Cache & performance

Exercise 1

A computer has a 16 GB main memory address space and a 64 KB cache. What is the number of bits in tag, index and block offset fields when cache is organized as follows:

a) direct mapping, block size 64 bytes

b) set associative mapping, block size 64 bytes, four blocks per set (4-way set associative)

c) set associative mapping, block size 64 bytes, eight blocks per set (8-way set associative)

d) Processor that has a cache organization of exercise 1-C accesses memory at address 0xA8F3ED5.

   1. Calculate block number (block address) for the memory reference

   2. Calculate tag for the memory reference

Exercise 2

Compute average memory access times in clock cycles for the two processors below assuming that caches are 8-way set associative and that miss rate for caches over 512KB is 0,004 for miss rate of other sizes use the table on the next page. Assume that miss penalty of a level is equal to lower level latency and the miss penalty for the lowest cache level is the main memory access time.

	Intel Nehalem	**Intel Penryn**
L1 Size / L1 Latency	64KB / 4 cycles	64KB / 3 cycles
L2 Size / L2 Latency	256KB / 11 cycles	6MB / 15 cycles
L3 Size / L3 Latency	8MB / 39 cycles	N/A
Main Memory Latency (DDR3-1600 CAS7)	107 cycles (33.4 ns)	160 cycles (50.3 ns)

Cache size (KB)	Degree associative	Total miss rate
4	1-way	0.098
4	2-way	0.076
4	4-way	0.071
4	8-way	0.071
8	1-way	0.068
8	2-way	0.049
8	4-way	0.044
8	8-way	0.044
16	1-way	0.049
16	2-way	0.041
16	4-way	0.041
16	8-way	0.041
32	1-way	0.042
32	2-way	0.038
32	4-way	0.037
32	8-way	0.037
64	1-way	0.037
64	2-way	0.031
64	4-way	0.030
64	8-way	0.029
128	1-way	0.021
128	2-way	0.019
128	4-way	0.019
128	8-way	0.019
256	1-way	0.013
256	2-way	0.012
256	4-way	0.012
256	8-way	0.012
512	1-way	0.008
512	2-way	0.007
512	4-way	0.006
512	8-way	0.006

Exercise 3

The cache simulator project sources can be found in CacheSimulator.zip. The simulator has been tested with Visual Studio. However, the simulator should compile with other toolchains with minor modifications.

Unzip the attached CacheTraces.zip. The file contains traces of memory references that have been recorded with various applications.

The two most important classes in the project are Memory and Cache. Memory is a simple class that simulates access to main memory. When created the object is initialized with access time of the memory. The value is given as an integer – you can consider this value to be nanoseconds or clock cycles (or any unit that can be expressed as an integer).

When cache object is created it is initialized with access time (hit time) of the cache. The other parameters are size of the cache memory in bytes, cache block size and rate of associativity (setting associativity to one implies direct mapped cache). The last parameter is a pointer to a lower-level memory object. In a simple case with a single cache the lower level is the main memory.

With multiple caches you need to create the object starting from the lowest level. First create main memory object. Then create lowest cache level, for example L2. L2 gets a pointer to main memory. Then create next level, for example L1. L1 gets a pointer to L2.

Simulator prints a summary that shows the average access time and some key figures from caches. You need to modify that part to suit your tests.

What to test and to report

Run the traces (five files) with the values from previous exercise. Assume the following rates of associativity:

Nehalem: L1 2-way, L2 4-way, L3 8-way

Penryn: L1 2-way, L2 8-way

Report your findings and elaborate the results (access times, hits rates) per file and processor.