# *Breve Histórico*

## Recomendações…

Tanenbaum, Capítulo 1: Evolução dos computadores e surgimento do SO
youtube.com
O jogo da imitação
Estrelas além do tempo (FORTRAN / IBM)

## 1° Computador Digital (“mecânico”)

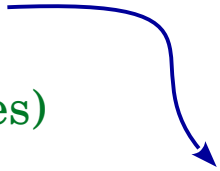
- Charles Babbage
- Ada Lovelace

## Eletrônica

### 1ª geração: 1945 – 1955 (Válvulas)

- II Guerra Mundial
- John Atanasoff, Z3, Colossus, Mark I, ENIAC
- Alguns programáveis (binário ou plugues)
- ∄ SO

### 2ª geração: 1955 – 1965 (Transistores)




- Mainframe
- Assembly / Fortran
- Processamento em Lote (Batch)
- IBM 1401 (computação comercial)
- IBM 7094 (computação científica)
- SO: FMS, IBSYS

## 3ª geração: 1965 – 1980 (CIs)

- Minicomputador
- Multiprogramação
- Timesharing
- IBM System 360 (computação comercial e científica) → SO: OS/360
- SO: MULTICS – Multiplexed Information and Computing Service
    - Versão “castrada” para um único usuário: UNICS → UNIX
      UNIX... Inicialmente escrito em Assembly
      A linguagem C “nasceu” para ele
      CPL → BCPL → B → C

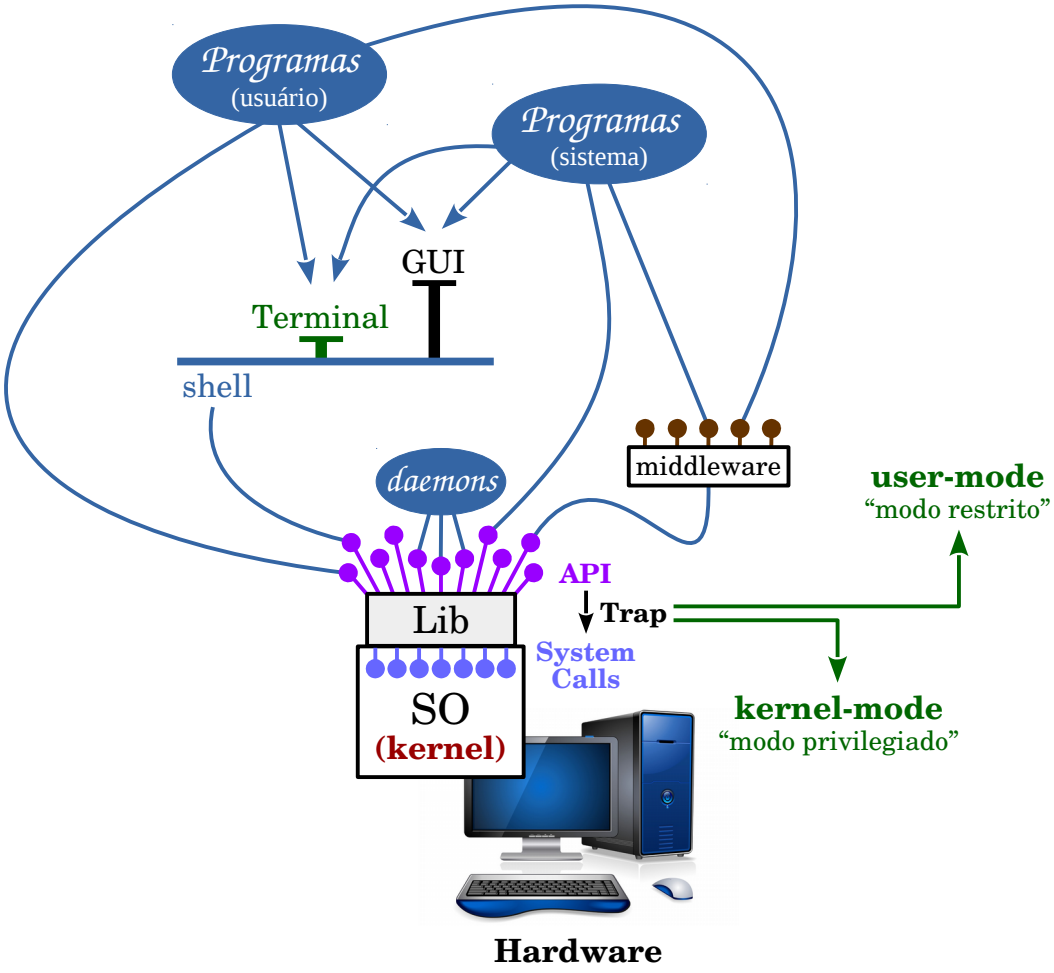
## 4ª geração: 1980 – Hoje (CIs em larga escala)

