

# Aula 1

## Organização de Computadores

Profa. Débora Matos

# Objetivos a disciplina

- Caracterização de desempenho das arquiteturas. Organização de processadores: bloco operacional e bloco de controle.
- Organização serial e paralela (pipeline) da CPU.
- Estudo de sistema de memória (hierarquia da memória, memória cache e memória virtual).
- Métodos para aumento de desempenho: organização de pipelines, máquinas super-escalares.
- Estudos de caso de processadores contemporâneos. Ferramentas para análise e projeto de organizações.

# Cronograma das aulas

Data	Assunto
31/07	Aula 1 – Apresentação da disciplina. Plano de Ensino. Revisão de conceitos. Introdução a microprocessadores: revisão dos conceitos de organização.
07/08	Aula 2 – Avaliação de desempenho. Lei de Amdahl. Análise de Potência. Como calcular o CPI, o MIPS e o tempo de CPI e análise de potência.
14/08	Aula 3 – Entendendo as instruções do MIPS e as unidades funcionais requeridas para execução das mesmas.
21/08	Aula 4 – Caminho de dados do MIPS, unidades funcionais, implementação de ciclo único e multiciclo.
28/08	Aula 5 – Implementação do MIPS com pipeline. Entendendo a melhora da performance com a técnica de pipeline.
04/09	Aula 6 – Tipos de conflitos de pipeline: Hazards de dados, hazards de controle. Soluções para os conflitos.
11/09	Aula 7 – Hazards de controle. Implementação de previsão dinâmica de desvios. Implementação de stalls, forwarding e exceções.

18/09	Aula 8 – Continuação sobre conflitos. Definição do trabalho a ser desenvolvido em VHDL.
25/09	PROVA 1
02/10	Aula 9 – Correção da prova. Hierarquia de memória, tipos de memória, tecnologia utilizada, capacidade de armazenamento, velocidade, memória cache.
09/10	Aula 10 – Hierarquia de memória, memória cache, tipos de mapeamentos da memória principal para a cache, técnicas de substituição de blocos, exercícios com memória cache.
16/10	Aula 11 –Correção dos exercícios sobre hierarquia de memória, cálculo do tamanho da cache, cálculo de CPI.
23/10	Aula 12 – Memória Virtual
30/10	Aula 13 – Processadores Paralelos, superpipeline, GPU
06/11	Aula 14: Processadores Superescalares e VLIW
13/11	PROVA 2
20/11	Entrega das notas das provas. Resolução da prova. Entrega por parte dos alunos da parte 1 do trabalho, orientação quanto a parte 2 do trabalho.
27/11	Apresentação da parte 2 do trabalho. Entrega dos relatórios.
04/12	RECUPERAÇÃO

# Avaliações

- PROVA 1 (P1)
- PROVA 2 (P2)
- Trabalhos (T) = Neander (40%) + MIPS-based (60%)
- Nota final =  $(P1 + P2 + T)/3$
- No caso do aluno não atingir a média mínima, será possível a realização da recuperação para substituir a nota de **uma das provas**;
- Na recuperação, entra todo o conteúdo visto durante a disciplina;
- Para poder realizar a prova de recuperação, a média total do aluno deverá ser **maior ou igual a 4**.

# Bibliografia

## Bibliografia Básica

- PATTERSON, David A; HENNESSY, John L. Organização e Projeto de Computadores: a Interface Hardware/Software. 3. Ed. Rio de Janeiro: Elsevier, 2005. xvii, 484 p.
- STALLINGS, W. Arquitetura e organização de computadores. 5° ed., São Paulo, 2010.

## Bibliografia Complementar

- TANENBAUM, Andrew, Organização estruturada de computadores, 4ª ed.
- Hennessy, John. L.; Patterson, David A. Arquitetura de Computadores: Uma abordagem quantitativa, 4ª Edição, 2007. Campus.

# Revisão de alguns conceitos

# Arquitetura X Organização

- **Arquitetura:**

Refere-se aos atributos de um sistema que são visíveis para o programador, ou seja, aos atributos que tem impacto direto sobre a execução lógica de um programa.

Exemplos:

- ✓ conjunto de instruções,
- ✓ número de bits utilizados para representar os tipos de dados,
- ✓ mecanismos de entrada e saída
- ✓ as técnicas de endereçamento à memória.



# Arquitetura X Organização

- **Organização:** refere-se as unidades operacionais e suas interconexões que implementam as especificações da arquitetura.

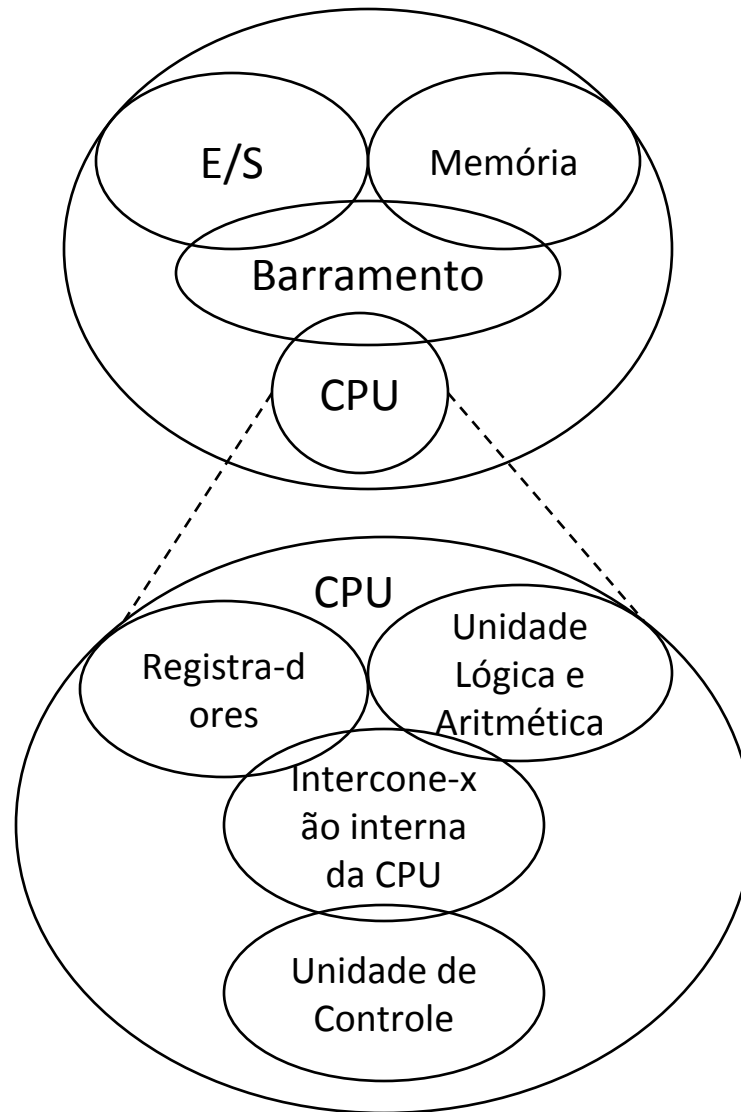
Exemplos:

- ✓ Detalhes de hardware transparentes aos programador: sinais de controle, periféricos, tecnologia de memória utilizada, número de ULAS, interconexões entre os componentes, etc.

# Arquitetura X Organização

- Definir se um computador deve ou não ter uma instrução de multiplicação é uma decisão do projeto de sua **arquitetura** ou **organização**?
- Definir se essa instrução será implementada por uma unidade especial de multiplicação ou por um mecanismo que utiliza repetidamente sua unidade de soma é uma decisão do projeto da sua **arquitetura** ou **organização**?

# Componentes de microcomputador



# **Microprocessadores: um breve histórico**

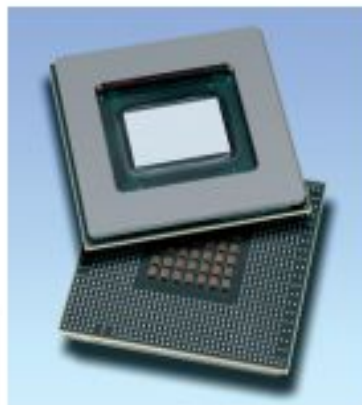
**Quais são alguns exemplos de fabricantes de processadores:**

# Microprocessadores: um breve histórico

Microprocessadores vem sendo desenvolvidos e produzidos por um grande número de fabricantes:

- Intel, AMD, Motorola, Texas, IBM e várias outros
- Várias empresas já foram participantes ativos no mercado de microprocessadores (como Zilog) e desapareceram, perderam importância ou foram adquiridas pelos concorrentes

microprocessador ARM



IBM Cell

Motorola



[www.cpu-world.com](http://www.cpu-world.com)

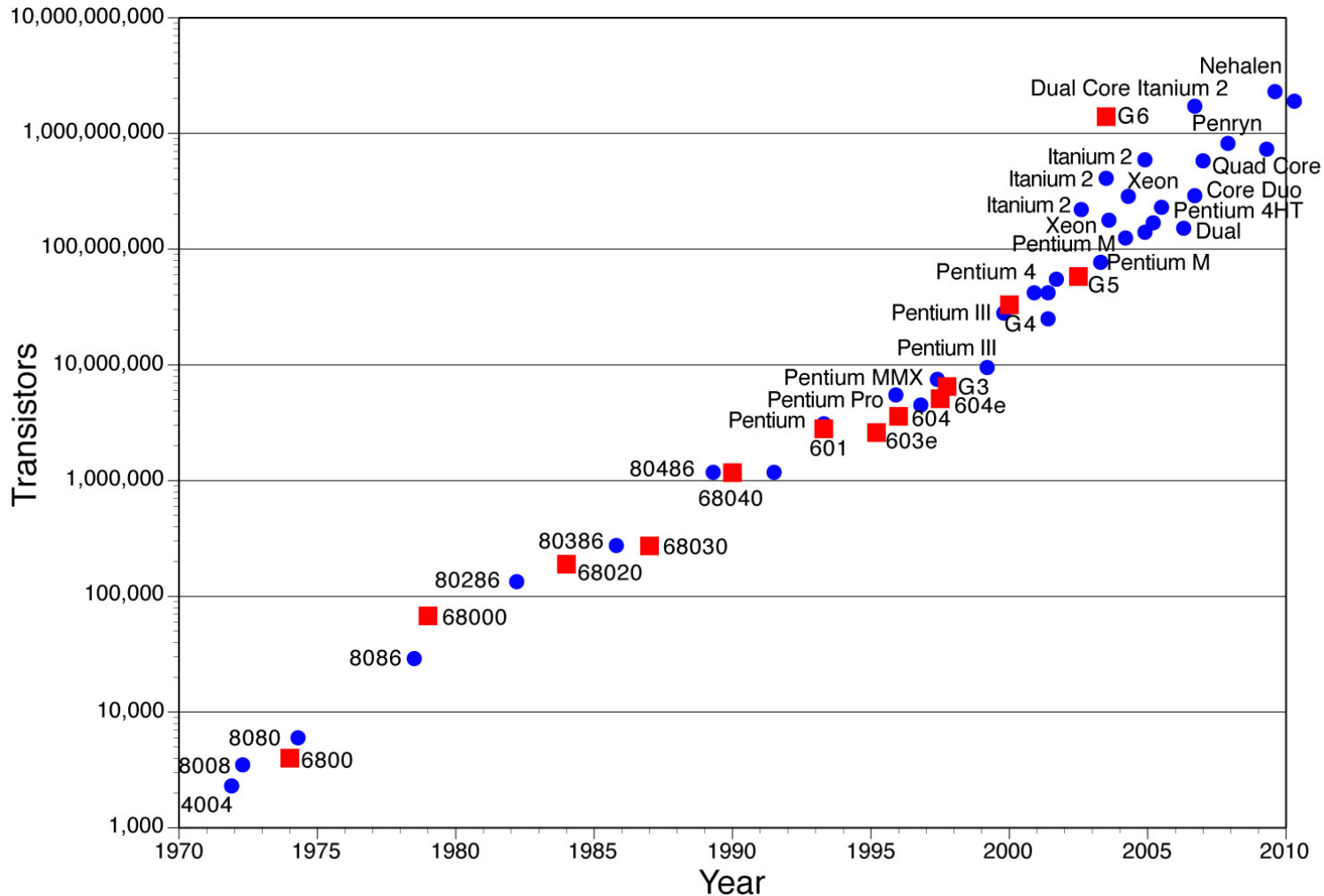


Microprocessadores de diferentes fabricantes e mesmo microprocessadores do mesmo fabricante mas de diferentes modelos não são intercambiáveis



