第4章 存储器-层次结构设计

- 4.1 引言
- 4.2 Cache性能
- 4.3 Cache优化技术
- 4.4 主存储器技术
- 4.5 虚拟存储器
- 4.6 谬误和易犯的错误
- 4.7 小结

Cache性能分析

CPU执行时间与访存延迟的关系

- AMAT = Average Memory Access Time
- $\text{CPI}_{\text{ALUOps}}$ 不包括访存指令

$$CPUtime = IC \times \left(\frac{AluOps}{Inst} \times CPI_{AluOps} + \frac{MemAccess}{Inst} \times AMAT \right) \times CycleTime$$

$$\begin{aligned} AMAT &= HitTime + MissRate \times MissPenalty \\ &= \left(HitTime_{Inst} + MissRate_{Inst} \times MissPenalty_{Inst} \right) + \\ &\quad \left(HitTime_{Data} + MissRate_{Data} \times MissPenalty_{Data} \right) \end{aligned}$$

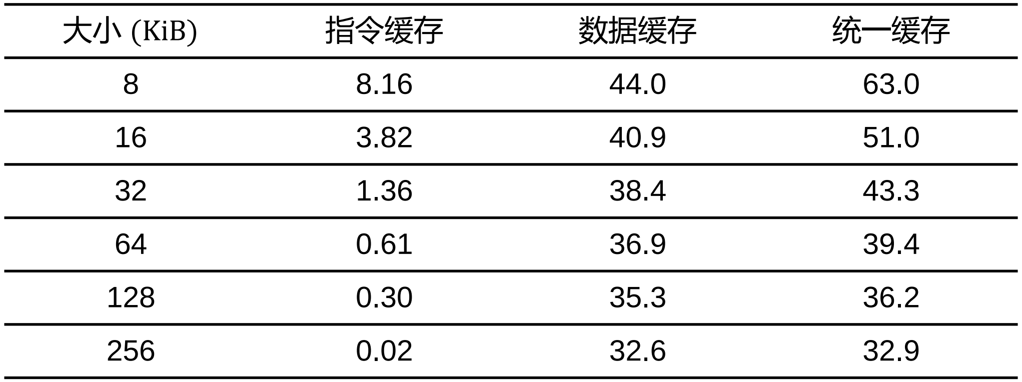
Example 1

- Which has the lower miss rate: a 16 KB instruction cache with a 16 KB data cache or a 32 KB unified cache? Use the miss rates in Figure B.6 to help calculate the correct answer, assuming 36% of the instructions are data transfer instructions. Assume a hit takes 1 clock cycle and the miss penalty is 100 clock cycles. A load or store hit takes 1 extra clock cycle on a unified cache if there is only one cache port to satisfy two simultaneous requests. Using the pipelining terminology of Chapter 3, the unified cache leads to a structural hazard. What is the average memory access time in each case? Assume write-through caches with a write buffer and ignore stalls due to the write buffer.

- 16KB指令缓存加上16KB 数据缓存相对于一个 32KB统一缓存, 哪一种的缺失率较低? 利用图B-6中的缺失率数据来帮助计算正确答案, 假定36%的指令为数据传输指令。假定一次命中需要1个时钟周期, 缺失代价为100 个时钟周期。对于统一缓存, 如果仅有一个缓存端口来满足两个同时请求, 一次载入或存储命中另需要一个时钟周期。利用第3章的流水线技术, 统一缓存会导致结构性冒险。每种情况下的存储器平均访问时间为多少? 假定采用具有写入缓冲区的直写缓存,忽略由于写入缓冲区导致的停顿。

Example 1

- 16KB指令缓存加上16KB 数据缓存相对于一个 32KB统一缓存, 哪一种的缺失率较低? 利用图B-6中的缺失率数据来帮助计算正确答案, 假定36%的指令为数据传输指令。假定一次命中需要1个时钟周期, 缺失代价为100 个时钟周期。对于统一缓存, 如果仅有一个缓存端口来满足两个同时请求, 一次载入或存储命中另需要一个时钟周期。利用第3章的流水线技术, 统一缓存会导致结构性冒险。每种情况下的存储器平均访问时间为多少? 假定采用具有写入缓冲区的直写缓存,忽略由于写入缓冲区导致的停顿。



