ITC Introdução a tecnologia da computação

Arquiteturas Von Neumann X Havard

Professor Me. Vinícius Tessele

John Von Neumann

- **O modelo:** O nome refere-se ao matemático John Von Neumann, que foi considerado o criador dos computadores da forma como são projetados até hoje.
- A ideia do modelo surgiu da necessidade de armazenar programas em um computador, pois, até então, ainda não haviam formas de armazenamento de programas em um computador. Von Neumann e outros pesquisadores descobriram que, utilizando dispositivos de memória em formas de linha de retardo de mercúrio, poderiam armazenar instruções de programas.

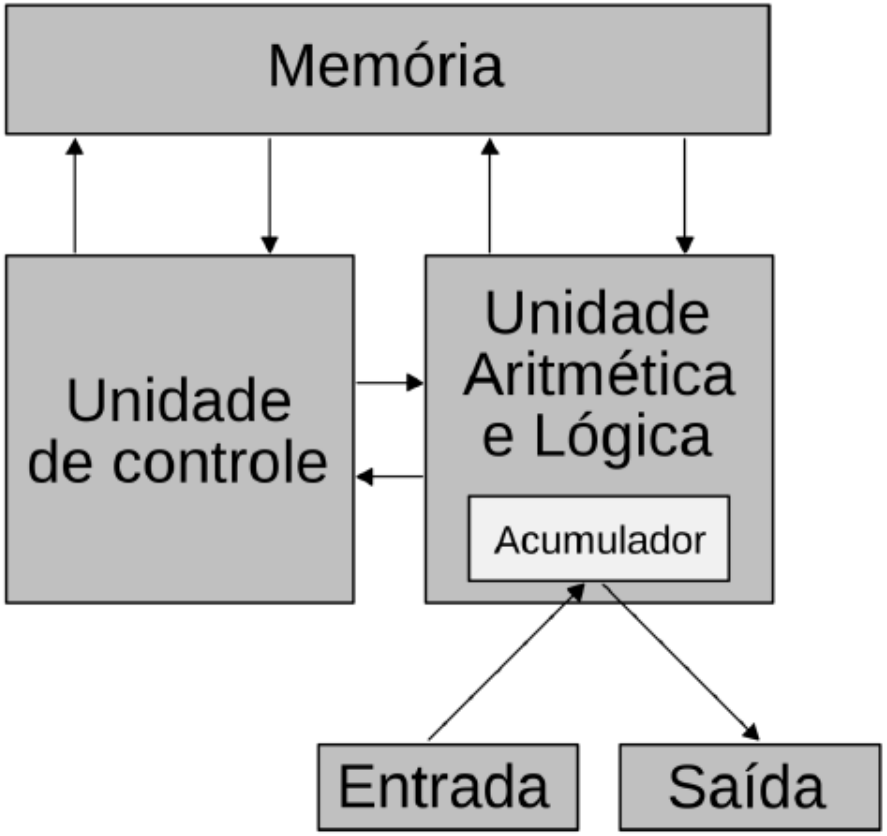
John Von Neumann

- A arquitetura de Von Neumann surgiu a partir de 1946, quando John von Neumann e sua equipe desenvolveram um novo projeto de "computador de programa armazenado". Projetado pela IAS, este computador foi largamente difundido, influenciando muitos projetos subsequentes de outras máquinas.

Von Neumann

- Base de praticamente todas as máquinas atuais;

- Componentes:
  - Memória;
  - Unidade de Controle (UC);
  - Unidade Lógica e Aritmética (ULA)
    - Aritimetic and Logic Unit (ALU);
  - Dispositivos de entrada/saída (E/S) – input/output (I/O).