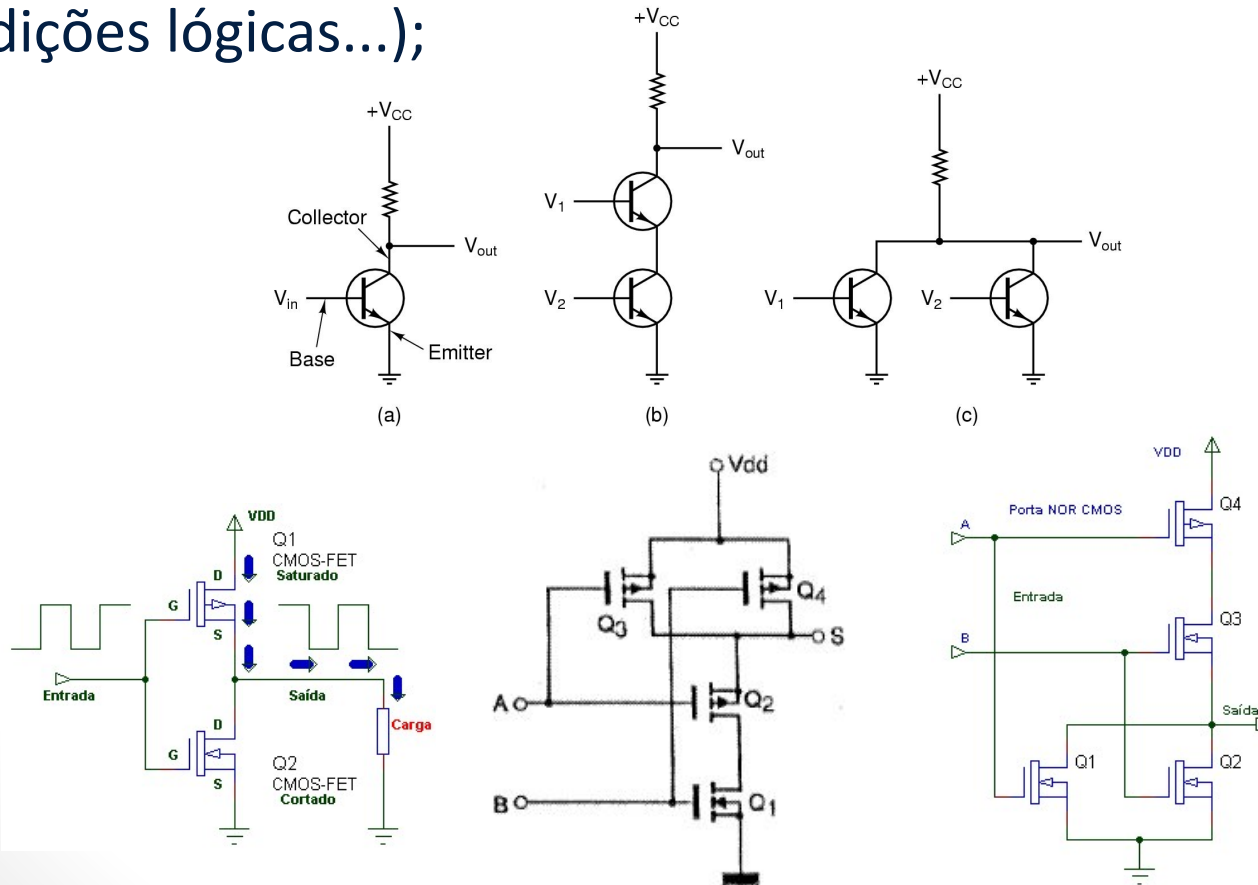
# Lei de Moore

A lei de Moore diz que a quantidade de transistores em um circuito integrado dobra a cada 18 meses.



# Lei de Moore

- Um processador é um circuito integrado composto por milhões de transistores (ex. core i7 possui o número de transistores próximo a 1 bilhão);
- Estes transistores são agrupados para desempenharem determinadas funções (cálculos aritméticos, registro, condições lógicas...);

