀 2010–Presente: A Era da Eficiência e Especialização

- Smartphones popularizam ARM (RISC), que se torna padrão em dispositivos móveis.
- Apple migra seus computadores de Intel (CISC) para seus próprios chips ARM (M1, M2, M3) com ganhos em eficiência energética e desempenho.
- Surgimento de arquiteturas abertas como **RISC-V**, que permite que empresas projetem seus próprios chips livremente.
- CPUs e GPUs se tornam mais integradas e especializadas para IA, machine learning e processamento paralelo.

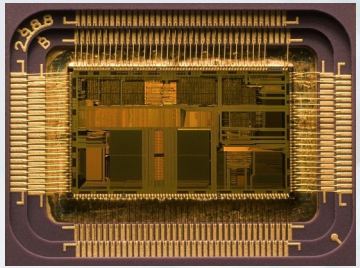
Década	Marco Principal	Arquitetura Dominante
1940–50	Computadores de válvula	Pré-Von Neumann
1960	Arquitetura de Von Neumann	Von Neumann
1970	Microprocessadores	CISC inicial
1980	Surgimento do conceito RISC	RISC vs CISC
1990	Consolidação dos PCs	Intel (x86 CISC)
2000	Crescimento ARM em celulares	ARM (RISC)
2010+	Transição Apple e IA, RISC-V surge	ARM (RISC) e híbridos

🧠 CISC (Complex Instruction Set Computer)

- CISC é um modelo de arquitetura de processadores que possui um **conjunto extenso e complexo de instruções**, algumas delas bastante sofisticadas.

- A ideia central do CISC surgiu nas décadas de 1970 e 1980, quando a memória era cara e limitada. Nesse contexto, projetar CPUs que pudessem fazer mais com menos linhas de código era uma vantagem.

- As instruções CISC são capazes de executar operações complexas — como acessar memória, realizar cálculos e manipular dados — em um **único comando de linguagem de máquina**.



🧠 CISC (Complex Instruction Set Computer)

💡 **Analogia:**

Imagine que você quer preparar uma refeição. No **modelo CISC**, você diria:
👉 **“Cozinhe arroz, frite ovo e prepare uma salada.”**
O processador executaria tudo com esse único comando.

No **modelo RISC**, você teria que dizer:
👉 **“Pegar arroz. Ligar fogão. Colocar água. Cozinhar arroz. Pegar ovo. Ligar frigideira. Fritar ovo...”**

🧠 CISC (Complex Instruction Set Computer)

🔍 Características Principais do CISC

- 🔡 **Grande conjunto de instruções:**
  - Centenas, às vezes mais de mil instruções diferentes.

- 🎯 **Instruções complexas:**
  - Uma única instrução pode realizar múltiplas operações (buscar da memória, executar cálculo e armazenar o resultado).

- 📦 **Vários modos de endereçamento:**
  - Flexibilidade na forma como os dados são acessados na memória.

🧠 CISC (Complex Instruction Set Computer)

🔍 Características Principais do CISC

- ⏳ **Execução variável:**
  - Instruções podem ter tempos de execução muito diferentes — algumas simples, outras muito demoradas.

- 🧠 **Maior dependência de hardware:**
  - O hardware precisa ser mais sofisticado e complexo para decodificar e executar instruções diversas.

- 💾 **Foco na redução de linhas de código:**
  - Menor quantidade de instruções no programa binário, já que cada instrução faz muito.

🧠 CISC (Complex Instruction Set Computer)

🖥️ Exemplos de Processadores CISC

- **Intel:** Linha **x86** (Pentium, Core i3, i5, i7, i9, Xeon).
- **AMD:** Ryzen, Threadripper, EPYC (todos também baseados em x86).
- Processadores antigos: **Motorola 68000**, muito usado em Macintosh, Amiga e Atari ST.
- Outros exemplos históricos: **VAX (DEC)**, muito usado em mainframes e servidores.

🧠 CISC (Complex Instruction Set Computer)

✅ Vantagens do CISC

- ➕ **Menos código de máquina:**
  - Como uma instrução faz mais, o programa ocupa menos espaço na memória.
- ➕ **Facilidade de programação (antigamente):**
  - Menor necessidade de escrever várias instruções para uma tarefa.
- ➕ **Maior compatibilidade legada:**
  - Arquiteturas como x86 carregam décadas de compatibilidade com sistemas operacionais e softwares antigos.
- ➕ **Bom desempenho em tarefas com muitos acessos à memória:**
  - Operações complexas otimizadas diretamente no hardware.

