Computer Architecture and Engineering - 3rd Year - ETSINF - UPV

Exercises and solutions of Unit 3. “Memory subsystem”

Exercise 3.1.

In the design phase of a computer cache, two alternatives are being considered. The first one is to use a direct mapping, which results in a miss rate of 1.4 % and leads to a clock period of 1 ns. The second is to use an associative mapping, which reduces the miss rate to 1 % but increases the clock period to 1.25 ns. In case of hit, one clock cycle is required for reading or writing the data. If the programs are executed with a CPI of 2, with 1.5 memory accesses per instruction, and the miss penalty is 75 ns, decide what is the best alternative.

Solution:

The memory access time on a system with cache memories is:

$T_{access} = Ht + MR \times MP$

In this case:

- Direct mapping:
  $T_{access} = 1 + 0.014 \times 75 = 2.05$ ns.
- Associative mapping:
  $T_{access} = 1.25 + 0.01 \times 75 = 2$ ns.

The access time is better for the cache with associative mapping.

Next, we compare the execution time for both alternatives. The execution time of a program on a computer with cache memory is given by:

$T_{execution} = I \times CPI \times T + I \times API \times MR \times MP \times T$. Substituting:

- Direct mapping:
  As the memory access time is 75 ns and the clock period is $T = 1$ ns, the miss penalty is $MP = 75$ cycles:
  $T_{execution} = I \times 2 \times 1 + I \times 1.5 \times 0.014 \times 75 \times 1 = 3.58I$ ns.
- Associative mapping:
  As the memory access time is 75 ns and the clock period is $T = 1.25$ ns, the miss penalty is $MP = 60$ cycles:
  $T_{execution} = I \times 2 \times 1.25 + I \times 1.5 \times 0.01 \times 60 \times 1.25 = 3.63I$ ns.

Thus, direct mapping, with shorter clock period, reduces the execution time.

□

Exercise 3.2.

Analyze the execution time of a program on a pipelined computer with CPI = 1 under the following cache configurations:

1. No cache memory.
2. Using a cache with a miss rate of 2 %.
3. Using a perfect cache.

The main memory has an access time of 100 clock cycles and the program has a 50 % of memory access instructions.

Solution:

The memory access time on a system with cache memories is:

$T_{access} = Ht + MR \times MP$

The execution time of a program on a computer with cache memory is given by:

$T_{execution} = I \times CPI \times T + I \times API \times MR \times MP \times T.$

Substituting:

1. No cache memory.
   $T_{access} = 1 + 1(100\%) \times 100 = 101$ cycles
   $T_{execution} = I \times 1 \times T + I \times 1.5 \times 1(100\%) \times 100 \times T = 151I$ cycles.
2. Using a cache with a miss rate of 2 %.
   $T_{access} = 1 + 0.02 \times 100 = 3$ cycles
   $T_{execution} = I \times 1 \times T + I \times 1.5 \times 0.02 \times 100 \times T = 4I$ cycles.
3. Using a perfect cache.
   $T_{access} = 1 + 0.0 \times 100 = 1$ cycle
   $T_{execution} = I \times 1 \times T + I \times 1.5 \times 0.0 \times 100 \times T = I$ cycles.

□

Exercise 3.3.

An engineer wants to compare the performance of two memory subsystems. The first one has separate instruction and data caches, each one with capacity for 16 KB, while the second one has a single cache memory of 32 KB. After running some test programs, it has been obtained that the miss rate for the instruction cache is 3.82, for the data cache is 40.9, and for the unified cache is 43.3 (misses per 1,000 executed instructions), and that 36 % of the executed instructions are memory accesses. If the hit time and the miss penalty are 1 and 100 clock cycles, respectively:

1. Compute the memory access time in both schemes.
2. If the computer is pipelined and the test program consists of n instructions, compare both alternatives based on the execution time.