大小 (KiB)	指令缓存	数据缓存	统一缓存
8	8.16	44.0	63.0
16	3.82	40.9	51.0
32	1.36	38.4	43.3
64	0.61	36.9	39.4
128	0.30	35.3	36.2
256	0.02	32.6	32.9

图B.6 对于不同大小的指令缓存、数据缓存与统一缓存, 每千条指令的缺失数

Example 1

- 16KB指令缓存加上16KB 数据缓存相对于一个 32KB统一缓存, 哪一种的缺失率较低? 利用图B-6中的缺失率数据来帮助计算正确答案, 假定36%的指令为数据传输指令。假定一次命中需要1个时钟周期, 缺失代价为100 个时钟周期。对于统一缓存, 如果仅有一个缓存端口来满足两个同时请求, 一次载入或存储命中另需要一个时钟周期。利用第3章的流水线技术, 统一缓存会导致结构性冒险。每种情况下的存储器平均访问时间为多少? 假定采用具有写入缓冲区的直写缓存,忽略由于写入缓冲区导致的停顿。

大小 (KiB)	指令缓存	数据缓存	统一缓存
8	8.16	44.0	63.0
16	3.82	40.9	51.0
32	1.36	38.4	43.3
64	0.61	36.9	39.4
128	0.30	35.3	36.2
256	0.02	32.6	32.9

图B.6 对于不同大小的指令缓存、数据缓存与统一缓存, 每千条指令的缺失数

$$\text{缺失率} = \frac{\frac{\text{缺失数}}{1000\ \text{条指令}}/1000}{\frac{\text{存储器访问次数}}{\text{指令}}}$$

首先将每千条指令的缺失数转换为缺失率。

由于每次指令访问都正好有一次存储器访问进行取指, 所以指令缓存缺失率为:

Miss rate16 KB instruction = (3.82 / 1000) / 1.0 = 0.00382

由于 36% 的指令为数据传输指令, 所以数据缓存缺失率为:

Miss rate16 KB data = (40.9 / 1000) / 0.36 = 0.11361

统一缓存缺失率需要考虑指令访问和数据访问 :

Miss rate32 KB unified = (43.3 / 1000) / (1 + 0.36) = 0.03184

Example 1

- 16KB指令缓存加上16KB 数据缓存相对于一个 32KB统一缓存, 哪一种的缺失率较低? 利用图B-6中的缺失率数据来帮助计算正确答案, 假定36%的指令为数据传输指令。假定一次命中需要1个时钟周期, 缺失代价为**100** 个时钟周期。对于统一缓存, 如果仅有一个缓存端口来满足两个同时请求, 一次载入或存储命中另需要一个时钟周期。利用第3章的流水线技术, 统一缓存会导致结构性冒险。每种情况下的存储器平均访问时间为多少? 假定采用具有写入缓冲区的直写缓存,忽略由于写入缓冲区导致的停顿。

Miss rate16 KB instruction = (3.82 / 1000) / 1.0 = 0.00382
Miss rate16 KB data = (40.9 / 1000) / 0.36 = 0.11361
Miss rate32 KB unified = (43.3 / 1000) / (1 + 0.36) = 0.03184

存储器平均访问时间公式可分为指令访问和数据访问:

存储器平均访问时间=指令百分比 × (命中时间 + 指令缓存缺失率 × 缺失代价) +
数据百分比 × (命中时间 + 数据缓存缺失率 × 缺失代价)

Average memory access timesplit=64%*(1+0.00382 * 100)+36%*(1+0.11361 * 100)
=5.33444

Average memory access timeunified= 64%*(1+0.03184*100)+ 36%*(1+1+0.03184 * 100)
=4.544