🧠 CISC (Complex Instruction Set Computer)

❌ Desvantagens do CISC

- ➖ **Hardware complexo:**
  - Circuitos maiores e mais difíceis de projetar, testear e otimizar.
- ➖ **Maior consumo de energia:**
  - Decodificar e executar instruções complexas exige mais energia, especialmente em chips móveis.
- ➖ **Menor desempenho em operações simples e repetitivas:**
  - Instruções complexas podem ser mais lentas que várias instruções simples otimizadas (como no RISC).
- ➖ **Dificuldade em paralelizar:**
  - A variabilidade nas instruções complica a implementação eficiente de pipelines e execução paralela.

🧠 CISC (Complex Instruction Set Computer)

Característica	CISC
🔤 Conjunto de Instruções	Extenso, com instruções complexas
🎯 Complexidade	Alta (tanto no hardware quanto nas instruções)
🧠 Hardware	Mais complexo, mais consumo de energia
📜 Código	Mais compacto, menos linhas
🖥️ Exemplos	Intel x86, AMD Ryzen, Motorola 68000
✅ Vantagens	Menos código, compatibilidade, tarefas complexas
❌ Desvantagens	Hardware complexo, mais energia, difícil paralelizar

🧠 CISC (Complex Instruction Set Computer)

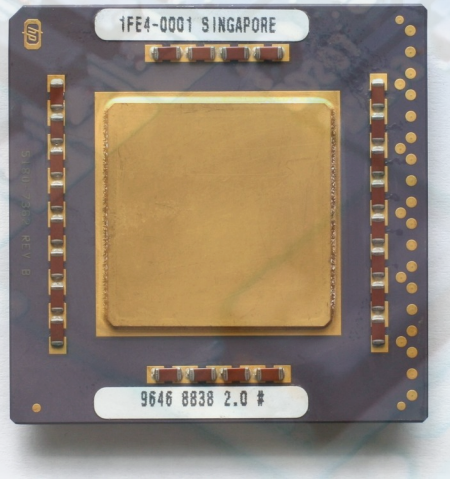
🚩 Curiosidade Importante

Embora os processadores Intel e AMD sejam arquiteturalmente CISC (**x86**), **internamente eles usam microarquiteturas que quebram as instruções complexas em microinstruções mais simples, semelhantes a RISC.** Isso é uma forma de combinar os benefícios dos dois mundos.

🧠 RISC — Reduced Instruction Set Computer

RISC é uma arquitetura de processadores baseada na simplificação do conjunto de instruções. Ao invés de ter instruções complexas e multifuncionais (como na arquitetura CISC), o RISC adota um conjunto pequeno, simples e otimizado de instruções, todas executáveis em poucos ciclos de clock — geralmente **um único ciclo**.

👉 A ideia central é que, ao reduzir a complexidade das instruções, o processador pode ser mais rápido, eficiente e escalável.



🧠 RISC — Reduced Instruction Set Computer

🚀 Características Principais do RISC

Característica	Descrição
✅ **Conjunto reduzido de instruções**	Instruções simples, com operações básicas, como soma, subtração, deslocamento, carregamento e armazenamento.
✅ **Instruções de tamanho fixo**	Facilita o pipeline e a decodificação, pois cada instrução tem o mesmo tamanho (ex.: 32 bits).
✅ **Pipeline eficiente**	A simplicidade permite dividir a execução em etapas (fetch, decode, execute...), aumentando o desempenho.
✅ **Foco em operações Load/Store**	Apenas instruções específicas acessam a memória (LOAD e STORE). Operações aritméticas ocorrem nos registradores.

🧠 RISC — Reduced Instruction Set Computer

🚀 Características Principais do RISC

Característica	Descrição
✅ **Execução rápida**	A maioria das instruções é executada em **um ciclo de clock**.
✅ **Mais registradores**	Uso intensivo de registradores internos, reduzindo o acesso à memória RAM (mais lenta).
✅ **Simplicidade no hardware**	O design do processador é mais simples, o que facilita melhorias em desempenho e eficiência energética.