Solution:

1. Compute the memory access time in both schemes.
   The memory access time in a cache memory system is:
   $$T_{access} = Ht + MR \times MP$$
   In our case:
   MR (misses/access) = (misses/instruction)/(accesses/instruction). Susbtituting:
   $MR_I = \frac{3.82/1000}{1} = 0.004$ $MR_D = \frac{40.9/1000}{0.36} = 0.1144$ $MR_U = \frac{43.3/1000}{1.36} = 0.0318$
   Separate caches:
   A program consisting of 100 instructions accesses data from memory 36 times on average, with 136 total accesses. Therefore, the miss rate is given by:
   $MR_{I+D} = \frac{100}{136}MR_I + \frac{36}{136}MR_D = 0.74 \cdot 0.004 + 0.26 \cdot 0.1144 = 0.0324$. Therefore:
   $T_{access} = Ht + MR \times MP = 1 + 0.0324 \cdot 100 = 4.24$ cycles
   Unified cache:
   In this case, we should take into account that an access to data in memory (26% of all accesses) will cause an structural hazzard that will lead to a stall cycle:
   $T_{access} = Ht + MR \times MP = 1 + 0.26 \cdot 1 + 0.0318 \cdot 100 = 4.44$ cycles

2. Comparison of both alternatives based on the execution time.

   The execution time of a program on a computer with cache memory is given by:
   $T_{execution} = I \times CPI \times T + I \times API \times MR \times MP \times T$. Susbtituting:

   Separate caches:

   $T_{execution} = n \times 1 \times T + n \times 1.36 \times 0.0324 \times 100 \times T = 5.41n$ cycles.

   Unified cache:

   $T_{execution} = n \times (1 + 0.36) \times T + n \times 1.36 \times 0.0318 \times 100 \times T = 5.49n$ cycles.

□

Exercise 3.4.

Consider an ideal pipelined computer with a memory hierarchy consisting of two cache levels and main memory. Their features are:

First level: direct mapping Hit time, $Ht_{L1} = 1$ cycle; local miss rate, $MR_{L1} = 0.04$.

Second level: associative mapping Hit time, $Ht_{L2} = 5$ cycles; local miss rate, $MR_{L2} = 0.50$.

Main memory Access time, $Ht_{MM} = 20$ cycles.

It is known that 25% of the executed instructions are loads or stores.

The designers think that it is possible to improve main memory access time by implementing the *critical word first* technique, which reduces the effective main memory access time to $Ht_{MM} = 15$ cycles. Implementing this improvement involves increasing the clock period by 10%.

The increase in clock period allows us to increase the associativity of the second level cache, reducing the local miss rate to $MR_{L2} = 0.40$.

Compute the improvement achieved in the execution time.

Solution:

1. The access time of the original multilevel cache is:

$$T_{access} = Ht_{L1} + MR_{L1} \times \underbrace{(Ht_{L2} + MR_{L2} \times MP_{L2})}_{MP_{L1}}$$

$$MP_{L1} = Ht_{L2} + MR_{L2} \times MP_{L2} = 5 + 0.5 \times 20 = 5 + 10 = 15$$

   The execution time is:

$$T_{execution} = I \times CPI \times T + I \times API \times MR \times MP \times T$$

$$T_{execution} = I \times (CPI + API \times MR \times MP) \times T$$

   substituting:

$$T_{execution} = I \times (1 + 1.25 \times 0.04 \times 15) \times T = I \times 1.75 \times T$$

   The L1 miss penalty with the indicated improvement would be:

$$MP'_{L1} = Ht'_{L2} + MR'_{L2} \times MP'_{L2} = 5 + 0.4 \times 15 = 5 + 6 = 11$$

   Using this value for MP_{L1}, the execution time would be:

$$T'_{execution} = I' \times (CPI' + API' \times MR' \times MP') \times T'$$

   substituting:

$$T'_{execution} = I \times (1 + 1.25 \times 0.04 \times 11) \times 1.1 \cdot T$$

$$T'_{execution} = I \times 1.55 \times 1.1 \cdot T = I \times 1.705 \times T$$

Thus, the achieved improvement would be:

$$S = \frac{T_{execution}}{T'_{execution}} = \frac{I \times 1.75 \times T}{I \times 1.705 \times T} = 1.026$$

The execution time would improve by 2.6 %.

□

Exercise 3.5.

A 32-bit pipelined computer with separate instruction and data caches is being designed. The characteristics of the memory hierarchy are:

- Main memory: latency, 16 $cycles$; bandwidth, $W = 2\ bytes/cycle$. Thus, for example, the transfer time of a 6-word block would be: $T_{mm} = 16 + \frac{(6 \times 4)}{2} = 28\ cycles$.
- Data cache: Hit time, $Ht^D = 1\ cycle$; miss rate, $MR^D = 0.05$.
- Instruction cache: Hit time, $Ht^I = 1\ cycle$; miss rate, $MR^I = 0.02$.
- Block size, $B = 4\ words$.

