Organização de Computadores I
DCC006

Aula 11 – Memória

Prof. Omar Paranaiba Vilela Neto

UFMG - ICEx
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

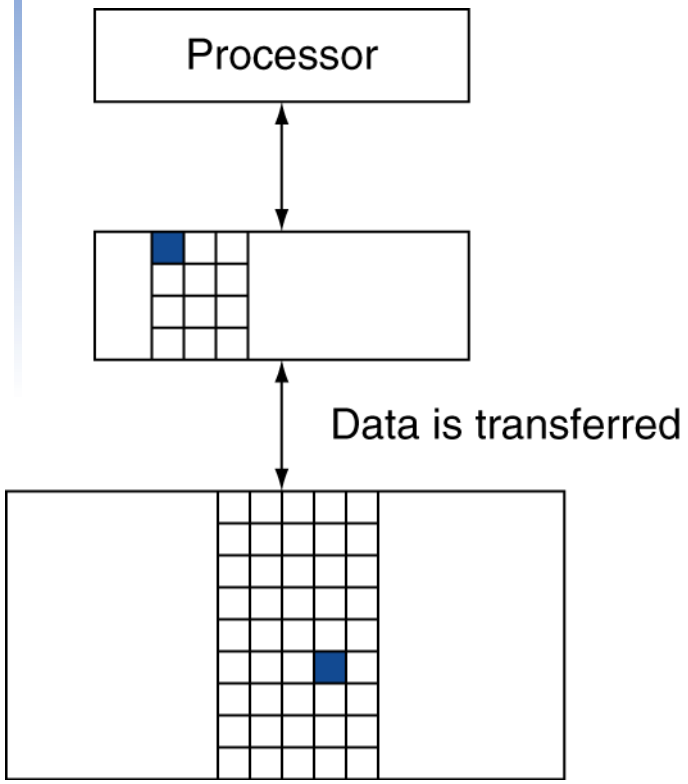
Princípio de Localidade

- Programas acessam uma pequena proporção do seu espaço de endereçamento por vez
- Localidade Temporal
  - Itens acessados recentemente provavelmente serão acessados em breve novamente
  - ex., instruções em loop; variáveis
- Localidade Espacial
  - Itens próximos aos acessados recentemente provavelmente serão acessados em breve
  - ex., acesso a instruções sequenciais; arrays

Aproveitando a Localidade

- Hierarquia de memória
- Guarde tudo no disco
- Copia dados acessado recentemente (e próximos) em uma memória DRAM menor
  - Memória pricipal
- Copia dados acessado recentemente (e próximos) em uma cache SRAM menor
  - Memória cache implementada na CPU

Níveis da Hierarquia de Memória



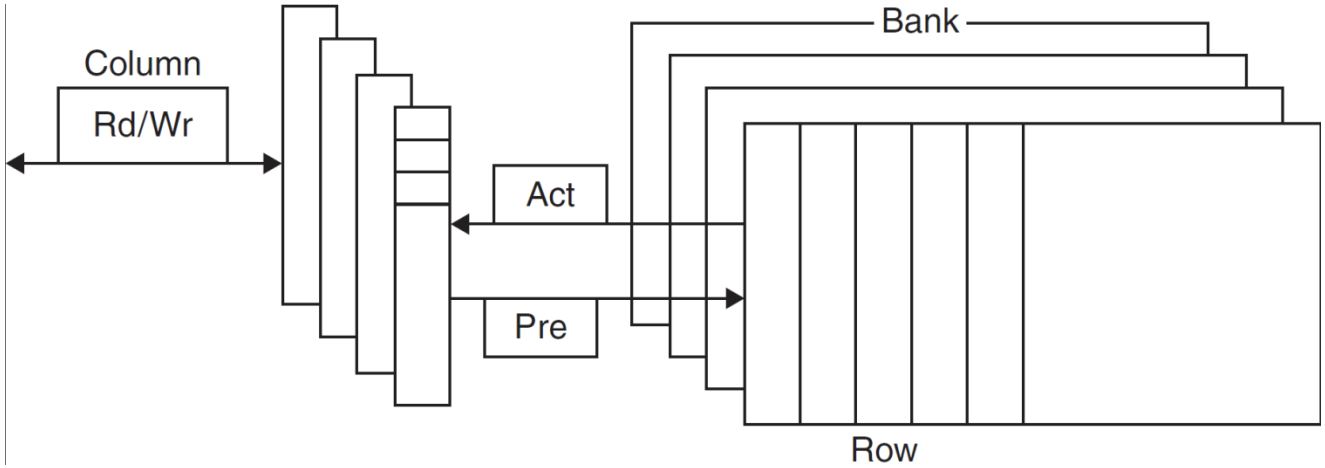
- Bloco(linha): unidade copiada
  - Pode ter múltiplas palavras
- Se dado acessado está no nível mais alto
  - Hit: satisfeito
    - Taxa de acerto: hits/acessos
- Se dado está ausente
  - Miss: copia bloco do nível mais baixo
    - Tempo necessário: miss penalty
    - Taxa de falha: misses/acesso = 1 – taxa de acerto
  - Em seguida acessa dado fornecido