注：教材中把缺少代价写成200，所以计算结果不正确。

Example 2

- Let’s use an in-order execution computer for this example. Assume that the cache miss penalty is 200 clock cycles, and all instructions normally take 1.0 clock cycles (ignoring memory stalls). Assume that the average miss rate is 2%, there is an average of 1.5 memory references per instruction, and the average number of cache misses per 1000 instructions is 30. What is the impact on performance when behavior of the cache is included? Calculate the impact using both misses per instruction and miss rate.
- 本例使用顺序执行计算机。假定缓存缺失代价为 200 个时钟周期, 所有指令通常都占用 1.0 个时钟周期（忽略存储器停顿）。假定平均缺失率为 2%, 每条指令平均有 1.5 次存储器访问, 每千条指令的平均缓存缺失数为 30 。如果考虑缓存的行为特性, 对性能的影响如何? 使用每条指令的缺失数及缺失率来计算此影响。

Example 2(中文解答)

- 本例使用顺序执行计算机。假定缓存缺失代价为 200 个时钟周期, 所有指令通常都占用 1.0 个时钟周期（忽略存储器停顿）。假定平均缺失率为 2%, 每条指令平均有 1.5 次存储器访问, 每千条指令的平均缓存缺失数为 30 。如果考虑缓存的行为特性, 对性能的影响如何? 使用每条指令的缺失数及缺失率来计算此影响。

解：

$$\text{CPU 时间} = \text{IC} \times \left(\text{CPI}_{\text{执行}} + \frac{\text{存储器停顿时钟周期}}{\text{指令}}\right) \times \text{时钟周期时间}$$

其性能 (包括缓存缺失) 为:

$$\text{CPU 时间包括缓存} = \text{IC} \times [1.0 + (30/1000 \times 200)] \times \text{周期时钟时间}$$

$$= \text{IC} \times 7.00 \times \text{时钟周期时间}$$

现在使用缺失率计算性能:

$$\text{CPU 时间} = \text{IC} \times \left(\text{CPI 执行} + \text{缺失率} \times \frac{\text{存储器访问数}}{\text{指令}} \times \text{缺失代价}\right) \times \text{时钟周期时间}$$

$$\text{CPU时间包括缓存} = \text{IC} \times [1.0 + (1.5 \times 2\% \times 200)] \times \text{时钟周期时间}$$

$$= \text{IC} \times 7.00 \times \text{时钟周期时间}$$

在有、无缓存情况下，时钟周期时间和指令数均相同。因此，从“完美缓存”到“有可能产生缺失的缓存”, CPI 从 1.00 增加到 7.00。在根本没有任何存储器层次结构时, CPI 将再次升高到 1.0+200×1.5=301, 比带有缓存的系统长出 40 多倍。

Example 2(Solution)

Let's use an in-order execution computer for the first example. Assume that the cache miss penalty is **200** clock cycles, and all instructions normally take **1.0** clock cycles (ignoring memory stalls). Assume that the average miss rate is **2%**, there is an average of **1.5** memory references per instruction, and the average number of cache misses per **1000** instructions is **30**. What is the impact on performance when behavior of the cache is included? Calculate the impact using both misses per instruction and miss rate.

Answer

$$CPU\ time = IC * \left(CPI_{execution} + \frac{Memory\ stall\ clock\ cycles}{Instruction}\right) * Clock\ cycle\ time$$

The performance, including cache misses, is

$$\begin{aligned} \text{CPU time}_{\text{with cache}} &= \text{IC} \times [1.0 + (30/1000 \times 200)] \times \text{Clock cycle time} \\ &= \text{IC} \times 7.00 \times \text{Clock cycle time} \end{aligned}$$

Now calculating performance using miss rate:

$$CPU\ time = IC * \left(CPI_{execution} + Miss\ rate\ * \frac{Memory\ accesses}{Instruction} * Miss\ penalty\right) * Clock\ cycle\ time$$

$$\begin{aligned} \text{CPU time}_{\text{with cache}} &= \text{IC} \times [1.0 + (2\% \times 1.5 \times 200)] \times \text{Clock cycle time} \\ &= \text{IC} \times 7.00 \times \text{Clock cycle time} \end{aligned}$$

