ECE 485/585: Microprocessor System Design

Spring 2016 TERM PROJECT

The project will be carried out in groups of two or three students. Final project report is due on Friday, June 10 11:59:59 PM.

Project Objective

The goal of the project is to simulate a single level of cache in software. You will write your own simulator in a software platform of your own choice (such as C, C++, JAVA, VHDL, Verilog etc.). The simulator will take the provided input address trace as its input. It will implement the workings of the cache. Specifically, it will keep track of which blocks are brought into the cache and which blocks are being evicted. At the completion of the trace, the simulator will provide statistics about cache hit ratio, read traffic, write traffic etc.

Note that there is no need to model the actual data stored in the cache. The input trace contains information only about the addresses that are being accessed. Since there is no data information in the trace, there is no way to model the data contents any way.

Project Details

Please read the following details carefully and follow them, when implementing your simulator:

1. **Input trace:** The simulator will take an address trace as its input. This trace captures the sequence of accesses that are being made to the cache. For each access, the trace provides the type of access and the memory address that is being accessed.
   Each line in the trace has the following format:

   Access_type Hex_Address

   Access_type signifies the type of the memory operation (read or a write). It is encoded as a binary digit. Access_type = 0 indicates a read and Access_type = 1 indicates a write.

   Hex_address is the address which is being accessed. The address is encoded in the hexadecimal (base-16) format. Each address is a 32-bit number (8 hexadecimal digits) and signifies the byte which is being accessed.

   The two fields in each line are separated by a whitespace.

   Example 1: Let us assume that the first line in the address trace is as follows:
   0 00A53C00
   This represents a read request to the byte address 0x00A53C00

   Example 2: Let us assume that the second line in the address trace is as follows:
   0 0044DB20
   This represents a write request to the byte address 0x0044DB20

2. **Cache Parameters:** Your model should support the following cache parameters to be configurable:

- Number of sets
- Associativity (Number of ways)
- Cache line size

To simplify the implementation, you should assume that each of the above three parameters will always be a power of 2. Also, the maximum associativity is 8 and the cache line size should range from 32 bytes to 128 bytes.

In addition, the modeled cache should implement a (true) LRU replacement policy. When a new line is brought into the cache, it should first look to occupy an invalid cache block. If there is no invalid block, then the LRU line in the cache should be evicted to make room for the new line.

As far as writes are concerned, you should assume a write-back cache with a write-allocate policy.

You don't need to implement any timing details (such as hit time, miss penalty etc.) for the cache.

3. **Simulator output:** After the last access in the trace has been simulated, the simulator should output the following statistics:

    - Total number of cache accesses
    - Number of cache reads
    - Number of cache writes
    - Number of cache hits
    - Number of cache misses
    - Cache hit ratio
    - Cache miss ratio
    - Number of evictions
    - Number of writebacks

4. **Project Evaluation:** Once you have understood all the above details, you are ready to implement the simulator. Once you have implemented your simulator, you'll be provided a set of input traces (called evaluation traces) which will be used to evaluate your project submissions. In the meantime, you can develop your own short test traces (following the trace format described above) to validate the correctness of your simulator.

    A major portion of your project grade will be based on the accuracy of your simulator output for the evaluation traces. You will be required to include your simulator results in the final project report. The evaluation traces and the details regarding which cache configuration to evaluate will be provided later (approximately a week before the project submission deadline).

Project Deadlines

- Inform instructor about your project group: Friday, May 13
- Project Q&A: Thursday, June 2 (automatic 1-week extension, if needed)
- Project report due: Friday, June 10