Tutorial Conclusion

Abraham Gonzalez

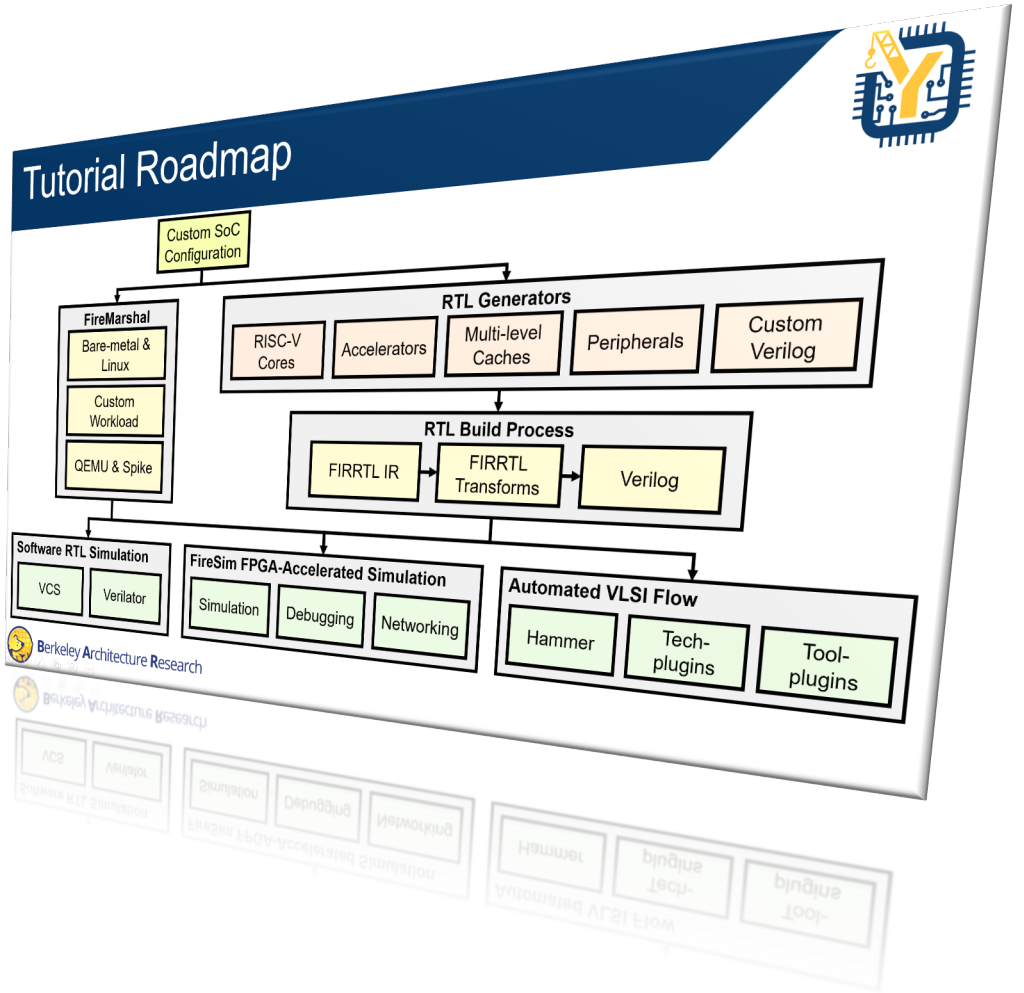
UC Berkeley

abe.gonzalez@berkeley.edu

Berkeley Architecture Research

Recap

- Chipyard Basics
  - Composing SoC using generators
  - Adding custom accelerators
  - Simulation
  - Prototyping
  - VLSI flow
- FireSim
  - Full-system FPGA-accelerated simulation
  - Linux-based software workloads
  - Debugging and instrumentation