Tecnologia de Memória

- Static RAM (SRAM)
  - 0.5ns – 2.5ns, $2000 – $5000 por GB
- Dynamic RAM (DRAM)
  - 50ns – 70ns, $20 – $75 por GB
- Magnetic disk
  - 5ms – 20ms, $0.20 – $2 por GB
- Memória Ideal
  - Tempo de acesso da SRAM
  - Capacidade e custo do disco

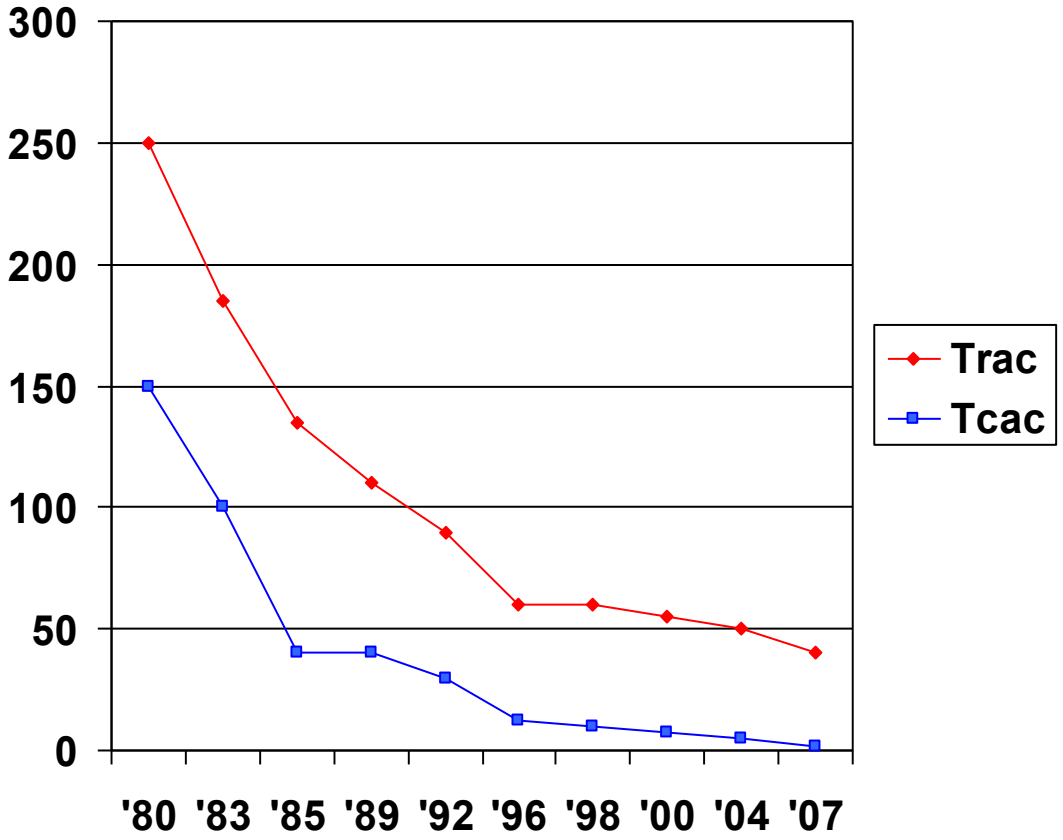
Tecnologia DRAM

- Dado armazenado como uma carga no capacitor
  - Um transistor para acessar a carga
  - Deve refrescar periodicamente
    - Lê e escreve novamente
    - Executa em uma linha DRAM



