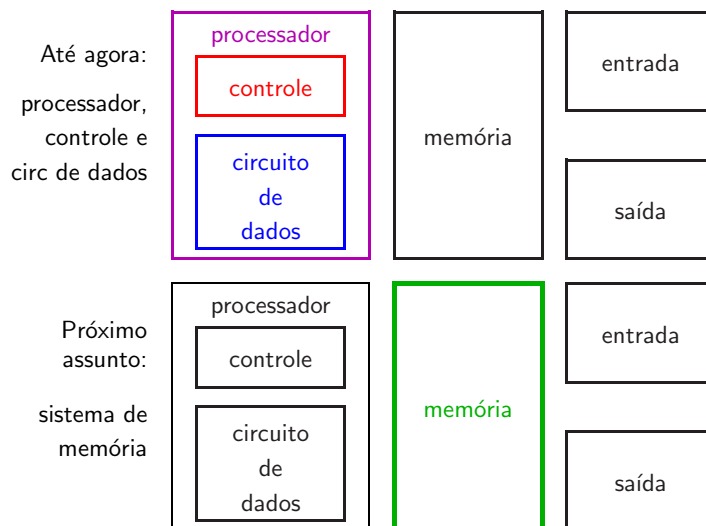
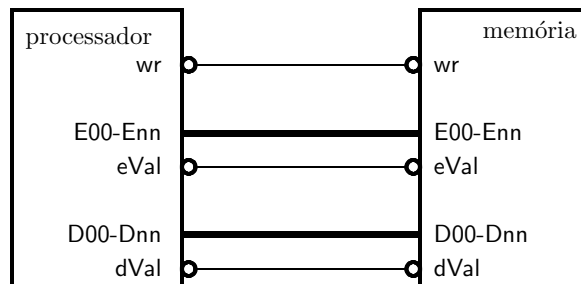


## Contexto

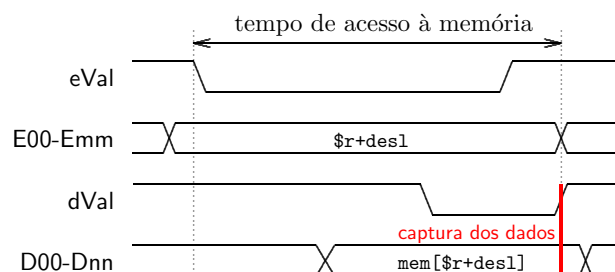
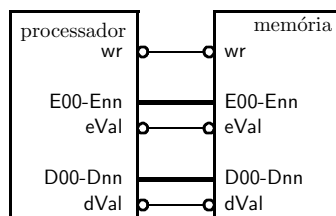


