Interactive High-Performance Processor Understanding Via the Web

Augustus K. Uht
Sean Langford
Dept. of Electrical and Computer Engineering

David Morano
Dept. of Electrical and Computer Engineering

UNIVERSITY OF Rhode Island

SSGRRw January 23, 2002
Acknowledgements

• Work supported by the Intel Corporation, URI, NSF, Mentor Graphics, Xilinx, VCC.
• Simulator by Dave Morano, Ali Khalafi and Marcos de Alba.
• Other members of the Levo team:
  – Tom Wenisch, Prof. David Kaeli (NEU)
• Constant advice and editing:
  – Laurette Bradley
Outline

1 Motivation
2 CPU Performance Basics
3 Instruction Level Parallelism (ILP)
4 CPU Overview
5 Architecture
6 Examples
7 Online or Offline Demo
8 Summary
Motivation

- CPU chip *complexity* high and growing
  - 10’s of millions of transistors → billions
- → *functional verification* costs growing
- → *time-to-market* excessive
- → *education* difficult
- → *debugging* difficult
Related Work

• Many specialized simulators, e.g., IBM BRAT
  – Not readily adaptable to other machines
• “General Purpose” visualizers, e.g., Stanford Rivet
  – May be adaptable to many types of systems
  – Scalability is an issue: much state
  – Often, adaptability is through custom scripts
    • LevoVis based on std. XML and SVG
  – Not readily accessible; LevoVis is Web-based
CPU Performance Basics

- Two elements to processor performance - $P$:
  - Clock frequency – $f$
    • Technology dependent
  - Instructions executed per cycle – $IPC$
    • Architecture and Implementation dependent

- Fundamental relation:
  $$P = f \times IPC$$

- Focuses on high IPC via ILP →
Instruction Level Parallelism (ILP)

• Execute more than 1 instruction per cycle
• Example:

1. A = B + C
2. D = E + F
3. G = A + H

instructions 1 and 2 can execute in parallel;
1 and 3 cannot (data dependency)
CPU Overview

• Uses modification of Tomasulo algorithm
  – (The original algorithm dates to 1964 and is used today in the Intel Pentium Pro, II, III & 4.)
• Instruction *time tags* enforce dependencies
• *Active Stations (AS)* hold instructions & data
• Communication buses segmented
  – → Short delay, high *f*
  – *Register Filter/Forwarding Units (RFUs)* link segments
Overview

- Based on XML and SVG
- SVG used to generate graphics
- XML links graphics with simulation data
- Simulation data files very large
  - Kept on server
  - Data for individual cycles (10) brought over Web as needed
- **User able to navigate to arbitrary cycles**
- **Arbitrary display of machine elements**
Architecture
Data-to-Graphic Mapping Example

<table>
<thead>
<tr>
<th>Simulation Data XML</th>
<th>SVG Graphic Component XML</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;uid&gt;</code></td>
<td><code>&lt;g id=&quot;uid&quot;&gt;</code></td>
</tr>
<tr>
<td><code>ffbe8d20</code></td>
<td><code>&lt;tspan x=&quot;36&quot; y=&quot;0&quot;&gt;ffbe8d20&lt;/tspan&gt;</code></td>
</tr>
<tr>
<td><code>&lt;/uid&gt;</code></td>
<td><code>&lt;/g&gt;</code></td>
</tr>
</tbody>
</table>

Gives

![Instruction Request Graphic](image)
Online or Offline Demonstration

• ONLINE: Goto → WWW LevoVis

• OFFLINE: Goto → local LevoVis
Summary

- Many flexible state visualization capabilities
- Ideal for complex CPU’s
- Usable for any synchronous digital system
- Gives understanding, analysis and debugging for researchers, students and engineers
- Allows world-wide concurrent access (Web-based)
- Adaptable to new systems or system requirements
Relevant Web Sites

Levo links:

www.ele.uri.edu/~uht
Or: www.levo.org

LevoVis direct:

ovel.ele.uri.edu:8080