🧠 RISC — Reduced Instruction Set Computer

🏗️ Exemplos de Processadores com Arquitetura RISC

Familia/Processador	Contexto
ARM (Cortex, Snapdragon, Apple M1/M2/M3)	Usados em smartphones, tablets, Apple Macs modernos e embarcados.
RISC-V	Arquitetura aberta, em expansão para IOT, HPC e PCs.
MIPS	Antigamente usado em roteadores, videogames (PlayStation 1 e 2).
SPARC (Sun/Oracle)	Usado em servidores de missão crítica (menos comum atualmente).
PowerPC (IBM, antigos Macs)	Usado em servidores, supercomputadores e consoles como Xbox 360 e PS3.

🧠 RISC — Reduced Instruction Set Computer

✅ Vantagens do RISC

Vantagem	Impacto
🔋 **Maior eficiência energética**	Perfeito para dispositivos móveis e embarcados.
⚡ **Maior velocidade por instrução**	Instruções simples e rápidas favorecem pipelines profundos.
🧠 **Design mais simples**	Menor custo de desenvolvimento, mais espaço para núcleos, caches e paralelismo.
🔄 **Facilidade de escalabilidade**	Permite criar processadores com muitos núcleos, otimizando desempenho para workloads modernos.
🔧 **Fácil implementação de paralelismo**	Arquitetura limpa favorece execução simultânea de várias instruções.

🧠 RISC — Reduced Instruction Set Computer

❌ Desvantagens do RISC

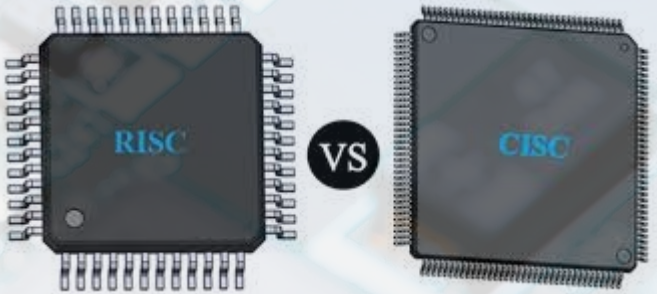
Desvantagem	Consequência
🧾 **Programas maiores (em instruções)**	Como não há instruções complexas, são necessárias mais instruções para realizar tarefas complexas.
📜 **Código de máquina mais extenso**	Pode exigir mais acesso à memória para buscar instruções.
🏗️ **Compiladores mais complexos**	O compilador precisa otimizar muito bem a sequência de instruções para maximizar desempenho.
🔧 **Nem sempre melhor em workloads específicos**	Alguns processos muito específicos podem se beneficiar de instruções complexas, presentes no CISC.

🔥 Comparativo Prático entre RISC e CISC

📜 Análise de Instruções

🔸 CISC (Complex Instruction Set Computer)

- **Características:**
  - Grande conjunto de instruções (centenas até milhares).
  - Cada instrução pode realizar várias operações (ex.: acessar memória + processar + armazenar).
  - Instruções de tamanho variável (em bytes).
  - Muitas instruções raramente usadas.



🔥 Comparativo Prático entre RISC e CISC

Exemplo Prático (CISC - Intel x86):

```
MUL AX, BX
```

→ Multiplica os conteúdos dos registradores AX e BX e armazena no registrador AX.

✅ Executa carregamento, multiplicação e armazenamento em uma única instrução.

🔥 Comparativo Prático entre RISC e CISC

RISC (Reduced Instruction Set Computer)

- **Características:**
  - Pequeno conjunto de instruções (geralmente menos de 100).
  - Cada instrução executa uma única operação simples.
  - Instruções de tamanho fixo (geralmente 4 bytes).
  - Forte dependência dos registradores (acesso à memória é separado da execução lógica).

🔥 Comparativo Prático entre RISC e CISC

Exemplo Prático (RISC - ARM):

```
LOAD R1, A    ; Carrega valor de A para registrador R1
LOAD R2, B    ; Carrega valor de B para registrador R2
MUL R3, R1, R2; Multiplica R1 e R2, salva em R3
STORE C, R3   ; Armazena o valor de R3 em C
```

✅ Mais instruções, mas cada uma é muito rápida e simples.

