The BOOM Project

Christopher Celio, Jerry Zhao, Abraham Gonzalez, Ben Korpan, Krste Asanovic, David Patterson
Outline

- Current state of the project
- How to get started
- How to contribute
- Things to work on
- Research ideas
- Conclusion and discussion
Project Maintainers

● Chris Celio is now at Esperanto Technologies
  ○ Works on the ET-Maxion out-of-order core

● Work at Berkeley
  ○ Jerry Zhao - Senior undergraduate
    ■ Maintenance + Keystone support + vectors
  ○ Abe Gonzalez, Ben Korpan - First-year graduate students
    ■ Ring Scalar design of BOOM
    ■ Keystone and security additions
    ■ General maintenance + Documentation updates
Recent/Ongoing Work

- Taped out as BROOM
- Integrated with FireSim - a open-source FPGA-accelerated simulation tool that runs on Amazon EC2 F1
  - Boot Linux on a multi-core BOOM with 16 GB DDR3, UART, Ethernet NIC in the cloud for 50 cents/hour at ~100 MHz
- Work towards BOOMv3 release
  - Memory model
  - RVC
  - Performance debugging
  - And more!
Outline

- Current state of the project
- *How to get started*
- How to contribute
- Things to work on
- Research ideas
- Conclusion and discussion
Getting Started

- Central location of information is the BOOM website and GitHub organization
  - Website: riscv-boom.github.io
  - GitHub: github.com/riscv-boom
- Website contents
  - Link to GitHub organization
  - BOOM v2 Specification
    - 70 pages of documentation on microarchitecture, future work, etc
  - Publications/Talks/Reports
  - Current team members
BOOM GitHub Organization

- **riscv-boom**
  - The core source code
  - Note: That elements of RocketChip are reused
- **boom-template**
  - Template for building new projects with the BOOM core
  - Includes all necessary materials to build/run the core
- **riscv-boom.github.io**
  - Project website
- **riscv-boom-doc**
  - Documentation for the core, project tools, etc
BOOM v2 Specification

- **Documentation on**
  - Core units/stages
    - Most detailed implementation resource outside of code
  - Repository layout and reused resources from RocketChip
  - Verification
  - Tools and tests
    - Using gem5 pipeline viewer
    - Torture tests and RISC-V tests
  - Physical realization information
  - Future work
    - Includes preliminary design ideas for each piece
- Will look to change to ReadTheDocs format
Outline

- Current state of the project
- How to get started
- How to contribute
- Things to work on
- Research ideas
- Conclusion and discussion
Contribute to the project

- **Bug reports**
  - Verify against Spike ISA simulator
  - Add issue with reproducible code snippets

- **Feature Additions**
  - Reach out to the Berkeley team to discuss major changes
    - Keep BOOM consistent
    - Avoid conflicting work
    - Bounce around ideas, get help, etc
  - Submit a pull request (PR)
    - Detailed description of changes
    - Don’t forget to update documentation alongside code
Outline

- Current state of the project
- How to get started
- How to contribute
- **Things to work on**
- Research ideas
- Conclusion and discussion
RoCC Support

- Enable support for co-processors and accelerators
  - Hwacha vector accelerator
  - Crypto units, DMA engines
  - + Your custom block
- Not trivial to implement
  - Adapt RoCC to an OoO machine
  - Where do instruction bits live?
  - One-at-a-time, stall until RoCC instruction is ready to commit
  - BoCC?
Memory System Changes

- Total Store Ordering and Weak Memory Ordering
  - Currently implements the relaxed consistency model
  - Update to support RVWMO, order loads to the same address

- Prefetchers

- Memory Disambiguation Predictor
  - Predict dependencies between different memory accesses

- Multi-ported data-cache
  - Commit multiple loads per cycle
RISC-V Compressed Instructions

- Enables smaller 16b encodings of common instructions
  - All 16b instructions map to a 32b longer instruction
  - 32b instructions can now be aligned to the half-word boundary

- Challenges
  - A few 16b instructions exhibit different behaviour than 32b instructions
    - Ex. JAL may return PC+2 or PC+4 depending on which version of the instruction it was
    - Must track if it was a 16b/32b instruction through the pipeline
  - Require code changes in several structures
    - Reorder Buffer (ROB)
    - Branch Predictor
    - More!
Verification

- **Tests**
  - Unit tests for particular sub-components/stages
  - More coverage on pipeline
- **Add co-simulation of commit trace against an ISA simulator**
- **Initial things to verify**
  - Specification compliance
  - Branch Prediction
    - TAGE
    - GShare
  - Much more!
- **Work on improving simulation visibility**
General maintenance

- Documentation additions
  - Update documentation for new additions
  - Transition to ReadTheDocs

- Centralized GitHub organization updates
  - Move all relevant BOOM repo’s here for easy access and visibility

- Refactoring
  - Update code for clarity, concision and maintainability
Outline

- Current state of the project
- How to get started
- How to contribute
- Things to work on
- Research ideas
- Conclusion and discussion
Microarchitectural Exploration

- Chisel + RISC-V ecosystem enables agile architecture research
  - Don’t settle on modelling; build and evaluate fully functional systems!
- BOOM is an important piece of groundwork
  - Can be leveraged to research high performance microarchitecture
  - 15,000+ lines of concise, partitioned code: feasible for one person to implement features
- Current ideas:
  - “RingScalar” integer execution core
  - Support for RISC-V vector extension
Hardware Security

- Good news? Not vulnerable to Meltdown or Foreshadow
- Bad news? Vulnerable to general Spectre style attacks
  - Actually good news
- BOOM - baseline for security research
- Leverage Chisel + RocketChip productivity to explore mitigations
  - Secure caches
  - Cache partitioning
  - Timing obfuscation
- A secure cache proof-of-concept is in the works
  - 1 man-week to block transient cache refills
- Enclaves are cool too
Outline

- Current state of the project
- How to get started
- How to contribute
- Things to work on
- Research ideas
- Conclusion and discussion
Conclusion

● Berkeley Out-of-Order Machine
  ○ Concisely expressed in ~15k LOC Chisel
  ○ RV64G + Priv v1.11
  ○ Synthesizable
  ○ Extensible
  ○ OPEN SOURCE!

● A lot of work to be done
  ○ Verification
  ○ Useful features

● Great opportunity to do make an impact
  ○ Architecture research
  ○ Work on an open source project
What do you want to do?

What would you like to see in BOOM and how can we help?
Questions and thanks!

- Get started today
  - Website: riscv-boom.github.io
  - GitHub: riscv-boom/riscv-boom

- Contact us
  - Google group: https://groups.google.com/forum/#!forum/riscv-boom

- Thanks to all who have worked on BOOM
Acknowledgements

- Research partially funded by DARPA Award Number HR0011-12-2-0016, the Center for Future Architecture Research, a member of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA, and ASPIRE and ADEPT Lab industrial sponsors and affiliates Intel, Google, Siemens, SK Hynix, Seagate, Huawei, Nokia, NVIDIA, Oracle, and Samsung. Approved for public release; distribution is unlimited. The content of this presentation does not necessarily reflect the position or the policy of the US government and no official endorsement should be inferred. Any opinions, findings, conclusions, or recommendations in this paper are solely those of the authors and does not necessarily reflect the position or the policy of the sponsors.

- Projects mentioned
  - Hwatcha: Website: http://hwacha.org/
  - RingScalar: Paper: https://dspace.mit.edu/handle/1721.1/34012