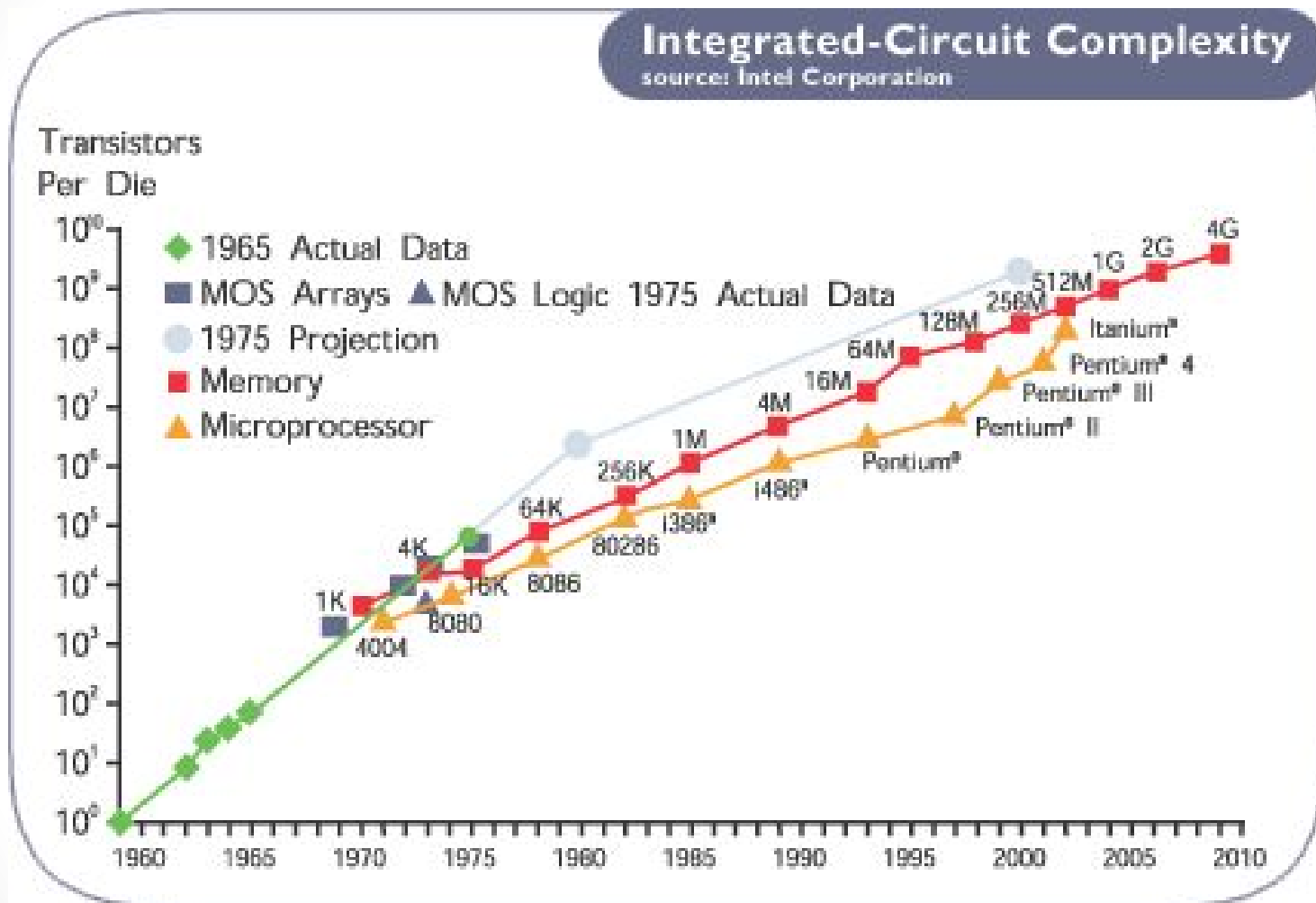


# Lei de Moore

- Sustentar a taxa de progresso definida pela lei de Moore durante mais de 40 anos exigiu incríveis inovações tecnológicas nas técnicas de fabricação.



# Memória DRAM




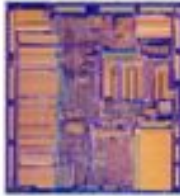

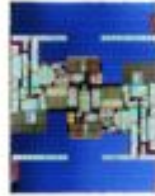


Capacidade por chip de DRAM ao longo do tempo

# Microprocessadores: um breve histórico

Tem evoluído, tornando-se menor e mais baratos

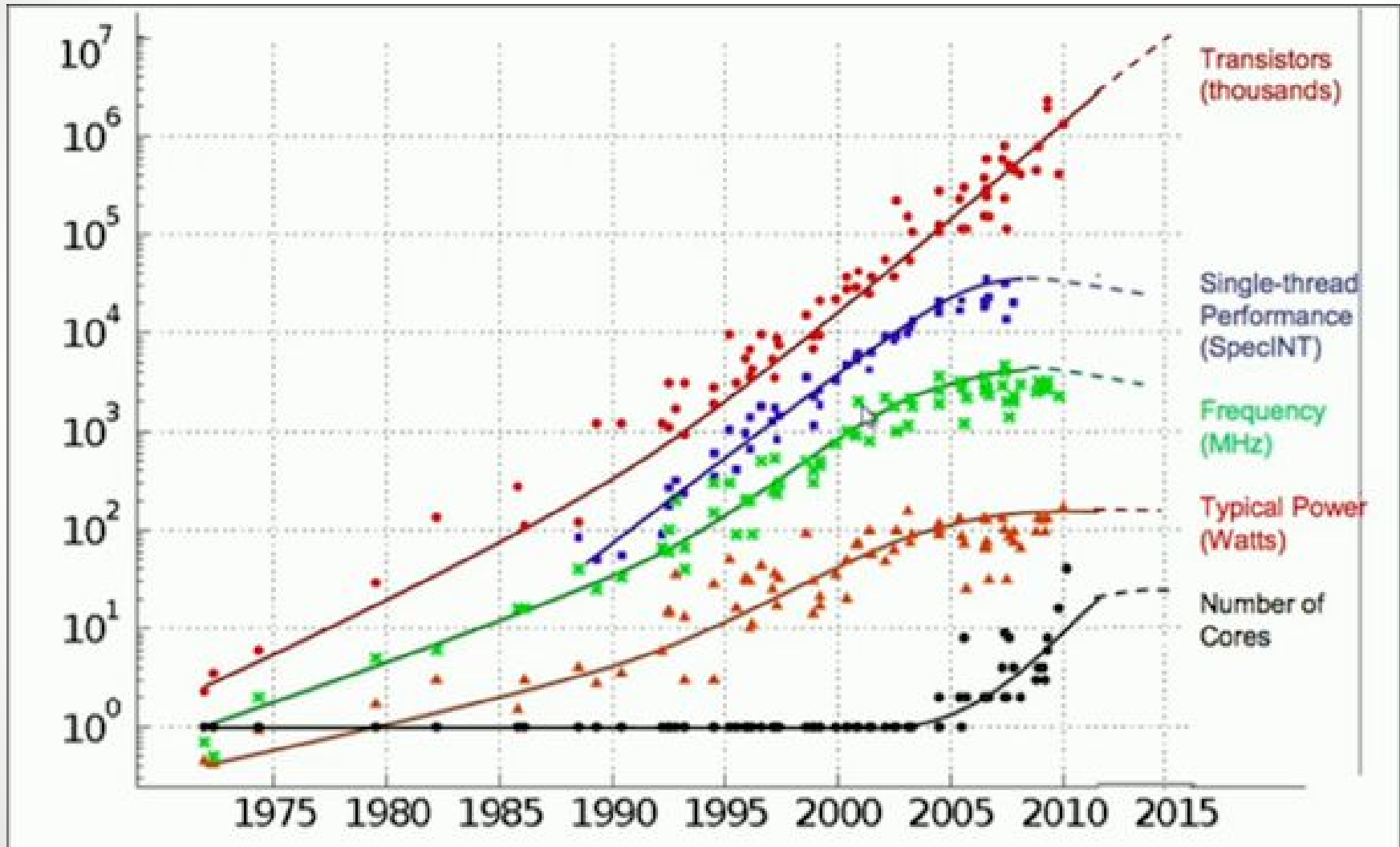
(<http://www.computerhistory.org/semiconductor/>)