🔥 Comparativo Prático entre RISC e CISC

Critério	🏛️ CISC	🚀 RISC
Conjunto de Instruções	Grande e complexo	Reduzido e simples
Tamanho das Instruções	Variável (1 a 15 bytes)	Fixo (geralmente 4 bytes)
Acesso à Memória	Direto nas instruções	Somente via instruções específicas
Pipeline	Difícil de implementar (instruções variáveis)	Fácil (instruções simples e regulares)
Número de Ciclos por Instrução	Varia (múltiplos ciclos)	1 ciclo na maioria das instruções
Complexidade do Hardware	Alta (decodificadores complexos)	Menor (decodificação simples)
Código Gerado	Mais compacto (menos linhas)	Mais longo (mais instruções)

🔥 Comparativo Prático entre RISC e CISC

⚙️ Comparação de Desempenho, Consumo e Complexidade

🔥 Desempenho

- **CISC:**
  - Menos instruções, mas cada uma pode demorar vários ciclos.
  - Melhor quando largura de banda da memória é limitada.
  - Bom para aplicações legadas (compatibilidade com software antigo).

- **RISC:**
  - Mais instruções, mas extremamente rápidas.
  - Otimizado para pipelines profundos, execução paralela e escalabilidade.
  - Ideal para arquiteturas modernas com muitos núcleos.

🔥 Comparativo Prático entre RISC e CISC

⚡ Consumo de Energia

- **CISC:**
  - Decodificadores e unidades de controle complexas consomem mais energia.
  - Ideal para desktops e servidores com menos restrição energética.

- **RISC:**
  - Arquitetura mais eficiente energeticamente.
  - Menor dissipação de calor.
  - Por isso domina dispositivos móveis, embarcados e notebooks ultrafinos (ARM, Apple M1/M2, Snapdragon).

🔥 Comparativo Prático entre RISC e CISC

🧠 Complexidade do Hardware

- **CISC:**
  - Hardware mais complexo (decodificação de instruções complexas).
  - Arquiteturas x86 possuem microtradutores internos que quebram instruções complexas em micro-operações semelhantes às de RISC.

- **RISC:**
  - Hardware mais simples e escalável.
  - Permite mais espaço para otimizações, como mais núcleos, caches maiores ou unidades especializadas (GPU, IA, etc.).

🔥 Comparativo Prático entre RISC e CISC

🔍 Cenário Atual: Mistura dos Dois Mundos

- **Processadores x86 (Intel/AMD):**
  - Embora sejam CISC, internamente convertem as instruções em micro-operações RISC.
  - Mistura características de ambos para melhorar desempenho.

- **Processadores ARM (Apple, Qualcomm, Samsung):**
  - Mantêm filosofia RISC pura.
  - Altamente eficientes, baixo consumo e excelente desempenho por watt.

🔥 Comparativo Prático entre RISC e CISC

📊 Benchmark de Mercado (Exemplos)

- 🔹 **Apple M1 (RISC ARM)** vs **Intel i7 (CISC x86):**
  → Apple entrega desempenho similar ou superior com consumo muito menor.

- 🔹 **Snapdragon (RISC ARM)** vs **Intel Atom:**
  → Snapdragon domina em dispositivos móveis graças à eficiência energética.









🔥 Comparativo Prático entre RISC e CISC

💡 Resumo da Análise

- ✅ **RISC:** Eficiência, simplicidade, desempenho escalável, baixo consumo.
- ✅ **CISC:** Compatibilidade com legado, código mais compacto, bom desempenho geral.

🌐 Por que o mundo está migrando para RISC?

- Celulares, tablets, IoT e até servidores estão adotando ARM (RISC) pela eficiência energética.

- A Apple migrou seus Macs de Intel (CISC) para ARM (RISC) com os chips Apple Silicon (M1, M2, M3) por ganhos significativos de desempenho e bateria.

🔥 Cenário Atual: RISC x CISC na Prática

📱 Por que ARM (RISC) domina dispositivos móveis?

- **Eficiência energética:**
  ARM é projetado para ser extremamente eficiente em consumo de energia. Dispositivos móveis, como smartphones e tablets, dependem de bateria — então, processadores que consomem menos energia são essenciais.

- **Simplicidade do hardware:**
  A arquitetura RISC permite processadores com menos transistores dedicados a lógica complexa, favorecendo menor geração de calor, menor custo de fabricação e mais espaço para outros recursos (como GPU, IA, modem 5G).

🔥 Cenário Atual: RISC x CISC na Prática

📱 Por que ARM (RISC) domina dispositivos móveis?

