

# Hierarquia de Memória I

Prof. Gustavo Girão

# Plano de Aula

- Motivação
- Introdução
- Hierarquia de memória
- Memória cache
  - Posicionamento
  - Substituição
  - Gravação

# Introdução

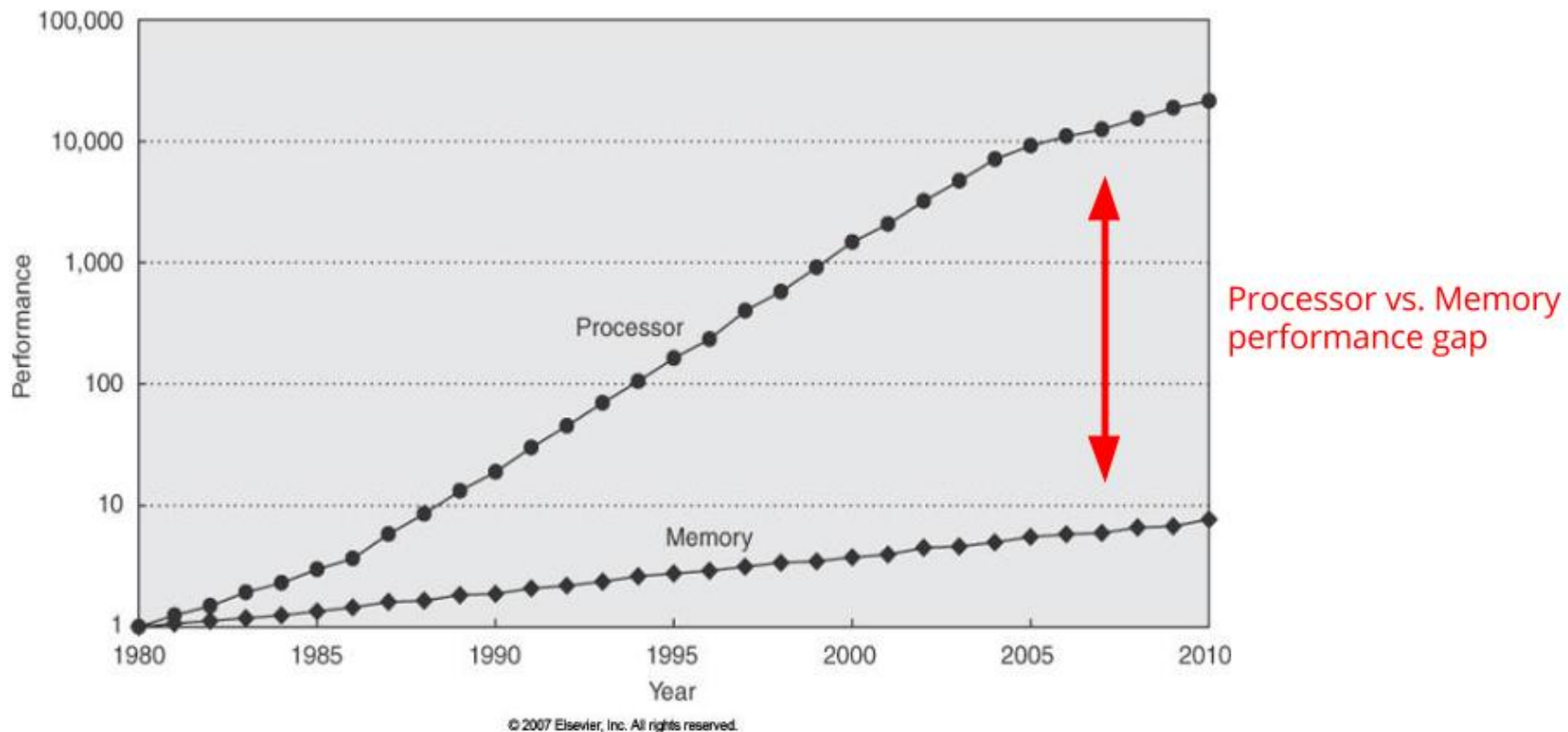
- ◆ Memória:
  - ◆ Pode ser definida como um local para **armazenamento de informações**, onde as duas únicas ações possíveis são a **leitura** e a **escrita**.
  - ◆ A informação pode ser representada pelo bit ou por um conjunto de **n bits** que possuem um **endereço** definido.
  - ◆ Em um sistema computacional, temos **diferentes tipos de memórias**, para diferentes finalidades, que se **interligam de forma estruturada** e que formam o subsistema de memória.

EM UM CENÁRIO IDEAL, ONDE PREÇO  
NÃO É UM PROBLEMA, QUAL O TAMANHO  
E A LOCALIZAÇÃO DA MEMÓRIA EM UM  
COMPUTADOR?

# Hierarquia de Memória

- ◆ Queremos memórias grandes e velozes, mas elas são caras
- ◆ O que temos são diferentes tecnologias de memória onde as mais velozes são mais caras e as mais baratas são mais lentas
- ◆ Essas memórias podem ser distribuídas dentro da arquitetura do computador
- ◆ A solução está em fazer com que as diferentes memórias combinadas pareçam ser uma memória grande e veloz a um custo razoável