The aim is to evaluate two potential improvements to the cache system. Here are the technical data of both alternatives.

1) Incorporate a second level cache with: Hit time, $Ht_{L2} = 6\ cycles$; local miss rate, $MR_{L2} = 0.5$. Block size is the same.
2) Increase the block size of both caches to 8 $words$, thus doubling its size. This change will affect the miss rate of both caches: $MR^D = 0.03$ and $MR^I = 0.015$

The CPI, ignoring cache misses, is 1.1 $cycles/inst$ and it is known that 25 % of the executed instructions are loads or stores. Analyze the influence of both alternatives on the program execution time, indicating which of the two alternatives is the most interesting one and quantifying the improvement obtained over the original design.

Solution:

The time required to bring a block of B words from main memory is given by the expression:

$T_{mm} = 16 + \frac{(B \times 4)}{2} = 16 + B \times 2$ cycles.

The execution time, considering the memory hierarchy, would be:

$$T_{exec} = I \times CPI \times T + I \times API \times MR \times MP \times T$$

In an architecture with data and instruction caches, the number of accesses per instruction and the miss rate can be separated:

$$T_{exec} = I \times CPI \times T + I \times \Big((API^I \times MR^I + API^D \times MR^D) \times MP\Big) \times T$$

Applying it to the original design of the memory subsystem:

$CPI = 1.1$, $API^I = 1$, $MR^I = 0.02$, $API^D = 0.25$, $MR^D = 0.05$ and $MP = 16 + 4 \times 2 = 24$ cycles. Substituting:

$$T_{exec} = I \times 1.1 \times T + I \times ((1 \times 0.02 + 0.25 \times 0.05) \times 24) \times T = I \times 1.88 \times T$$

Consider now the execution time with each of the proposed improvements:

- Two cache levels.
  In this case, the miss penalty is:
  $MP = Ht_{L2} + MR_{L2} \times T_{mm} = 6 + 0.5 \times 24 = 18$ cycles.
  Substituting in the execution time expression:

$$T_{exec} = I \times 1.1 \times T + I \times ((1 \times 0.02 + 0.25 \times 0.05) \times 18) \times T = I \times 1.69 \times T$$

- Increasing the block size.
  In this case, besides miss rates, the miss penalty is modified when the block size is changed:
  $MP = 16 + 8 \times 2 = 32$ cycles.
  Substituting in the execution time expression:

$$T_{exec} = I \times 1.1 \times T + I \times ((1 \times 0.015 + 0.25 \times 0.03) \times 32) \times T = I \times 1.82 \times T$$

Clearly, the first option is the best one. The speedup obtained with respect to the original design is $\frac{I \times 1.88 \times T}{I \times 1.69 \times T} = 1.1124$. Therefore, the use of two cache levels improves execution time by 11,24 %.

□

Exercise 3.6.

A computer has the following components:

A *load/store* processor similar to MIPS, with word size of 32 bits, clock frequency of 100 MHz, and CPI = 1.3. 20 % of the executed instructions are memory reads, and 10 % are memory writes.

A single level cache memory, split into 16 KB for instructions and 16 KB for data. The block size is 16 bytes and the mapping is two-way set associative. It implements the *write-through* and *no-write allocate* policies. The miss rate is 2 % for reads and 5 % for writes.

For writes, the processor has buffers that always give higher priority to reads.

A main memory with an access time for reading of 50 ns and a burst mode that provides successive words every 10 ns. That is, bursts follow the pattern 5-1-1-1-1 at 100 MHz. Initially, loading a block in the cache is conventional (that is, it does not apply *Critical Word First* nor *Early Restart*)

Compute:

1. The read miss penalty, expressed in clock cycles and in seconds, when conventional load is applied.
2. The read miss penalty, expressed in clock cycles and in seconds, when *Critical Word First* is applied.
3. The write miss penalty, expressed in clock cycles and in seconds.
4. The execution time for a program with 10 million instructions, assuming that conventional load is applied.

