The Berkeley Out-of-Order Machine (**BOOM!**): Computer Architecture Research Using an Industry-Competitive, Synthesizable, Parameterized RISC-V Processor

Christopher Celio, Krste Asanovic, David Patterson

2015 June

celio@eecs.berkeley.edu

ASPIRE UC Berkeley

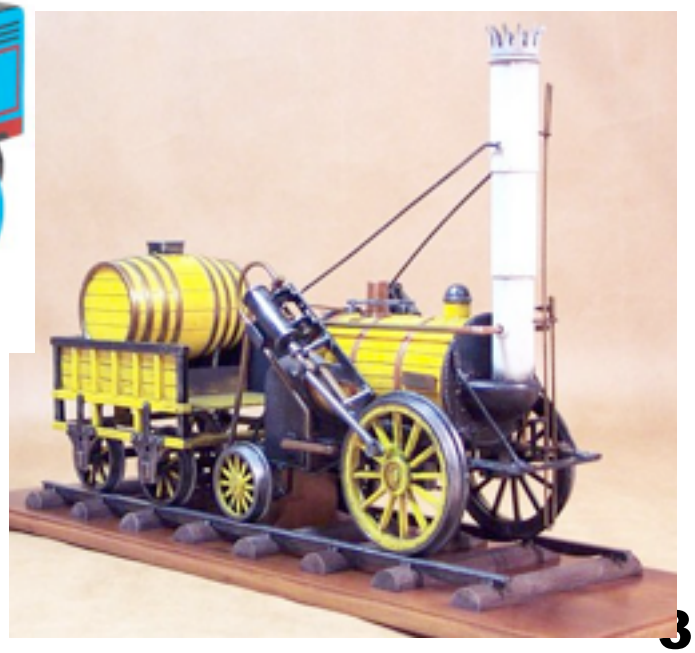
What is BOOM?

- superscalar, out-of-order processor written in Berkeley's Chisel RTL
- It is synthesizable
- It is parameterizable
- We hope to use it as a platform for architecture research

BOOM is a work-in-progress.
Results shown in the talk are preliminary and subject to change!

Other Berkeley RISC-V Processors

- Sodor Collection
  - RV32I - tiny, educational, not-synthesizable
- Z-scale
  - RV32IM - micro-controller
- Rocket
  - RV64G - in-order, single-issue application core
- BOOM
  - RV64G - out-of-order, superscalar application core



Why OoO?

- Great for **...**
  - tolerating **variable** latencies
  - finding **ILP** in code (instruction-level parallelism)
  - complex method for fine-grain data **prefetching**
  - plays nicely with **poor** compilers and **lazily** written code

Performance!