The clock cycle time and instruction count are the same, with or without a cache. Thus, CPU time increases sevenfold, with CPI from 1.00 for a "perfect cache" to 7.00 with a cache that can miss. Without any memory hierarchy at all the CPI would increase again to 1.0 + 200 × 1.5 or 301—a factor of more than 40 times longer than a system with a cache!

例3：直接映像与2路组相联的性能

假设： CPI=2（理想cache） clock cycle time＝1.0 ns

- 两种 caches 大小都是 64KB， cache块是 64 字节
- 直接映像 cache缺失率是1.4%， 2路组相联cache缺失率是 1.0%
- 2路组相联cache需要的多路选择器使 CPU clock cycle time 延长 1.25 倍
- 直接映像cache命中时间 1 个时钟周期，缺失开销是 75ns
- MPI（平均每条指令的访存次数）＝1.5

- 试比较两者的性能，先计算平均访存时间，然后计算CPU时间。

例3：直接映像与2路组相联的性能

假设： CPI=2（理想cache） clock cycle time＝1.0 ns

- 两种 caches 大小都是 64KB， cache块是 64 字节
- 直接映像 cache缺失率是1.4%， 2路组相联cache缺失率是 1.0%
- 2路组相联cache需要的多路选择器使 CPU clock cycle time 延长 1.25 倍
- 直接映像cache命中时间 1 个时钟周期，缺失开销是 75ns
- MPI（平均每条指令的访存次数）＝1.5

- 试比较两者的性能，先计算平均访存时间，然后计算CPU时间。

答案： 平均访存时间是

Average memory access time＝Hit time + Miss rate×miss penalty

因此，两种结构的平均访存时间是

Average memory access time$_{1\text{-way}}$＝1.0 ×1.0 + (0.014 ×75)＝2.05 ns

Average memory access time$_{2\text{-way}}$＝1.0× 1.0× 1.25 +(0.01 ×75)＝2.00 ns

2路组相联cache的平均访存时间更短。

例3：直接映像与2路组相联的性能

假设： CPI=2（理想cache）　　clock cycle time＝1.0 ns

- 两种 caches 大小都是 64KB， cache块是 64 字节
- 直接映像 cache缺失率是1.4%， 2路组相联cache缺失率是 1.0%
- 2路组相联cache需要的多路选择器使 CPU clock cycle time 延长 1.25 倍
- 直接映像cache命中时间 1 个时钟周期，缺失开销是 75ns
- MPI（平均每条指令的访存次数）＝1.5

- 试比较两者的性能，先计算平均访存时间，然后计算CPU时间。

$$CPUtime = IC \times \left(CPI_{execution} + \frac{Misses}{Instruction} \times Miss\,penalty \right) \times Clock\,cycle\,time$$

$$= IC \times \left[\left(CPI_{execution} \times Clock\,cycle\,time \right) + \left(Miss\,rate \times \frac{Memory\,accesses}{Instruction} \times Miss\,penalty \times Clock\,cycle\,time \right) \right]$$

条件中的 75 ns 就是 (miss penalty×Clock cycle time)，两种结构的性能是：

CPU time$_{1\text{-way}}$＝IC×(2×1.0 + (0.014×1.5 ×75))＝3.58 ×IC

CPU time$_{2\text{-way}}$＝IC×(2×1.0×1.25 + (0.010×1.5 ×75))＝3.63 ×IC

例3：直接映像与2路组相联的性能

假设： CPI=2（理想cache） clock cycle time＝1.0 ns

- 两种 caches 大小都是 64KB， cache块是 64 字节
- 直接映像 cache缺失率是1.4%， 2路组相联cache缺失率是 1.0%
- 2路组相联cache需要的多路选择器使 CPU clock cycle time 延长 1.25 倍
- 直接映像cache命中时间 1 个时钟周期，缺失开销是 75ns
- MPI（平均每条指令的访存次数）＝1.5

