Hardware Design

Tutorial 8

Instructor: Dr. Haiyu Mao
TA: Zihao Pu

20.03.2026

Conceptual Questions

Conceptual Questions

Q1: Which of the following best describes the principle of temporal locality?
A: Programs tend to access nearby memory locations
B: Programs tend to access the same memory location multiple times within a short period
C: Programs tend to access memory in sequential order
D: Programs tend to use more stack memory than heap memory

Q2: In a direct-mapped cache, a given memory block can be placed in:
A: Any cache block
B: Exactly one cache block
C: One of two cache blocks
D: One of four cache blocks

Conceptual Questions

- **Q3: Which of the following cache organizations has the HIGHEST associativity?**
  A: Direct-mapped
  B: 2-way set associative
  C: 4-way set associative
  D: Fully associative
- **Q4: What is the primary advantage of higher associativity in a cache?**
  A: Faster hit time
  B: Lower power consumption
  C: Reduced conflict misses
  D: Simpler hardware

Conceptual Questions

- **Q5: In a cache using LRU replacement, which block is evicted when a miss occurs?**
  A: The first block that was loaded
  B: The most recently accessed block
  C: The least recently accessed block
  D: A randomly selected block

- **Q6: Which write policy updates main memory on every write operation?**
  A: Write-back
  B: Write-through
  C: Write-allocate
  D: No-write-allocate

- **Q7: What is the purpose of a "dirty bit" in a cache block?**
  A: To indicate the block contains invalid data
  B: To indicate the block has been modified and needs to be written to main memory
  C: To indicate the block is the least recently used
  D: To indicate the block contains a tag mismatch
- **Q8: Which type of cache miss occurs when the working set of a program is larger than the cache?**
  A: Compulsory miss
  B: Conflict miss
  C: Capacity miss
  D: Coherence miss

Question 1

CACHE ORGANIZATION AND ADDRESS DECOMPOSITION

Cache Design

- Consider a computer system with a 16-bit byte-addressable address space. The system has a **4-way set-associative cache** with the following specifications:Total cache size: 2 KB (2048 bytes), Block size: 32 bytes, 4-way set associative
- **(a)** Determine the number of blocks and the number of sets in the cache.
- **(b)** For a 16-bit memory address, calculate the width (in bits) of each field:
  - Byte Offset
  - Set Index
  - Tag
- **(c)** Given the memory address 0x5A3B, determine:
  - The byte offset
  - The set index
  - The tag value

Cache Design

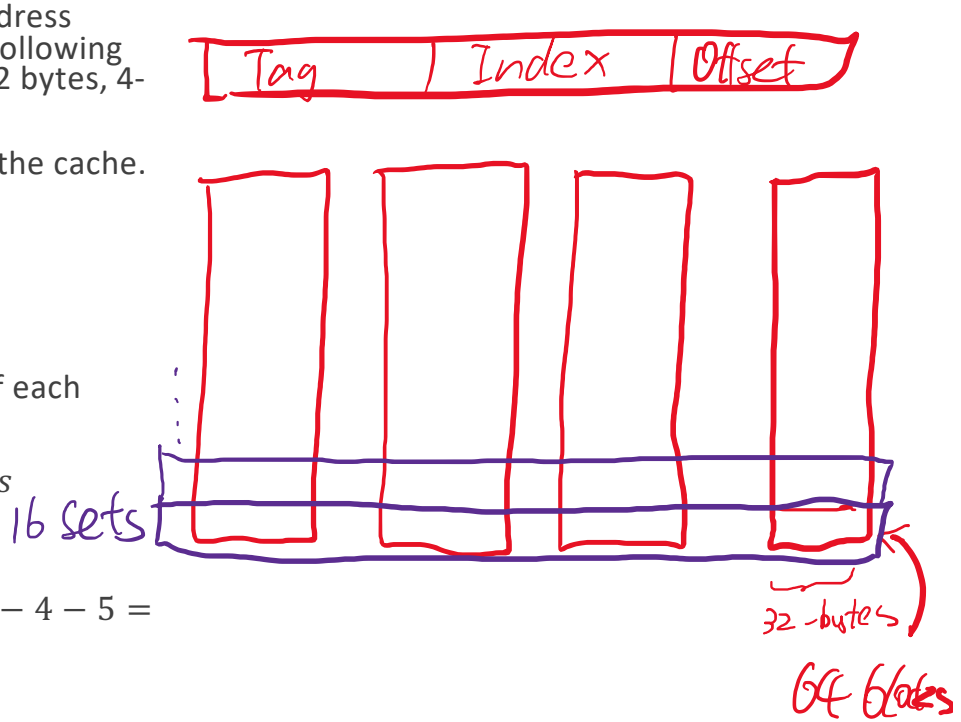
- **(d)** How many total bits are required to implement this cache if each block has 1 valid bit and 1 dirty bit?

