# LECTURE - 20

# LECTURE - 20

# Topic for Today

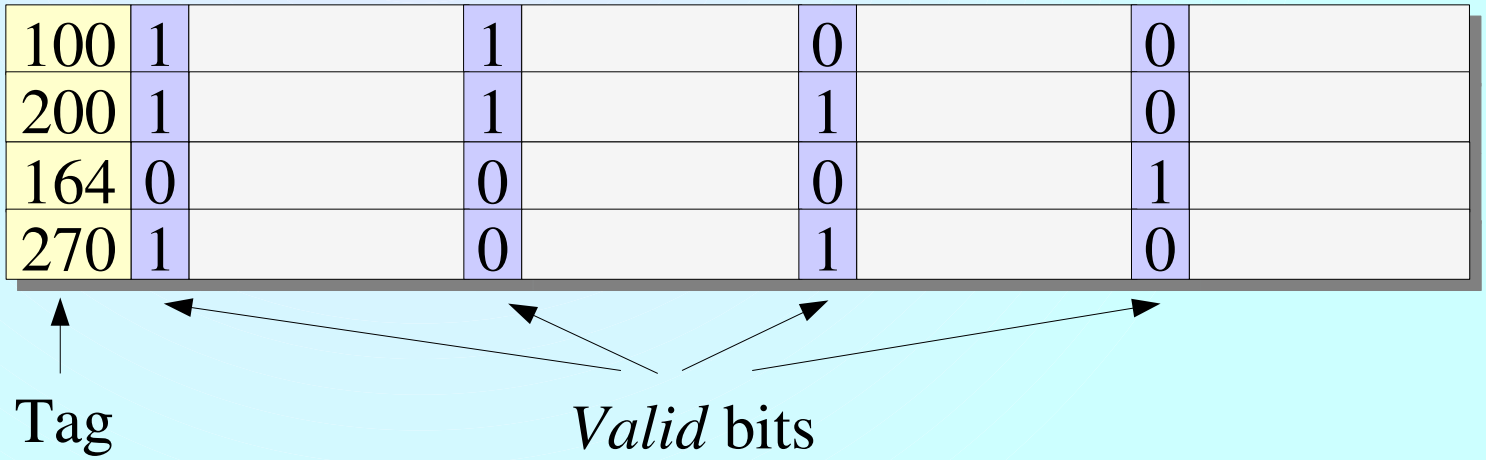
- Reducing Cache Miss Penalty
- 
- **Scribe?**

# Technique-1: Prioritize Read Misses over Writes

- Write-through cache ==> write-buffer
  - Beware of consistency
  - Example: store x, load y, load x →x and y in the same block
- Possible solution: wait for write-buffer to clear before processing any read miss
- Better (but more complex) solution: check write buffer, and process read miss first
- Write-back cache: write-back dirty block after processing read miss

# Technique-2: Sub-Block Placement

- *Sub-block:* units smaller than the full block
  - *Valid* bits added to sub-blocks
  - Only a sub-block read on cache miss




- How is this different from just using a smaller block size?
  - Tag length is reduced (good for on-chip cache)