1950s	1960s	1970s	1980s	1990s	2000s
Silicon Transistor	TTL Quad Gate	8-bit Microprocessor	32-bit Microprocessor	32-bit Microprocessor	64-bit Microprocessor
					
<b>1</b> Transistor	<b>16</b> Transistors	<b>4500</b> Transistors	<b>275,000</b> Transistors	<b>3,100,000</b> Transistors	<b>592,000,000</b> Transistors

microprocessadores de 8 a 64 bits

8 a 64 indica o número de bits da palavra de dados. Quando maior o número de bits, maior a capacidade de representação de inteiros.

# Microprocessadores ao longo dos anos

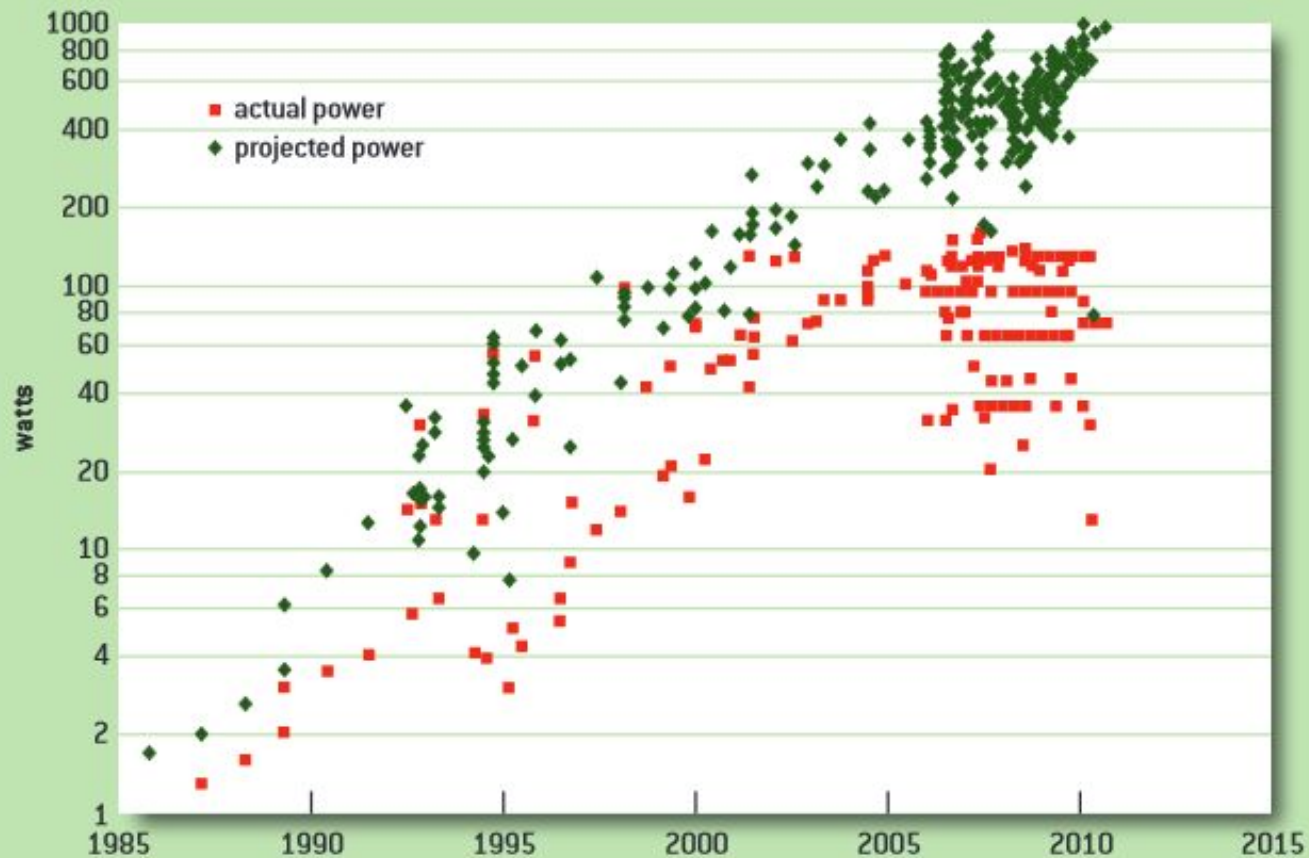


Fonte: Data Processing in ExaScale-Class Computer Systems (C. Moore, April 2011).

# Microprocessadores ao longo dos anos

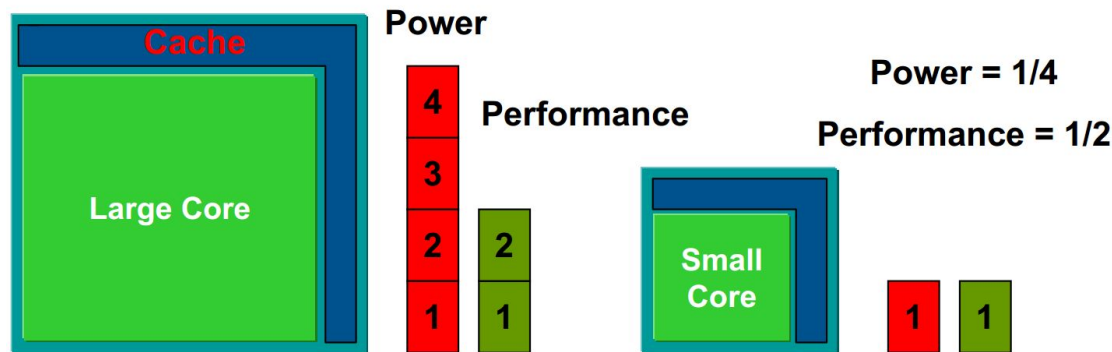
FIGURE 9

How Power Should Have Scaled



# Microprocessadores ao longo dos anos

- Solução: substituição do aumento da frequência pelo aumento do número de cores.