Cache Design

- **(e)** Consider the following sequence of block addresses (these are block addresses, not byte addresses): 4, 12, 8, 4, 20, 12. Assume all these blocks map to the same set and the cache uses LRU replacement. The cache is initially empty.
  - How many hits and how many misses occur?
  - Which blocks are in the cache after this sequence?

Cache Design - Answer

- Consider a computer system with a 16-bit byte-addressable address space. The system has a **4-way set-associative cache** with the following specifications: Total cache size: 2 KB (2048 bytes), Block size: 32 bytes, 4-way set associative
- **(a)** Determine the number of blocks and the number of sets in the cache.
  - $Number\ of\ blocks = \frac{Cache\ Size}{Block\ Size} = \frac{2048}{32} = 64\ blocks$
  - $Number\ of\ sets = \frac{Number\ of\ blocks}{Associativity} = \frac{64}{4} = 16\ sets$
- **(b)** For a 16-bit memory address, calculate the width (in bits) of each field:
  - $Offset\ bits = \log_2(\text{Block SIze}) = \log_2(32) = 5\ bits$
  - $Index\ Bits = \log_2(Num_{Sets}) = \log_2(16) = 4bits$
  - $Tag\ Bits = Addr\ Bits - IndexBits - Offset\ Bits = 16 - 4 - 5 = 7\ Bits$
- **(c)** Given the memory address 0x5A3B
- 0 1 0 1 1 0 1 0 0 0 1 1 1 0 1 1
- [Tag][Set Index][ByteOffset]



Cache Design

- **(d)** How many total bits are required to implement this cache if each block has 1 valid bit and 1 dirty bit?

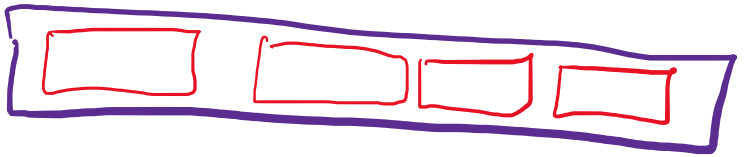
In each block

1 + 1 + 17 + 32 x 8 = 265 bits / block

Total: 265 x 64 = 16960 bits

Cache Design

- **(e)** Consider the following sequence of block addresses (these are block addresses, not byte addresses): 4, 12, 8, 4, 20, 12. Assume all these blocks map to the same set and the cache uses LRU replacement. The cache is initially empty.
  - How many hits and how many misses occur?
  - Which blocks are in the cache after this sequence?



LRU

Init: { x x x x }
Cycle 1: { x x x 4 } miss
Cycle 2: { x x 12 4 } miss
Cycle 3: { x 8 12 4 } miss
Cycle 4: { x 4 8 12 } hit
Cycle 5: { 20 4 8 12 } miss
Cycle 6: { 12 20 4 8 } hit

2 hit, 4 miss

Question 2

CACHE PERFORMANCE AND AMAT

Calculating Average Memory Access Time

- A 1GHz processor has the following memory hierarchy:
  - L1 cache: Hit time = 1 cycle, Miss rate = 5%
  - L2 cache: Hit time = 10 cycles, Miss rate = 20%
  - Main memory: Access time = 100 cycles
- **(a)** Calculate the Average Memory Access Time (AMAT) for the L1 cache in cycles.

