Computer Organization

COMP2120

Qi Zhao

February 28, 2024

Cache Memory Part II

Comparison

- Direct-map, k-way set associative, fully associative
- Direct-map

Tag	Line number	Offset
block id	cache line number	word/Byte id

- k-way set associative

Tag	set number	Offset
block id	Cache set number	word/Byte id

- Fully associative

Tag	Offset
block id	word/Byte id

- Higher hit ratio, slower

An exercise

Consider a machine with a byte addressable main memory of 2^{16} bytes and block size of 8 bytes.

Assume that a direct mapped cache consisting of 32 lines is used with this machine.

- How is a 16-bit memory address divided into tag, line number, and byte number?
- Into what line would bytes with each of the following addresses be stored?

$$0001000100011011$$
$$1100001100110100$$
$$1101000000011101$$

- Suppose the byte with address 0001101000011010 is stored in the cache. What are the addresses of the other bytes stored along with it?
- How many total bytes of memory can be stored in the cache?

Replacement Algorithm

- No replacement algorithm required for direct map organization, since there is only one line.
- There are 2-8 lines in each set for n-way set associative organization ($2 \leq n \leq 8$), need to choose one to replace when the needed block is not in cache.
- **FIFO** – first-in-first-out, or **LRU** – least recently used
- Random replacement (**RR**)

Write policies

- Maintain data consistency between cache memory and main memory.
- Problem only for **write**. If a cache line has not been written, then one can just discard it when it is replaced by another block.
- Two write policies — write through and write back

Write policies

- **Write through** — when writing cache, also write main memory.
  - Writing main memory is slower than cache.
  - However, we can let the memory write and continue execution of next instruction.
  - Cache and main memory always consistent.
  - Difficult to manage when we need to access memory, (e.g. A cache miss) if the previous write has not finished yet.
- **Write Back** — write back the memory block only when it is replaced.
  - Faster, no need to write to memory in between.
  - Main memory is inconsistent with cache, difficult to manage when I/O operation is needed. i.e. I/O to cache or I/O to main memory.

Multi-level caches

- memory in the same chip as the CPU will be much faster than memory on the mother board. The connection is in the chip and speed is fast. No need to go outside the CPU chip.
- We cannot have a too large on-chip cache (chip space limited), hence we need a Level 2 cache.
- If we have a very fast on chip cache L1, and a (still quite) fast cache, the overall access time can be improved.
- average access time =

$$
\begin{aligned}
&= \text{hit rate} \times \text{ hit time} + \text{ miss rate} \times \text{ average time when cache miss} \\
&= \text{hit time} + \text{ miss rate} \times \text{ miss penalty}
\end{aligned}
$$

- Miss penalty: extra time when cache miss occurs
- A fast Level 2 cache can reduce the miss penalty.

Multi-level caches

- Example: Assuming main memory access time of 50ns, L1 cache 1ns with a miss rate of 10%, L2 cache 5ns with a miss rate of 5%, and L3 cache 10ns with a miss rate of 2%.
- No cache, access time = 50ns
- Only L1 cache, access time $= 1 + 0.1 \times 50 =$ 6ns.
- Only L1+L2 cache, access time $= 1 + 0.1 \times (5 + 0.05 \times 50) = 1.75$ns
- L1+L2+L3 cache, access time $= 1 + 0.1 \times (5 + 0.05 \times (10 + 0.02 \times 50)) = 1.555$ns
- A more realistic model is that you have to consider the miss penalty of different levels. Very complicated.

Multi-level caches

Two-level cache perfor- mance as a function of cache size [GENU04]

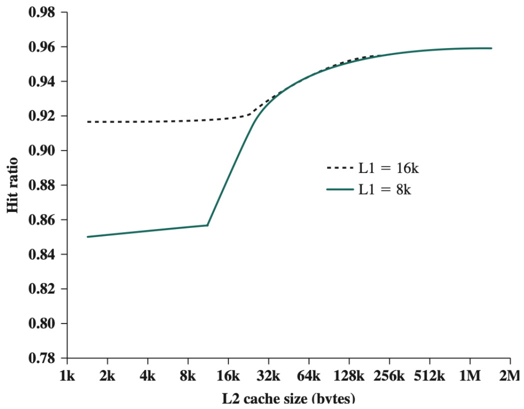
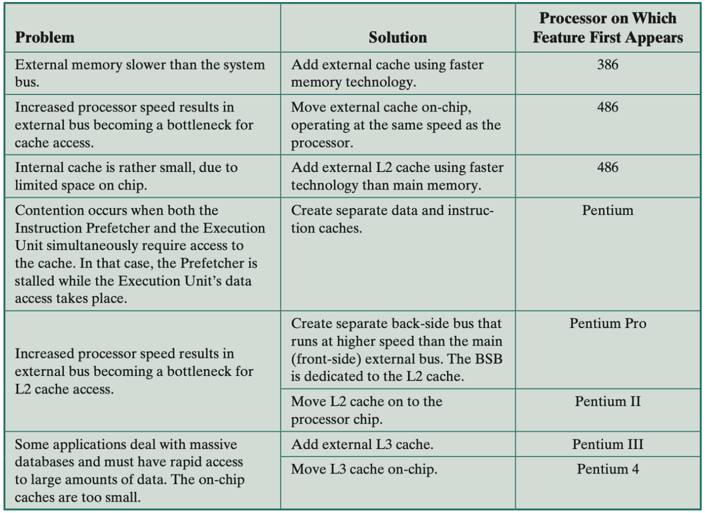