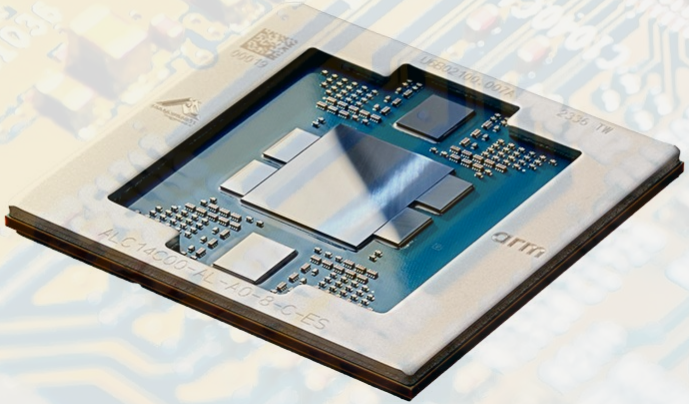
- **Desempenho adequado:**
  Embora usem instruções mais simples, os chips ARM compensam isso com clock elevado, paralelismo massivo e mais núcleos de execução, alcançando alto desempenho.

- **Evolução do Ecossistema:**
  ARM licenciou sua arquitetura para centenas de fabricantes (Qualcomm, Samsung, MediaTek, Apple, etc.), promovendo diversidade de designs e inovações específicas.

🔥 Cenário Atual: RISC x CISC na Prática

📱 Por que ARM (RISC) domina dispositivos móveis?

- **Exemplos:**
  - Chips Snapdragon (Android)
  - Apple A17 (iPhones)
  - Raspberry Pi (educação e IoT)
  - AWS Graviton (data centers)





🔥 Cenário Atual: RISC x CISC na Prática

- Até 2020, Macs usavam Intel x86 (arquitetura CISC).
- Apple lançou o **M1 (ARM)** como parte de sua própria linha de chips, abandonando Intel.

🔥 Cenário Atual: RISC x CISC na Prática

Por que Apple fez isso?

- **Controle total do hardware + software:** Otimização mais agressiva, já que a Apple controla o chip, o sistema operacional (macOS), e os aplicativos.

- **Eficiência energética:** MacBooks passaram a ter autonomia de bateria muito superior — de 10h para até 20h em alguns modelos.

- **Desempenho absurdo:** O M1 superou muitos Intel i7 e i9 em tarefas single-core e multi-core, além de integrar GPU, memória unificada e aceleradores de IA no mesmo chip.

🔥 Cenário Atual: RISC x CISC na Prática

🔥 Por que Apple fez isso?

- **Menor aquecimento:** Dispensa cooler em alguns modelos, com operação silenciosa.

🔧 M2, M3 e além:

- Seguem o mesmo caminho, aumentando núcleos, frequência, desempenho gráfico e integração de IA, mantendo baixíssimo consumo.

📌 Impacto na indústria:

- Forçou Intel e AMD a repensarem eficiência energética.
- Popularizou o conceito de SoC (System on Chip) em computadores, não só em celulares.

🔥 Cenário Atual: RISC x CISC na Prática

🖥️ Processadores Intel/AMD — Otimizações modernas do CISC

- Embora baseados na arquitetura **x86 (CISC)**, as CPUs modernas usam uma estratégia híbrida:
    - **Decodificadores RISC internos:** As instruções complexas x86 são traduzidas em micro-operações (μops) simples, muito semelhantes ao estilo RISC.
    - Isso permite que, internamente, o processador opere de forma mais eficiente, mas mantenha compatibilidade com décadas de software.

🔥 Cenário Atual: RISC x CISC na Prática

🚀 Inovações recentes:

- **Arquitetura híbrida:**
  Intel Alder Lake e AMD Ryzen adotam núcleos de dois tipos:
  - **P-Cores (Performance):** Alta potência e desempenho.
  - **E-Cores (Efficiency):** Menor consumo, otimizados para tarefas em background.
- **Aumento de paralelismo:**
  Muitos núcleos, múltiplas threads, instruções vetoriais (AVX-512, etc.).

🔥 Cenário Atual: RISC x CISC na Prática

🚀 Inovações recentes:

- **Foco em IA:**
  Novos chips incluem aceleradores dedicados para IA, ML e segurança (Intel NPU, AMD AI Engine).

- **Energia ainda é desafio:**
  Embora muito otimizados, os chips x86 ainda consomem mais energia que ARM, especialmente em dispositivos móveis e laptops.

