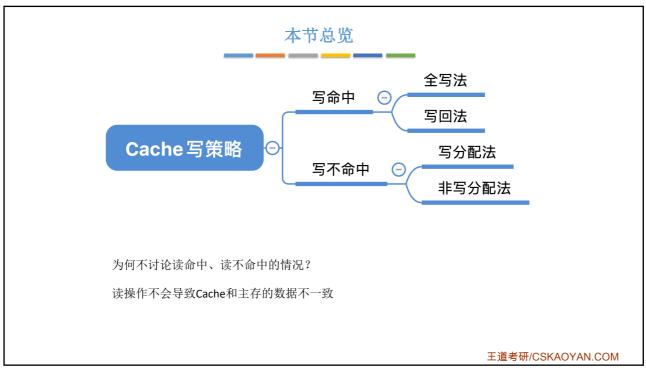
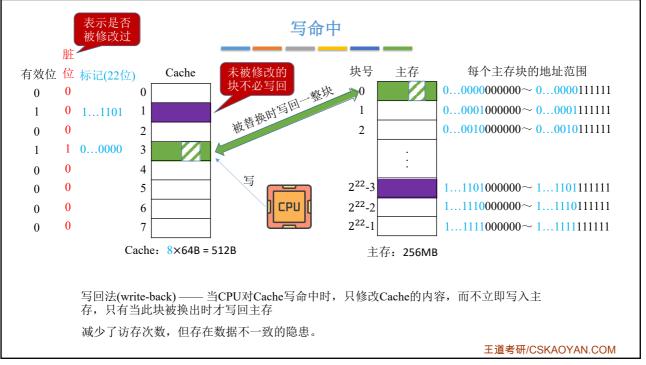
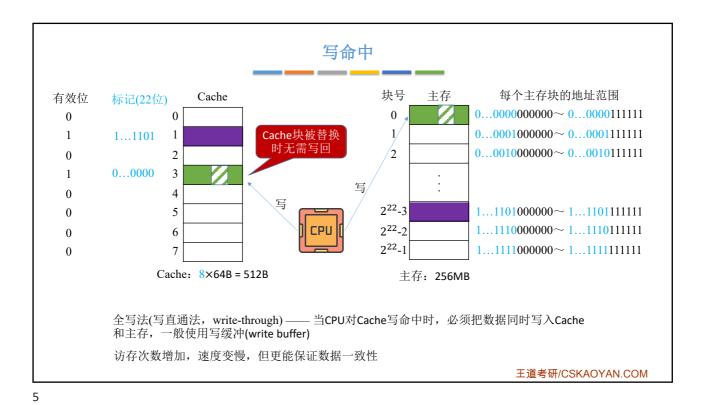
Cache 写策略

有待解决的问题 主存块号 注意:每次被访问的主存块, Cache块号 一定会被立即调入Cache 0 1KB 0 1KB 1 1KB 1KB 1 2 1KB 主存的地址共22位: 2 1KB 3 1KB 块号 块内地址 12位 10位 6 1KB 4093 1KB 4M=2²², 1K=2¹⁰ 1KB 4094 1KB 整个主存被分为 212 = 4096 块 Cache (8KB) 4095 1KB 主存(4MB) • 如何区分 Cache 与 主存 的数据块对应关系? ——Cache和主存的映射方式 • Cache 很小,主存很大。如果Cache满了怎么办? --替换算法 • CPU修改了Cache中的数据副本,如何确保主存中数据母本的一致性? ——Cache写策略 王道考研/CSKAOYAN.COM



3





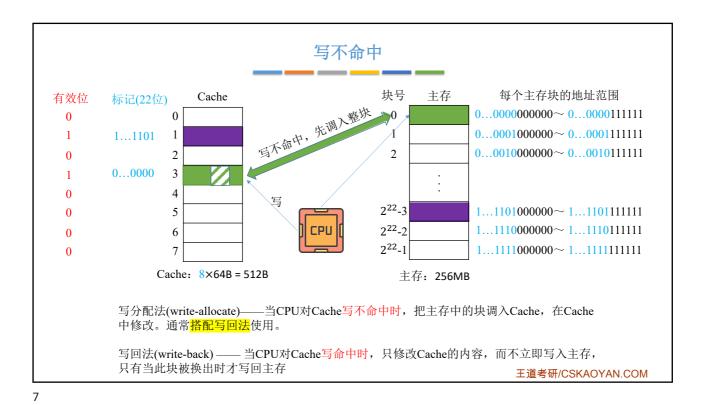
写命中 SRAM实现 的FIFO队列 写缓冲 块号 Cache 主存 每个主存块的地址范围 有效位 标记(22位) 0 0 0 $0...0001000000 \sim 0...00011111111$ 1 1 1...1101 写 $0...0010000000 \sim 0...0010111111$ 0 2 0...00003 写 4 0 2^{22} -3 5 $1...11010000000 \sim 1...11011111111$ 0 $2^{22}-2$ $1...11100000000 \sim 1...11101111111$ 6 0 $2^{22}-1$ 0 7 $1...1111000000 \sim 1...1111111111$ Cache: $8 \times 64B = 512B$ 主存: 256MB 全写法(写直通法, write-through) — — 当CPU对Cache写命中时,必须把数据同时写入Cache 和主存,一般使用写缓冲(write buffer)

6

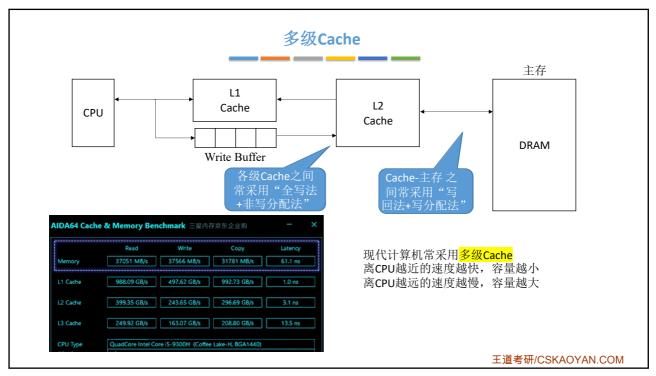
因为写缓冲饱和而发生阻塞

王道考研/CSKAOYAN.COM

使用写缓冲,CPU写的速度很快,若写操作不频繁,则效果很好。若写操作很频繁,可能会



写不命中 块号 主存 每个主存块的地址范围 Cache 有效位 标记(22位) 0 0 0 $0...0001000000 \sim 0...00011111111$ 1 1 1...1101 $0...0010000000 \sim 0...0010111111$ 2 0 2 3 0 4 0 $2^{22}-3$ 5 $1...11010000000 \sim 1...11011111111$ 0 $2^{22}-2$ $1...11100000000 \sim 1...11101111111$ 6 0 $2^{22}-1$ 0 7 $1...1111000000 \sim 1...1111111111$ 只有"读"未命 Cache: $8 \times 64B = 512B$ 主存: 256MB 非写分配法(not-write-allocate)——当CPU对Cache写不命中时只写入主存,不调入Cache。 <mark>搭配全写法</mark>使用。 — 当CPU对Cache<mark>写命中时</mark>,必须把数据同时写入Cache和 全写法(写直通法, write-through) -主存,一般使用写缓冲(write buffer) 王道考研/CSKAOYAN.COM



9