Solution:

1. Read miss penalty with conventional load:
   If the word width is 32 bits and the block size of 16 bytes, the bursts to load a block into the cache are 4 words long. The required time is $MP = (5 + 1 + 1 + 1 + 1) cycles = 9 \times 10ns = 90ns$
2. Read miss penalty when *Critical Word First* is applied:
   The required time is $MP = 5 + 1 cycles = 6 \times 10ns = 60ns$
3. Write miss penalty: none, because with the *no-write allocate* policy and the write buffers, the processor never has to wait for memory transfers.
4. The execution time is $T = I \cdot CPI \cdot t_C + I \cdot RPI \cdot MR^R \cdot MP^R = 10 \cdot 10^6 \cdot 1.3 \cdot 10ns + 10 \cdot 10^6 \cdot 1.2 \cdot 0.02 \cdot 90ns$

□

Exercise 3.7.

A computer has the following components:

- A *load/store* processor similar to MIPS, with word size of 32 bits, clock frequency of 100 MHz, and CPI = 1.3. 20 % of the executed instructions are memory reads, and 10 % are memory writes.
- A single level cache memory, split into 16 KB for instructions and 16 KB for data. The block size is 16 bytes and the mapping is two-way set associative. It implements the *write-through* and *no-write allocate* policies. The miss rate is 2 % for reads and 5 % for writes.
- For store instructions, the processor implements buffers
- The main memory is an SDR SDRAM whose timing parameters are $CL - t_{RCD} - t_{RP} = 1 - 2 - 2$. It has been measured that the requested row is open in 90% of the accesses.

Compute:

1. The read miss penalty, expressed in clock cycles and in seconds, when conventional load is applied.
2. The read miss penalty, expressed in clock cycles and in seconds, when *Critical Word First* is applied.
3. The write miss penalty, expressed in clock cycles and in seconds.
4. The execution time for a program with 10 million instructions, in two cases:
   - Conventional load is applied
   - *Critical Word First* is applied

Solution:

If the word width is 32 bits and the block size is 16 bytes, the bursts to load a block into the cache are 4 words long ($B = 4$). Also, since it is an SDR SDRAM, $B_{wc} = 1$ word/cycle.

1. Read miss penalty with conventional load:

$$MP \text{ (in cycles)} = L_c \cdot (1 - ML) + L_{rc} \cdot ML + \frac{1}{B_{wc}} \cdot B$$

$$L_c = t_{RP} + t_{RCD} + CL \text{ cycles} = 5 \text{ cycles}$$

$$L_{rc} = CL \text{ cycles} = 1 \text{ cycle}$$

Thus,

$$MP \text{ (in cycles)} = 5 \cdot (1 - 0.9) + 1 \cdot 0.9 + \frac{1}{1} \cdot 4 = 5.4 \text{ cycles} = 54 \text{ ns}$$

2. Read miss penalty when *Critical Word First* is applied. It is equivalent to have $B = 1$. Thus:

$$MP \text{ (in cycles)} = 5 \cdot (1 - 0.9) + 1 \cdot 0.9 + \frac{1}{1} \cdot 1 = 2.4 \text{ cycles} = 24 \text{ ns}$$

3.

4. Write miss penalty: none, because with the *no-write allocate* policy and the write buffers, the processor never has to wait for memory transfers.
5. The execution time for a program with 10 million instructions is

$$T_{ex} = I \times CPI \times T + I \times RPI \times RMR \times RMP$$

- With conventional block load:

$$\begin{aligned} T_{ex} &= 10 \cdot 10^6 \text{ instructions} \times 1.3 \text{ cycles/instruction} \times 10 \cdot 10^{-9} \text{ s/cycle} + \\ &+ 10 \cdot 10^6 \text{ instructions} \times 1.2 \text{ reads/instruction} \times \\ & \times 0.02 \text{ misses/read} \times 54 \cdot 10^{-9} \text{ s/miss} \\ &\simeq 130 \text{ ms} + 13 \text{ ms} = 143 \text{ ms} \end{aligned}$$

- With *Critical Word First* only the miss penalty changes. Therefore:

$$T_{ex} = 130 \text{ ms} + 10 \cdot 10^6 \times 1.2 \times 0.02 \times 24 \cdot 10^{-6} \text{ ms} \simeq 130 + 6 = 136 \text{ ms}$$

□