## Sistemas de Memória – interf CPU-mem

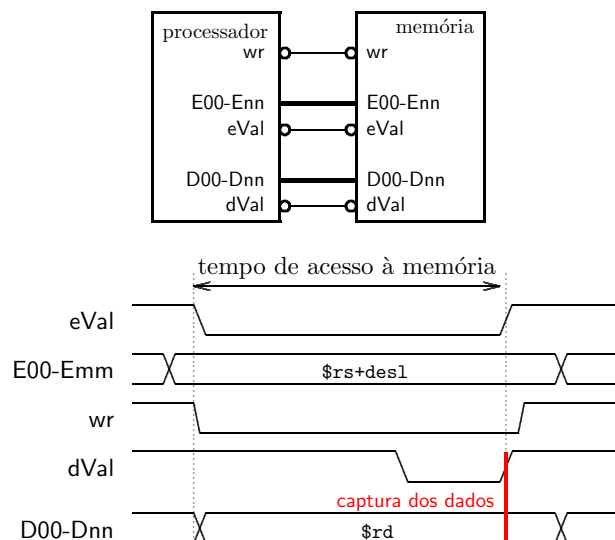


- eVal = endereço válido
- dVal = dados válidos
- wr = write

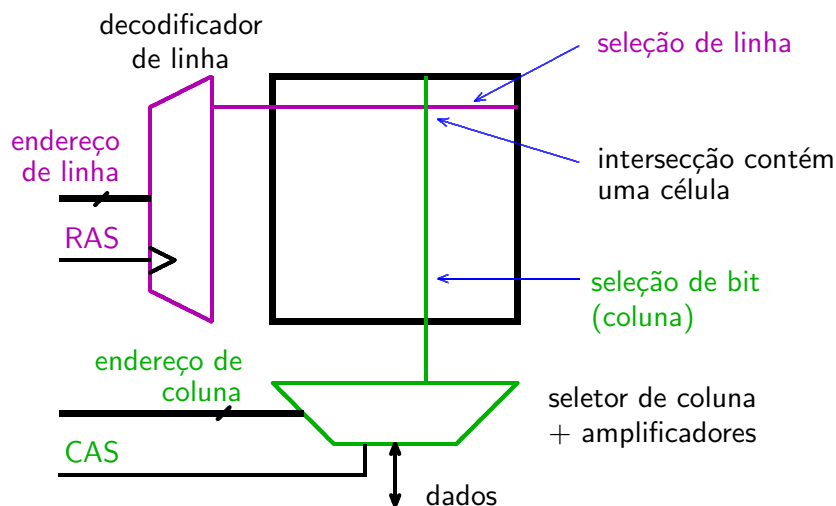
## Sistemas de Memória – leitura



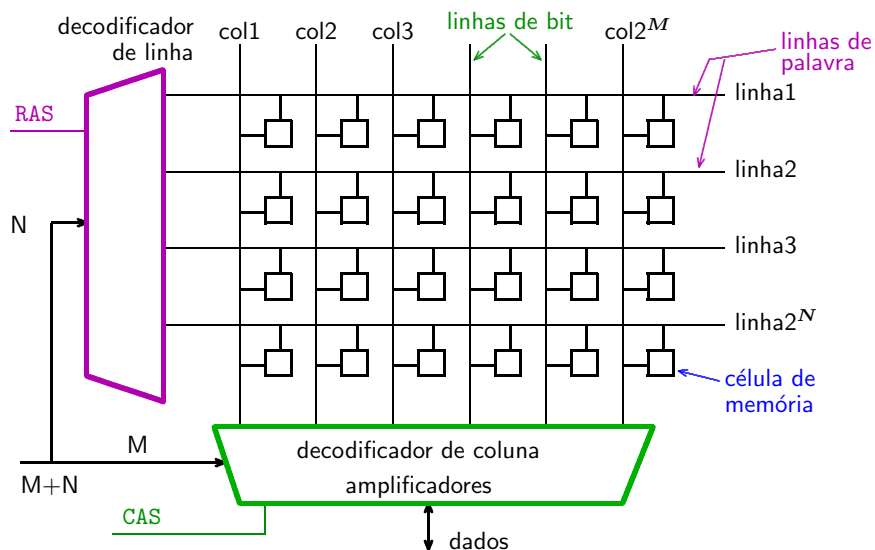
## Sistemas de Memória – escrita



## Sistemas de Memória – organização do CI



## Memória Dinâmica – matriz



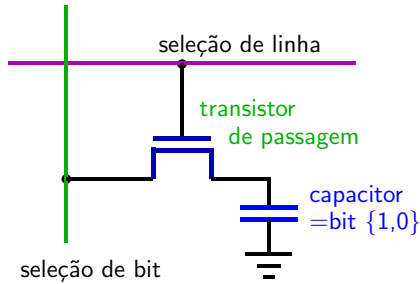
## Memória Dinâmica – célula DRAM

### Escrita:

1. ativa seleção de bit (bit line)
2. seleciona linha

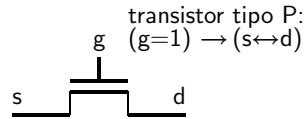
### Leitura:

1. carrega linha de bit até  $V_{dd}/2$
2. seleciona linha
3. linha de bit e capacitor dividem carga
4. amplifica diferença de voltagem  
amplificador sente diferença de  $10^6$  elétrons
5. escreve e reforça valor



### Refresh:

1. lê conteúdo de cada célula



## Memória Dinâmica – escrita/leitura

### Escrita:

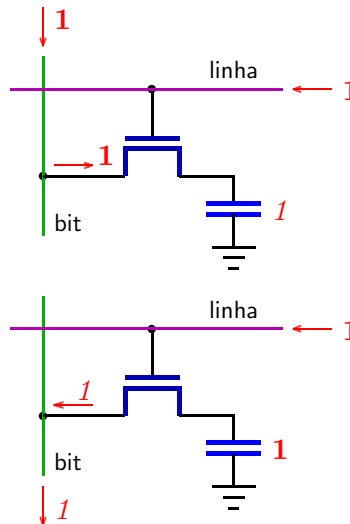
1. força valor na linha de bit
2. seleciona linha
3. capacitor mantém valor por 60ms, então refresca

