ss-viz

A SimpleScalar Visualizer

SimpleScalar 4.0 Micro34 Tutorial

Bill Yoder
Doug Burger
Steve Keckler

Jacob Sarvela
Pradeep Desai
Jinhuo Liang

December 2, 2001
University of Texas at Austin

Supported by
NSF CADRE

ss-viz
Project Goals

- Provide an easy-to-use graphical front-end to the SimpleScalar engine.
- Serve both researchers and students.
- Illustrate resource usage and identify bottlenecks.
- Let users examine μprocessor behavior without having to understand simulator internals.
- Support tinkering with different processor configurations.
- Run at reasonable speeds.

Supported by
NSF CADRE

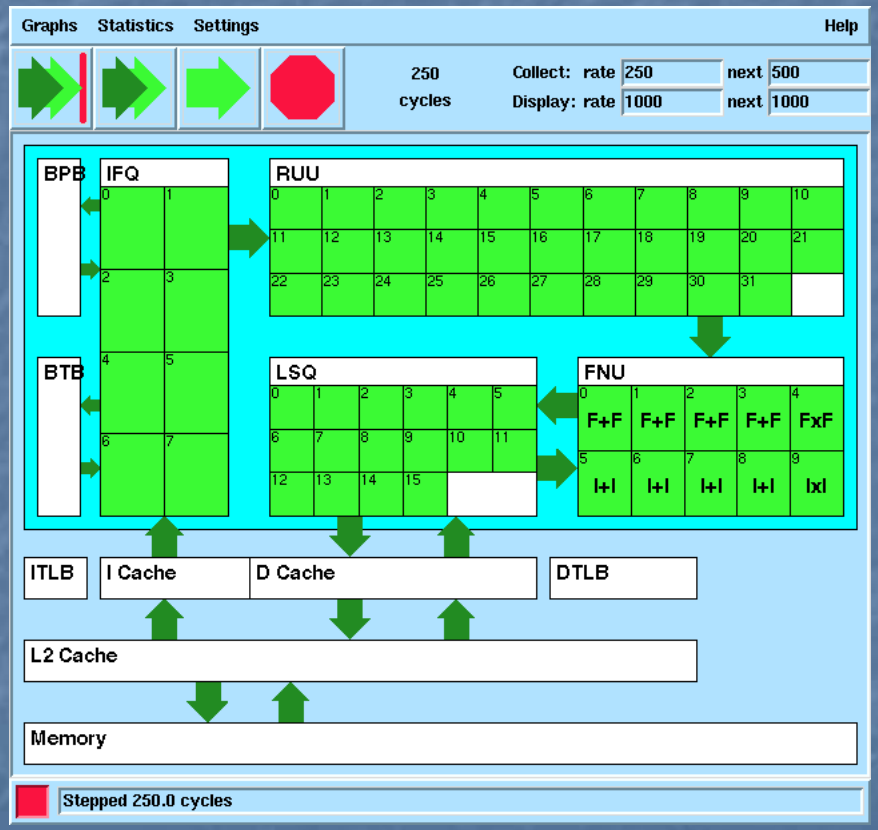
Visualizer Features

- Provides single-stepping, discrete stepping, and click-to-pause execution.
- Animates the activity of the IFQ, RUU, LSQ, and arithmetic units.
- Provides real-time graphical output.
- Loads and runs all SimpleScalar benchmarks.
- Provides statistics from each execution run.
- Includes on-line help.