Gerações de DRAM

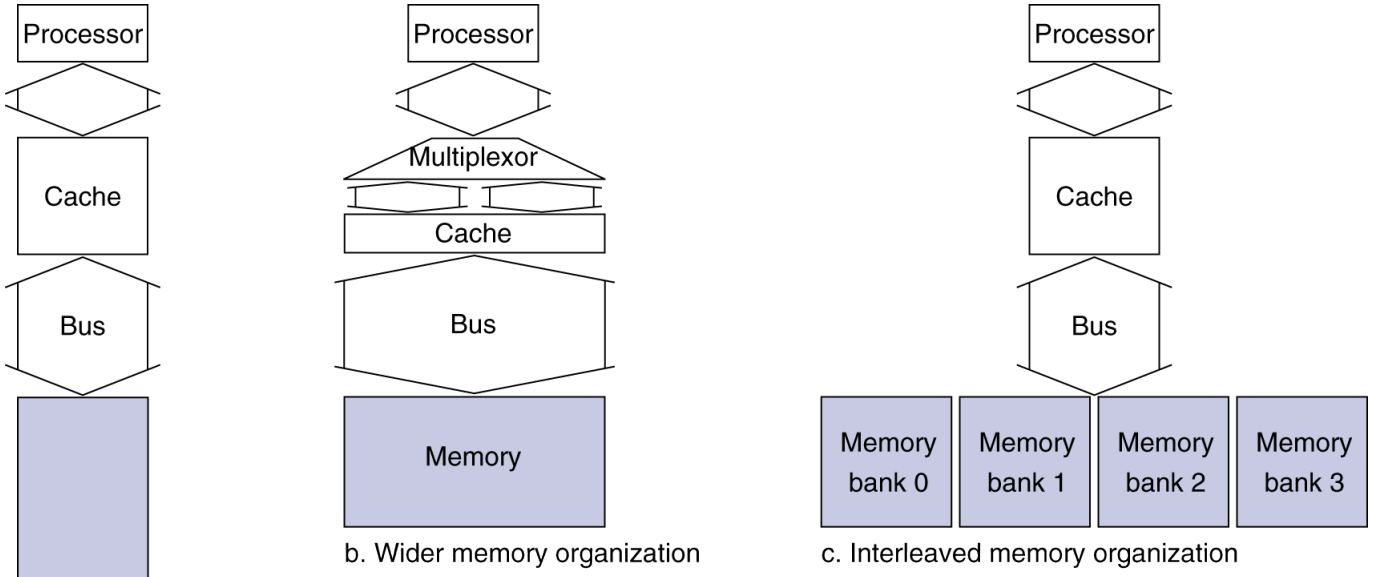
Ano	Capacidade	$/GB
1980	64Kbit	$1500000
1983	256Kbit	$500000
1985	1Mbit	$200000
1989	4Mbit	$50000
1992	16Mbit	$15000
1996	64Mbit	$10000
1998	128Mbit	$4000
2000	256Mbit	$1000
2004	512Mbit	$250
2007	1Gbit	$50



Desempenho de DRAM

- Row buffer
  - Permite que várias palavras sejam lidas e atualizadas em paralelo
- DRAM Síncrona
  - Permite acesso consecutivo em conjunto sem ser necessário enviar cada endereço
  - Largura de banda melhor
- DRAM banking
  - Perimite acesso simultâneo em múltiplas DRAMs
  - Largura de banda melhor