### Leitura:

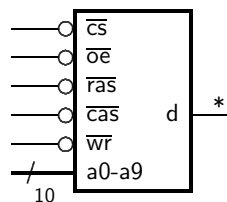
1. carrega linha de bit até  $V_{dd}/2$
2. seleciona linha
3. linha de bit e capacitor dividem carga
4. amplificador detecta valor (1/0)
5. re-escreve valor

### Refresh:

igual a leitura



## Memória Dinâmica – circuito integrado



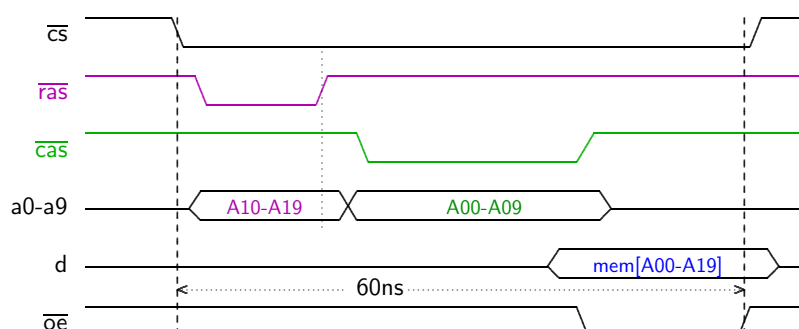
cs = chip select  
oe = output enable  
wr = write

ras = row address strobe  
cas = column address strobe

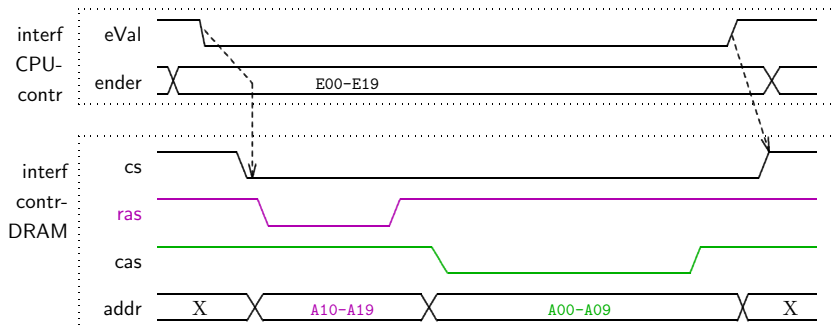
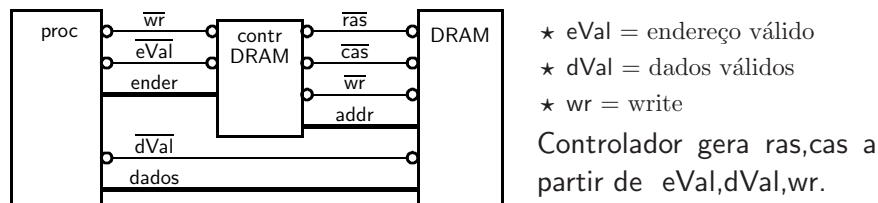
Um acesso à memória custa  $\approx 60\text{ns}$ :

$20\text{ns RAS} + 20\text{ns CAS} + 20\text{ns OE}$

60ns é otimista: acesso aleatório  $\approx 110\text{ns}$



## Sistemas de Memória — interf CPU-DRAM



## Sistemas de Memória



Em geral, o desempenho de computadores é limitado pela *latência* da memória e pela *vazão* de/para memória

**latência** é o tempo de um único acesso [s]  
tempo de acesso à memória  $\gg$  ciclo do processador

**vazão** é o número de acessos por unidade de tempo [coisa/s]

se uma fração  $m$  das instruções acessam a memória (lw/sw)  
ocorrem  $1 + m$  referências/instrução  
→ CPI=1 **se e só se** ocorrerem  $1 + m$  referências/ciclo