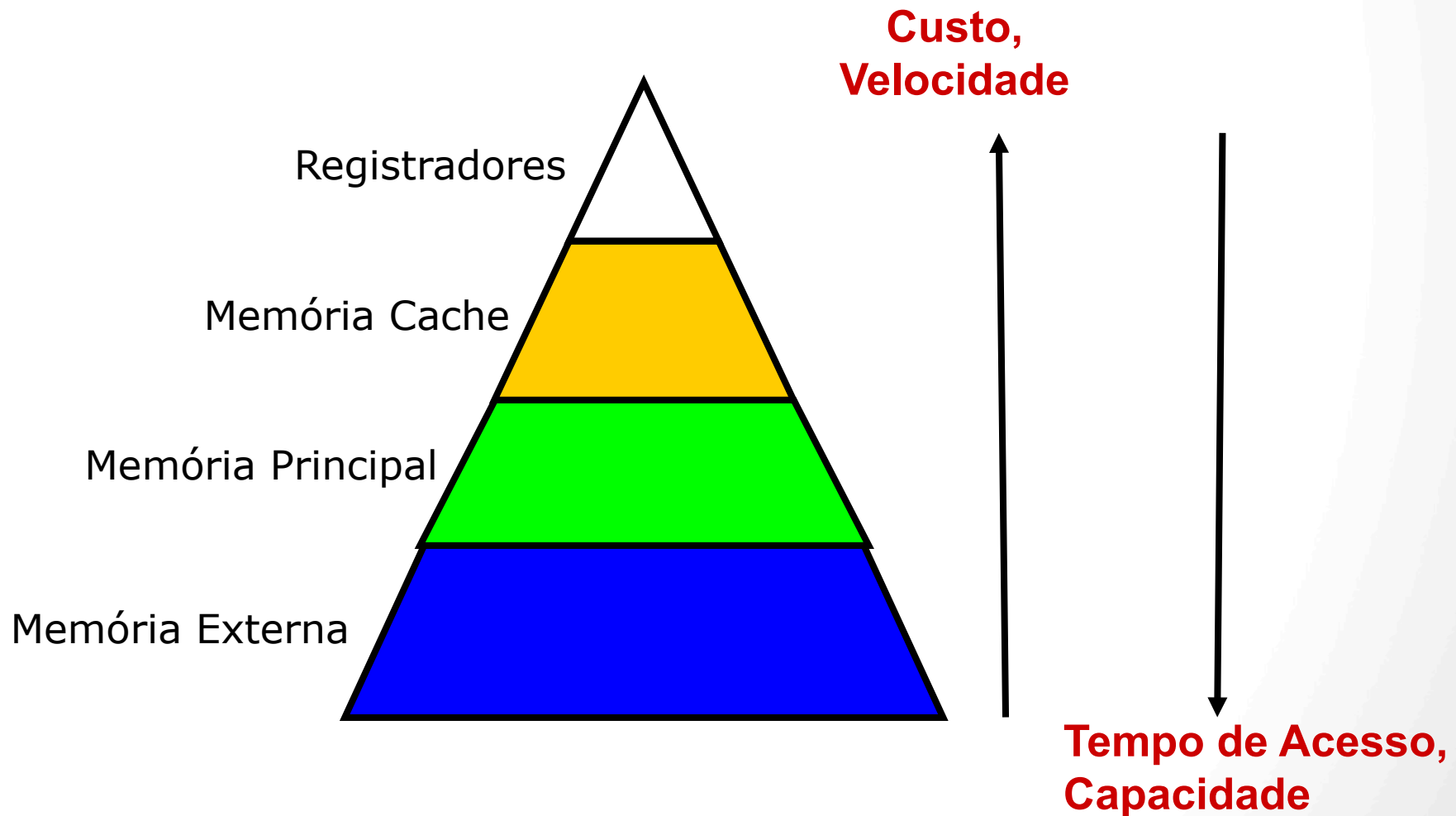
# Memory Wall



Memory speed lags behind CPU speed

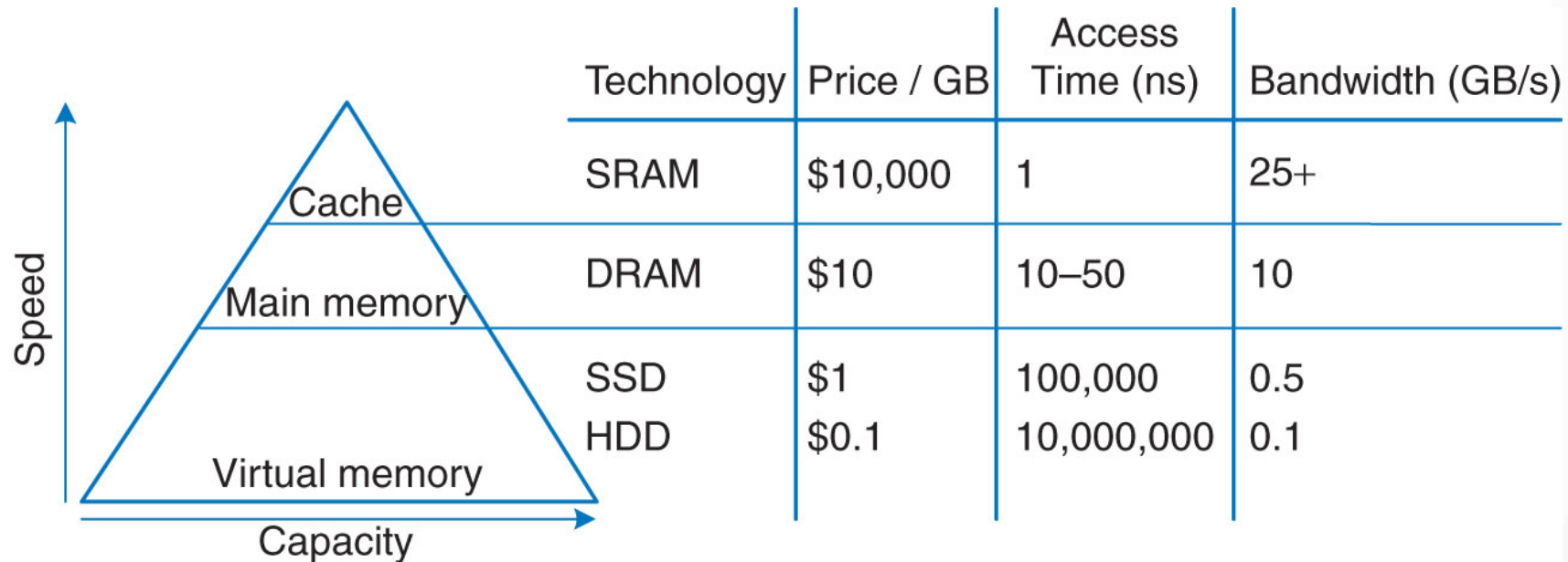
# Hierarquia de Memória

- ◆ Hierarquia de Memória
  - ◆ Inclusiva: cada nível é um sub-conjunto do nível inferior