🤔 Reflexão Final: Existe um Vencedor?

🏆

Não. A escolha depende do contexto e das necessidades.

📌 **Quando escolher RISC (ex.: ARM, RISC-V):**

- Dispositivos móveis, IoT, wearables, carros, sistemas embarcados.
- Quando autonomia, aquecimento e eficiência são mais importantes que compatibilidade de software legado.
- Data centers buscando eficiência energética (Ex.: AWS Graviton).

🤔 Reflexão Final: Existe um Vencedor?

📌 **Quando escolher CISC (x86 — Intel/AMD):**

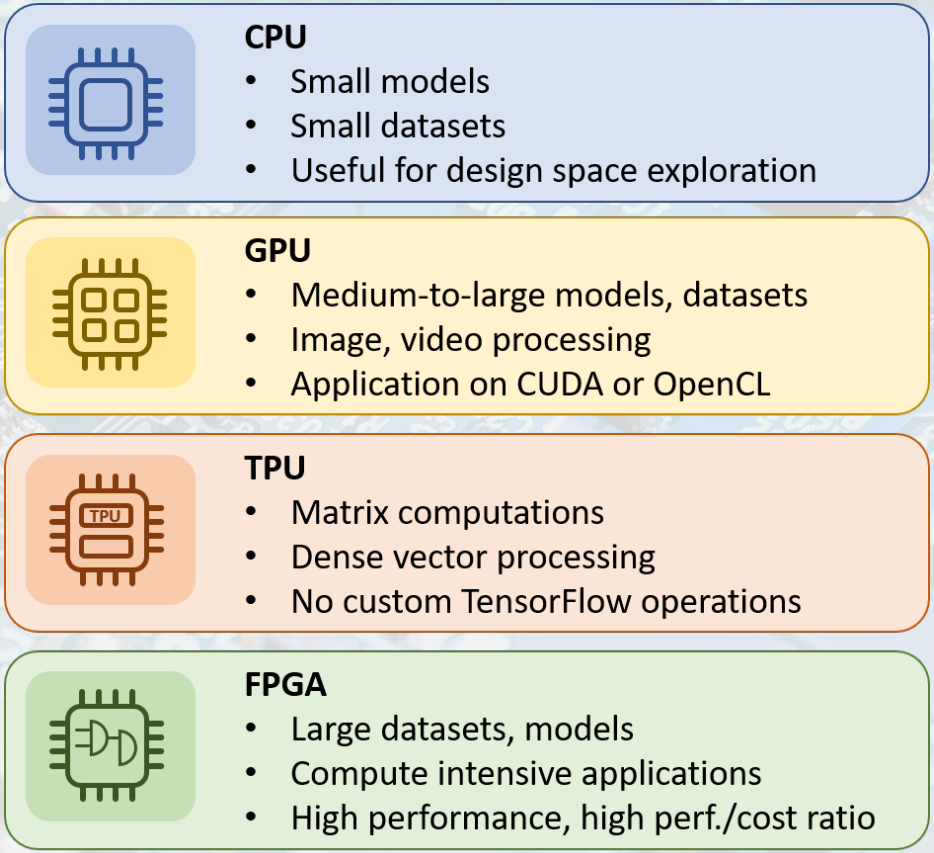
- Computadores que rodam software legado (Windows, programas otimizados para x86).
- Ambientes corporativos e servidores que dependem de ferramentas muito antigas ou com pouca adaptação a ARM.
- Gamers e usuários que precisam de GPUs dedicadas e ainda aproveitam as vantagens dos PCs tradicionais.