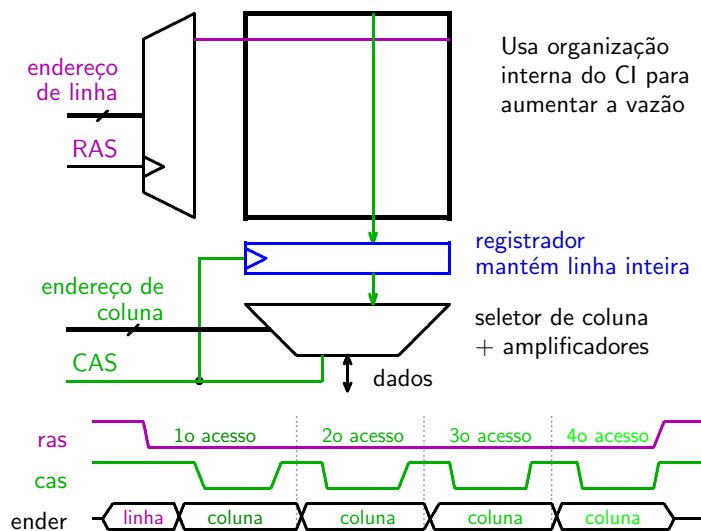
## Vazão e Latência

- **Latência** da memória é o intervalo desde a requisição **pelo** processador até a disponibilidade **para** o processador [s]
- **Vazão** é a taxa de transferência de/para a memória [req/s]  
 $bandwidth = largura\ de\ banda$  [xfer/s]
- Vazão e latência são intimamente relacionadas:  
Se  $R$  é o número de requisições que a memória pode atender simultaneamente, então

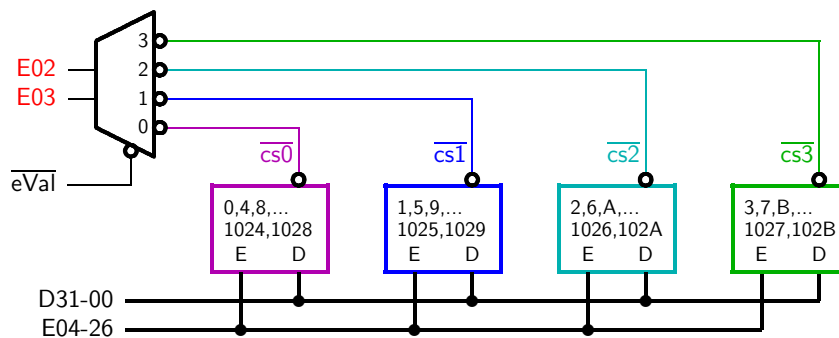
$$V = R/L \quad [req/s = req / s]$$

Vazão pode ser aumentada com dinheiro (p.ex barramento mais largo)  
Latência depende da velocidade da luz, mas pode ser escondida

## Melhorar vazão: Fast Page Mode



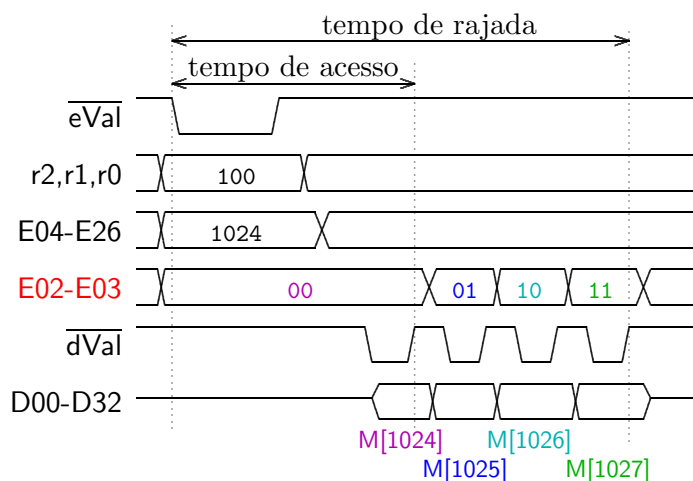
## Melhorar vazão: Memória Intercalada em Bancos



Memória intercalada (em bancos) para acessos em rajada

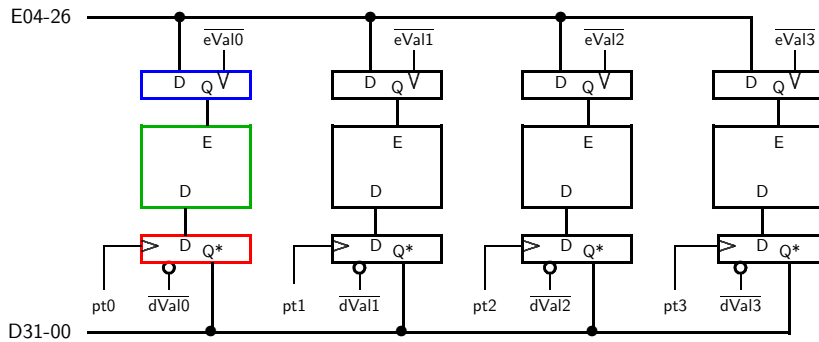
Cada **banco** mantém palavras com endereços separados de **númBancos**  
acesso inicia simultaneamente em todas as memórias  
mas transferências são em rajada