# Hierarquia de Memória

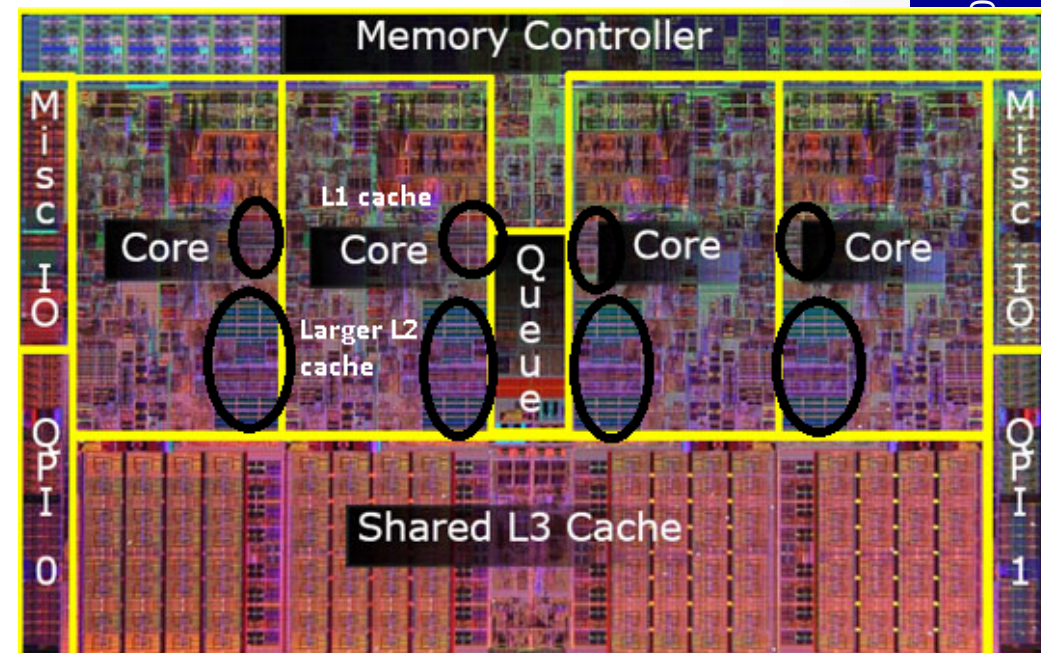
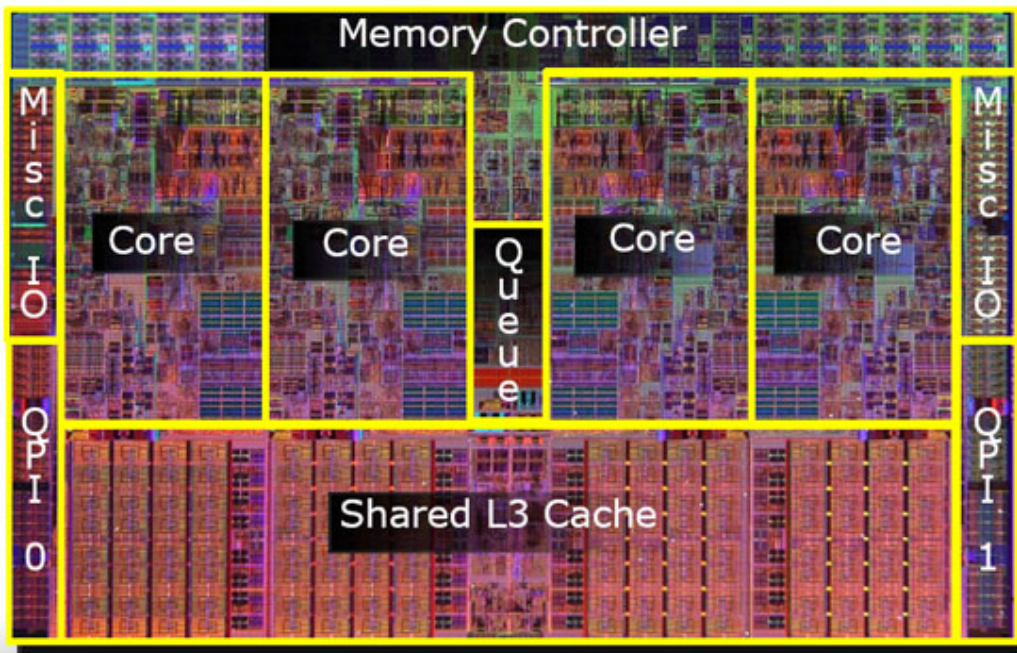
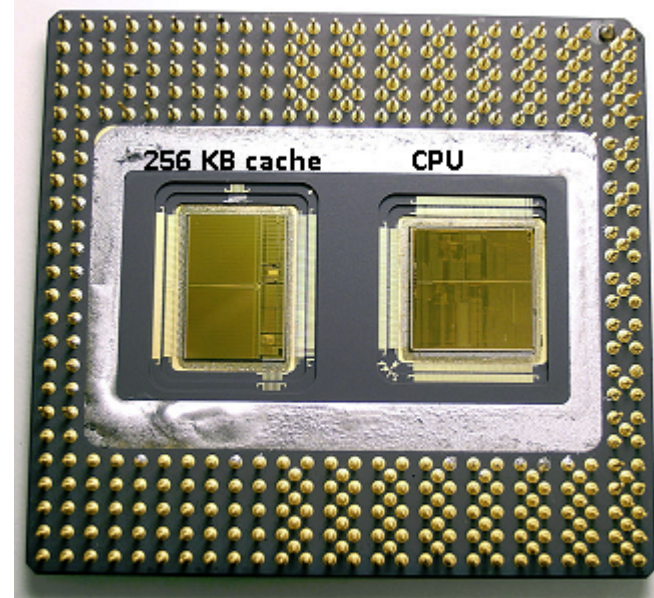
- ◆ Hierarquia de Memória
  - ◆ Componentes tecnológicos