- 试比较两者的性能，先计算平均访存时间，然后计算CPU时间。

$$\text{Average memory access time}_{1\text{-way}} = 1.0 \times 1.0 + (0.014 \times 75) = 2.05\ \text{ns}$$

$$\text{Average memory access time}_{2\text{-way}} = 1.0 \times 1.0 \times 1.25 + (0.01 \times 75) = 2.00\ \text{ns}$$

$$\text{CPU time}_{1\text{-way}} = IC \times (2 \times 1.0 + (0.014 \times 1.5 \times 75)) = 3.58 \times IC$$

$$\text{CPU time}_{2\text{-way}} = IC \times (2 \times 1.0 \times 1.25 + (0.010 \times 1.5 \times 75)) = 3.63 \times IC$$

Example 4

- What is the impact of two different cache organizations on the performance of a processor? Assume that the CPI with a perfect cache is 1.6, the clock cycle time is 0.35 ns, there are 1.4 memory references per instruction, the size of both caches is 128 KB, and both have a block size of 64 bytes. One cache is direct mapped and the other is two-way set associative. Figure B.5 shows that for set associative caches we must add a multiplexor to select between the blocks in the set depending on the tag match. Since the speed of the processor can be tied directly to the speed of a cache hit, assume the processor clock cycle time must be stretched 1.35 times to accommodate the selection multiplexor of the set associative cache. To the first approximation, the cache miss penalty is 65 ns for either cache organization. (In practice, it is normally rounded up or down to an integer number of clock cycles.) First, calculate the average memory access time and then processor performance. Assume the hit time is 1 clock cycle, the miss rate of a direct-mapped 128 KB cache is 2.1%, and the miss rate for a two-way set associative cache of the same size is 1.9%.
- 两种缓存组织方式对处理器性能的影响如何? 假定完美缓存的 CPI 为 1.6 , 时钟周期时间为 0.35ns, 每条指令有1.4 次存储器访问, 两个缓存的大小都是 128KiB, 两者的块大小都是 64 字节。一个缓存为直接映射, 另一个为两路组相联。图 B-5 显示,对于组相联缓存, 必须添加一个多路选择器, 以根据标记匹配在组中的块之间做出选择。由于处理器的速度直接与缓存命中的速度联系在一起, 所以假定必须将处理器时钟周期时间延长1.35 倍, 才能与组相联缓存的选择多路选择器相适应。对于一级近似, 每一种缓存组织方式的缓存缺失代价都是65ns。(在实践中, 通常会舍入为整数个时钟周期。) 首先, 计算存储器平均访问时间, 然后再计算处理器性能。假定命中时间为1个时钟周期, 128KiB 直接映射缓存的缺失率为 2.1%, 同等大小的两路组相联缓存的缺失率为1.9%

Example 4

What is the impact of two different cache organizations on the performance of a processor? Assume that the CPI with a perfect cache is 1.6, the clock cycle time is 0.35 ns, there are 1.4 memory references per instruction, the size of both caches is 128 KB, and both have a block size of 64 bytes. One cache is direct mapped and the other is two-way set associative. Figure B.5 shows that for set associative caches we must add a multiplexor to select between the blocks in the set depending on the tag match. Since the speed of the processor can be tied directly to the speed of a cache hit, assume the processor clock cycle time must be stretched 1.35 times to accommodate the selection multiplexor of the set associative cache. To the first approximation, the cache miss penalty is 65 ns for either cache organization. (In practice, it is normally rounded up or down to an integer number of clock cycles.) First, calculate the average memory access time and then processor performance. Assume the hit time is 1 clock cycle, the miss rate of a direct-mapped 128 KB cache is 2.1%, and the miss rate for a two-way set associative cache of the same size is 1.9%.