## Melhorar vazão: Memória Intercalada em Bancos



Aumenta a vazão, mas não diminui a latência

## Melhorar vazão: Pipelining + Bancos

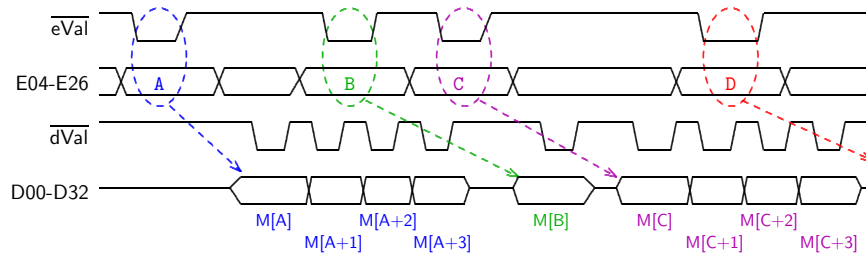


Interface com bancos de memória em pipeline de 3 estágios:

endereço acesso transferência

Após o tempo de acesso dado é armazenado no registrador de dados, e quando barramento fica livre é transferido para o processador

## Melhorar vazão: Pipelining + Bancos



Pipeline desacopla (1) endereçamento de acesso à mem e (2) acesso à mem de transferência pelo barramento

Barramento suporta várias transações concorrentes  $4 \leq \#tr \leq 8$

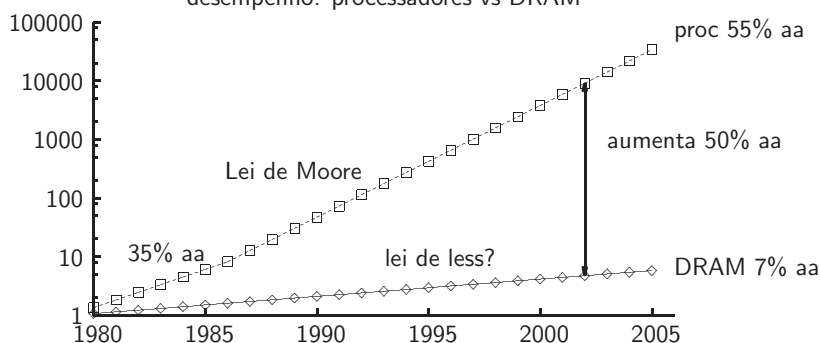
Cada transação viaja com sua etiqueta (para casar pedido com resposta)

Aumenta a vazão, mas não diminui a latência

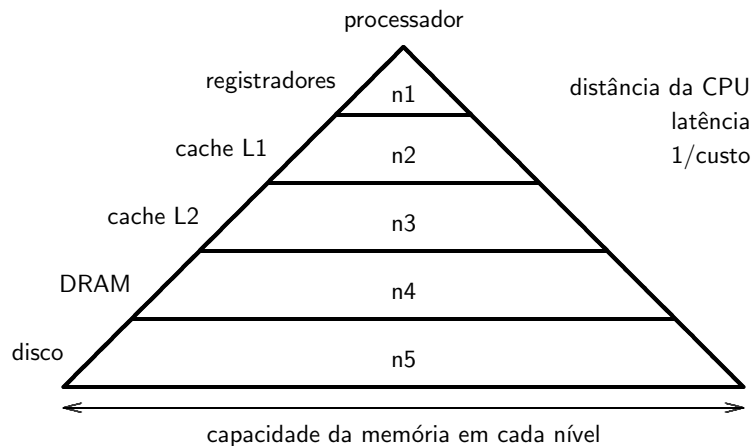
## Hierarquias de Memória

- Velocidade de processadores cresce  $\approx 50\%$  ao ano  $2x$  em  $1\frac{1}{2}$  anos
- Velocidade de memória cresce  $\approx 7\%$  ao ano  $2x$  em 10 anos
- Solução: inserir memória pequena e rápida entre CPU e DRAM.