Fonte: Keynote presentation (L. Benini, RSP 2010).

- MPSoC: sistema composto por múltiplos cores, hierarquia de memória e componentes de entrada/saída (I/O) em um CI (Circuito Integrado).

# Microprocessadores ao longo dos anos

DRAM		
Year	Size	Cycle Time
1980	64 Kb	250 ns
1983	256 Kb	220 ns
1986	1 Mb	190 ns
1989	4 Mb	165 ns
1992	16 Mb	145 ns
1995	64 Mb	120 ns

**1000:1!** (Size growth from 1980 to 1995)

**2:1!** (Cycle Time reduction from 1980 to 1995)

Fonte: Presentation CS252 (Prof. Kurt Keutzer, 2000)

# Quais são alguns dos avanços que temos visto na arquitetura dos computadores?

- Aumento da capacidade de armazenamento
- Aumento da frequência de operação do processador
- Aumento do número de núcleos de processamento
- Aumento do paralelismo das organizações de computadores
- Processadores mais tolerante à falhas

# Qual o objetivo desses avanços?

- Acelerar as aplicações, aplicativos, softwares;
- Prover maior capacidade de armazenamento;
- Possibilitar o uso de mais recursos;
- Reduzir o consumo de energia;
- Aumentar a eficiência energética;
- Reduzir o número de falhas;



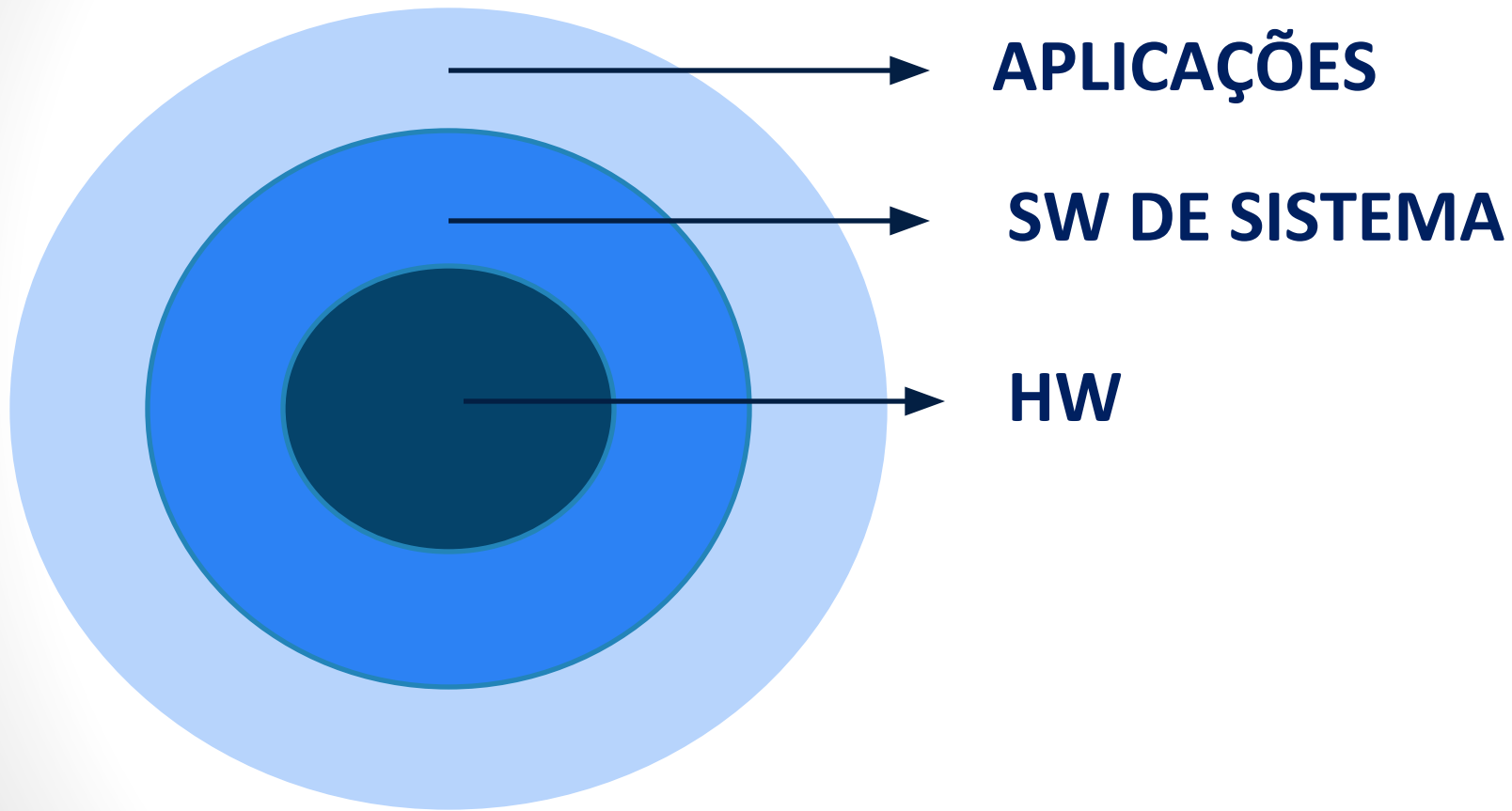
# O que afeta o desempenho de um computador?

- Requisitos de HW e SW
  - ✓ Algoritmos dos programas
  - ✓ Linguagens de programação
  - ✓ Compiladores
  - ✓ Sistema operacional
  - ✓ Outros softwares de sistemas
  - ✓ O projeto do processador
  - ✓ Sistema de entrada e saída, dispositivos

# HW x SW

- O hardware de um computador só pode executar instruções de baixo nível, operações extremamente simples.
- De uma aplicação complexa até as instruções simples, envolve várias camadas de SW que interpretam ou traduzem operações de alto nível em instruções simples de computador.