Answer $Average\ memory\ access\ = Hit\ time + Miss\ rate * Miss\ penalty$

$$Average\ memory\ access_{1\ \_\ \mathrm{way}} = 0.35 + 2.1\% * 65 = 1.715\ (\mathrm{ns})$$

$$Average\ memory\ access_{2\ \_\ \mathrm{way}} = 0.35 * \ 1.35 + 1.9\% * 65 = 1.7075\ (\mathrm{ns})$$

The average memory access time is better for the two-way set-associative cache.

两路组相联缓存的存储器平均访问时间更优。

$$CPU\ time = IC * \left(CPI_{execution} + \frac{Misses}{Instruction} * Miss\ penalty\right) * Clock\ cycle\ time$$

$$= IC * \left(CPI_{execution} * Clock\ cycle\ time + \frac{Misses}{Instruction} * Miss\ penalty * Clock\ cycle\ time\right)$$

$$= IC * \left(CPI_{execution} * Clock\ cycle\ time + Miss\ rate * \frac{Memory\ accesses}{Instruction} * Miss\ penalty * Clock\ cycle\ time\right)$$

What is the impact of two different cache organizations on the performance of a processor? Assume that the CPI with a perfect cache is 1.6, the clock cycle time is 0.35 ns, there are 1.4 memory references per instruction, the size of both caches is 128 KB, and both have a block size of 64 bytes. One cache is direct mapped and the other is two-way set associative. Figure B.5 shows that for set associative caches we must add a multiplexor to select between the blocks in the set depending on the tag match. Since the speed of the processor can be tied directly to the speed of a cache hit, assume the processor clock cycle time must be stretched 1.35 times to accommodate the selection multiplexor of the set associative cache. To the first approximation, the cache miss penalty is 65 ns for either cache organization. (In practice, it is normally rounded up or down to an integer number of clock cycles.) First, calculate the average memory access time and then processor performance. Assume the hit time is 1 clock cycle, the miss rate of a direct-mapped 128 KB cache is 2.1%, and the miss rate for a two-way set associative cache of the same size is 1.9%.

Answer $CPU\ time = IC * \left(CPI_{execution} * Clock\ cycle\ time + Miss\ rate * \frac{Memory\ accesses}{Instruction} * Miss\ penalty * Clock\ cycle\ time \right)$

$$CPU\ time_{1\ \_\ \mathrm{way}} = IC * (1.6 * 0.35 + 2.1\% * 1.4 * 65) = 2.471*\mathrm{IC}$$

$$CPU\ time_{2\ \_\ \mathrm{way}} = IC * (1.6 * 0.35 * 1.35 + 1.9\% * 1.4 * 65) = 2.485*\mathrm{IC}$$

In contrast to the results of average memory access time comparison, the direct-mapped cache leads to slightly better average performance because the clock cycle is stretched for *all* instructions for the two-way set associative case, even if there are fewer misses. Since CPU time is our bottom-line evaluation and since direct mapped is simpler to build, the preferred cache is direct mapped in this example.

与存储器平均访问时间的对比结果相反, 直接映射缓存的平均性能略好一些, 这是因为尽管两路组相联的缺失数较少，但针对所有指令延长了时钟周期。由于 CPU 时间是我们的基本评估标准，而且直接映射的构建更简单一些，所以本示例中直接映射更有优势。

例5：缺失代价与乱序执行处理器

What is the impact of two different cache organizations on the performance of a processor? Assume that the CPI with a perfect cache is 1.6, the clock cycle time is 0.35 ns, there are 1.4 memory references per instruction, the size of both caches is 128 KB, and both have a block size of 64 bytes. One cache is direct mapped and the other is two-way set associative. Figure B.5 shows that for set associative caches we must add a multiplexor to select between the blocks in the set depending on the tag match. Since the speed of the processor can be tied directly to the speed of a cache hit, assume the processor clock cycle time must be stretched 1.35 times to accommodate the selection multiplexor of the set associative cache. To the first approximation, the cache miss penalty is 65 ns for either cache organization. (In practice, it is normally rounded up or down to an integer number of clock cycles.) First, calculate the average memory access time and then processor performance. Assume the hit time is 1 clock cycle, the miss rate of a direct-mapped 128 KB cache is 2.1%, and the miss rate for a two-way set associative cache of the same size is 1.9%.