desempenho: processadores vs DRAM



## Hierarquias de Memória

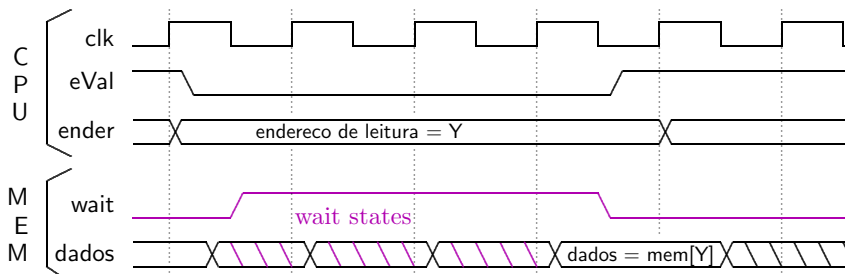
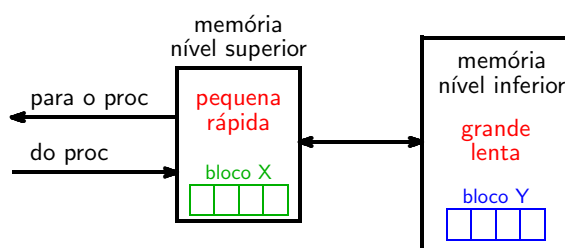


**Requisito:** sistema de memória com capacidade similar à do disco, com latência similar ao acesso a registradores

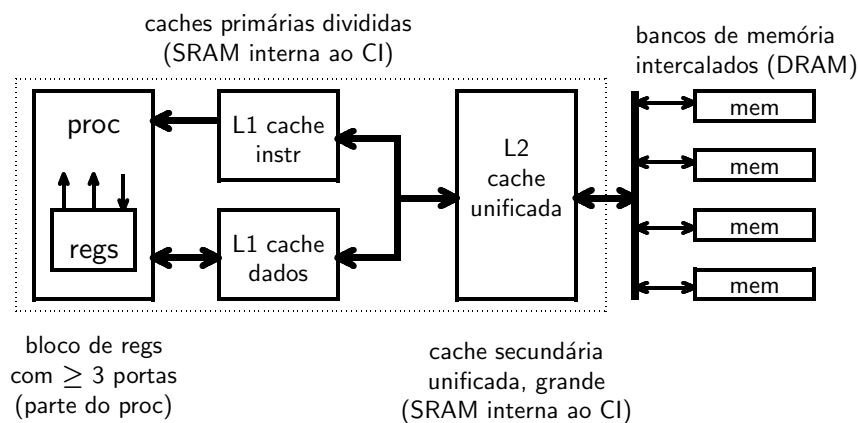
## Idéia: porta de memória com latência variável

Dados no nível superior retornam com baixa latência

Dados no nível inferior retornam com maior latência



## Sistemas de Memória Típicos



## Exemplo de Hierarquia de Memória – PowerPC 970

iMAC G5, 1.6GHz, US\$1300

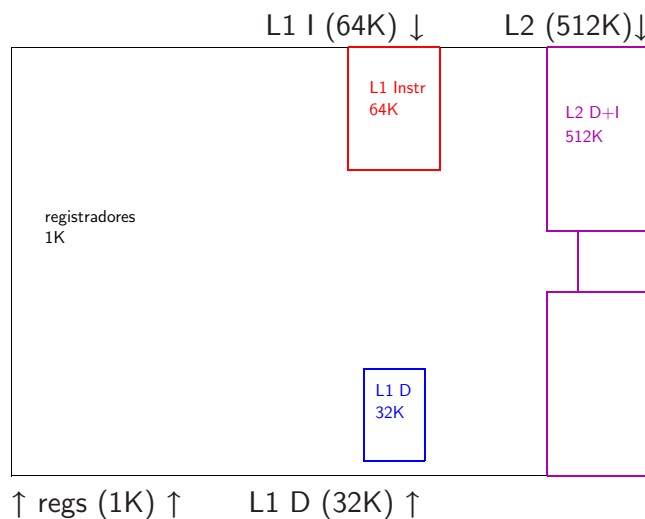
	regs	L1 I	L1 D	L2 i+d	DRAM	disco
capacid [B]	1K	64K	32K	512K	256M	80G
latência [c]	1	3	3	11	88	$10^6$
gerenciado	compil	hw	hw	hw	hw	$\alpha$

$\alpha$ : SO, hw, aplicativo

**Requisito:** sistema de memória com capacidade similar à do disco, com velocidade similar à latência do acesso aos registradores

**Objetivo de projeto:** ilusão de memória grande, rápida e barata

## Exemplo de Hierarquia de Memória – PowerPC 970



## Latência sob lente de aumento

	regs	L1 I	L1 D	L2 i+d	DRAM	disco
capacid [B]	1K	64K	32K	512K	256M	80G
latência [c]	1	3	3	11	88	$10^6$
latência [s]	0.6n	1.9n	1.9n	6.9n	55n	12.5m
1/lat [Hz]	1.6G	533M	533M	145M	18M	80

O que fazer quanto a latência:

### 1) Paralelismo

requisita dados a N memórias de 1 bit ao mesmo tempo.