Figure 4.17 Total Hit Ratio (L1 and L2) for 8-kB and 16-kB L1

Multi-level caches

Table 4.4 Intel Cache Evolution



Problem	Solution	Processor on Which Feature First Appears
External memory slower than the system bus.	Add external cache using faster memory technology.	386
Increased processor speed results in external bus becoming a bottleneck for cache access.	Move external cache on-chip, operating at the same speed as the processor.	486
Internal cache is rather small, due to limited space on chip.	Add external L2 cache using faster technology than main memory.	486
Contention occurs when both the Instruction Prefetcher and the Execution Unit simultaneously require access to the cache. In that case, the Prefetcher is stalled while the Execution Unit's data access takes place.	Create separate data and instruction caches.	Pentium
Increased processor speed results in external bus becoming a bottleneck for L2 cache access.	Create separate back-side bus that runs at higher speed than the main (front-side) external bus. The BSB is dedicated to the L2 cache.	Pentium Pro
	Move L2 cache on to the processor chip.	Pentium II
Some applications deal with massive databases and must have rapid access to large amounts of data. The on-chip caches are too small.	Add external L3 cache.	Pentium III
	Move L3 cache on-chip.	Pentium 4

Unified caches vs Split caches

- **Split caches** – caches are divided into instruction cache and data cache.
- **Unified caches** – no such division.

Unified caches

- a unified cache will automatically balance between instruction and data, i.e. if you access more data, more data will be in the cache and vice versa.
- Will have memory contention problem on parallel and pipeline execution of instructions.

Split caches

- the sizes of instruction cache and data cache are fixed. Hence even though one of the cache is full while the other cache still has space available, you cannot use the other cache.
- No pipeline hazard between instruction and data. (Will be discussed in later chapters)
- The trend is towards split cache

Example

- Consider a hypothetical machine with 512 words of cache memory.
- two-way set associative organization, with cache block size of 64 words,
- LRU replacement algorithm.
- Suppose the cache hit time is 10ns.
- Suppose the machine can access 4 words of memory in parallel, and the time to transfer the first 4 words is from main memory to cache is 60ns, while each subsequent 4 words require 12ns.
- Consider the following read pattern (in blocks of 64 words, and block No. starts from 0)
- assume each block contains an average of 24 memory references. Total 29×24 memory references.

```
 0  1  2  5  3  2  5  3  11  7
9  0  6  0  7  9  8  7  9  11
12  2  4  5  12  15  12  13  15
```

Questions

1. What is the total No. of blocks in cache memory?
2. What is the number of sets in cache memory?
3. What is the cache miss penalty (i.e., time to transfer one block of data from main memory to cache memory)?
4. Write down the content of the cache memory (for all the blocks) at the end of the memory references, assuming that the cache is empty at the beginning
5. What is the cache hit rate?
6. Calculate the average memory access time

Answers

1. $512/64 = 8.$
2. $8/2 = 4$
3. Cache miss penalty

$$= 60 + 1215 = 240ns$$

4. Block 0 maps to set 0, Block 1 maps to set 1, block 5 maps to set 1, block 11 maps to set 3, $11\%4 = 3$.

Set 0	Set 1	Set 2	Set 3
0	1	2	3
	5		

Answers

- read pattern `0 1 2 5 3 2 5 3 11 7`
  Number of cache misses = 6

Set 0	Set 1	Set 2	Set 3
0	1	2	3
	5		11

- LRU Number of cache misses = 7

Set 0	Set 1	Set 2	Set 3
0	1	2	$3 \rightarrow 7$
	5		11

Answers

- read pattern second row `9 0 6 0 7 9 8 7 9 11`
  Number of cache misses = 8

Set 0	Set 1	Set 2	Set 3
0	$1 \to 9$	2	7
	5		11

- Number of cache misses = 9

Set 0	Set 1	Set 2	Set 3
0	9	2	7
	5	6	11

Answers

- read pattern second row `9 0 6 0 7 9 8 7 9 11`
  Number of cache misses = 10

Set 0	Set 1	Set 2	Set 3
0	9	2	7
8	5	6	11

- read pattern the third row `12 2 4 5 12 15 12 13 15`
  Number of cache misses = 11

Set 0	Set 1	Set 2	Set 3
$0 \rightarrow 12$	9	2	7
8	5	6	11

Answers

- read pattern the third row `12 2 4 5 12 15 12 13 15`
  Block 4, Number of cache misses = 12

Set 0	Set 1	Set 2	Set 3
12	9	2	7
$8 \to 4$	5	6	11

- Block 15, Number of cache misses = 13

Set 0	Set 1	Set 2	Set 3
12	9	2	$7 \to 15$
4	5	6	11

Answers

- read pattern the third row `12 2 4 5 12 15 12 13 15`
  Block 13, Number of cache misses = 14

Set 0	Set 1	Set 2	Set 3
12	$9 \rightarrow 13$	2	15
4	5	6	11

- End

Set 0	Set 1	Set 2	Set 3
12	13	2	15
4	5	6	11

Answers

5. We have 14 cache hit misses.
   Number of blocks accessed $= 29$
   Number of memory access $= 29 \times 24 = 696$
   each block average 24 memory access
   Cache hit rate $= 1 - 14/696 = 97.9885\%$.
6. Calculate the average memory access time.
   Average Memory Access Time

$$= 10 + (1 - 0.979885) \times 240 = 14.8276ns$$