Supported by
NSF CADRE

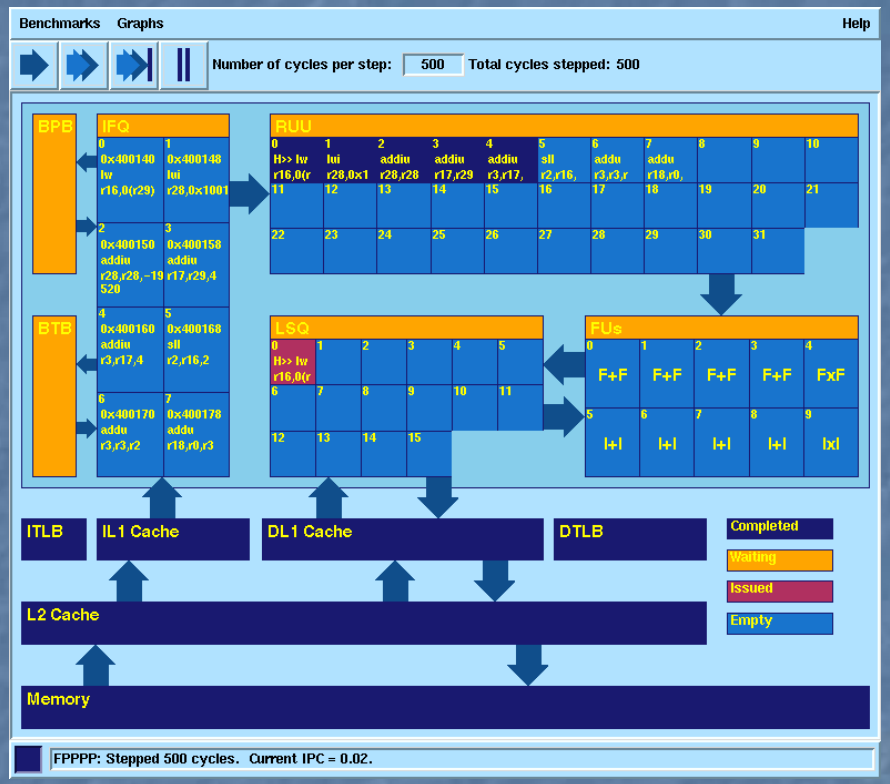
Feedback on the Prototype

Spring 2001: From two dozen engineering students



- ✓ UI concept
- ✓ Execution graphs
- ✓ Statistical info
- ☹ Graphic design
- ☹ Operation
- ☹ Online help

Today's Status



- GUI refurbished with better colors.
- Simplified start-up and user interaction.
- Four units animated (IFQ, RUU, LSQ, FUs).
- HTML help page.
- Various bugs fixed.

Software Design

- The Visualizer back-end is the SimpleScalar out-of-order issue superscalar processor (sim-outorder) with a 2-level memory system and speculative execution support, implemented in UNIX/C.
- The GUI is written as an X11R6 Windows application using the Tcl/Tk toolkit (`libtcl8.3.a` and `libtk8.3.a`).
- The Tcl/C interface probes the simulator for run-time configuration information, statistics, and machine state.
- The front-end displays this information using the Tcl interpreter and the Tk canvas widget.
- Dialogs, push buttons, and menus invoke UI callback functions to control application behavior (e.g., to resume program execution) and modify settings (e.g., graph units).

Supported by
NSF CADRE

Software Block Diagram

main.c → sswish.c
sim_main() → Tk_Main()

Ss_Init.c (hooks into SimpleScalar)

sim-outorder.c
- options db
- statistics db
- IFQ
- RUU
- LSQ
- FUs
- sim_step(num_steps)

Tcl – **language and interpreter**
- **system**
- **strings**
- **math**
- **unit structures**
- **stepping**
- **stats**

Tk – **windows and widgets**
- **tool buttons**
- **menus**
- **graphs**
- **fonts, colors**
- **canvas**
- **pop-ups**

*.tcl source files

X Windows
- Display
- Keyboard
- Mouse

Benchmarks and input data

Supported by NSF CADRE

Future Development

- Portability
  - Package for Solaris/Sparc.
  - Port to Linux/x86.
- Functionality
  - Animate more units, e.g., the L1 and L2 caches.
  - Expose more simulator resources for easy configuration (e.g., the number and type of FUs).
  - Expand on-line help.
  - Enable back-stepping (?!)
- Robustness
  - Improve Tk window management of graphs and window re-sizing.
  - Maintain GUI at benchmark termination.
- Truth and Beauty
  - Code cleanup
  - Internal documentation

(Feedback welcome!)

Supported by
NSF CADRE

Demo Notes

1. Use the VNC viewer on a laptop in order to connect to the VNC display server running on a SPARCstation.
2. Begin with the initial display, pointing out the components, menus, messages, and controls.
3. Show block stepping, single stepping, and continuous execution.
4. Show cell updates, with text and color fills.
5. Show statistics for the various units.
6. Show graphs and their dynamic updates.

Supported by
NSF CADRE