Acesso simultâneo aos N bits. Vazão N vezes maior.

### 2) Segmentação da memória

Se memória tem latência de N ciclos,

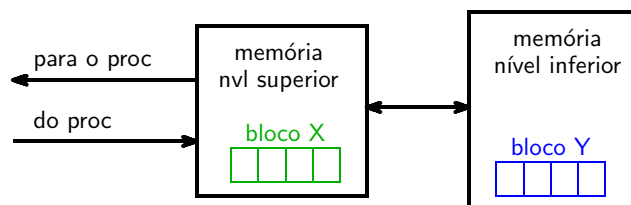
emite uma requisição por ciclo e recebe resposta N ciclos depois.



## Algoritmo para reduzir latência

**Localidade temporal:** mantenha os dados acessados recentemente perto do processador  
escolhe o que expurgar da cache

acessos à variáveis locais na pilha, índices de laços



**Localidade espacial:** mova blocos contíguos do espaço de endereçamento para os níveis superiores  
escolhe o que trazer para a cache

instruções, elementos de vetor ou matriz

## Terminologia

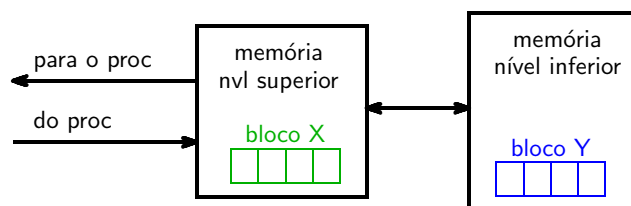
**Acerto:** dado está no nível superior (bloco X)

*hit*

**Taxa de acertos:** fração das referências encontradas no nível superior

**Tempo de acerto:** tempo para acessar nível superior

inclui verificação de acerto/falta



**Falta:** é necessário buscar dado do nível inferior (bloco Y)

*miss*

**Taxa de faltas:**  $1 - \text{TaxaDeAcertos}$

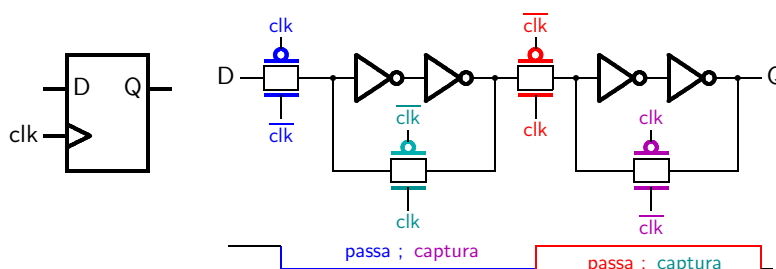
**Penalidade por falta:** tempo para carregar bloco no nível superior, e entregar para CPU