- Microcomputador
- Era dos PCs
- SO: CP/M, MS-DOS, Apple OS, Windows, MINIX, Linux

## 5ª geração: 1990 – Hoje (Computação móvel)

- Tablet, Smartphone
- SO: Android, iOS, Windows Phone

# *Sistema Operacional*




Camada de software mais fundamental, responsável por gerenciar o hardware e os programas em execução.

O software do SO abstrai os detalhes dos componentes de hardware oferecendo aos programas um **modelo computacional de alto nível** suportado por um conjunto de primitivas essenciais (chamadas do sistema / system calls).

**Processos, threads, arquivos, ...**

"Ao invés de criar um arquivo, considere escrever diretamente no hardware de disco: cilindro, trilha, setor."

O código do SO responsável pelas funções essenciais do sistema e que executa em modo privilegiado na CPU é denominado **kernel**.

Acesso total ao hardware
(kernel-mode)

Contudo, parte da funcionalidade de um sistema operacional pode ser executada em modo restrito.

Acesso limitado ao hardware
(user-mode)

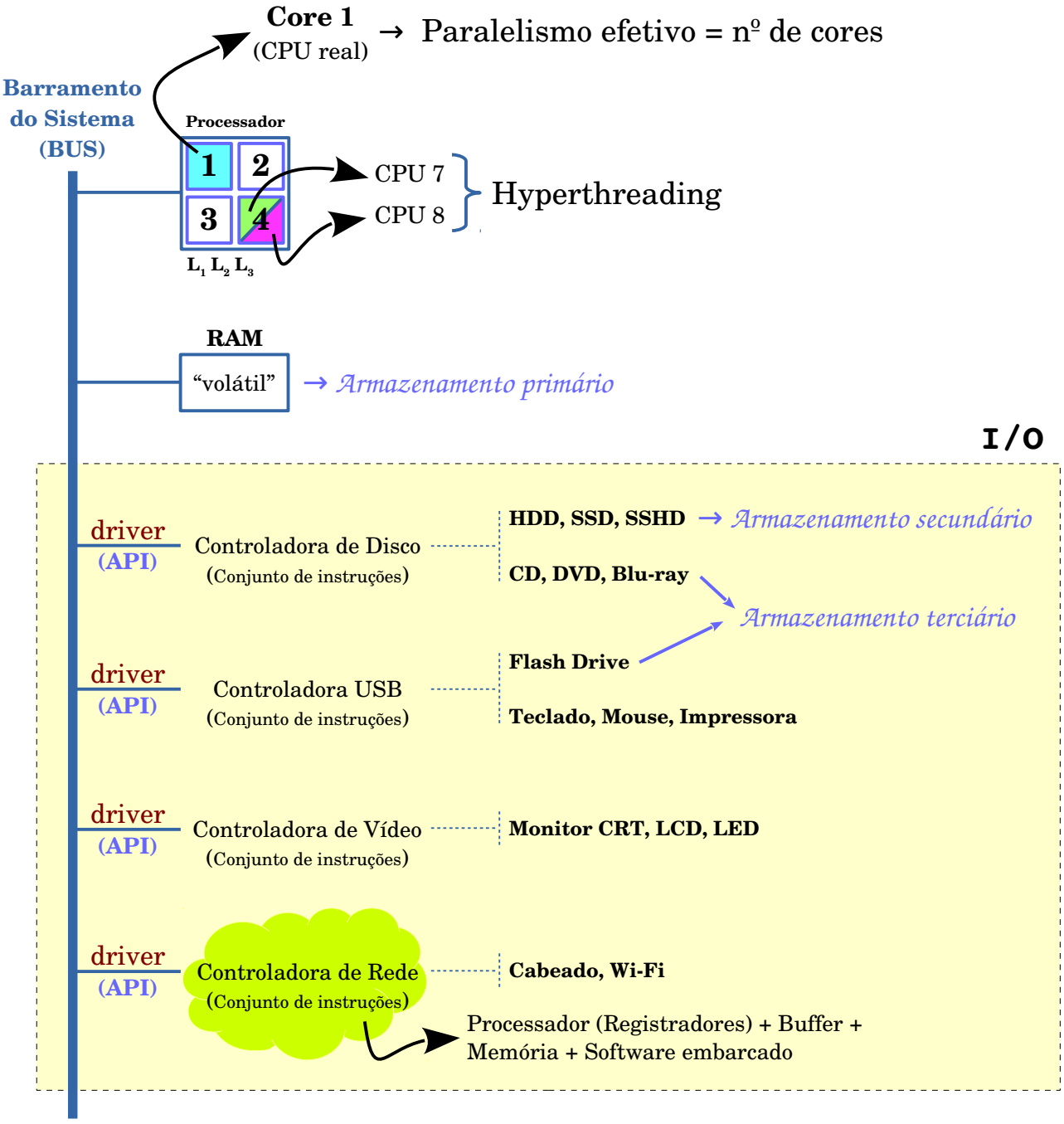
SO → Oferece um ambiente onde os programas são executados de forma controlada e utilizando os recursos de hardware da melhor maneira possível.