# Quem são?

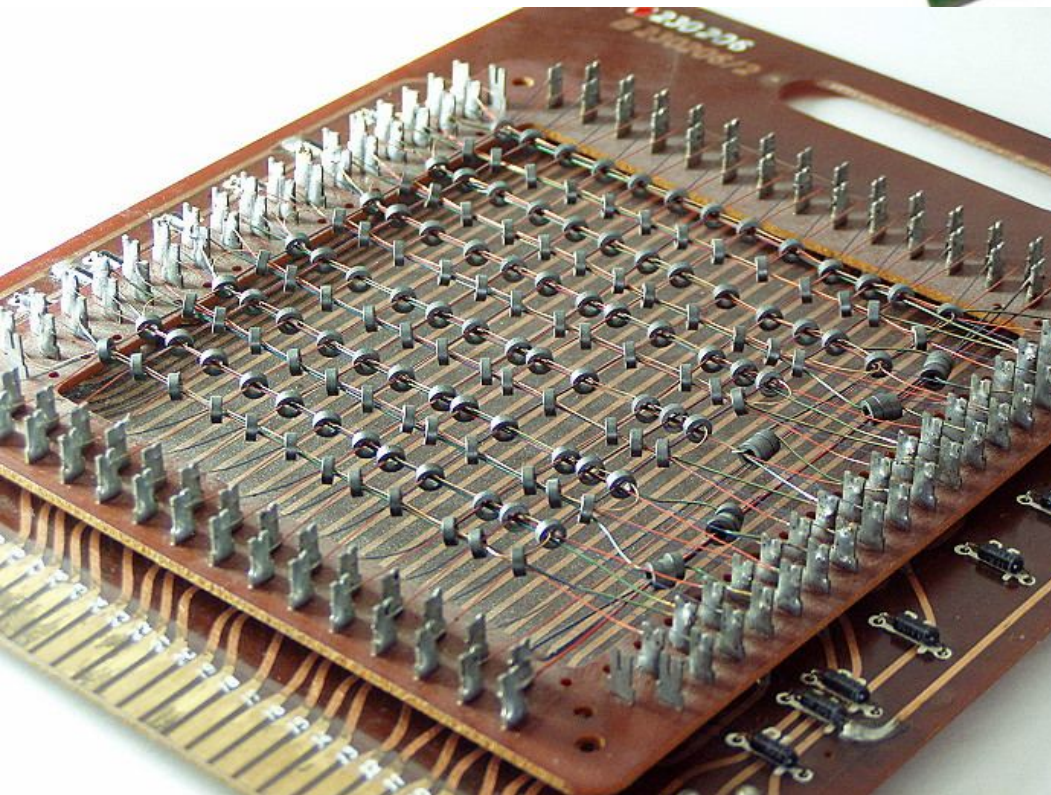
- Caches





# Quem são?

- Memória Principal
  - Memória RAM



# Quem são?

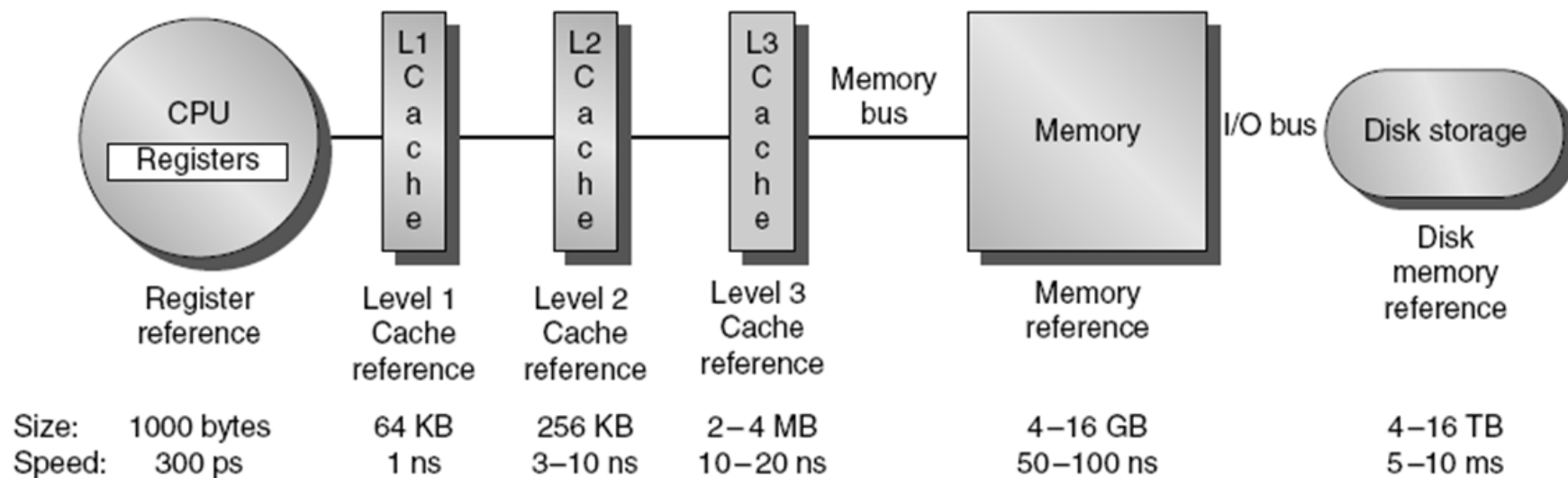
- Memória Secundária
- Discos, Fitas, Cartões





# Hierarquia de Memória

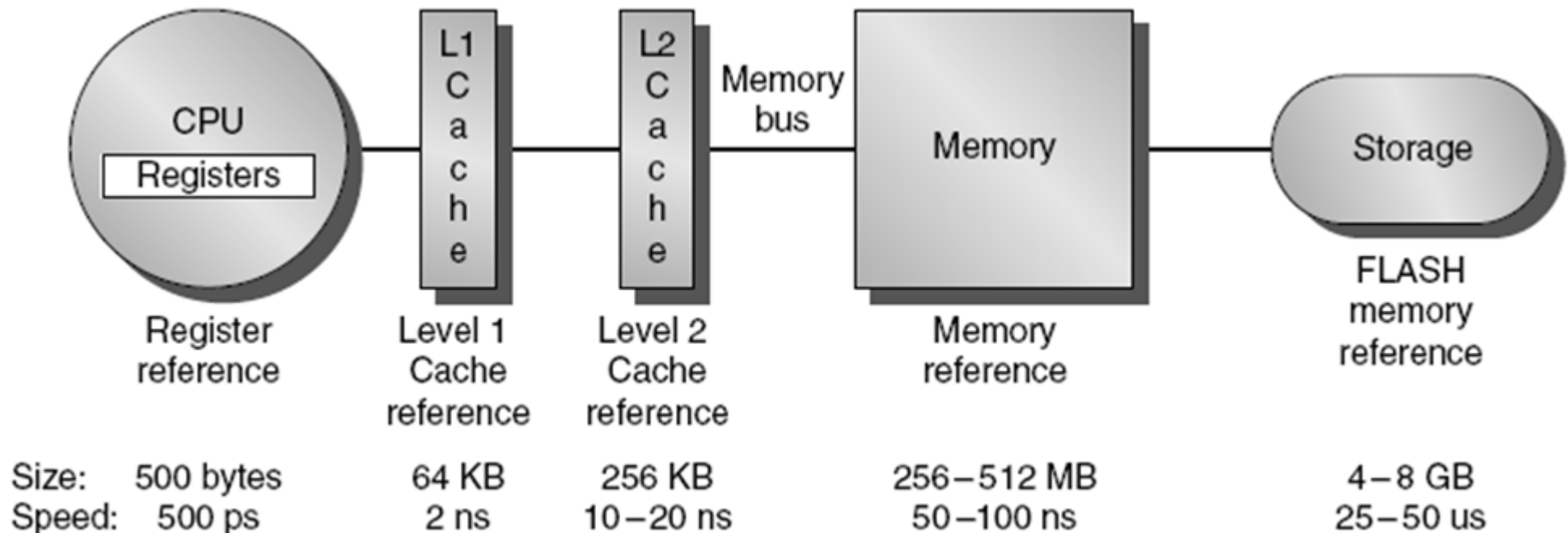
- Tamanhos e latência
  - Para servidores