A Arquitetura de von Neuman

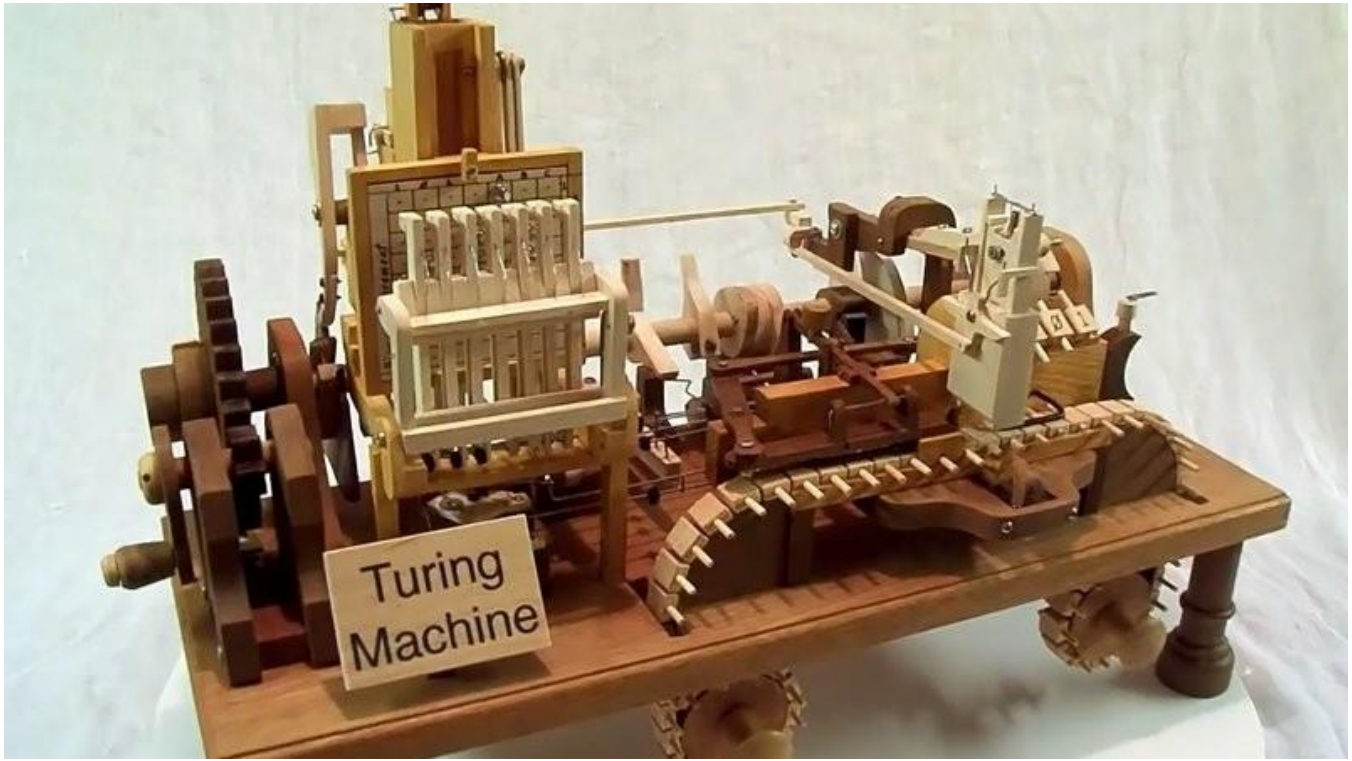
- É uma arquitetura de computador que se caracteriza pela possibilidade de uma máquina digital armazenar seus programas no mesmo espaço de memória que os dados, podendo assim manipular tais programas. Esta arquitetura é um projeto modelo de um computador digital de programa armazenado que utiliza uma unidade de processamento (CPU) e uma de armazenamento ("memória") para comportar, respectivamente, instruções e dados.

Von Neumann

- A máquina proposta por Von Neumann reúne os seguintes componentes
- Uma memória
- Uma unidade aritmética e lógica (ALU)
- Uma unidade central de processamento (CPU), composta por diversos registradores, e
- Uma Unidade de Controle (UC), cuja função é a mesma da tabela de controle da **Máquina de Turing** universal: buscar um programa na memória, instrução por instrução, e executá-lo sobre os dados de entrada.