Melhorando largura de banda



a. One-word-wide memory organization

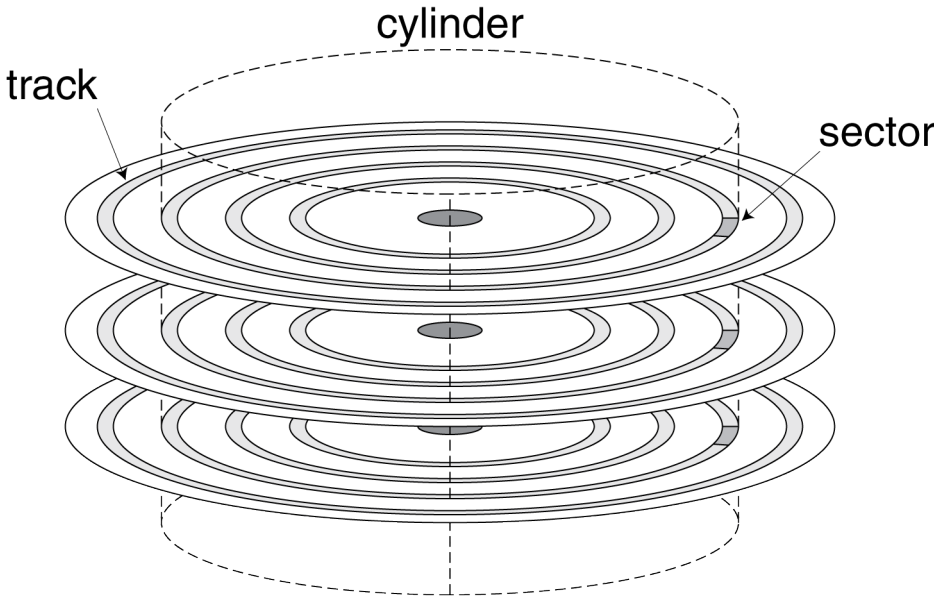
b. Wider memory organization

c. Interleaved memory organization

- 4-word wide memory
  - Miss penalty = 1 + 15 + 1 = 17 bus cycles
  - Bandwidth = 16 bytes / 17 cycles = 0.94 B/cycle
- 4-bank interleaved memory
  - Miss penalty = 1 + 15 + 4×1 = 20 bus cycles
  - Bandwidth = 16 bytes / 20 cycles = 0.8 B/cycle

Disco de armazenamento

- Não volátil, magnético rotacional



Setores do Disco e Acesso

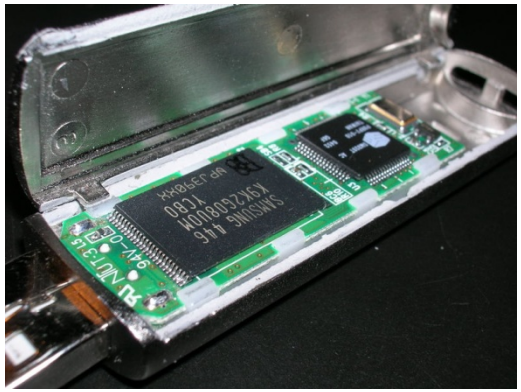
- Cada setor armazena
  - ID do setor
  - Data
  - Código de correção de erro (ECC)
    - Usado para esconder defeitos e armazenar erros
  - Campos de sincronização e gaps
- Acesso a um setor envolve
  - Atraso de fila, se outro acesso pendente
  - Busca: move a cabeça
  - Latência de rotação
  - Transferência de dados
  - Custo de controle

Questões de desempenho do disco

- Fabricantes dizem tempo média de busca
  - Baseado em todas as possíveis buscas
  - Localidade de agendamento do SO reduzem esse tempo
- Controlador inteligente aloca setor físicos no disco
- Drives de disco incluem cache
  - Antecipa acessos
  - Evitam busca e rotações desnecessárias

Armazenamento em Flash

- Armazenamento não volátil com semicondutor
  - 100× – 1000× mais rápido que disco
  - Menos, energeticamente favorável, robusta
  - Porem mais cara (entre disco e DRAM)

