Computer architecture

Homework week 7

Instructions

Submit by e-mail to the lecturer, as a PDF document with your name and student ID near the beginning. You can work in groups of 2, however, use different groups than week 4, 5 and 6. Use the English language. Deadline: Oct 21st, 23:59.

Answers to bonus questions are counted only if you get at least half the points to each previous question.

Question 1 - Cache vocabulary (2pt)

For each of the following pairs of terms, describe in 1 sentence the difference between the two:

- *direct mapped* vs. *set-associative* caches.
- *write policy* vs. *replacement policy*.
- *hit* vs *miss*.
- line *index* vs *tag*;

Question 2 - LRU variants (2pt)

There are many variants to the pure LRU replacement policy for set-associative caches. For example, 2Q, LRU/2, Least Frequently Used, Random Replacement, etc.

Using own research, select 2 possible variants of your own choice, determine by whom and when they were devised, and a short summary (1-2 sentences) of their known advantages and limitations.

Question 3 - 2-way associative caches (2pt)

A 2-way set associative cache with LRU policy can be much simpler and cheaper to implement than a cache with higher associativity. Why? (max. 1 paragraph)

Question 4 - Caching and virtual addressing (2pt)

Virtual address translation entails a question of whether the tags in cache use virtual or physical addressing.

Explain the terms *homonyms* and *synonyms* in the context of caching and virtual addressing in your own words (1-2 sentences for each). Provide examples to illustrate.

Question 5 - Hardware vs software TLBs (2pt)

Explain the difference between *hardware-managed* and *software-managed* TLBs in your own words (1-2 sentences for each). Give 2 example names of processor architectures for each.

Question 6 (Bonus) - Victim caches (2pt)

Explain what a *victim cache* is in your own words (max 1 paragraph). Give the name of one architecture where it was used in hardware.

Question 7 (Bonus) - Address to cache index translation (2pt)

Background

MGSim offers multiple algorithms to select which bits in an address serve as index to select a cache set.

The "simplest" agorithm was seen during the lecture, it is called "DIRECT" in MGSim. It simply extracts the low-order bits of the address to serve as index.

The other algorithms, eg "RMIX", "XORFOLD", "XORLSB", combine more bits from the address. You can see how this is performed in the source code for the file `arch/BankSelector.cpp`.

Question

Explain in your own words why it can be desirable to use such an index selection algorithm, although it is more expensive to implement in hardware.