ECE 485/585
Fall 2009
Final Project Description

Your team is responsible for the design and simulation of an L2 cache for a new core 32-bit processor with shared memory. You've decided on a 4-way set associative cache using write-back and write allocate policies. However, your team has not decided on a replacement policy and wants to explore alternatives.

Design and simulate a cache as described above using the following three difference replacement policies and measure the performance of the cache on each.

- Random
- LRU
- Pseudo-LRU

Your design does not need to support a shared memory protocol (e.g. MESI).

Describe and simulate your cache in Verilog, C, or C++. If using Verilog, your design does not need to be synthesizable. (Note, if using the Synapticad tools you'll need Verilogger Extreme for Verilog 2001 support for file I/O).

Maintain and report key statistics of cache usage and print them upon completion of execution of each trace. These include the total number of memory references, the number of reads, number of writes, total number of cache hits and misses, and hit ratio.

You must submit a report which includes a design specification that describes the interface to the cache module (both processor side and front side bus side), relevant internal design documentation, the source modules for your cache, any associated modules used in the validation of the cache (including the testbench if using Verilog), and your simulation results. You do not need to worry about arbitration.

Make (and document) reasonable assumptions about the processor and memory subsystem and their interfaces. The report should justify these assumptions as well as any design decisions you make. Be sure to state any assumptions about the L1 cache as well. Your cache line size must be greater than four bytes.

Your testbench must read memory accesses from a text file of the following format:

n address

Where n is
0 read data request from L1 cache
1 write data request from L1 cache
2 instruction fetch (treated as a read request from L1 cache)

The address will be a hex value. For example:

2 408ed4
0 10019d94
2 408ed8
1 10019d88
2 408edc

Your grade will be based upon:

- External specification
- Completeness of the solution (adherence to the requirements above)
- Correctness of the solution
- Quality and readability of the project report (e.g. specifications, design decisions)
- Validity of design decisions
- Quality of implementation
- Structure and clarity of design
- Readability
- Maintainability
- Testing
- Presentation of results

Extra Credit

There are two possibilities for extra credit. They are not mutually exclusive; you can do both. However, separate submissions indicating which extra credit is being sought are required for each.

Extra Credit A

You can receive up to 10% extra credit by modifying your design to support a shared memory multiprocessor environment through the use of the MESI protocol. In this case, your trace files should look like the following:

n address

Where n is

0 read data request from L1 cache
1 write data request from L1 cache
2 instruction fetch (treated as a read request from L1 cache)
3 snooped read from another processor
4 snooped invalidate (write or read with intent to modify) from another processor

Your design must respond appropriately to snooped transactions and produce signals which can be snooped by others.

Extra Credit B

Using any cache organization you like (line size, associativity) and any combination of policies (e.g. write allocate/write-no allocate, write-back/write-through) but a maximum of 2MB (including data, tags, and any other required storage), achieve the best hit ratio you can. You do not need to support the MESI protocol. Those achieving better than the class median on the basic assignment will receive up to 10% extra credit. The team with the best hit ratio will also receive a prize. Submissions exceeding the 2MB of storage constraint will not be considered.