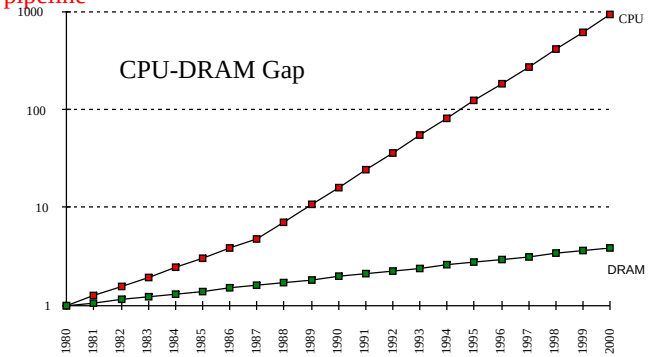
Hierarquia de Memória — Motivação, Definições, Quatro Perguntas Fundamentais

Hierarquia do Sistema de Memória

Cache: a safe place for hiding or storing things...

Quem se Preocupar com a Hierarquia de Memória?

- Somente vimos processadores até agora...
  - Custo/performance de CPU cost/performance, ISA, Execução com pipeline

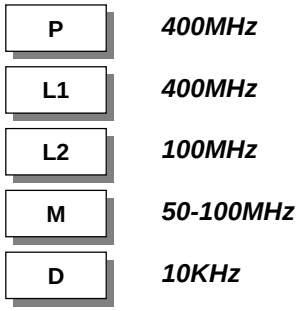


- 1980: não haviam caches em processadores (386)
  1995: caches de 2 níveis no PC (486,Pentium)

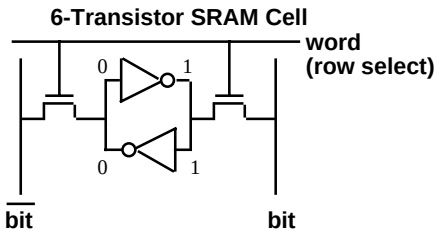
Princípios Gerais

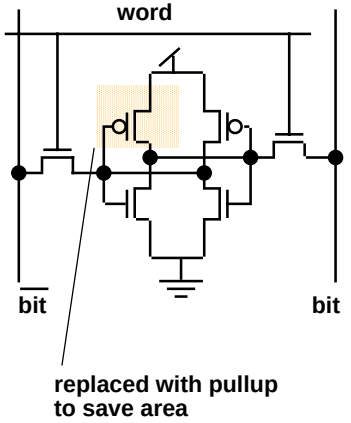
- Localidade
  - *Localidade temporal*: dados serão referenciados novamente
  - *Localidade espacial*: itens serão referenciados na vizinhança
- Localidade + HW menor é mais rápido == hierarquia de memória
  - *Níveis*: cada menor e mais rápida é mais cara que a de nível mais baixo
  - *Inclusive*: dado encontrado no topo também é encontrado no nível mais baixo
- Definições
  - *Nível mais alto* é mais próximo ao processador
  - *Bloco*: unidade mínima de dados (também conhecido como *linha*)
  - Endereço = *endereço do bloco* + *offset dentro do bloco*
  - *Hit time*: Tempo para acessar dado no nível mais alto

Princípios Gerais de Caches



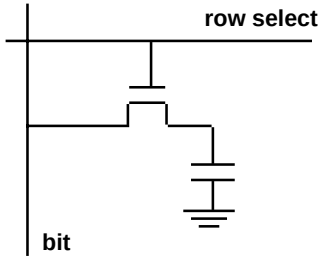
Static RAM Cell





- Write:
  1. Drive bit lines (bit=1, bit=0)
  2.. Select row
- Read:
  1. Precharge bit and bit to Vdd
  2.. Select row
  3. Cell pulls one line low
  4. Sense amp on column detects difference between bit and

Transistor Memory Cell (DRAM)



- Write:
  - 1. Drive bit line
  - 2.. Select row
- Read:
  - 1. Precharge bit line to Vdd
  - 2.. Select row
  - 3. Cell and bit line share charges
    - Very small voltage changes on the bit line
  - 4. Sense (fancy sense amp)
    - Can detect changes of ~1 million electrons
  - 5. Write: restore the value
- Refresh
  - 1. Just do a dummy read to every cell.

DRAMs over Time

	DRAM Generation					
1st Gen. Sample	**'84**	**'87**	**'90**	**'93**	**'96**	**'99**
Memory Size	**1 Mb**	**4 Mb**	**16 Mb**	**64 Mb**	**256 Mb**	**1 Gb**
Die Size (mm^2)	**55**	**85**	**130**	**200**	**300**	**450**
Memory Area (mm^2)	**30**	**47**	**72**	**110**	**165**	**250**
Memory Cell Area (μm^2)	**28.84**	**11.1**	**4.26**	**1.64**	**0.61**	**0.23**

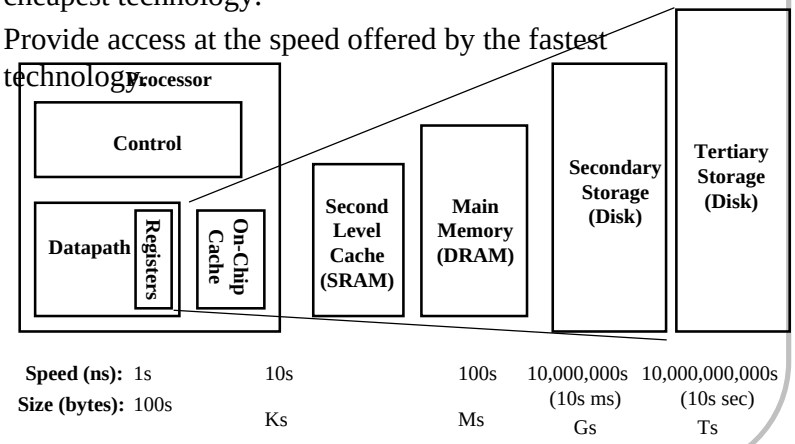
(from Kazuhiro Sakashita, Mitsubishi)