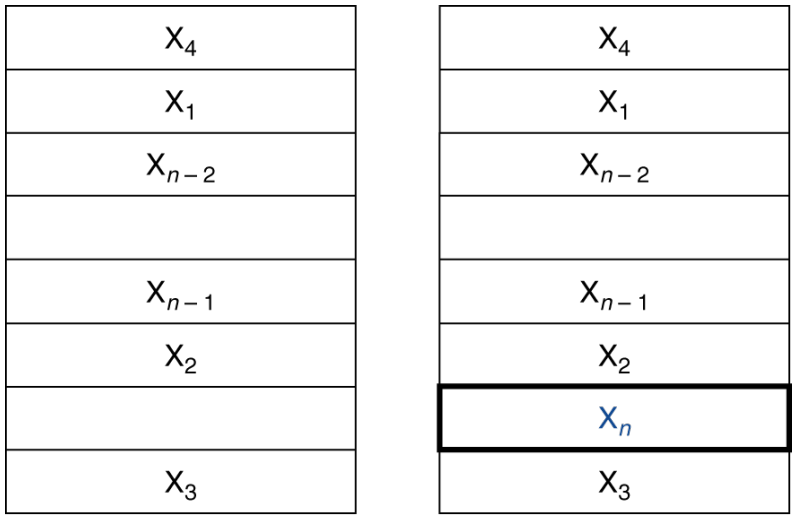


Tipos de Flash

- NOR flash: bit cell como uma porta NOR
  - Usado para memória de instruções em sistemas embarcados
- NAND flash: bit cell como uma NAND
  - Densa (bits/area), mas bloqueia em acesso
  - Mais barata por GB
  - Usado em USB e mídia de armazenamento
- Bits desgastam depois de 1000 acessos
  - Não é útil para substituir RAM ou disco

Memória Cache

- Nível da hierarquia de memória mais próxima da CPU
- Dado acessos a X_1, …, X_{n-1}, X_n

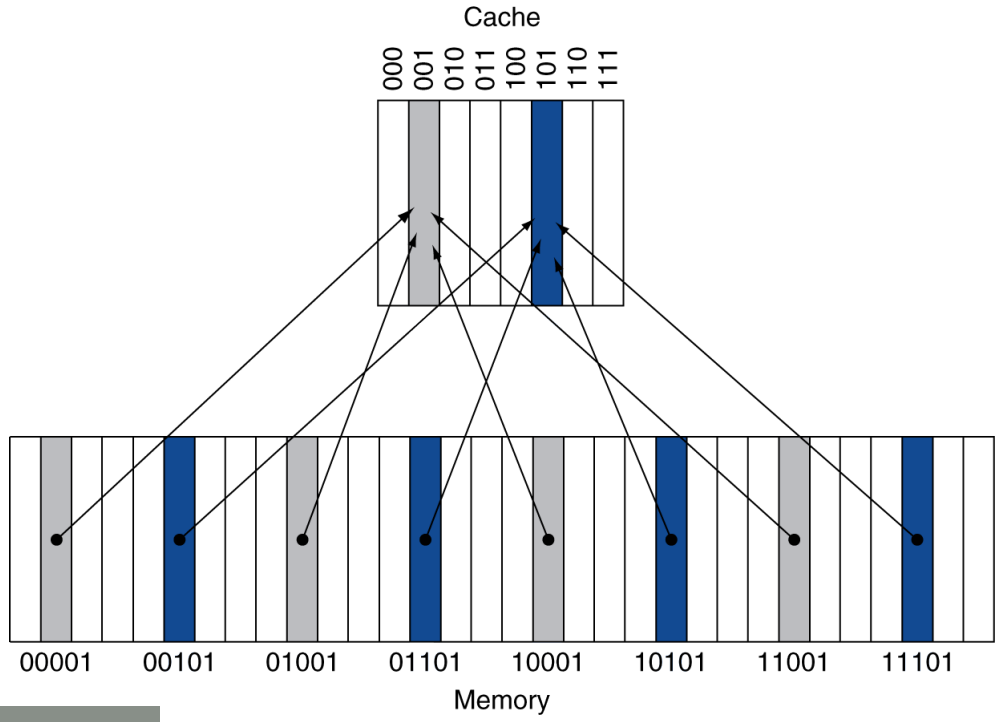


a. Before the reference to X_n b. After the reference to X_n

- Como sabemos se dado está presente?
- Onde encontramos o dado?