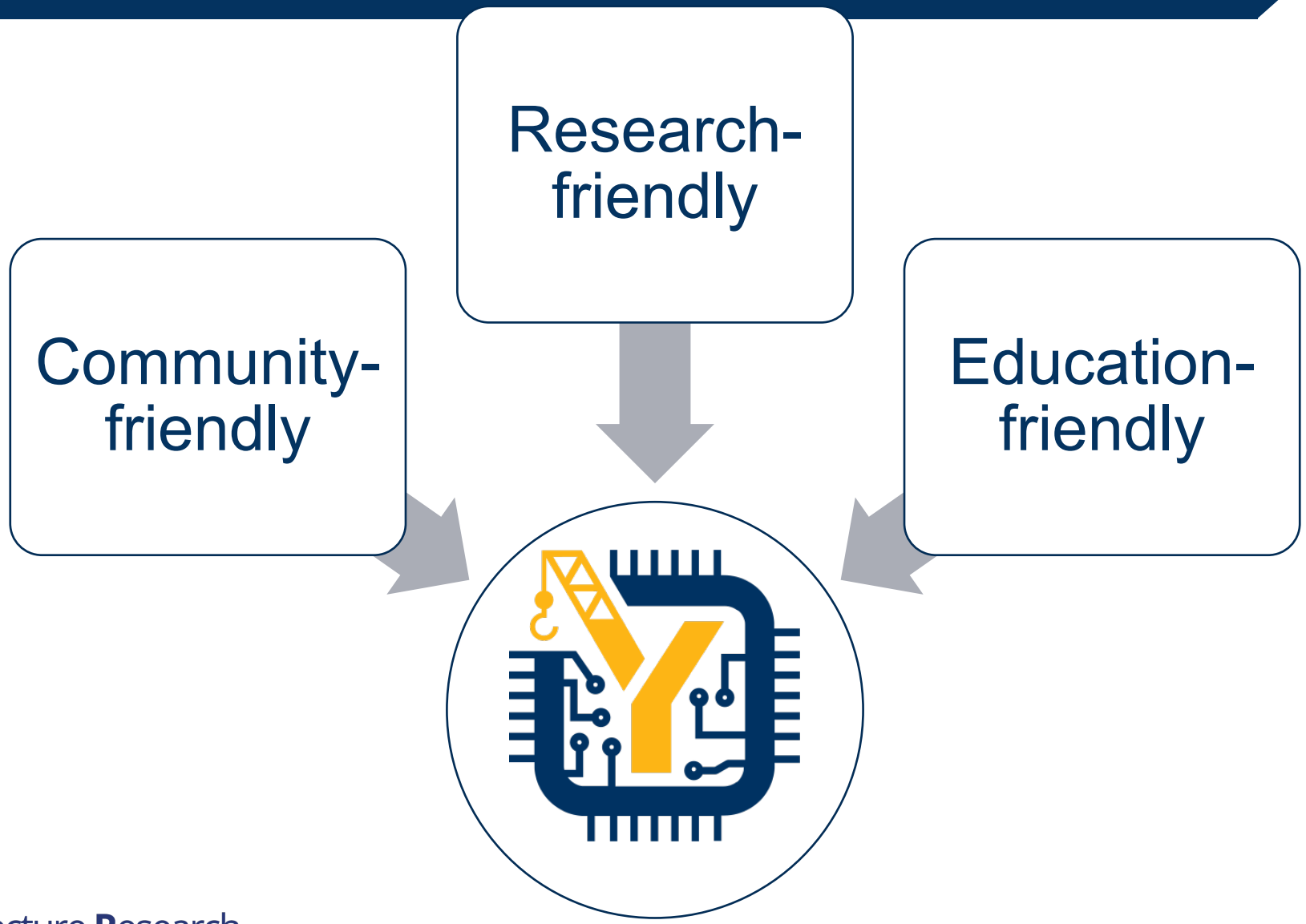


Future Development

- Local FPGA support w/ FireSim
- Better support for open-source EDA and fabrication
  - OpenRoad
- New cores + accelerators
- Mixed signal integration

Credit: https://medium.com/@jondishotsky/talking-houses-and-flying-cars-55c431c7f2ec

Chipyard Goals



Chipyard is Community-Friendly

Documentation:

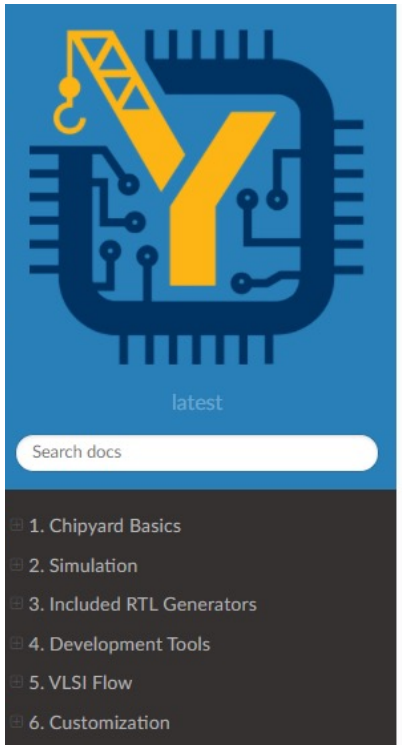
- https://chipyard.readthedocs.io/
- 133 pages ++
- Most of today's tutorial content is covered there

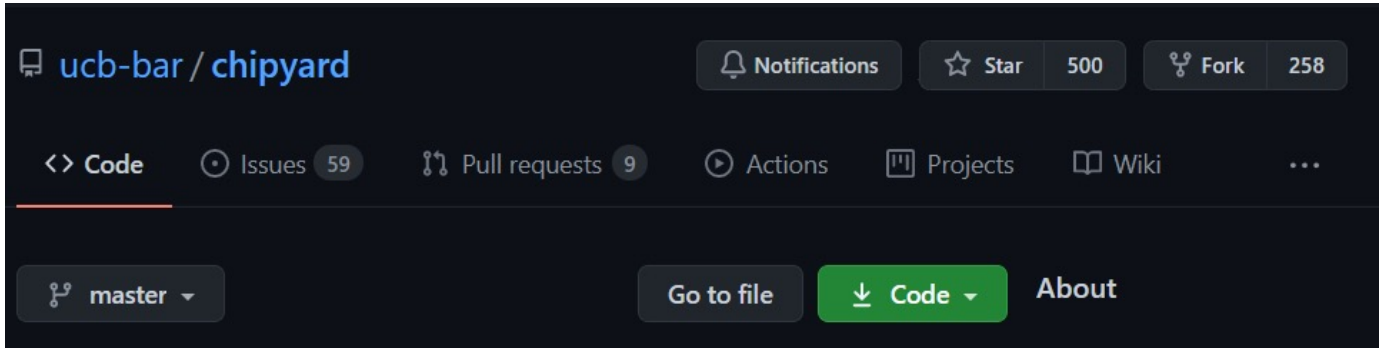
Mailing List:

- google.com/forum/#!forum/chipyard

Open-sourced:

- All code is hosted on GitHub
- Issues, feature-requests, PRs are welcomed





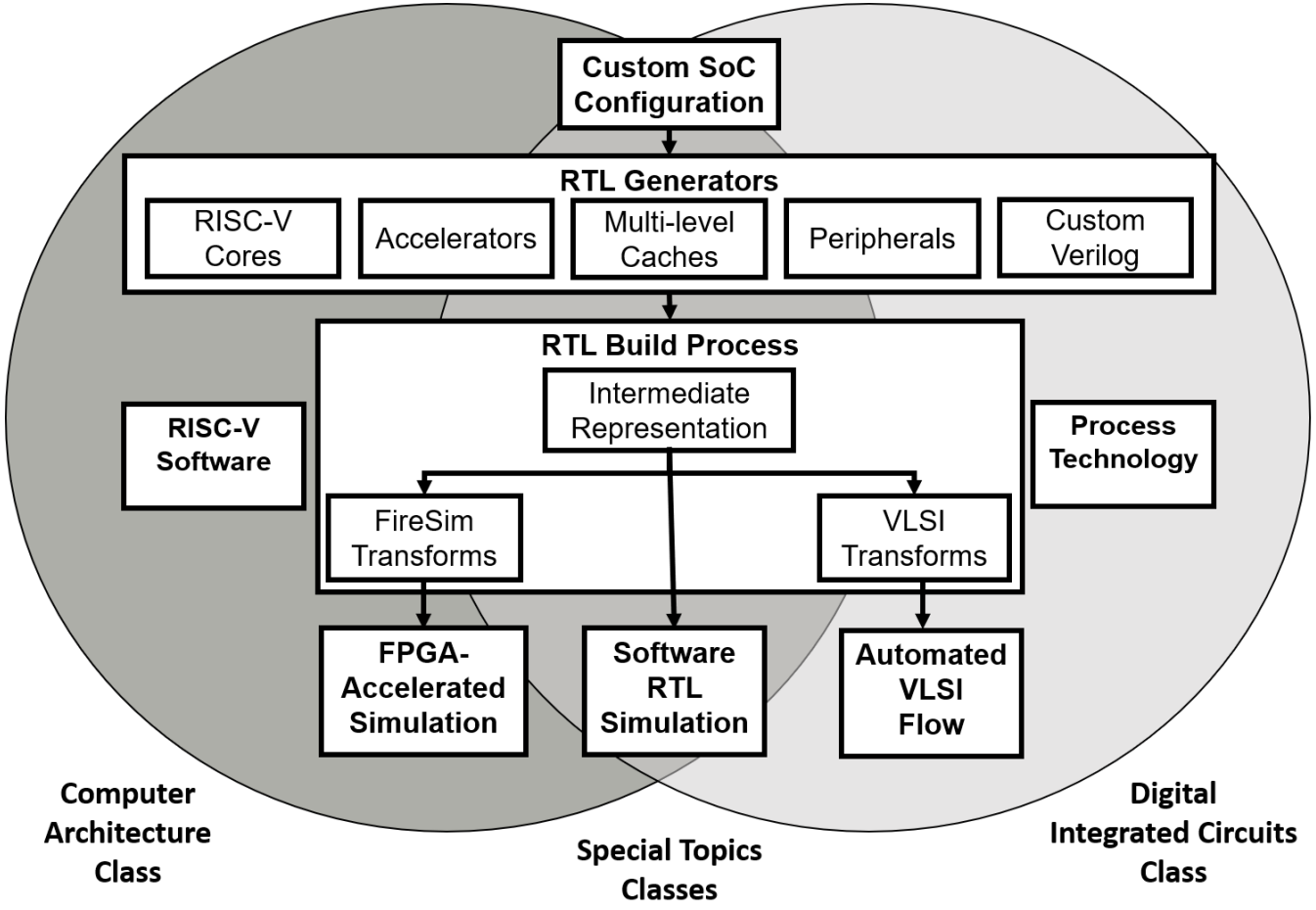
Chipyard is Education Friendly

Proven in many Berkeley HW/Architecture courses

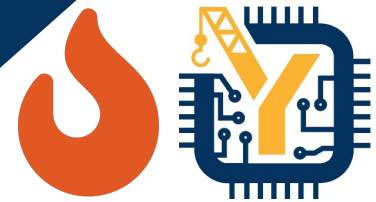
- Hardware for Machine Learning
- Undergraduate Computer Architecture
- Graduate Computer Architecture
- Advanced Digital ICs
- Tapeout HW design course

Advantages of common shared HW framework

- Reduced ramp-up time for students
- Students learn framework once, reuse it in later courses
- Enables more advanced course projects (tapeout a chip in 1 semester)



Chipyard is Research-Friendly



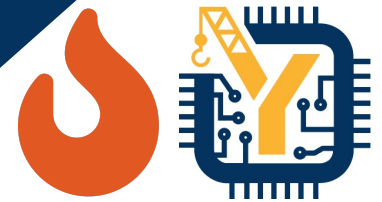
- Add new accelerators/custom instructions
- Modify OS/driver/software
- Perform design-space exploration across many parameters
- Test in software and FPGA-sim before tape-out