Máquina de Turing

- A **Máquina de Turing** é um dispositivo teórico conhecido como *máquina universal*, que foi concebido pelo matemático britânico Alan Turing (1912-1954), muitos anos antes de existirem os modernos computadores digitais



Alan Turing

- Vídeo
- Alan Turing, o gênio trágico
- Jogo da imitação

Máquina de von Neumann

- Ciclo de execução:
  - Busca
    - A instrução é lida da memória;
  - Decodificação:
    - Determina-se a instrução a ser executada, geralmente usase lógica combinacional para esta tarefa;
  - Execução:
    - Para cada tipo de instrução é realizada sua execução, conforme o necessário.

Máquina de von Neumann

Conceitos Importantes:

- Registradores:
  - Estruturas de memória interna da CPU, especializadas para funções específicas;
  - Podem armazenar endereços de memória com função específica ou dados temporários;
- OP Code:
  - Conjunto de instruções com formatação e funções específicas, de acordo com a arquitetura da máquina.

Máquina de von Neumann

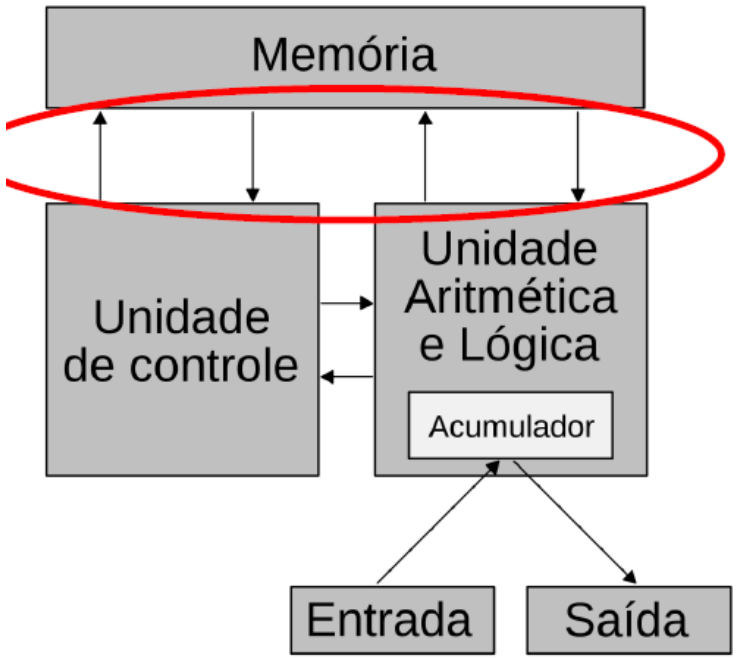
- Relógio (Clock):
  - Referência de tempo necessária à CPU;
  - Um circuito eletrônico oscilador gera uma onda quadrada, essencial para o sequenciamento das operações realizadas pela CPU;
  - Está relacionado com a frequência (taxa) de operação do processador;
- Interrupções:
  - Sinais de controle externos, usados para reconhecimento de ações externas, por parte da CPU;
  - Geralmente são disparados por dispositivos de E/S (teclado, impressora, etc.).

Máquina de von Neumann

- Endereçamento:
  - As informações na memória são localizadas por meio de sua posição;
  - Cada posição da memória possui um endereço de localização;
- Instruções e dados:
  - Armazenados em memória, cada qual em seu espaço de endereçamento específico;
  - Instruções executadas sequencialmente, exceto em casos especiais (saltos ou sub-rotinas);
- Barramento (Bus):
  - Estrutura para transporte das informações (dados, instruções ou sinais de controle).

