Computer Architecture

Fall, 2025
Week 14
2025.12.01

組別：__________ **簽名：**____

[group 1]

1. Qustion:

Compare the following cache types: direct-mapped cache, N-way set-associative cache, and fully associative cache.

1. Which cache type has the longest data access time, and why?

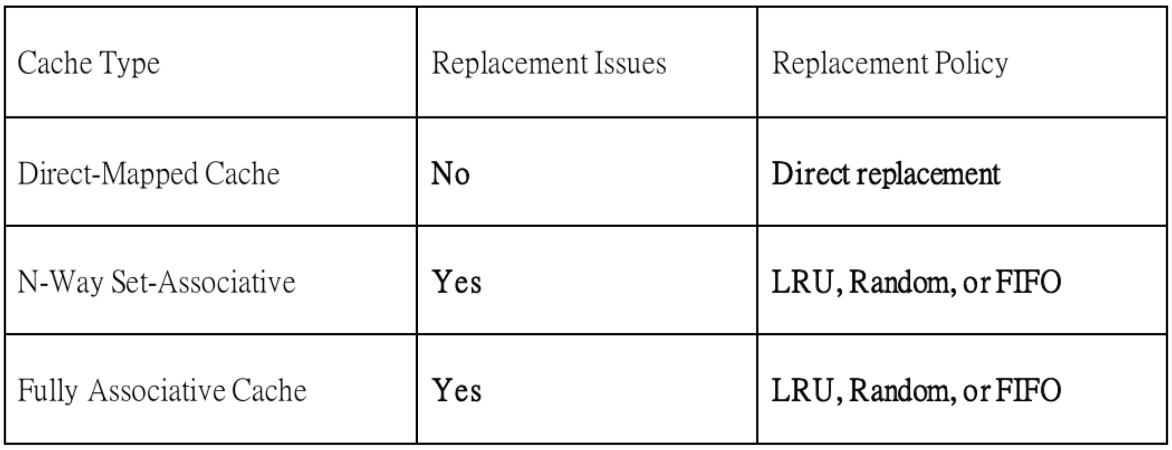
2. Which cache types encounter data replacement issues? Please suggest a replacement policy for each. Fill in the answer in the given table

Cache Type	Replacement Issues	Replacement Policy
Direct-Mapped Cache		
N-Way Set-Associative		
Fully Associative Cache		

Ans:

1. Fully Associative Cache, because the data can be in any cache line. The system must search the entire cache (using associative lookup) to find the data.

2.



Cache Type	Replacement Issues	Replacement Policy
Direct-Mapped Cache	No	**Direct replacement**
N-Way Set-Associative	Yes	**LRU, Random, or FIFO**
Fully Associative Cache	Yes	**LRU, Random, or FIFO**

[group 3]

2. Question:

True/False

Q1. In a set-associative cache, each memory block can map only to one specific set, but can be placed in any way within that set.

Q2. Increasing the associativity of a cache generally reduces conflict misses but may increase access time.

Q3. Under an inclusive cache policy, every block in L1 must also exist in L2, and evicting a block from L2 may force the eviction of the same block in L1.

Q4. In a multi-level cache system, the L1 cache is typically larger and slower than the L2 cache.

Q5. When an L1 cache miss occurs, the processor usually searches the L2 cache and main memory simultaneously to reduce latency.

Ans:

1. True

In a set-associative cache, a memory block maps to exactly one set, but may be placed in any way of that set.

2. True

Higher associativity usually reduces conflict misses but increases access time due to more comparators.

3. True

In an inclusive policy, any block in L1 must also exist in L2, and evicting from L2 may force eviction from L1.

4. False

L1 cache is typically smaller and faster, not larger and slower than L2.

5. False

On an L1 miss, the processor normally checks L2 first; main memory is accessed only if L2 also misses.

[group 4]

3. Question:

Please list at least three ways to improve cache performance.

Ans:

1. Reduce the time to hit in the cache.
2. decreasing the miss ratio.
3. decreasing the miss penalty.

[group 5]

4. Question:

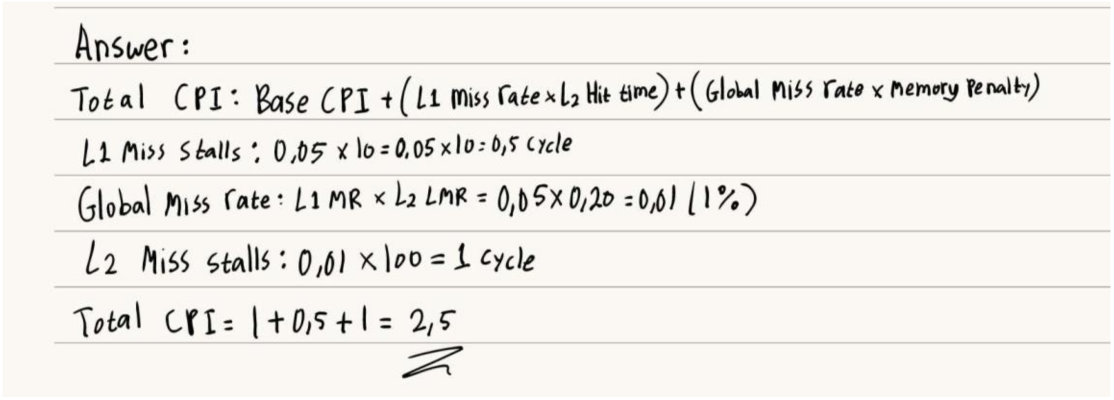
A CPU has a Base CPI of 1.0 (assuming ideal memory). You are evaluating the performance of a system with two levels of cache (L1 and L2) versus Main Memory.

