Computer Design & Organization
Assignment 5
Due: Wednesday, December 6

The purpose of this assignment is to give you experience in organizing a cache memory hierarchy. To this end, you will: profile a benchmark suite, explore the cache hierarchy design space, and choose the best design given a set of design constraints.

You should work in groups of three and, at the end of your report, describe who worked on what. Also explain your output file naming convention. You will be using three different simulators in this project. You can find everything you need to know about how to use them in the SimpleScalar Toolset Documentation on the course web page (except for how to use the victim cache, which is explained in a later section of this assignment).

Your benchmark suite consists of the following programs.

Name	Input
compress95.ss	< inputs/compress.in
perl.ss	inputs/charcount inputs/all_gre_words

Your study will involve the following three components.

Benchmark Characterization with `sim-profile`

You will be using the `sim-profile` simulator to study the instruction set usage of these benchmarks. `sim-profile` does not need a configuration file; you will simply pass options `–iclass` and `–iprof` on the command line. This initial component of the study should give you an idea of how much these applications will rely on the memory system to achieve good performance.

Issues you should investigate, analyze and discuss include:

- instruction class profile (-iclass option)
- type and frequency of memory instructions (-iprof option)

At the conclusion of this part of your study, you should have a solid understanding of how much and in what way each benchmark will depend on the memory system. Do these programs represent a typical workload, or is their instruction usage (or instruction mix) significantly different? You can find examples of typical instruction mixes (for a different instruction set) in H&P, on page 105. Based on this comparison, do you think a novel or radical memory system is called for? That is, based on what you have learned from H&P, is a standard cache design likely to suffice? Your report should include the results of your experiments and a discussion of what this characterization tells you about the importance and nature of a good memory system.

Exploring the Design Space with `sim-cache`

You will be using the `sim-cache` simulator to explore the range of options available for your cache design. The `sim-cache` simulator efficiently simulates the memory system and, for our purposes, only provides useful miss rates for caches. Since it does not reflect changes in execution time for programs, you will only use the simulator to investigate the effect of cache configuration parameter choices on cache performance (i.e., miss rate) independent of design constraints (after all, these can always change). Later in this assignment, you will balance the benefits against the costs of achieving improved cache miss rates when you consider the specific constraints imposed on your design.

The `sim-cache` simulator does not use a full-fledged configuration file; the configuration options are limited to those that are memory-specific. You configure your cache memory system by setting `-cache` arguments either in a configuration file or on the command line. We recommend that you set these parameters on the command line rather than creating a configuration file for each unique simulation you perform. The argument syntax, as well as several examples, can be found in the toolset documentation on the course web page. For these experiments, you only need to define your L1 and L2 caches; you can leave the TLB and other settings alone.

In your experiments, the following choices/parameters are fixed:

- Use an L1/L2 hierarchy
- L1 blocks should be 32 (bytes)
- L2 blocks should be 64 (bytes)
- Use separate instruction and data L1 caches
- Use the same configuration for your instruction and data L1 caches (same size and associativity)
- Use a unified L2 cache
- Use an LRU replacement policy

For the L1 and L2 caches in your system, you should find reasonable values for the following (both of which should be powers of two):

- cache size (= number of sets * associativity * block size)
- degree of associativity (find diminishing returns)

You will need to strike a balance between compress and perl performance; you are looking for a design that serves the needs of both benchmarks (**not** one cache design for each benchmark). The question that drives your experiments should be: what is the ideal choice for this cache parameter? Hint 1: For help getting started, review the cache chapter in your text. Hint 2: study the L1 caches first, then move on to the L2.

Your report should justify the parameter ranges you considered and clearly describe how you arrived at your reasonable parameter values (e.g., graphically depict reaching a point of diminishing returns). This component of the study should help you develop candidate designs worth considering under the constraints described in the next section.

Designing for Performance with `sim-outorder`

In the final component of your study, you will apply your expert knowledge and understanding of the memory characteristics of the benchmark suite to the task of developing a cache memory system under a set of design constraints. Rather than seeking to minimize miss rate solely, as in

the previous section, you will vary the cache memory configuration of `sim-outorder` in order to minimize total execution time. All other non-cache configuration parameters will remain at the default simulator values (that is, you don't need to set them). The same set of fixed choices/parameters from the last section hold here as well.

The simulators `sim-cache` and `sim-outorder` use the same cache configuration argument syntax. Since `sim-outorder` measures execution time, you also need to specify the hit latencies for your caches in the manner specified in the SimpleScalar toolset documentation. You will set the latencies according to the rules in the following constraints table:

Cache Option	Value	Hit Time (cycles)	Cycle Time (ns)
size	≤32KB	1	20
	<128KB and >32KB	2	20
	≥128KB	6	20
L1 associativity	direct-mapped	no penalty	+0
	2-way		+2
	4-way		+4
	8-way		+6
4-entry victim data cache	present	2	+0

Note that cycle time is not a simulator parameter. You will use this value in your calculation of total execution time. Penalty values preceded by "+" should be added to the size baseline cycle time value. **There is also an area constraint for your cache hierarchy: your total cache byte budget is 512KB** (you can use less, but no more).

You are permitted to use a 4-entry data victim cache in your design; there is an entry in the constraints table for a 4-entry victim cache which gives its hit latency. If necessary, review the victim cache material in H&P. The victim cache is enabled by passing the parameters `–cache:dvictim 4 –cache:dvictimlat 2"` to `sim-outorder` (these set the number of entries and latency, resp.).

In your report, *clearly describe* and j*ustify each parameter decision* in your final cache design proposal. You should describe the approach you took in choosing the best design. Compare your final design, which considers hit time, cycle time and chip area constraints, to what you would have chosen just based on miss rate. Discuss which decisions impacted overall performance the most. In which cases did the constraints keep you from choosing the ideal parameter value? How successful was miss rate in choosing the right parameter choice given these constraints? Whether you included it in your final design or not, what effect did the victim cache have on performance?