# HW x SW



# Exemplos de SW de sistema:

- ✓ Sistema Operacional
- ✓ Compilador
- ✓ Montadores
- ✓ Interpretador

# O que faz o Sistema Operacional?

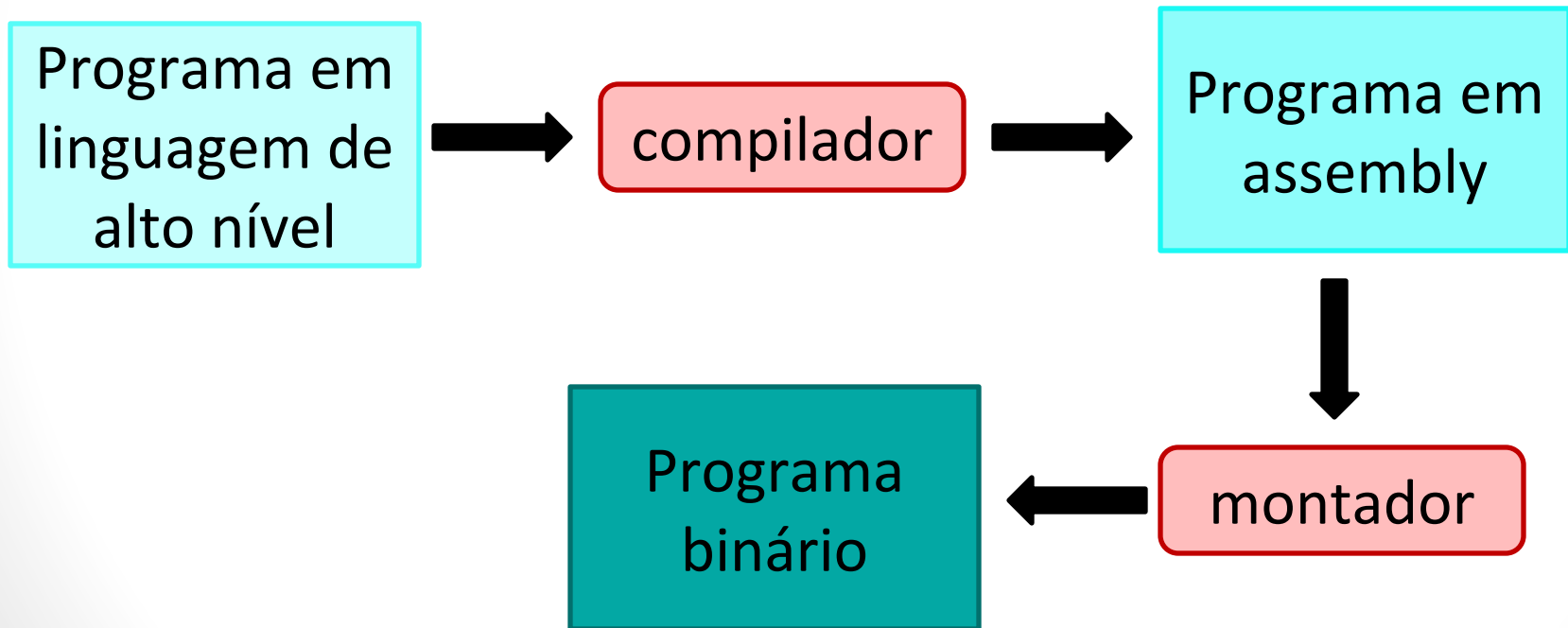
Gerencia os recursos do computador:

- Manipula operações básicas de entrada e saída;
- Atende interrupções;
- Aloca armazenamento e memória;
- Possibilita e controla o compartilhamento do processador entre aplicações que rodam simultaneamente.

# O que fazem os compiladores?

Realizam a tradução de um programa descrito em linguagem de programação de alto nível em linguagem de máquina.

# Função dos compiladores e montadores



# Função dos compiladores e montadores

Programa em linguagem  
de alto nível (em C)

```
swap(int v[], int k)
{ int temp;
  temp = v[k];
  v[k] = v[k + 1];
  v[k + 1] = temp;
}
```

Compilador

Programa em linguagem  
assembly (para MIPS)

```
swap:    multi $2, $5, 4
         add $2, $4, $2
         lw $15, 0($2)
         lw $16, 4($2)
         sw $16, 0($2)
         sw $15, 4($2)
         jr $31
```

Assembler/  
Montador

Programa em linguagem  
de máquina (para MIPS)

```
00001010110000110101111001010100
10111001001000111001001010100100
01110101101010100101110101001010
1010100101010101010111101011111
00010100010001010101101111110100
01010100010010101010010101001001
01010010111001011110010101101000
```



# Princípios básicos

- O endereço representa uma posição particular na memória e pode ser formado de várias maneiras;
- A Unidade Lógica e Aritmética (ULA) é responsável por realizar ações indicadas nas instruções, executando operações numéricas (aritméticas) e não numéricas (lógica);
  - Preparação de informações de desvios do programa;
- O controle do programa e a ULA formam a unidade central de processamento, ou

# Busca-Decodificação-Execução de instruções

## ➤ **BUSCA:**

PC – (program counter) – contador de instruções contém sempre a posição da próxima instrução a ser executada. A instrução apontada pelo PC é trazida da memória para uma área de armazenamento chamada registrador de instruções.

➤ **DECODIFICAÇÃO:** Sinais de controle são gerados de acordo com a informação presente no campo de operação.

➤ **EXECUÇÃO:** A execução da instrução se dá ao final da operação determinada após a decodificação. Ao término da execução da instrução, o ciclo é repetido.

# Busca-Decodificação-Execução de instruções

➤ **DECODIFICAÇÃO:** Normalmente realizada por lógica combinacional.

➤ **EXECUÇÃO:** Exemplos de operações:

- Cálculo do endereço de operandos;
- Busca de operandos na memória;
- Seleção de operação da ULA;
- Carga de registradores;
- Escrita de operandos na memória;
- Atualização do PC para desvios.

O controle do ciclo busca-decodificação-execução é feito pela unidade de controle.