**"O sistema operacional existe para os programas.**
**Os programas existem para os usuários."**

O SO de hoje só é possível graças ao hardware de hoje...

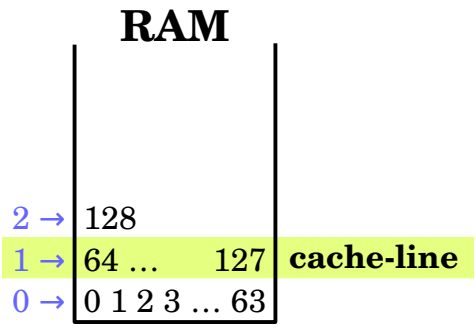
# Arquitetura Computacional

## "Visão simplificada"




E se não houvesse um SO?

# *Cache*



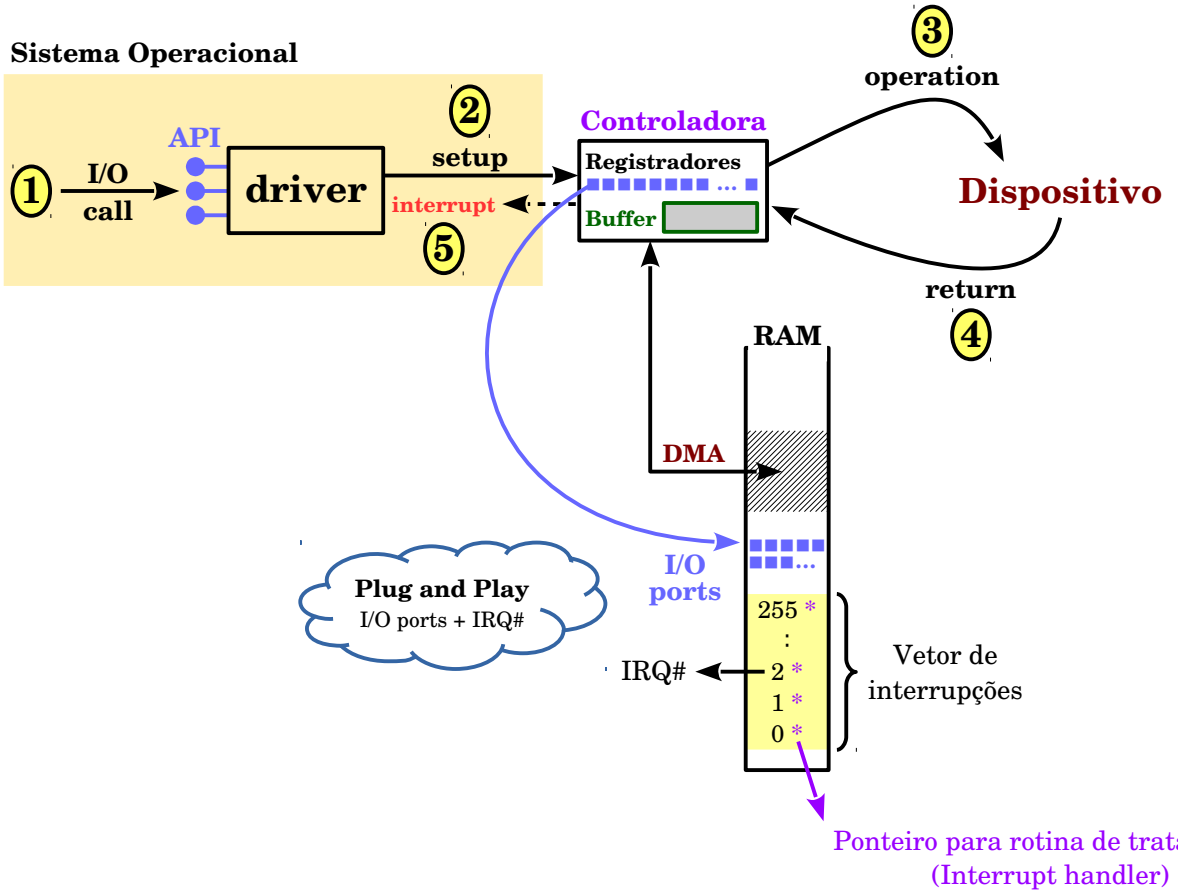

**Problema:** CPU (speed) x RAM (access time)
**Solução:** Memória mais próxima e mais rápida (cache)

A memória cache explora o princípio da localidade espacial e temporal.

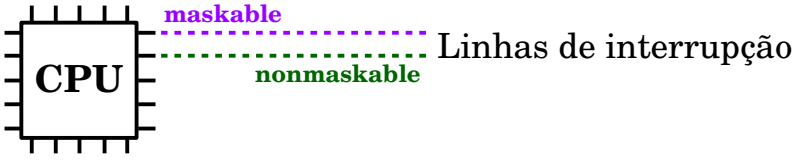
Embora o SO não tenha controle sobre todos os aspectos do hardware (ex.: cache), seu código deve levar em consideração suas nuances para um melhor gerenciamento dos recursos disponíveis.

O que é mais rápido, uma busca linear ou binária?

# *Interrupções*




**Interrupção** → Notifica CPU da ocorrência de um evento. A notificação ocorre por meio de um sinal. A CPU interrompe o processamento atual para responder ao evento, a não ser quando o processamento é crítico.




maskable: Utilizada pelas controladoras de dispositivos. Linha desativada pela CPU quando uma sequência de instruções críticas deve ser executada (operação atômica).

## Vetor de interrupções (Intel)...

0..31 = nonmaskable (condições de erro: divisão por zero, opcode inválido, ponto flutuante, etc.)
32..255 = maskable

## Interrupções são geradas em...

Hardware: - Controladoras
- Timer

Software: - Divisão por zero
- Acesso inválido à memória
- Uma chamada explícita ao sistema (syscall) → TRAP

Interrupções aninhadas e controle de prioridade → SO (real-time)

Algumas funcionalidades do SO (gerenciamento de interrupções) são implementadas em Assembly. **Motivo:** A linguagem C não fornece o acesso de baixo nível necessário.

Implementação de um SO → C/C++; Assembly