RISC-V Simulator
Final Project

Martin Schoeberl

Instruction Set Simulator

- A processor simulation in software
- To have a reference implementation
  - For compiler writer without access to the real hardware
  - As a golden reference for the hardware implementation
- Explore instruction usage
  - Easier to collect statistics in software
  - Simulate other features (e.g., caches)
- A valuable tool for computer architects
  - But keep in mind that a simulation is always a simplification

RISC-V Simulator

- Will give you a deep knowledge of the bleeding edge technology RISC-V
- You can execute any RISC-V program in your simulator
  - You could extend it till you can boot Linux ;-)
- Develop yourself what you have used so far

Organization

- Just instruction simulation, no pipeline
- Store the state of the processor
  - Registers, PC, memory
- Main execution loop:
  - Read instruction from memory
  - Decode instruction (switch statements)
  - Execute the instruction
- Finish with either end of program or ecall 10
- Use: Java, C, C++, Scala, …

We Provide

- A description of the assignment at:
  - https://github.com/schoeberl/cae-lab/tree/master/finasgmt
- Organized tasks
  - To help you to organize your work
  - With test cases
- A starting point in Java (or C++ or Chisel)
  - But feel free to start from scratch
  - Use any programming language you like!

Or in Chisel

- Single-cycle processor is basically an ISA simulator
- Code as shown in the book (Figure 4.15)
- Not a big case statement
  - But blocks with connections (datapath and control)
  - A steppingstone to a pipelined version
- You can also do this in VHDL or (System)Verilog

- Show Java and Chisel code (run)

Deliverable

- Report handed in
  - DTU Inside: Final Assignment
  - Due Su 04/12
  - Report as pdf plus source as .zip file
- Check working simulator with a TA
  - TA will provide test programs
  - Due latest 28/11
- Group of 1 to 2

Tips

- Print out register content for debugging
- Write your own assembler test cases
  - Share them with your fellow students!
  - Be social
- However, do not share your simulator code
  - Use a private repository for your group
  - Sharing is helping cheating
    - Sorry, that's the DTU rule

Want to Do More?

- If this is too easy, there are several ways to extend your work
  - Ecalls as in Ripes
  - Add a cache simulation
  - Further RISC-V instructions
  - Visual representation (like Venus)
  - Boot Linux
  - Simulate the pipeline
    - Including visualization
  - Make it a tool for future CAE teaching here and elsewhere in the world
    - Ripes started as a CAE simulator

Summary

- Instruction simulator is a useful tool
  - For compiler writer
  - For computer architects
- You will program your own RISC-V simulator
- The details are on the GitHub page
- We provide test programs for you
- Show your working simulator to a TA
- Hand in report and source code