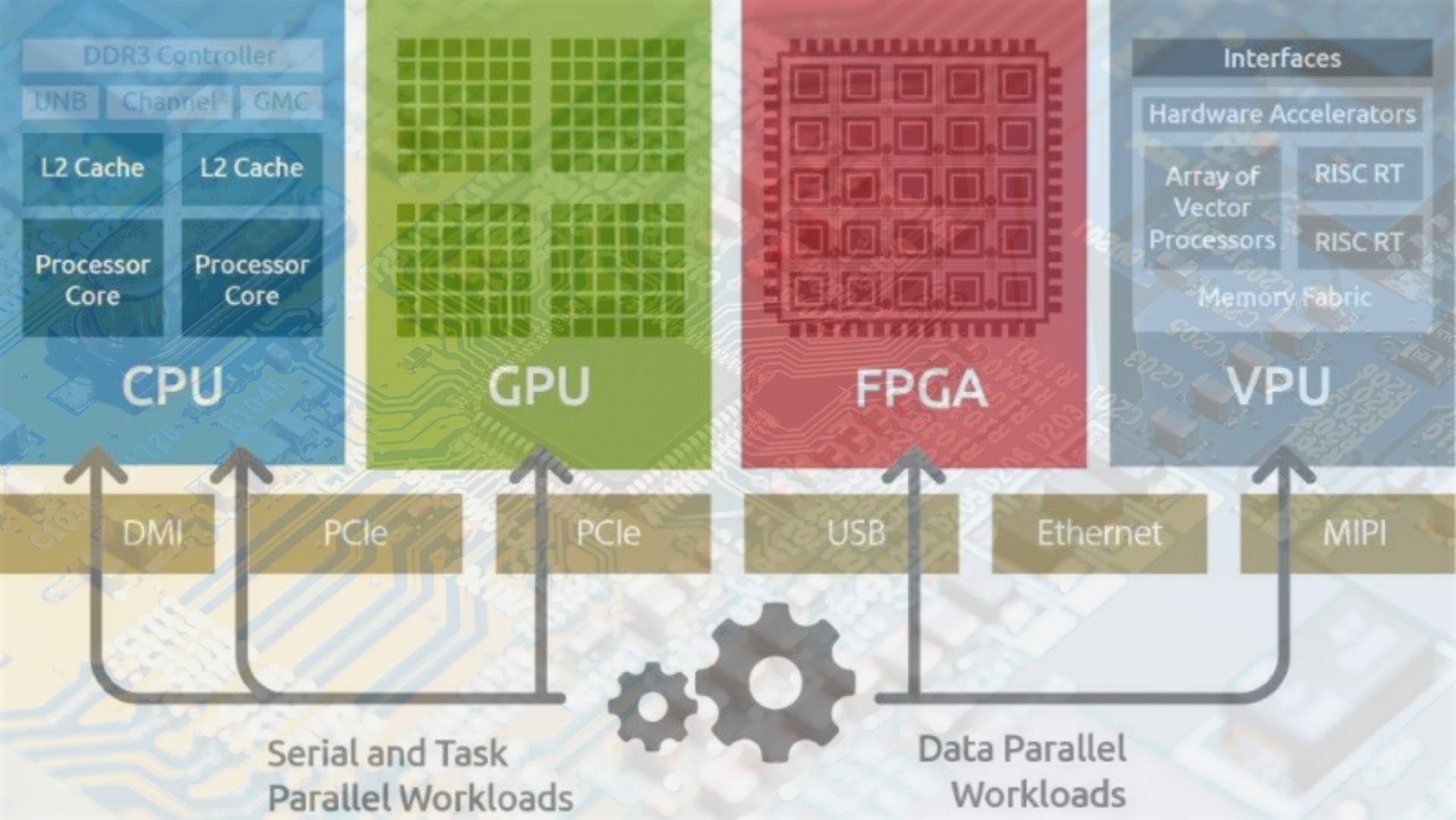
Tendência futura

- O futuro não é RISC *ou* CISC, mas sim **sistemas heterogêneos**, onde há processadores especializados convivendo (CPU, GPU, NPU, FPGA).

- IA, computação em nuvem e edge computing estão pressionando por mais eficiência, o que favorece ARM e RISC-V.

- x86 continua forte, mas enfrenta pressão crescente fora do ambiente desktop.




DDR3 Controller
UNB
Channel
GMC
L2 Cache
L2 Cache
Processor Core
Processor Core
CPU
GPU
FPGA
Interfaces
Hardware Accelerators
Array of Vector Processors
RISC RT
RISC RT
Memory Fabric
VPU
DMI
PCIe
PCIe
USB
Ethernet
MIPI
Serial and Task Parallel Workloads
Data Parallel Workloads

“Na computação, não há soluções únicas. Há apenas escolhas que dependem dos problemas que queremos resolver.”


Processador
RISC
CISC
intel
AMD

Eclesiastes 12:1 – "Lembra-te do teu Criador nos dias da tua mocidade."

A juventude é o melhor tempo para firmar valores que guiarão sua vida.

UniFAP
CENTRO UNIVERSITÁRIO PARAÍSO