Numerous research projects built on Chipyard/FireSim…
Including MICRO'21 award winners

Best Paper Runner Up - TIP: Time-Proportional Instruction Profiling
Distinguished Artifact - A Hardware Accelerator for Protocol Buffers

https://dl.acm.org/doi/10.1145/3466752.3480058
https://doi.org/10.1145/3466752.3480051

Diversity of Uses



System Research

Keystone: An Open Framework for Architecting Trusted Execution Environments

Dayeol Lee
dayeol@berkeley.edu

David Kohlbrenner
dkohlbre@berkeley.edu

Shweta Shinde
shwetas@berkeley.edu

The nanoPU: Redesigning the CPU-Network Interface to Minimize RPC Tail Latency

Stephen Ibanez, Alex Mallery, Serhat Arslan, Theo Jepsen, Muhammad Shahbaz, Nick McKeown, Changhoon Kim

Stanford University

Abstract

The nanoPU is a new networking-optimized CPU designed to minimize tail latency for RPCs. By bypassing the cache and memory hierarchy, the nanoPU directly places arriving messages into the CPU register file. The wire-to-wire latency through the application is just 65ns, about 13× faster than the current state-of-the-art. The nanoPU moves key functions from software to hardware: reliable network transport, congestion control, core selection, and thread scheduling. It also supports a unique feature to bound the tail latency experienced by high-priority applications.

Our prototype nanoPU is based on a modified RISC-V CPU; we evaluate its performance using cycle-accurate simulations of 324 cores on AWS FPGAs, including real applications (MICA and chain replication).

paper is the *wire-to-wire latency*, defined as the time from when the first bit of an RPC request message arrives at the NIC, until the first bit of the processed RPC response leaves the NIC. The best reported median wire-to-wire latency is around 850ns [28]. Our goal is to reduce *both* median and tail numbers to below 100ns, making it worthwhile to run "nanoServices"; short RPCs requiring less than 1μs of work.

Many prior attempts to reduce RPC overhead have included low-latency and lossless switches [30, 35, 8], a reduced number of network tiers, and specialized libraries [28]. The current fastest approaches deploy dedicated NIC and switch hardware, but these are hard to program [25, 24, 26, 50].

Our work asks the question: *Can we design a future CPU core that is easy to program, yet can serve RPC requests with the absolute minimum overhead and tail latency?* Our design, which we call the *nanoPU*, can be seen as a model

Chips

Berkeley engineering students pull off novel chip design in a single semester

Class shows successful model for expanding entry into field of semiconductor design

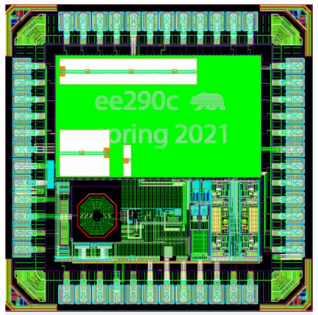
June 17, 2021

In what could have important implications for engineering education as well as the field of chip design, a class of Berkeley E… novel chip tha… course, 19 stu… end of a four-…

The current g… industry news… manufacturin… semiconducto… semiconducto…

The term "tape-out" refers to the process of recording a chip's final design and delivering it for fabrication — in this case, to the Taiwan Semiconductor Manufacturing Company. This used to be handled via magnetic tape but nowadays happens electronically, a digital file converted to a physical chip.

With the support of Apple, Nikolić, fellow professors Kris Pister and Ali Niknejad, and graduate student instructors Dan Fritchman and Aviral Pandey led 10 undergrads, five master's students and four Ph.D. candidates through the design and successful tape-out of a novel chip within the span of a single semester. It had never before been done at Berkeley.



Shown is a "tape-out" of a novel chip design completed by Berkeley Engineering students.

"It's a testament to our students, the teaching assistants and the faculty that we were able to pull it off, but also to the infrastructure for chip design that Berkeley has put together over the last decade," said Pister. "It's really quite remarkable, and most people that I talk to don't believe me when I tell them what we did."