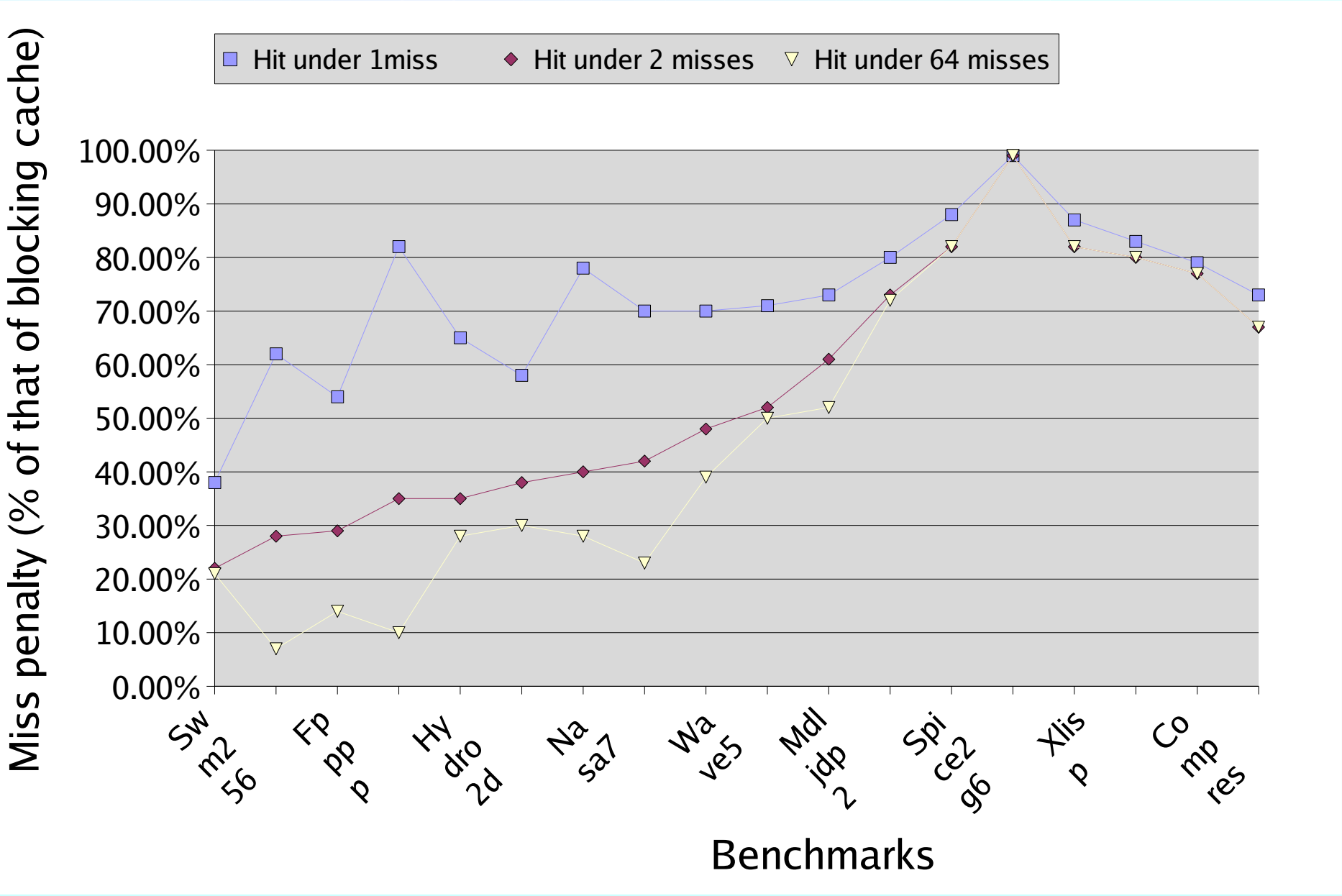
# Technique-3: Restart CPU ASAP

- *Early restart:* CPU can proceed as soon as the requested word is loaded onto cache
- *Critical word first:* The requested word is fetched first
  - A.k.a *wrapped* fetch, or *requested word first*
- These are good for caches with large blocks
- What if another access to same block, before it is fully loaded?
  - Stall if that portion of block not yet loaded

# Technique-4: Non-blocking Cache

- For OOO CPUs (e.g. Tomasulo)
  - No point in stalling the CPU on a miss
  - *Hit-under-miss* allows hits while the cache is processing a miss
  - *Hit-under-multiple-miss* can benefit more
  - *Miss-under-miss* makes sense if main memory can handle more than one request in parallel
- This significantly increases complexity of cache controller