(a) Memory hierarchy for server

# Hierarquia de Memória

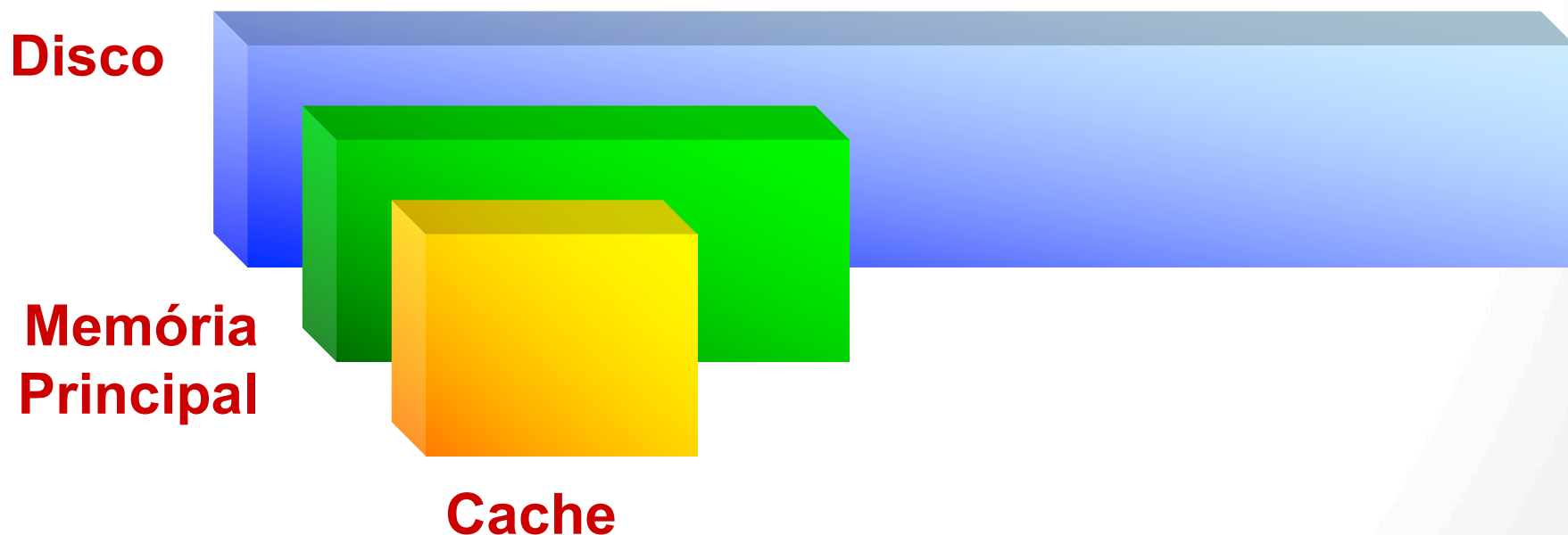
- Tamanhos e latência
  - Para dispositivos móveis



(b) Memory hierarchy for a personal mobile device

# Hierarquia de Memória

- ◆ Os dados contidos num nível mais próximo do processador são sempre um sub-conjunto dos dados contidos no nível anterior. (hierarquia de memória inclusiva!)
- ◆ O nível mais baixo contém a totalidade dos dados.



# Atenção

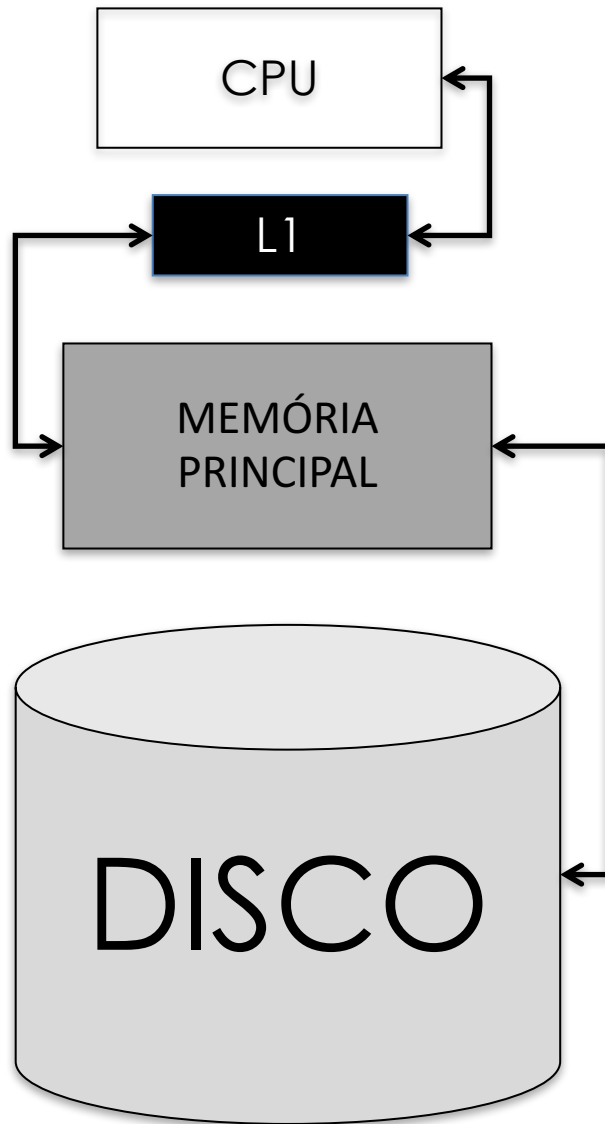
- Memória Primária (principal): memória que o processador pode endereçar diretamente;
- Memória Secundária: não podem ser endereçadas diretamente. Ou seja, a informação precisa ser primeiro carregada na memória principal para depois ser enviada para a memória secundária.

# Cache Hit e Cache Miss

- Quando o processador busca uma informação (dado ou instrução) na **memória principal**, ele vai primeiro na **cache**
- Se a informação está na cache, chama-se *cache hit*
  - A informação é buscada e será usada pelo processador
- Se a informação NÃO está na cache, chama-se *cache miss*
  - A informação precisa ser buscada na memória principal

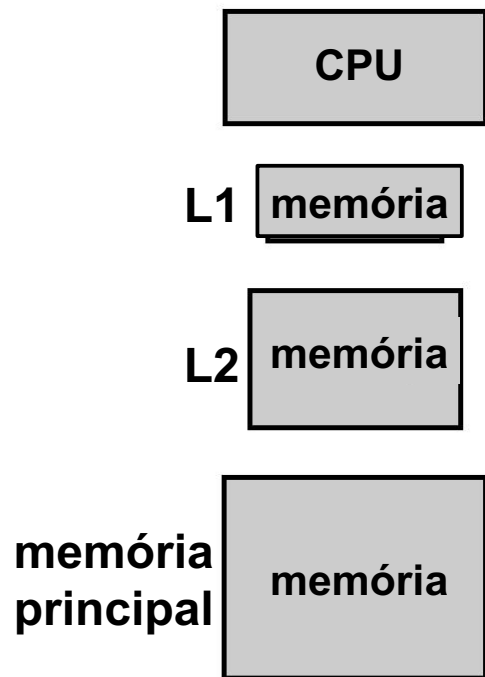


# Operação da Hierarquia de Memória



1. Acessa L1 p/ instrução ou dado. Se encontra a informação (cache hit), executa;
2. Senão (cache miss), busca na memória principal. Se acha, coloca em L1 e repete o passo 1;
3. Senão, obtém através de E/S.