$\text{TempoDeAcerto} \ll \text{PenalidadePorFalta}$

## Revisão – como armazenar um bit

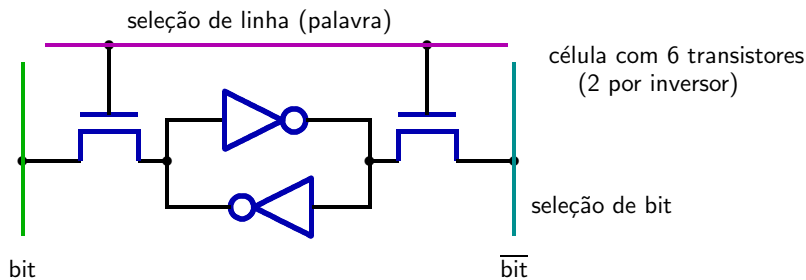


Outros elementos no circuito de memória controlam leit/escr  
Exemplo: FF-D



circuito com 16 transistores, maioria implementa semântica RD/WR

## Memória Estática – SRAM



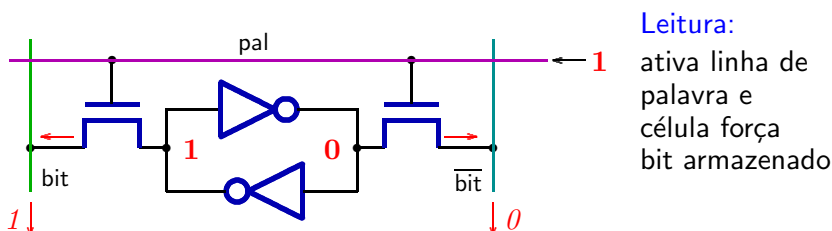
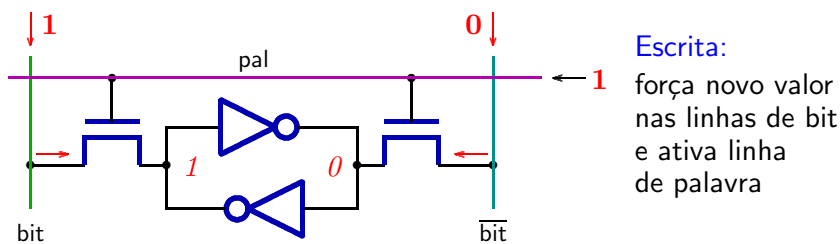
### Escrita:

1. ativa linhas de bit  
bit=1,  $\overline{\text{bit}}=0$  ( $\text{FF} \leftarrow 1$ )
2. seleciona linha/palavra

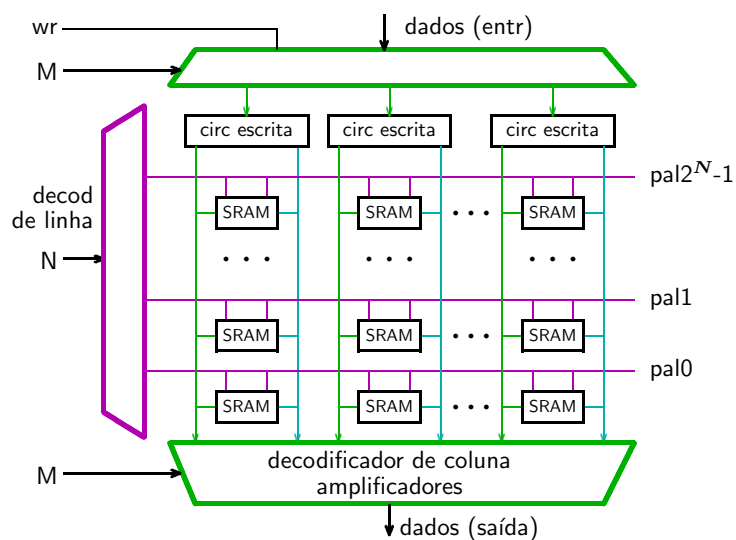
### Leitura:

1. carrega linhas de bit até  $V_{dd}$  ou  $V_{dd}/2$
2. seleciona linha/palavra
3. célula puxa uma das linhas para zero
4. amplificador detecta diferença entre bit e  $\overline{\text{bit}}$

## Memória Estática – escrita/leitura



## Memória Estática – matriz



## Memória Estática – projeto

- Arquiteto especifica número de linhas e de colunas → capacidade
- Tempo de acesso aumenta proporcionalmente ao comprimento das linhas de bit e linhas de palavra maior → mais-lento
- Quanto maior o número de linhas de E/S, maior a vazão na interface

Memórias cache são implementadas com memória estática  
mais transistores por célula → menor densidade  
acesso mais rápido porque não multiplexa linhas de endereço

## Resumo

- Memória dinâmica – matriz acessada em 2 fases (linha-coluna)
  - ★ bit armazenado em capacitor que “esquece” seu valor (refresh)
  - ★ pode aproveitar estrutura interna para aumentar vazão: fast page mode, bancos
  - ★ latência é sempre grande por conta do tamanho das estruturas
- Memória estática – acesso em uma fase
  - ★ bit armazenado em latch – menor densidade, maior velocidade
  - ★ usada para implementar memórias rápidas (cache)
- Memória cache – memória pequena e rápida entre CPU e DRAM
  - ★ localidade temporal – escolhe o que expurgar da cache
  - ★ localidade espacial – escolhe o que carregar na cache
  - ★ taxa de acerto, tempo de acerto, penalidade por falta