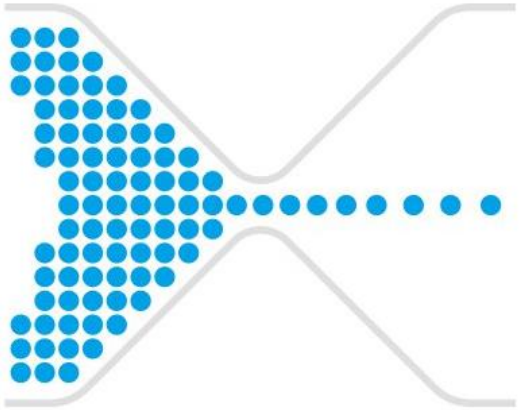
Gargalo de von Neumann

- Tráfego intenso no barramento do sistema:
- Principal rota de informação: entre CPU e memória (ponto crítico);
- Constante fluxo de dados e instruções;
- Gera desperdício de tempo (CPU em espera);
- Agrava-se gradativamente pelo aumento do gap de velocidade entre a memória principal e a CPU.



Gargalo de von Neumann

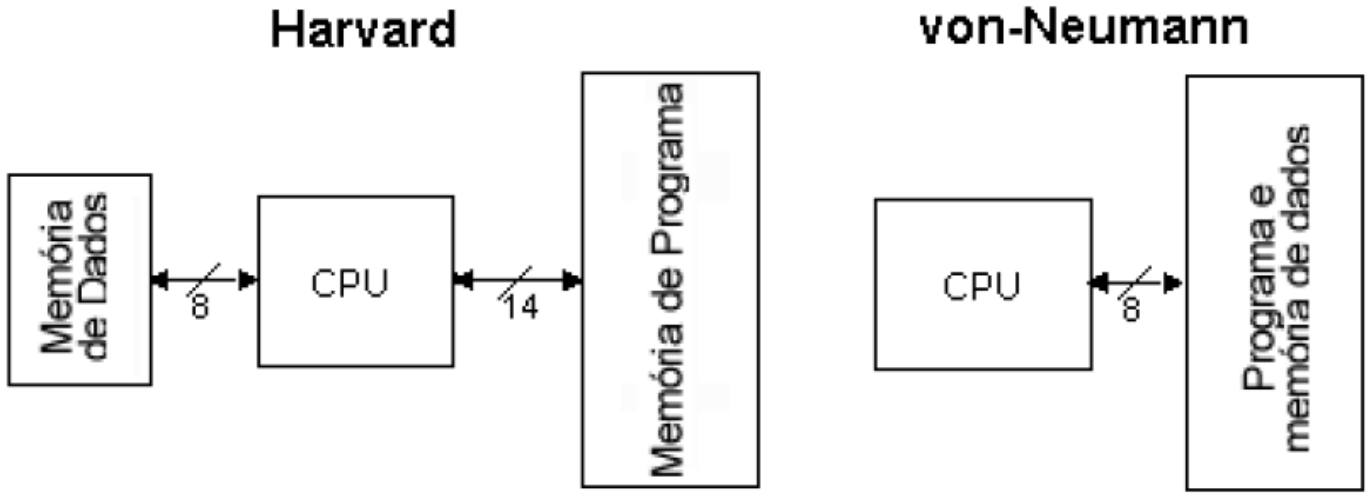
- Limitação da taxa de transferência entre a CPU e a memória em comparação com a quantidade de memória. Esta transferência é menor do que a taxa com que o processador consegue trabalhar e menor do que a quantidade de memória em geral disponível. Isto faz com que a CPU seja forçada a esperar por dados que precisam ser transferidos para ou a partir da memória. Gera desperdício de tempo (CPU em espera).