- L1 Miss Rate: 5% (0.05)
- L2 Hit Time: 10 cycles
- L2 Miss Rate (Local): 20% (0.20)
- Main Memory Miss Penalty: 100 cycles

Question:

Calculate the Total Effective CPI for this system.

Ans:



Answer:

Total CPI: Base CPI + (L_1 miss rate × L_2 Hit time) + (Global miss rate × memory penalty)

L_1 Miss Stalls: 0,05 × 10 = 0.05 × 10 = 0,5 cycle

Global Miss rate: L_1 MR × L_2 LMR = 0,05 × 0,20 = 0,01 (1%)

L_2 Miss stalls: 0,01 × 100 = 1 cycle

Total CPI = 1 + 0,5 + 1 = $\underline{\underline{2,5}}$

[group 6]

5. Question:

Explain how the least recently used (LRU) works for cache block replacement for two-way set associative cache?

Ans:

In a two-way set-associative cache, a pointer is used to keep track of which block to replace. The pointer moves from one block to the other each time a block is accessed. When a replacement is needed, the cache evicts the block that the pointer is currently pointing to.

[group 8]

6. Question:

A cache has 1024 total blocks and is 4-way set associative. How many sets are in the cache? How many index bits (base-2)?

Ans:

Total blocks = 1024
Associativity = 4 → sets = 1024 / 4 = 256 sets
Index bits = $\log_2(256)$ = 8 bits

[group 9]

7. Question:

For caches with the same total capacity, why does increasing associativity usually reduce conflict misses?
And why can excessively high associativity potentially hurt performance?

Ans:

(1) In a direct-mapped cache, each memory block can only be placed in exactly one cache location. If two blocks map to the same index, they continuously replace each other, causing conflict misses.

In an n-way set associative cache, each index contains n entries, giving each block multiple placement options. This reduces placement conflicts, and therefore decreases conflict misses.

(2)

- More tag comparators are required → higher hardware cost
- Larger multiplexers → longer access delay
- Hit data becomes available later because the hit/miss decision must be made first

Thus, although increasing associativity reduces misses, too much associativity increases hit time, hardware complexity, and may lower overall system performance.

[group 11]

8. Question:

Which of the following statements are true?

A. Virtual memory "block" is called a page.

B. When CPU performance increases, the miss penalty becomes more significant.

C. In a fully associative cache, each memory block is placed in a specified cache location.

D. The main reason for adding an L2 cache between the L1 cache and main memory is to reduce the average hit time.

Ans:

A, B

C) In fully associative cache, each memory block can be placed in any cache location.

D) The main reason is to reduce the miss rate.

[group 13]

9. Question:

Compare data Available time(Direct mapped cache ,N-way set-associative cache)

Ans:

In terms of data availability time, a direct-mapped cache provides the fastest access because each memory block can only be placed in a single cache location. As a result, only one tag comparison is required, and data can typically be retrieved within one clock cycle. The hardware structure is simple, but this design is prone to conflict misses, leading to a higher miss rate.
In contrast, an N-way set-associative cache allows a memory block to be stored in any of the N cache lines within the same set. This requires performing N tag comparisons in parallel, increasing hardware complexity and slightly delaying data access, typically to around 1–2 clock cycles. Although its access time is slightly slower than that of a direct-mapped cache, it significantly reduces conflict misses and generally improves overall performance. Modern CPUs commonly use 2- to 8-way set-associative caches for the L1 level to balance access latency and hit rate.

[group 14]

10. Question:

Design a 128KB direct-mapped data cache that uses a 32-bit address and 16 bytes per block. Calculate the following:

(1) How many bits are used for the byte offset?

(2) How many bits are used for the index field?

(3) How many bits are used for the tag?

Ans:

(1) 2 bits

(2) 128K/16=8K = 2^{13} -> 13 bits

(3) 32 – 13 – 4 = 15 bits

[group 12]

11. Question:

For a fixed size n-way associative cache, when n increases, do the number of tag bits increase or decrease?

Ans:

increase, when n increases, the number of sets decreases, so the index bits decrease and the tag bits increase.