Let’s redo the example above, but this time we assume the processor with the longer clock cycle time supports out-of-order execution yet still has a direct-mapped cache. Assume 30% of the 65 ns miss penalty can be overlapped; that is, the average CPU memory stall time is now 45.5 ns.

Miss Penalty and Out-of-Order Execution Processors

- 两种缓存组织方式对处理器性能的影响如何? 假定完美缓存的 CPI 为 1.6 , 时钟周期时间为 0.35ns, 每条指令有 1.4 次存储器访问, 两个缓存的大小都是 128KiB, 两者的块大小都是 64 字节。一个缓存为直接映射, 另一个为两路组相联。图 B-5 显示,对于组相联缓存, 必须添加一个多路选择器, 以根据标记匹配在组中的块之间做出选择。由于处理器的速度直接与缓存命中的速度联系在一起, 所以假定必须将处理器时钟周期时间延长 1.35 倍, 才能与组相联缓存的选择多路选择器相适应。对于一级近似, 每一种缓存组织方式的缓存缺失代价都是65ns。(在实践中, 通常会舍入为整数个时钟周期。) 首先, 计算存储器平均访问时间, 然后再计算处理器性能。假定命中时间为1个时钟周期, 128KiB 直接映射缓存的缺失率为 2.1%, 同等大小的两路组相联缓存的缺失率为1.9%
- 让我们重做上面的例题,但这一次假定具有较长时钟周期时间的处理器支持乱序执行，仍采用直接映射缓存。假定 65ns 的缺失代价中有 30% 可以重叠，也就是说，CPU 存储器平均停顿时间现在为 45.5ns。

Miss Penalty and Out-of-Order Execution Processors

What is the impact of two different cache organizations on the performance of a processor? Assume that the CPI with a perfect cache is 1.6, the clock cycle time is 0.35 ns, there are 1.4 memory references per instruction, the size of both caches is 128 KB, and both have a block size of 64 bytes. One cache is direct mapped and the other is two-way set associative. Figure B.5 shows that for set associative caches we must add a multiplexor to select between the blocks in the set depending on the tag match. Since the speed of the processor can be tied directly to the speed of a cache hit, assume the processor clock cycle time must be stretched 1.35 times to accommodate the selection multiplexor of the set associative cache. To the first approximation, the cache miss penalty is 65 ns for either cache organization. (In practice, it is normally rounded up or down to an integer number of clock cycles.) First, calculate the average memory access time and then processor performance. Assume the hit time is 1 clock cycle, the miss rate of a direct-mapped 128 KB cache is 2.1%, and the miss rate for a two-way set associative cache of the same size is 1.9%.

Let's redo the example above, but this time we assume the processor with the longer clock cycle time supports out-of-order execution yet still has a direct-mapped cache. Assume 30% of the 65 ns miss penalty can be overlapped; that is, the average CPU memory stall time is now 45.5 ns.

Answer $Average\ memory\ access\ = Hit\ time + Miss\ rate * Miss\ penalty$

$$Average\ memory\ access_{1\ \_\ \text{way}_{,}\ ooo} = 0.35 * 1.35 + 2.1\% * 45.5 = 1.428\ (\text{ns})$$

$$CPU\ time = IC * \left(CPI_{execution} * Clock\ cycle\ time + Miss\ rate * \frac{Memory\ accesses}{Instruction} * Miss\ penalty * Clock\ cycle\ time\right)$$

$$CPU\ time_{1\ \_\ \text{way}_{,}\ ooo} = IC * (1.6 * 0.35 * 1.35 + 2.1\% * 1.4 * 45.5) = 2.0937\text{*IC}$$