TDTS08: Advanced computer architecture

Lesson

2010
Outline

- Lab organization and goals
  - SimpleScalar architecture and tools
  - Assignment on multiprocessor systems
  - Break
  - Exercises
Organization

- Assistants
  - Group A: Soheil Samii
  - Group B: Ke Jiang
  - Group C: Jakob Rosén

- Web page
  - http://www.ida.liu.se/~TDTS08
  - Check the lab pages
Organization

- Sign up in Webreg (you have already done that!)
- Deadline for all lab assignments: January 9, 2011
- Rules: Read them! (linked from the lab pages)
Examination

- Written report for each lab
  - Hand in the report enclosed in a lab cover; it must be signed by all group members
    - Hand in at a lab session
    - Put in the box outside your assistant’s office
  - Returned in the box outside your assistant’s office
Labs

- 5 lab assignments
  1. Cache memories
  2. Pipelining
  3. Superscalar architectures
  4. VLIW processors
  5. Multiprocessor systems

- Labs homepage
  - http://www.ida.liu.se/~TDTS08/labs
Goals

- Obtain knowledge about computer organization and architectures
- Insights in various trade-offs involved in the design of a processor
- Become familiar with a set of tools necessary for evaluation of computer architectures
  - Simulation tools!
Environment

- Unix
- Simulations are started from the command line

⚠️ If you are not familiar with the Unix environment:
  - Search the Internet
  - Tutorials
  - List of basic commands
  - Make sure that you learn the basic commands in order to be able to work in a command-line environment
Outline

- Lab organization and goals
  - SimpleScalar architecture and tools
- Assignment on multiprocessor systems
- Break
- Exercises
Introduction

Issues covered in our lab:

- Cache memories
- Instruction pipelining
- Superscalarity
- VLIW (Very Long Instruction Word) processors
- Multiprocessor and Multi-computer systems
Architecture Simulation

Software

Hardware

SimpleScalar

Sun ULTRAx 10 PC, etc.

Real HW

HW simulator (software)
SimpleScalar: Literature

- “The SimpleScalar Tool Set, Version 2.0” by Doug Burger and Todd M. Austin
  - Very important!! Read it as a preparation for the labs
  - Used all the time during all labs!

- User’s and Hacker’s guide (slides by Austin)

- Linked from the lab pages
SimpleScalar Architecture

- Virtual architecture derived from MIPS-IV
- SimpleScalar ISA semantics are a superset of MIPS
  - Control (j, jr,..., beq, bne,...)
  - Load/Store (lb, lbu, ...)
  - Integer Arithmetic (add, addu, ...)
  - Floating Point Arithmetic (add.s, add.d, ...)
  - Miscellaneous (nop, syscall, break)
- little/big-endian instruction set definition
### SimpleScalar Architecture (cont’d)

#### Instruction encodings (64 bits)

**Register format**

<table>
<thead>
<tr>
<th>Field</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>16-annotate</td>
<td>16</td>
</tr>
<tr>
<td>16-opcode</td>
<td>16</td>
</tr>
<tr>
<td>8-rs</td>
<td>8</td>
</tr>
<tr>
<td>8-rt</td>
<td>8</td>
</tr>
<tr>
<td>8-rd</td>
<td>8</td>
</tr>
<tr>
<td>8-ru/shamt</td>
<td>8</td>
</tr>
</tbody>
</table>

**Immediate format**

<table>
<thead>
<tr>
<th>Field</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>16-annotate</td>
<td>16</td>
</tr>
<tr>
<td>16-opcode</td>
<td>16</td>
</tr>
<tr>
<td>8-rs</td>
<td>8</td>
</tr>
<tr>
<td>8-rt</td>
<td>8</td>
</tr>
<tr>
<td>16-imm</td>
<td>16</td>
</tr>
</tbody>
</table>

**Jump format**

<table>
<thead>
<tr>
<th>Field</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>16-annotate</td>
<td>16</td>
</tr>
<tr>
<td>16-opcode</td>
<td>16</td>
</tr>
<tr>
<td>6-unused</td>
<td>6</td>
</tr>
<tr>
<td>26-target</td>
<td>26</td>
</tr>
</tbody>
</table>
SimpleScalar Architecture (cont’d)

- Registers
  - 32 integer registers + PC, HI, LO
  - 32 floating-point registers + FCC

- Virtual memory:
  - 0x00000000 – 0x003fffff  unused
  - 0x00400000 – 0x0fffffff  text (code)
  - 0x10000000 – .............  data
  - ............. – 0x7ffffc000  stack
  - 0x7ffffc000 – 0x7fffffff  Args and Env
SimpleScalar: tools

- Several simulators
  - **Sim-fast**: Fast, only functional simulation (no timing)
  - **Sim-safe**: Sim-fast + memory checks
  - **Sim-cache**: Sim-safe + cache simulation and various timing properties (simulation time, measured time, …)
  - **Sim-cheetah**: Simulation of multiple cache configurations
  - **Sim-outorder**: Superscalar simulator
SimpleScalar Tools (cont’d)

- Tool set installed in ~TDTS08/bin
- Configurable through command-line arguments or files (recommended):
  - dumpconfig <filename>
  - config <file-name>
- gcc cross-compiler available for generating binaries to be executed on SimpleScalar
- Binaries have been generated and are available in the course directory
Demonstration

1. Autoset the path

```bash
gedrix <348> echo setenv PATH \"\$PATH\":/home/TDTS08/bin >> .login
```

2. Assignment 2

   a) Copy the files from the course folder

```bash
gedrix <319> mkdir lab1
gedrix <320> cd lab1/
gedrix <321> cp ~TDTS08/www-pub/labs/cache_memories/2/* ./
gedrix <322> ls
cache1.cfg Makefile test1.ss test2.ss
cache2.cfg test1.c test2.c
```