# Operação da Hierarquia de Memória



- (1) Busca L1 p/ instrução ou dado.  
Se acha (*cache hit*), executa.
- (2) Senão (*cache miss*), busca L2.  
Se acha, coloca em L1 e repete (1).
- (3) Senão, busca na memória principal.  
Se acha, coloca em L2 e repete (2).
- (4) Senão, obtém através de I/O.

**Passos (1) - (3) são executados em hardware:**

- 1-3 ciclos p/ obter da cache L1.
- 5-20 ciclos p/ obter da cache L2.
- 50-200 ciclos p/ obter da memória principal.

**Passo (4) é executado pelo SO**

# Princípio da Localidade

- ◆ Localidade: **estatística de comportamento** do acesso aos dados da memória.
- ◆ Se um item é referenciado,
  - ◆ Ele tende a ser referenciado em **breve** novamente
  - ◆ Os itens na sua **vizinhança** também o serão

# Princípio da Localidade

O princípio da localidade faz com que a hierarquia de memória funcione na prática.

# Localidade

- ♦ Localidade temporal:
  - ♦ Um item referenciado tende a ser referenciado novamente em um curto espaço de tempo

EXEMPLO?

- ♦ Localidade espacial:
  - ♦ Os itens próximos na memória a um item referenciado tendem a ser também referenciados

EXEMPLO?

# Localidade

- ◆ Localidade temporal:
  - ◆ Um item referenciado tende a ser referenciado novamente em um curto espaço de tempo
  - ◆ Ex: loops
- ◆ Localidade espacial:
  - ◆ Os itens próximos na memória a um item referenciado tendem a ser também referenciados
  - ◆ Ex: matrizes e estruturas

# Localidade – Exemplo 1

```
int soma_linhas(int a[M][N]) {  
    int i, j, sum = 0;  
    for (i = 0; i < M; i++)  
        for (j = 0; j < N; j++)  
            sum += a[i][j];  
    return sum;  
}
```

- ◆ **Localidade temporal:**  
Mesmas instruções são executadas seguidamente  $M \times N$  vezes.
- ◆ **Localidade espacial:**  
Percorre-se a matriz linha a linha.  
Assim, faz-se sempre acesso a posições de memória contíguas.

# Localidade – Outro Exemplo

Baixa Localidade Temporal

Baixa Localidade Espacial



# Exemplo 2 – Baixa L. Espacial

```
int soma_coluna(int a[M][N]) {  
    int i, j, sum = 0;  
    for (j = 0; j < N; j++)  
        for (i = 0; i < M; i++)  
            sum += a[i][j];  
    return sum;  
}
```

- ◆ **Localidade temporal:**
  - ◆ Mesmas instruções são executadas seguidamente  $M \times N$  vezes.
- ◆ **Localidade espacial:**
  - ◆ Percurso da matriz é feito coluna a coluna
  - ◆ Portanto, dois acessos consecutivos não fazem acesso a posições de memória contíguas.

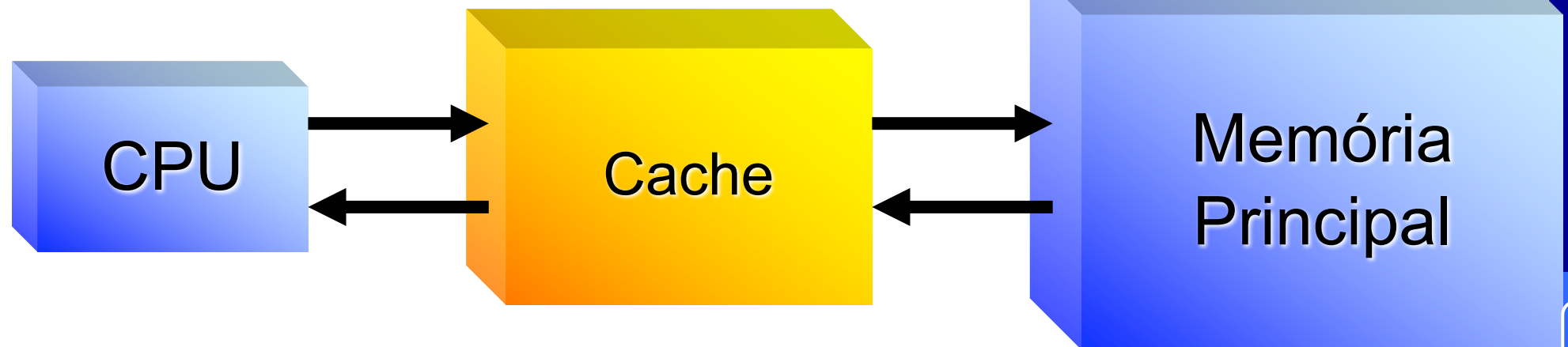
# Memória Cache

- ◆ Memória rápida e de pequena capacidade.
- ◆ Guarda somente os itens que serão **referenciados com frequência** em determinado momento.
- ◆ Para tal, **explora os conceitos** de Localidade **Temporal** e de Localidade **Espacial**.
- ◆ Tais itens podem ser tanto **dados**, como **instruções**.