Arquitetura de Harvard

Arquitetura de Harvard

- Arquitetura de Harvard:
  - É uma variação da arquitetura de von Neumann;
  - Barramentos são separados para instruções e dados;
  - Memórias separadas para dados e instruções
  - O termo Harvard foi originado dos computadores Mark I a Mark IV devido à Universidade de Harvard onde foram desenvolvidos.



von Neumann X Harvard

- A diferença entre a arquitetura de Von Neumann e a Harvard é que a última separa o armazenamento e o comportamento das instruções do CPU e os dados, enquanto a anterior utiliza o mesmo espaço de memória para ambos.

von Neumann X Harvard

- von Neumann:
  - Processa uma informação por vez;
  - Dados e instruções trafegam no mesmo barramento;
  - Mais simples, porém mais lenta;
- Harvard:
  - Dados e instruções trafegam em barramentos separados;
  - Componentes internos dispostos em locais distintos;
  - Mais rápida, porém mais complexa.

Máquinas não-von Neumann

- Máquinas não-von Neumann – outras arquiteturas:
- É qualquer máquina que não se enquadra nas características definidas por von Neumann;
- Máquinas paralelas:
  - Máquinas com várias unidades de processamento (CPUs) executando programas de forma cooperativa, com controle centralizado ou não;
- Máquinas de fluxo de dados:
  - Máquinas que não executam instruções de um programa, mas realizam operações de acordo com a disponibilidade dos dados envolvidos;

- Máquinas não-von Neumann – outras arquiteturas:
  - Redes neurais artificiais:
    - Também não executam instruções de um programa, trabalhando com um modelo onde os resultados são gerados a partir de respostas a estímulos de entrada;
  - Processadores sistólicos (VLSI):
    - O processamento ocorre pela passagem de dados por arranjo de células de processamento executando operações básicas, organizadas de forma a gerar o resultado desejado;

Arquitetura de processadores: RISC e CISC

RISC e CISC

- A arquitetura de processador descreve o processador que foi usado em um computador. Grande parte dos computadores vêm com identificação e literatura descrevendo o processador que contém dentro de si, arquitetura CISC e RISC.
- A CISC (em inglês: *Complex Instruction Set Computing*, Computador com um Conjunto Complexo de Instruções), usada em processadores Intel e AMD; suporta mais instruções no entanto, com isso, mais lenta fica a execução delas.

RISC e CISC

- A RISC (em inglês: *Reduced Instruction Set Computing*, Computador com um Conjunto Reduzido de Instruções) usada em processadores PowerPC (da Apple, Motorola e IBM) e SPARC (SUN); suporta menos instruções, e com isso executa com mais rapidez o conjunto de instruções que são combinadas.

Arquitetura RISC

- É um das inovações mais importantes e interessantes.
- RISC significa uma arquitetura com um conjunto reduzido de instruções (“Reduced Instruction Set Computer”).
- A maioria dos projetos RISC compartilham os seguintes elementos básicos:
  - Grande número de registradores de propósito geral.
  - Um conjunto de instruções simples e limitado
  - Enfoque na otimização da pipeline de instruções

Arquitetura RISC

Características de alguns processadores CISC, RISC

	Computador com um conjunto complexo de instruções (CISC)			Computador com um conjunto reduzido de instruções (RISC)	
Característica	IBM 370/168	VAX 11/780	Intel 80486	SPARC	MIPS R4000
Ano de desenvolvimento	1973	1978	1989	1987	1991
Número de instruções	208	303	235	69	94
Tamanho de uma instrução (bytes)	2-6	2-57	1-11	4	4
Modos de endereçamento	4	22	11	1	1
Número de registradores de propósito geral	16	16	8	40-520	32
Tamanho da memória de controle (Kbits)	420	480	246	—	—
Tamanho da cache (Kbytes)	64	64	8	32	128

CISC ?

- Características da execução de instrução
  - Uma das evoluções em computação foi a invenção das linguagens de programação. Na época, o custo do software era elevado, devido ao limitado número de programadores disponíveis uma vez que era difícil programar nas linguagens alto nível existentes. O software além de caro era pouco confiável, devido esta dificuldade.
  - Resposta dos pesquisadores e da indústria
    - Desenvolver linguagens de programação de alto nível cada vez mais poderosas e complexas, facilitando o desenvolvimento dos softwares.