Medições em Caches

- *Hit rate*: fração achada no nível
  - Tão alta que as vezes usamos *Miss rate*
- Tempo médio de acesso à memória = *Hit time* + *Miss rate* x *Miss penalty* (ns ou clocks)
- *Miss penalty*: tempo requerido para trocar um bloco vindo de um nível mais baixo, incluindo tempo até carregar dado na CPU
  - *tempo de acesso*: tempo para nível mais baixo = *f*(latência do nível mais baixo)
  - *tempo de transferência*: tempo para transferir bloco = *f*(BW nível + alto e + baixo, tamanho do bloco)

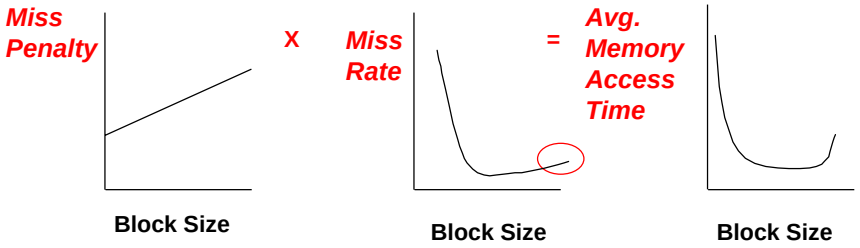
Memory Hierarchy

- By taking advantage of the principle of locality:
  - Present the user with as much memory as is available in the cheapest technology.
  - Provide access at the speed offered by the fastest technology.



Tamanho do Bloco vs. Performance de Caches

- Aumento do tamanho do bloco geralmente causa aumento do tempo de acesso médio

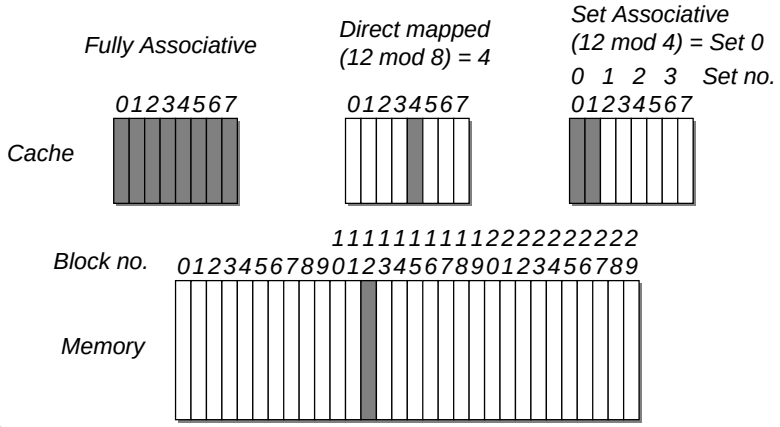


Quatro Perguntas Básicas sobre Hierarquia de Memória

- Q1: Onde o bloco vai ser colocado na memória de nível mais alto? *(Block placement)*
- Q2: Como bloco é encontrado na memória de nível mais alto? *(Block identification)*
- Q3: Quais blocos serão trocados em um miss? *(Block replacement)*
- Q4: O que acontece em uma escrita? *(Write strategy)*

Q1: Onde Bloco Deve Ser Colocado no Nível Mais Alto?

Onde bloco 12 deve ser colocado?



Q2: Como o Bloco é Encontrado no Nível Mais Alto?

- Tag para cada bloco
  - Não é necessário checar índice ou offset
- Aumento de associatividade reduz índice, aumenta tag

Block Address		Block offset
Tag	Index	

Q3: Qual Bloco Será Trocado Durante um Miss?

- Fácil de decidir para caches de mapeamento direto
- Para *Set Associative* ou *Fully Associative*:
  - Aleatório (associatividade alta)
  - LRU (associatividade baixa)

Associativity:	2-way		4-way		8-way	
Size	LRU	Random	LRU	Random	LRU	Random
16 KB	5.18%	5.69%	4.67%	5.29%	4.39%	4.96%
64 KB	1.88%	2.01%	1.54%	1.66%	1.39%	1.53%
256 KB	1.15%	1.17%	1.13%	1.13%	1.12%	1.12%

Q4: O Que Acontece Durante uma Escrita?

- *Write through*: A informação é escrita tanto para o bloco da cache quanto para a memória de nível mais baixo.
- *Write back*: A informação é escrita somente para o bloco da cache. Este bloco é escrito na memória quando ele for trocado.
  - Precisa de *dirty bit*
- Vantagens e Desvantagens:
  - WT: miss de leitura não resulta em escrita na memória
  - WB: reduz BW para memória de nível mais baixo
- WT está sempre combinada com *write buffers* de forma a não precisarmos esperar pelo nível mais baixo de memória

Q4: O Que Acontece Durante uma Escrita?

- *Write allocate (fetch on write)* : bloco é trazido para cache se ocorrer um miss de escrita, seguido por uma escrita com hit.
- *No-write allocate (write around)*: bloco é modificado no nível mais baixo somente e não é carregado na cache.
- WB é geralmente utilizado com *write allocate*
- WT é geralmente utilizado com *no-write allocate*

Resumo

- Gap entre CPU e memória é um dos maiores obstáculos para a performance de sistemas de computação
- Para melhorarmos a relação custo/benefício, utilizamos o princípio da localidade
- Para qualquer nível de memória, queremos saber a resposta para os 4 Qs