RISC-V Development

Presentation

TEE Hardware for RISC-V

This session shows an exploration of RISC-V hardware generation to implement hardware accelerators for a Trusted Execution Environment (TEE) application. The first exploration to talk about is the Rocket Chip Generator combined with CHIPYARD, which is compared with th… configurable RISC-V Rocket Cores with TileLink buses type and buses. Although the CHIPYARD libraries are l… independent, making the implementation of CHIPYAR… the Rocket scala libraries, is possible to implement pe… system. The TEE software is constructed under this h… keypair generation and data signing with and without … TEE, and includes the early-stage bootloader which pe… TEE executable demos to demonstrate the correct bel… Berkeley Out-of-Order Machine (BOOM) Processor, an… CHIPYARD. The implementation of the system can pe… technology nodes. Additional configurable options are… memory, Xilinx PCI-e IP inclusion, USB 1.1 Host inclus…

Key Takeaways

- Show the interaction between software and harc… main application.
- Clear the obscurity of using RISC-V hardware ge… inside focus on CHISEL-based designs (rocket-c…
- Present an insight of including fixed or semi-fixe… includes Verilog-based and system Verilog-base…
- Demonstrate the extensibility of the current RISC…

Hardware Cores/SoCs
Thursday, 10 December 2020 4:00pm - 5:00pm
PST (Pacific Standard Time, GMT-8)

SPEAKER

Ckristian Duran
Research Assistant
University of Electro-Communications

Presentation

Leveraging the RISC-V Eco-System to Put a Chip in Customer Hands in less than $10M

This talk will present the journey of Intensivate in developing the first commercial Cluster CPU, with a focus on how the RISC-V eco-system enables delivering a commercially viable chip, in a 12nm process node, into customer hands at less than $10M.

Audience members will hear the ways in which the cost to deliver such as chip has been reduced, including the role that the RISC-V software ecosystem played, the role of the Rocket-Chip RTL available from Chip Yard, the role of FireSim FPGA emulation system, and the role of the Chisel hardware language.

Key Takeaways

- If you have an idea, it is indeed possible to bring it to market on less than $10M.
- Cost reduction derived from: SW eco-system (OS, toolchain) -- Open Source RTL (Rocket-Chip, chipyard) -- Emulation (FireSim)
- Chisel plus FireSim enabled FPGA centric development -- time to modify RTL not far from time to modify software simulator -- but at a cost

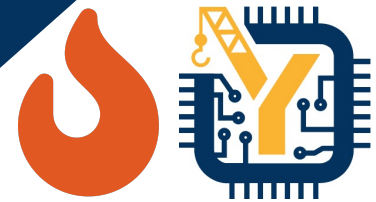
Community Ecosystem
Tuesday, 8 December 2020 11:00am - 11:20am
PST (Pacific Standard Time, GMT-8)

SPEAKER

Sean Halle
CEO
Intensivate

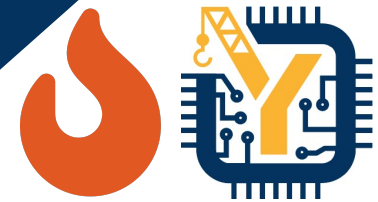
Berkeley Architecture Research

Join The Community!



- Used in industry and academia
- Development is all open-source and on Github
  - Stable `master` branch (latest release)
  - Less-stable `dev` branch with all the newest features
- Sub-projects managed using submodules
- Over 130 pages of documentation!
  - If something isn't clear, please let us know
- Lots of communication on the mailing list
- We appreciate feedback! We appreciate PRs even more!
- Thank you for attending!

Learn More



- Chipyard
  - Github: https://github.com/ucb-bar/chipyard/
  - Docs: https://chipyard.readthedocs.io/en/latest/index.html
  - Mailing List: https://groups.google.com/forum/#!forum/chipyard

- FireSim
  - Website: https://fires.im/
  - Github: https://github.com/firesim/firesim/
  - Docs: https://docs.fires.im/en/latest/
  - Mailing List: https://groups.google.com/forum/#!forum/firesim