# Non-Block Cache Performance



# Technique-5: Second-Level Caches

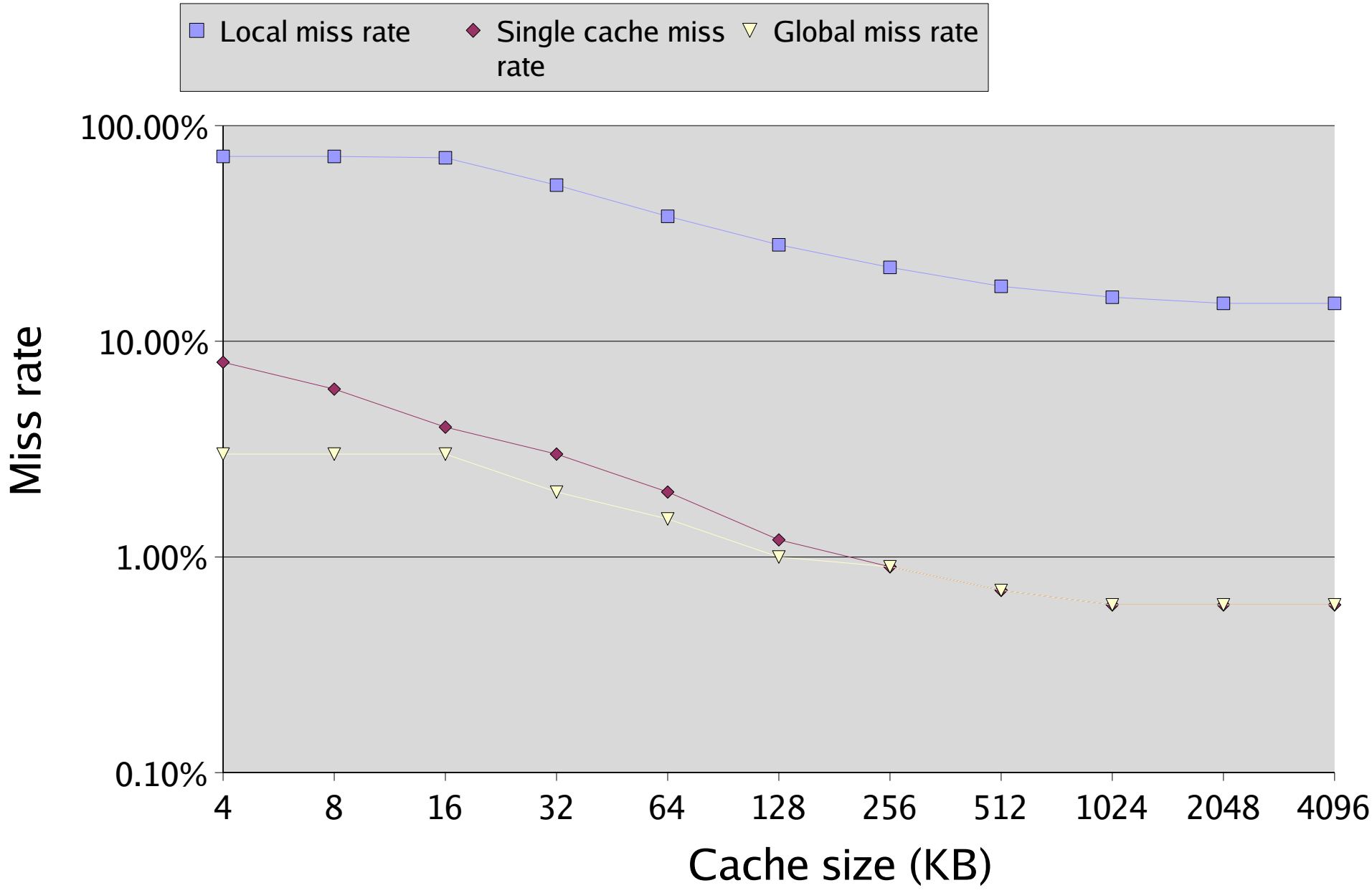
- L1 cache can be small and fast
- L2 cache can be larger, but faster than main memory

$$Avg.\ mem.\ access\ time = Hit\ time_{L1} + Miss\ rate_{L1} \times Miss\ penalty_{L1}$$

$$Miss\ penalty_{L1} = Hit\ time_{L2} + Miss\ rate_{L2} \times Miss\ penalty_{L2}$$

- *Local miss rate:* misses w.r.t. memory accesses to this cache
- *Global miss rate:* misses w.r.t. memory access by CPU

# Local and Global Miss Rates



# Second Level Cache Design

- L2 can be larger
  - Big enough to virtually eliminate capacity misses
- Higher associativity does not hurt
  - CPU clock cycle time is not affected
- Larger block size to further reduce misses
- *Multi-level inclusion property:* L2 contains all data that L1 contains
  - More work on a second-level miss