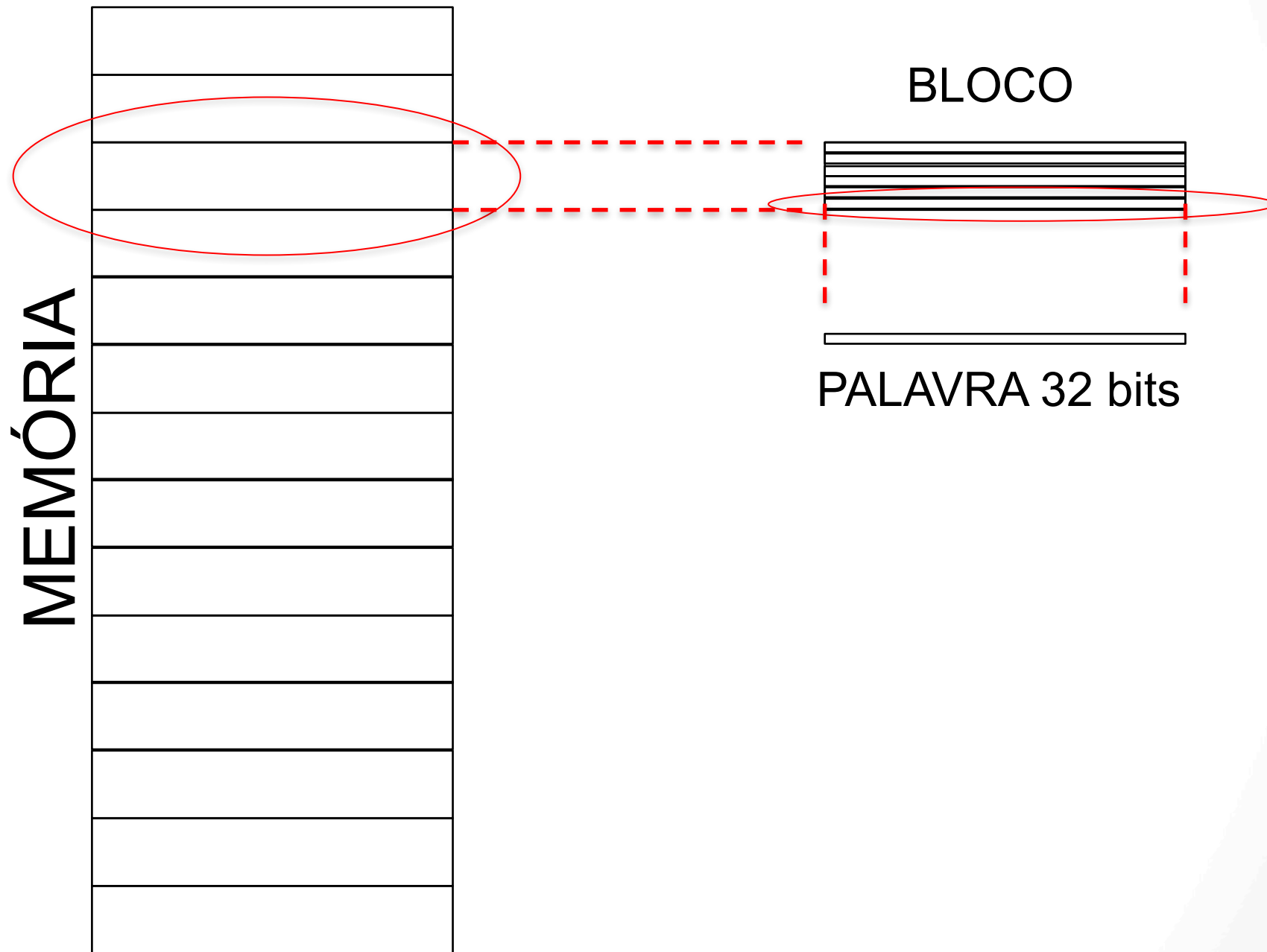
# Memória Cache

Transferência  
de palavras

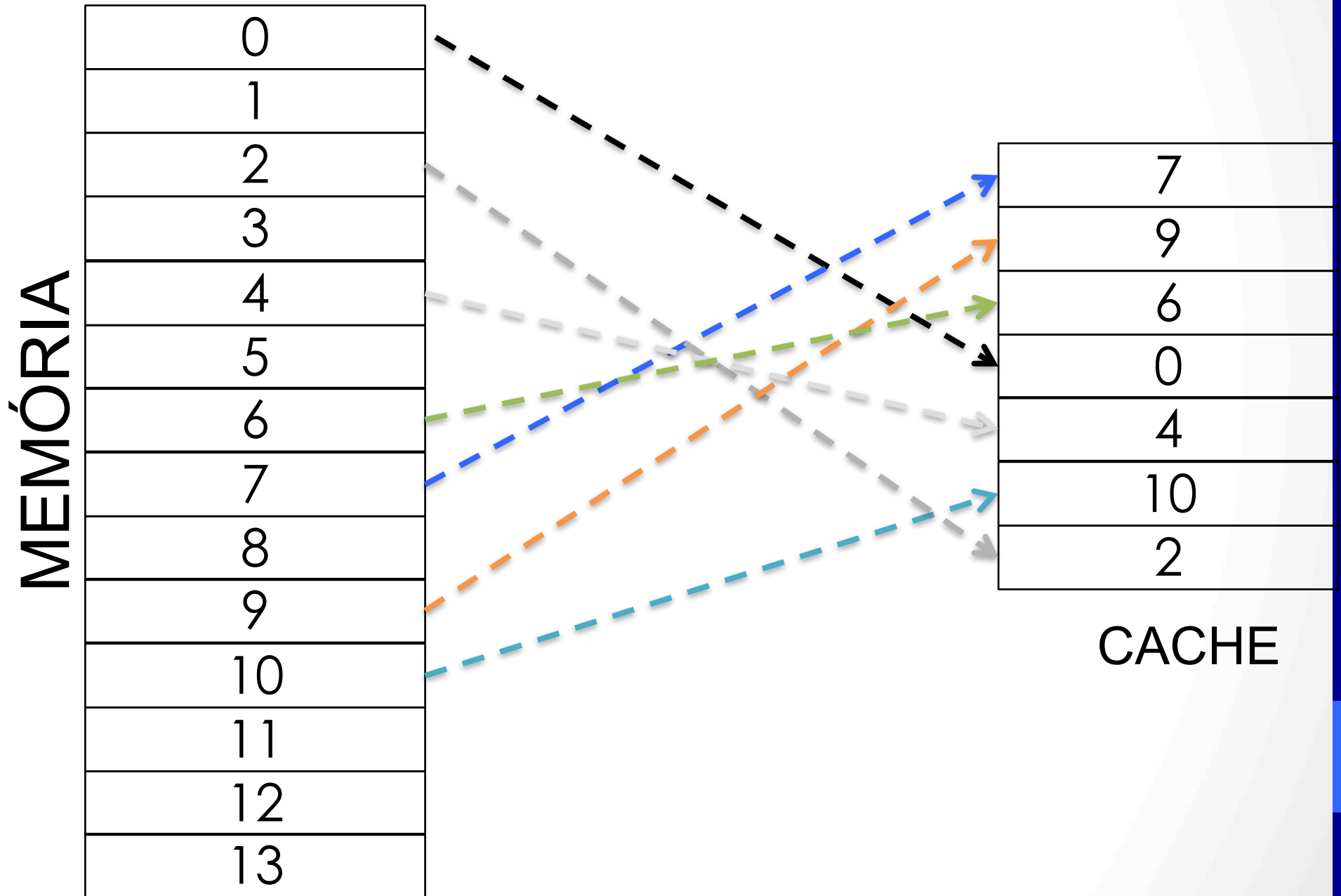
Transferência  
de blocos



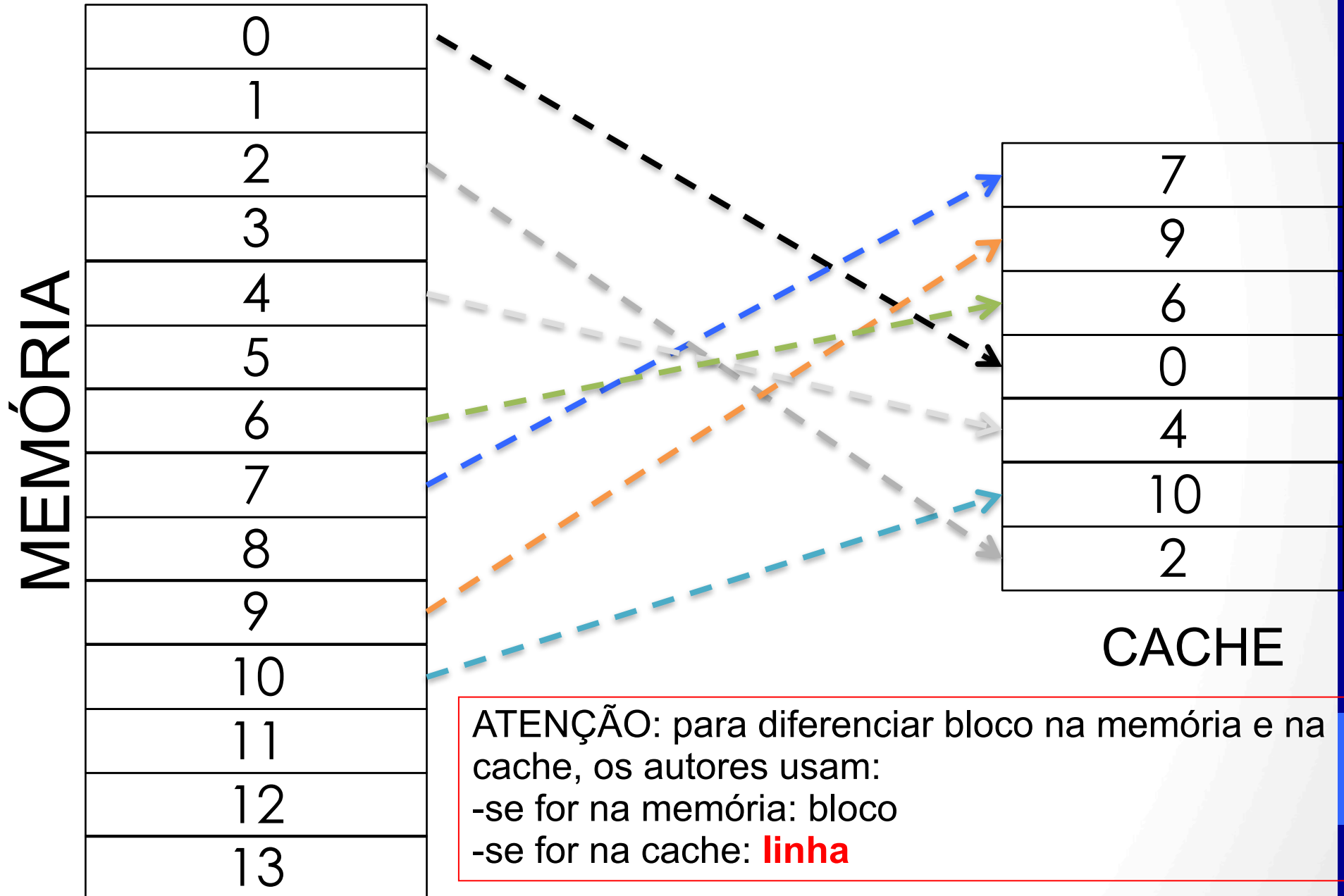
# Divisão da Memória



# Divisão da Memória



# Divisão da Memória



# Terminologia

- ♦ **Taxa de hit (hit rate):**  
% de hits ocorridos em relação ao total de acessos à memória
- ♦ **Taxa de miss (miss rate):**  
% de misses ocorridos em relação ao total de acessos à memória. Igual a 1-hit rate
- ♦ **Tempo de hit (hit time):**  
tempo para determinar se o acesso é um hit +  
tempo para acessar e entregar o dado de um nível inferior p/ CPU
- ♦ **Penalidade de miss (miss penalty):**  
tempo para determinar se o acesso é um miss +  
tempo para substituir o bloco no nível inferior +  
tempo de entregar o bloco à CPU

# Terminologia

- ♦ **Palavra:**  
Unidade do sistema. Algumas memórias podem ser acessadas por byte ao invés de palavra.
- ♦ **Bloco:**  
Um coletivo de palavras dentro da **memória**
- ♦ **Linha:**  
Um coletivo de palavras dentro da **cache**. Para facilitar, todo sistema adota o mesmo tamanho de Bloco para a Linha.
- ♦ **Tag:**  
Identificador único de um Bloco da memória. Utilizado para rotular a linha da cache identificando assim o bloco que atualmente está lá



# Questões Básicas na Hierarquia de Memória

## I. **Posicionamento do bloco.**

Onde o bloco deve ser colocado na memória de nível mais alto?

## II. **Substituição de bloco.**

Quais blocos serão trocados em um miss?

## III. **Estratégia de gravação.**

O que acontece em uma escrita?

# Questões Básicas na Hierarquia de Memória

## I. **Posicionamento do bloco.**

Onde o bloco deve ser colocado na memória de nível mais alto?

## II. **Substituição de bloco.**

Quais blocos serão trocados em um miss?

## III. **Estratégia de gravação.**

O que acontece em uma escrita?

**Discutiremos sobre isso na próxima aula!**



# Bibliografia

- ▶ PATTERSON, D.A. & HENNESSY, J. L.  
**Organização e Projeto de Computadores -**  
A Interface Hardware/Software. 3ª ed.  
Campus, 2005. **CAPÍTULO 7**
- ▶ STALLINGS, William. Arquitetura e  
organização de computadores. 8. ed. São  
Paulo: Pearson, 2010. **Capítulo 4**

# Próxima aula

- Memória Cache parte 2!