CISC ?

- Novo problema que surge (o gap semântico)
  - A distância semântica entre as operações disponíveis em linguagem de alto-nível e as operações disponibilizadas pelo hardware dos computadores (linguagem de máquina).
- Características da execução de instruções
  - Resposta ao problema (do gap semântico)
  - Desenvolvimento de arquiteturas que diminuíssem a distância entre as instruções de linguagens de alto nível e as instruções de máquina (em assembly), ou seja uma arquitetura com grande número de instruções em assembly, parecidas com as instruções em alto nível

CISC ?

- As característica destas arquiteturas incluem:
  - Grande conjunto de instruções
  - Muitos modos de endereçamento.
  - Uso intensivo da micro programação
- Esses complexos conjuntos de instruções (em linguagem de máquina) tinham o seguinte objetivo:
  - Facilitar o desenvolvimento de compiladores
  - Melhorar a eficiência na execução de programas, com a implementação de sequencia de operações em microcódigo
  - Oferecer suporte para linguagens de alto nível cada vez mais complexas

CISC x RISC

- Projetos RISC beneficiam-se da inclusão de algumas características RISC (e vice-versa).
- Não existe RISC ou CISC puro atualmente.
  - O PENTIUM inclui algumas características RISC
  - O Power PC inclui algumas características CISC

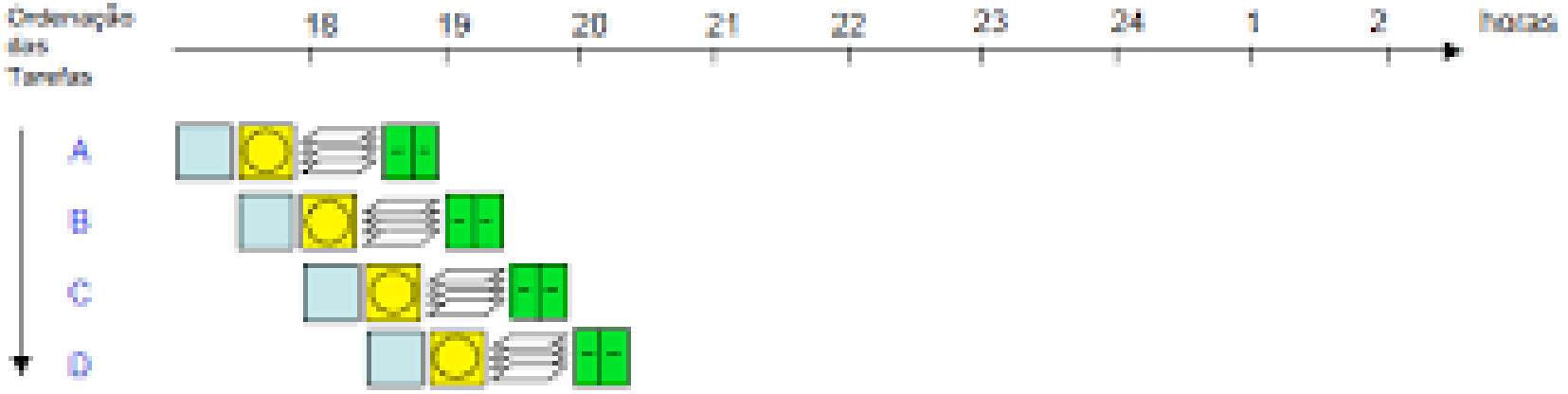
Pipeline

- **Modo de execução com Pipelining:** uma das características mais relevantes da arquitetura RISC é o uso de pipelining, mesmo sabendo que ela tem um funcionamento mais efetivo quando as instruções são todas bastante parecidas.
- A **segmentação de instruções** (em inglês, ***pipeline***) é uma técnica *hardware* que permite que a CPU realize a busca de uma ou mais instruções além da próxima a ser executada. Estas instruções são colocadas em uma fila de memória dentro do processador (CPU) onde aguardam o momento de serem executadas: assim que uma instrução termina o primeiro estágio e parte para o segundo, a próxima instrução já ocupa o primeiro estágio.