Cache de Mapeamento Direto

- Localização determinada pelo endereço
- Diretamente mapeada: só há uma opção
  - (Endereço do bloco) mod (#Blocos na cache)



- #Blocos é uma potência de 2
- Usa os bits menos significativos como endereço

Tags e Bits válidos

- Como saber se um bloco está armazenado na cache?
  - Armazene o endereço do bloco tal como o dado
    - Só os bits mais significativos do endereço
    - Chamado de **tag**
- E se não tiver nenhum dado no local?
  - Bit Valido: 1 = presente, 0 = não presente
  - Iniciado como 0

Cache Exemplo

- 8-blocos, 1 palavra/bloco, mapeamento direto
- Estado inicial

Index	V	Tag	Data
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	N		
111	N		

Cache Exemplo

Word addr	Binary addr	Hit/miss	Cache block
22	10 110	Miss	110

Index	V	Tag	Data
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	N		
111	N		

Cache Exemplo

Word addr	Binary addr	Hit/miss	Cache block
22	10 110	Miss	110

Index	V	Tag	Data
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Cache Exemplo

Word addr	Binary addr	Hit/miss	Cache block
26	11 010	Miss	010

Index	V	Tag	Data
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Cache Exemplo

Word addr	Binary addr	Hit/miss	Cache block
26	11 010	Miss	010

Index	V	Tag	Data
000	N		
001	N		
010	**Y**	**11**	**Mem[11010]**
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Cache Exemplo

Word addr	Binary addr	Hit/miss	Cache block
22	10 110	Hit	110
26	11 010	Hit	010

Index	V	Tag	Data
000	N		
001	N		
010	Y	11	Mem[11010]
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Cache Exemplo

Word addr	Binary addr	Hit/miss	Cache block
16	10 000	Miss	000

Index	V	Tag	Data
000	N		
001	N		
010	Y	11	Mem[11010]
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Cache Exemplo

Word addr	Binary addr	Hit/miss	Cache block
16	10 000	Miss	000

Index	V	Tag	Data
000	Y	10	Mem[10000]
001	N		
010	Y	11	Mem[11010]
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Cache Exemplo

Word addr	Binary addr	Hit/miss	Cache block
16	10 000	Miss	000
3	00 011	Miss	011

Index	V	Tag	Data
000	Y	10	Mem[10000]
001	N		
010	Y	11	Mem[11010]
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Cache Exemplo

Word addr	Binary addr	Hit/miss	Cache block
16	10 000	Miss	000
3	00 011	Miss	011

Index	V	Tag	Data
000	Y	10	Mem[10000]
001	N		
010	Y	11	Mem[11010]
011	Y	00	Mem[00011]
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Cache Exemplo

Word addr	Binary addr	Hit/miss	Cache block
16	10 000	Miss	000
3	00 011	Miss	011
16	10 000	Hit	000

Index	V	Tag	Data
000	Y	10	Mem[10000]
001	N		
010	Y	11	Mem[11010]
011	Y	00	Mem[00011]
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Cache Exemplo