(Use "cp -r" to copy the whole contents of a directory, including all subdirectories)
Demonstration

b) Run simulation of *test1.ss* with configuration *cache1.cfg*

```
ceedrix <354> sim-cache -config cache1.cfg test1.ss
```

c) You will get the following output

```
sim: ** simulation statistics **
sim_num_insn        505292260 # total number of instructions executed
sim_num_refs         87822850 # total number of loads and stores executed
sim_elapsed_time     89   # total simulation time in seconds
sim_inst_rate        5677441.1236 # simulation speed (in insts/sec)
dll1.accesses        87823201.0000 # total number of accesses
dll1.hits            87022637 # total number of hits
dll1.misses          800564 # total number of misses
dll1.replacements    799540 # total number of replacements
dll1.writebacks      274082 # total number of writebacks
dll1.invalidations   0   # total number of invalidations
dll1.miss_rate       0.0091 # miss rate (i.e., misses/ref)
dll1.repl_rate       0.0091 # replacement rate (i.e., repls/ref)
dll1.wb_rate         0.0031 # writeback rate (i.e., wrbks/ref)
dll1.inv_rate        0.0000 # invalidation rate (i.e., invs/ref)
```
Demonstration

3. Assignment 3

a) Dump the default configuration of *sim-cheetah* to *config-file* and look into the default configuration

```
gedrix <355> sim-cheetah -q -dumpconfig config-file
```

Part of the configuration file:

```
# reference stream to analyze, i.e., {inst|data|unified}
-ref  data

# replacement policy, i.e., lru or opt
-R  lru

# cache configuration, i.e., fa, sa, or dm
-C  sa

# min number of sets (log base 2, line size for DM)
-a  7

# max number of sets (log base 2, line size for DM)
-b 14

# line size of the caches (log base 2)
-l 4

# max degree of associativity to analyze (log base 2)
-n 1
```
**Demonstration**

b) Modify the configuration file according to the instructions using any text editor you prefer

c) Run sim-cheetah with modified config-file

```
$ gedrix <358> sim-cheetah -config config-file ~/TDTS08/spec95-big/go.ss 3 7
```

d) Sample results:

- **Plot the results**
  - OpenOffice
  - Gnuplot
  - Matlab
  - Excel
  - Your favorite plotting tool

<table>
<thead>
<tr>
<th>No. of sets</th>
<th>Associativity 1</th>
<th>Associativity 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>0.122619</td>
<td>0.096127</td>
</tr>
<tr>
<td>256</td>
<td>0.109566</td>
<td>0.093854</td>
</tr>
<tr>
<td>512</td>
<td>0.102443</td>
<td>0.092651</td>
</tr>
<tr>
<td>1024</td>
<td>0.097429</td>
<td>0.085947</td>
</tr>
<tr>
<td>2048</td>
<td>0.083881</td>
<td>0.064152</td>
</tr>
<tr>
<td>4096</td>
<td>0.065504</td>
<td>0.061119</td>
</tr>
<tr>
<td>8192</td>
<td>0.061401</td>
<td>0.058600</td>
</tr>
<tr>
<td>16384</td>
<td>0.059216</td>
<td>0.056511</td>
</tr>
</tbody>
</table>
Demonstration

![Graph showing miss ratio vs. number of sets for different associativities. The x-axis represents the number of sets in log2 scale, ranging from 128 to 16384. The y-axis represents the miss ratio in percentage, ranging from 0 to 12%. Two lines are plotted for different associativities: Associativity 1 and Associativity 2. The graph illustrates the decrease in miss ratio as the number of sets increases.]
Outline

- Lab organization and goals
- SimpleScalar architecture and tools
  - Assignment on multiprocessor systems
- Break
- Exercises
Lab 5: Multiprocessor Systems

- Assignment:
  - Select an article on a multi-core, multiprocessor, multi-computer system, or a graphics processor
  - Review the selected article
  - Write a review report on the article

- List of papers is available on the course page
- You may select other articles if your lab assistant agrees
Multiprocessor Systems (cont’d)

- Read and understand the paper
  - If the course literature does not help you, investigate the referenced papers.
  - Searching the Internet can help you find explanations of abbreviations and terms
Multiprocessor Systems (cont’d)

- Analyze the paper
  - Classify the architecture (MIMD, SIMD, NUMA)
- Possible questions to ask:
  - Why has the actual method/approach been selected?
  - What are the advantages and disadvantages?
  - What is the application area?
  - What has been demonstrated?
  - …
Multiprocessor Systems (cont’d)

- Write a summary
  - ~1000 words
- Submit, in PDF format, to your lab assistant
  - sohsa65.liu@analys.urkund.se (Soheil Samii)
  - kejiang.liu@analys.urkund.se (Ke Jiang)
  - jakro62.liu@analys.urkund.se (Jakob Rosén)
Outline

- Lab organization and goals
- SimpleScalar architecture and tools
- Assignment on multiprocessor systems
  ➔ Break
- Exercises
Exercises

- We have selected some assignments from the 8th edition of the course book (Stallings)
  - Review questions (page 146)
    - 4.4 and 4.8
  - Problems (pages 147–149)
    - 4.8 (preparation for lab 1, mandatory)
      - Include your solution for this assignment in the report for lab 1
  - Additional exercises in this order:
    - 4.15 (locality)
    - 4.22 (average memory-access time)
    - 4.17 (replacement policy)