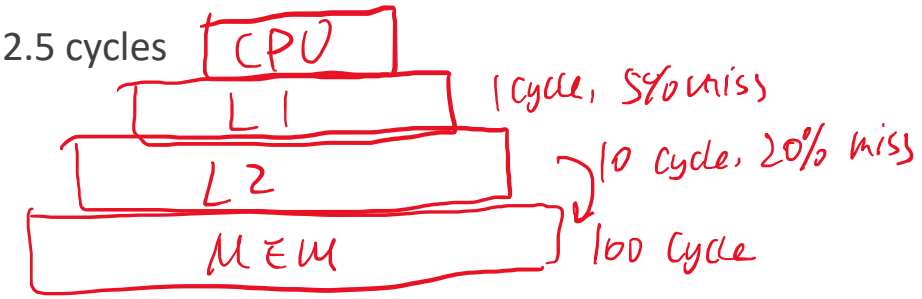
- **(b)** If the processor has a base CPI of 1.5 (assuming perfect instruction cache with zero memory access latency), and 30% of instructions are memory accesses (loads/stores), calculate the total CPI including memory stalls. Assume the base CPI does NOT account for any memory access time.

- **(c)** Now consider a different scenario. A programmer optimizes their code using blocking/tiling, which reduces the L1 miss rate from 5% to 2%. Calculate the new AMAT and the percentage improvement in AMAT.

- **(d)** Explain why reducing miss rate through blocking/tiling works. What type of misses does this optimization primarily target?

Calculating Average Memory Access Time

- **(a)** Calculate the Average Memory Access Time (AMAT) for the L1 cache in cycles.
- 2.5 cycles



$AMAT_{L1} = 1 + 5\% \times 30 = 2.5$ cycles
$AMAT_{L2} = 10 + 20\% \times 100 = 30$ cycles
AMAT: 100 cycle
→ 2.5ns

- **(b)** If the processor has a base CPI of 1.5 (assuming perfect cache with zero memory access latency), and 30% of instructions are memory accesses (loads/stores), calculate the total CPI including memory stalls. Assume the base CPI does NOT account for any memory access time.

Iron Law: $ExecTime = IC \times CT \times CPI$

$TotalExecTime = 70\% \times CT \times 1.5 + 30\% \times CT \times (2.5+1.5) = 1 \times CT \times 1.5 + 30\% \times CT \times 2.5$

$= 2.25 \times CT \times IC$

$Effective\ CPI = \frac{TotalExecTime}{IC \times CT} = 2.25$

- **(c)** Now consider a different scenario. A programmer optimizes their code using blocking/tiling, which reduces the L1 miss rate from 5% to 2%. Calculate the new AMAT and the percentage improvement in AMAT.

$AMAT_{L2} = 30$ cycles

$AMAT_{L1}^{new} = 1 + 2\% \times 30 = 1.6$ cycles $\rightarrow 1.6$ns

Improvement: $\frac{2.5 - 1.6}{2.5} = \frac{0.9}{2.5} = 36\%$

- **(d)** Explain why reducing miss rate through blocking/tiling works. What type of misses does this optimization primarily target?
- **Divides the working set:** The algorithm processes data in smaller chunks (tiles) that fit within the cache, rather than operating on the entire dataset at once.
- **Increases temporal locality:** By working on a tile completely before moving to the next, the same data is accessed multiple times while it's still in the cache.
- **Reduces capacity misses:** The primary target of blocking is **capacity misses**. When the working set is larger than the cache, data is constantly being evicted and reloaded. Blocking ensures each tile fits in the cache.