Word addr	Binary addr	Hit/miss	Cache block
18	10 010	Miss	010

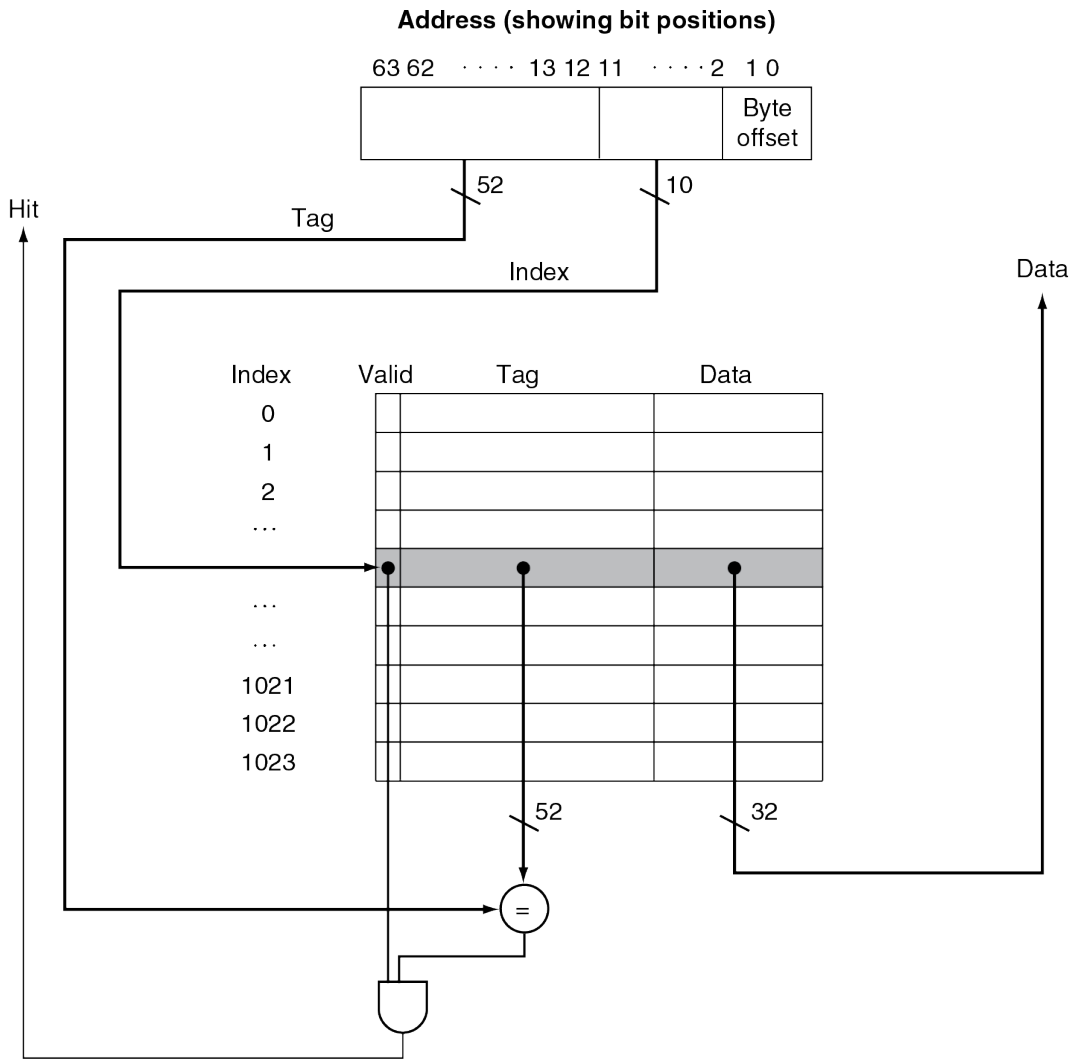
Index	V	Tag	Data
000	Y	10	Mem[10000]
001	N		
010	Y	11	Mem[11010]
011	Y	00	Mem[00011]
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Cache Exemplo

Word addr	Binary addr	Hit/miss	Cache block
18	10 010	Miss	010

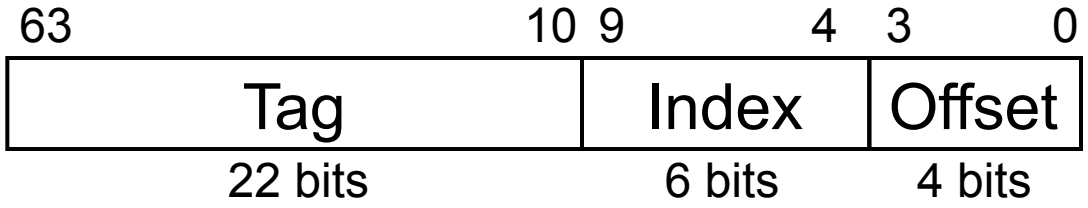
Index	V	Tag	Data
000	Y	10	Mem[10000]
001	N		
010	**Y**	**10**	**Mem[10010]**
011	Y	00	Mem[00011]
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Endereçamento



Exemplo: Tamanho do bloco

- 64 blocos, 16 bytes/bloco
  - Para qual bloco o endereço 1200 mapeia?
- Bloco endereço = $\lfloor 1200/16 \rfloor = 75$
- Bloco número = 75 mod 64 = 11



Considerações − tamanho do bloco

- Bloco maiores devem reduzis taxa de falha
  - Devido à localidade espacial
- Mas em uma cache de tamanho fixo
  - Blocos maiores ⇒ menos blocos
    - Mais competição ⇒ aumenta taxa de falhas
- miss penalty alto
  - Cancela benefício de taxa de falha pequena
  - Early restart and critical-word-first ajudam

Cache Misses

- Em Hit, CPU funciona normalmente
- Em falha (Miss)
  - Para o pipeline da CPU
  - Busca bloco no próximo nível
  - Miss na cache de instrução
    - Reinicia a busca
  - Miss na cache de dados
    - Espera para completar acesso ao dado