Pipeline

- Analogia com uma "Lavanderia doméstica" 1

- **Execução em pipeline**



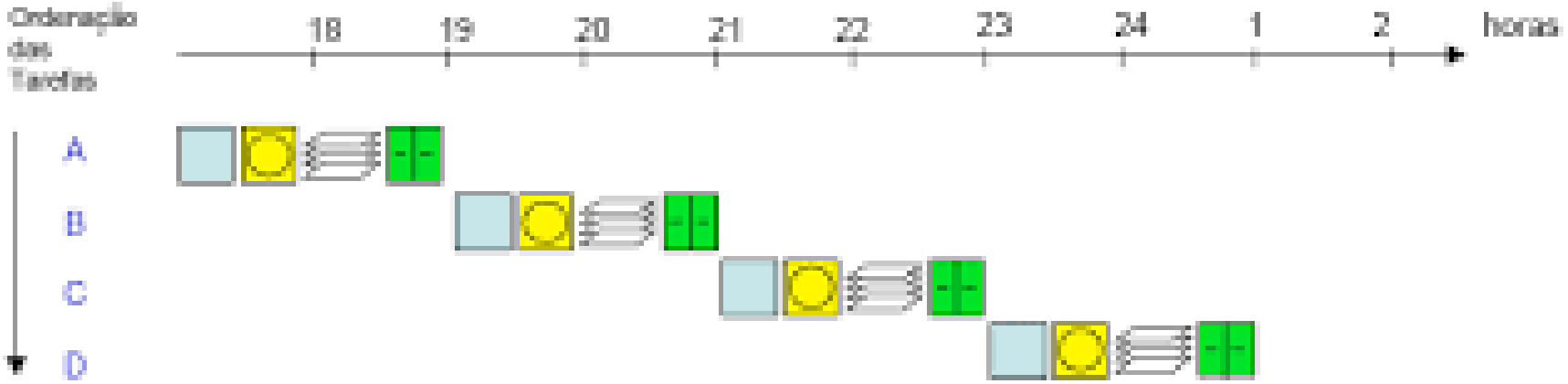
Tempo total: 3,5 horas

Tempo entre duas mudas de roupas prontas: 1/2 hora

Sequencial

- Analogia com uma "Lavanderia doméstica" 2

- **Execução seqüencial**



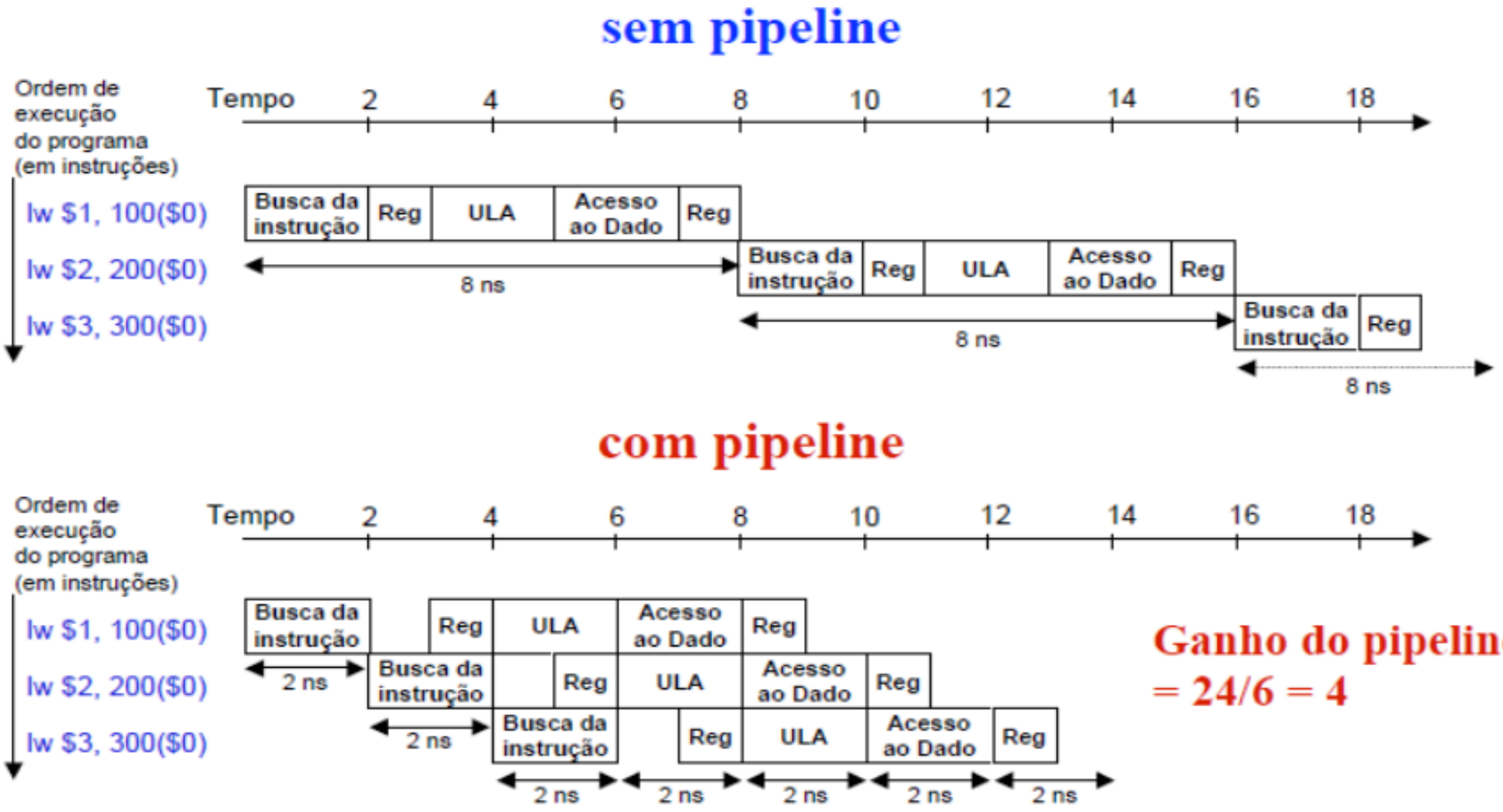
Tempo total: 8 horas

Tempo entre duas mudas de roupas prontas: 2 horas

Visão geral do pipeline

- Técnica aplicada para viabilizar o aumento de desempenho
- Busca de paralelismo em nível de instrução
- Suporte dado em nível de hardware
  - Solução "invisível" ao programador

Desempenho do pipeline



Desempenho do pipeline

- Sem pipeline
  - O tempo decorrido entre o início da primeira instrução e o início da quarta instrução é 3x8 ns = 24 ns
- Com pipeline
  - O tempo decorrido entre o início da primeira instrução e o início da quarta instrução é 3x2 ns = 6 ns

- **Logo, a melhora do desempenho advinda do uso do pipeline é de 24 ns/6 ns = 4 vezes!**

Desempenho do pipeline

- Qual efeito causado pelo pipeline que garante aumenta o desempenho?
  - Freqüência de relógio?
  - Vazão?
  - Tempo de resposta?
  - Feitiçaria?

Desempenho do pipeline

- O pipeline aumenta o desempenho por meio do aumento da vazão (throughput) das instruções, ou seja, aumentando o número de instruções executadas por unidade de tempo (e não por meio da diminuição do tempo de execução de uma instrução individual).