# Elementos funcionais básicos – unidade operacional

- **Unidade Operacional:**
  - Executa as transformações sobre **dados**, especificadas pelas **instruções do computador**;
  - Componentes: ULA, registradores de uso geral e específico, barramentos.
  - Cada organização possui:
    - número e tamanho de registradores variados em cada organização.
    - quantidade e tipo de operações variadas que a ULA realiza.

# Elementos funcionais básicos – unidade operacional

- **Unidade Lógica e Aritmética (ULA)**
  - Realiza operações lógicas e aritméticas;
  - Exemplo: Soma de dois operandos;
  - Negação de um operando;
  - Inversão de um operando;
  - Lógica de operando;
  - Rotação de um operando para a direita ou esquerda.
- As operações da ULA geralmente são bem simples;
- Funções complexas são realizadas pela ativação sequencial das várias operações básicas. Exemplo: multiplicação.

# Elementos funcionais básicos – unidade de controle

- **Lógica combinacional:**
  - os sinais de saída são função exclusiva dos sinais de entrada.
- **Lógica sequencial:**
  - os sinais de saída são função dos sinais de entrada e do estado anterior. A unidade de controle utiliza máquinas de Estado Finitas (FSM).
- As máquinas de estados são circuitos sequenciais que utilizam flip-flops com sinal de clock para que as informações armazenadas possam ser atualizadas sincronamente a intervalos regulares (a cada pulso de clock).
- Existem várias formas de implementar a lógica sequencial. Porém duas são usuais:
  - Organização convencional
  - Organização microprogramada (por descrição de hardware).

# Elementos funcionais básicos – registradores especiais

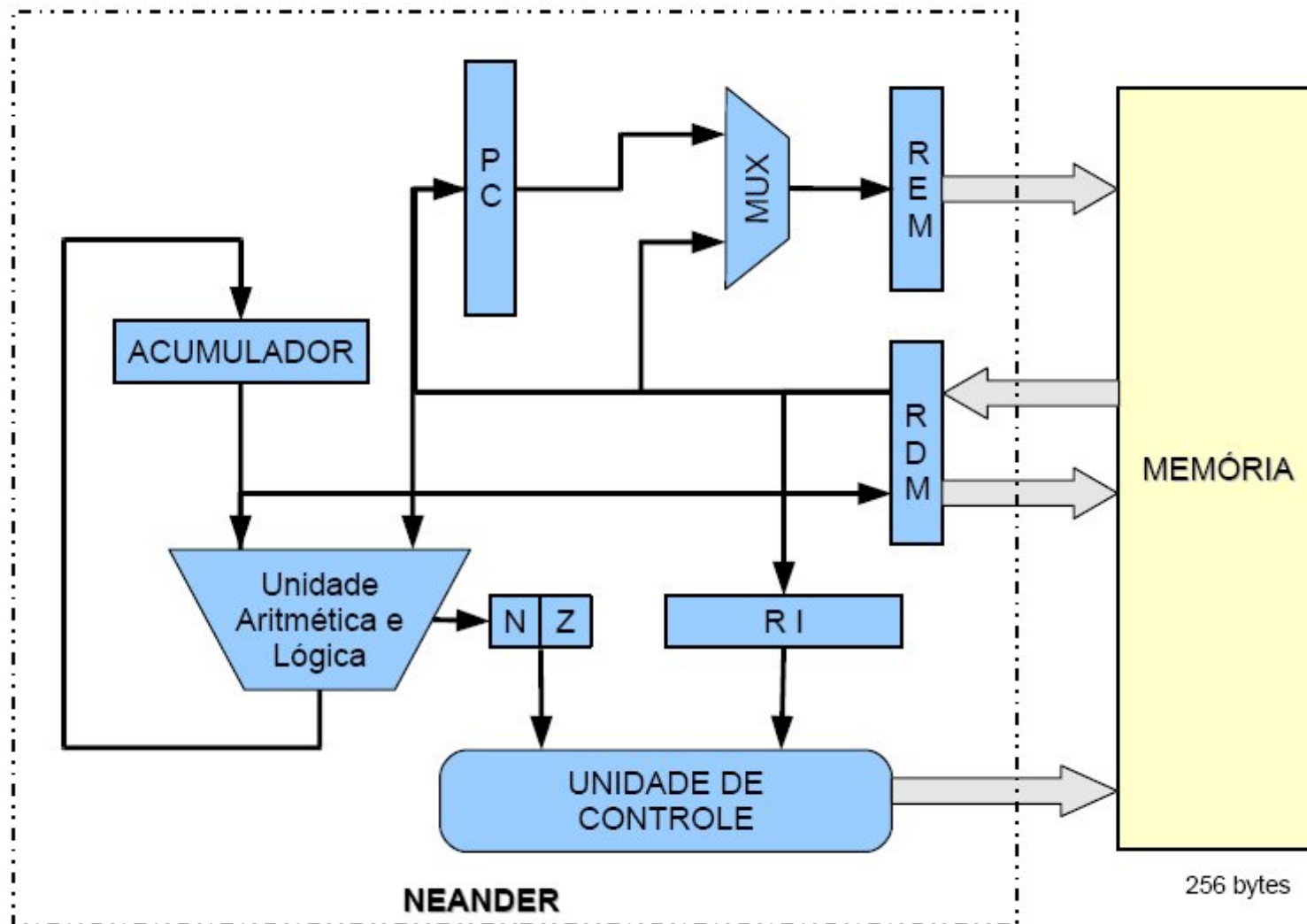
- Existem no computador alguns registradores com funções especiais. Depende da arquitetura e organização de cada máquina;
- Tipos:
  - Apontador de instruções (PC)
  - Registrador de instruções (IR)
  - Registrador de estado (RST)

# Elementos funcionais básicos – registradores especiais

- **Apontador de Instruções (PC)**
  - Tem como função manter atualizado o endereço de memória da próxima instrução;
- **Registrador de instrução (IR)**
  - Armazena a instrução que está sendo executada. De acordo com o conteúdo, a unidade de controle determina quais sinais deve ser gerados;
- **Registrador de estado (RST)**
  - Armazena códigos de condição gerados pela unidade lógica e aritmética;



# Organização do Neander



# Organização MIPS