OoO widely used in industry

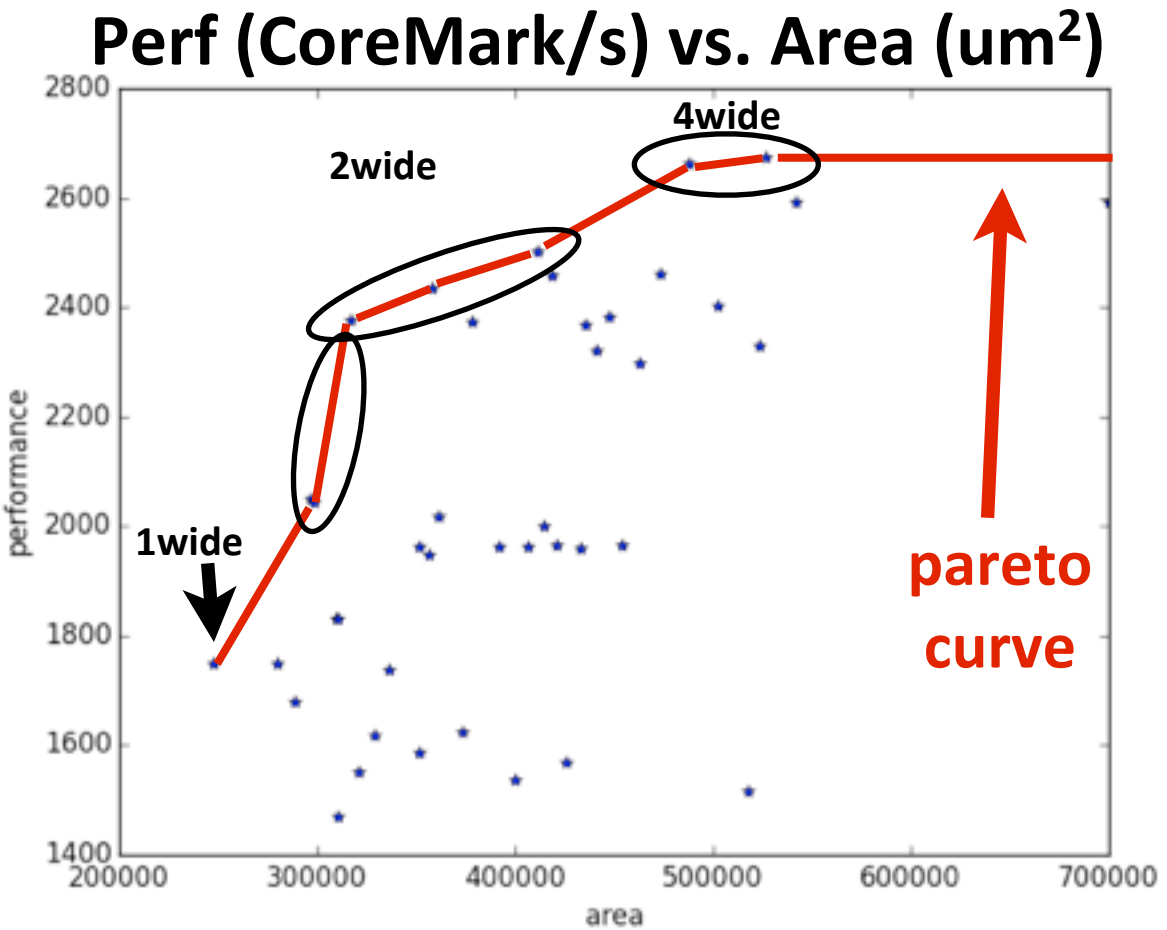
- Intel Xeon/i-series (10-100W)
- ARM Cortex mobile chips (1W)
- Intel Atom
- Sun/Oracle Niagara UltraSPARC
- Play Station

Academic OoO Research

- general lack of effort in academia to build, evaluate OoO designs
- most research uses software simulators
  - cannot produce area, power numbers
  - hard to trust, verify results
    - McPAT is calibrated against 90nm Niagara, 65nm Niagara 2, 65nm Xeon, and 180nm Alpha 21364
  - very slow
- Other Academic OoO RTL efforts...
  - Illinois Verilog Model, Princeton Sharing Architecture, NCSU FabScalar (Alpha, PISA)
  - other ISAs can be very challenging to implement fully
  - rely on SW simulators for performance numbers
  - hopefully RISC-V can make everybody's lives easier!

Design-space exploration

- ***Very preliminary***
- Parameters
  - fetch width
  - issue width
  - ROB size
  - IW size
  - LSU size
  - Regfile size
  - # of branch tags
- 3x range in area
- 2x range in performance



data collected by
Orianna DeMasi

Research Methodology

- Which benchmarks?
- How many cycles do we need to run?
- State of the art
  - "SimPoints"
  - run 4-10 snapshots per SPEC2000/2006 benchmark
  - each snapshot runs for ~10M instructions
- What other people do (ISCA 2014 results)
  - **~50M instructions / workload**
  - **~200B instructions / paper**
- What we can do
  - map design to an **FPGA**
  - run 50 MHz (~1T cycles/6hrs)
  - run full **reference** benchmark (~2 Trillion instructions avg)
  - run on FPGA cluster (~1-2 weeks simulation in one day, or **~30-60T instructions/day**)

Berkeley Architecture Research Infrastructure

- **RISC-V** ISA
- Chisel HCL (hardware construction language)
- Rocket-chip SoC generator

The RISC-V ISA is easy to implement!

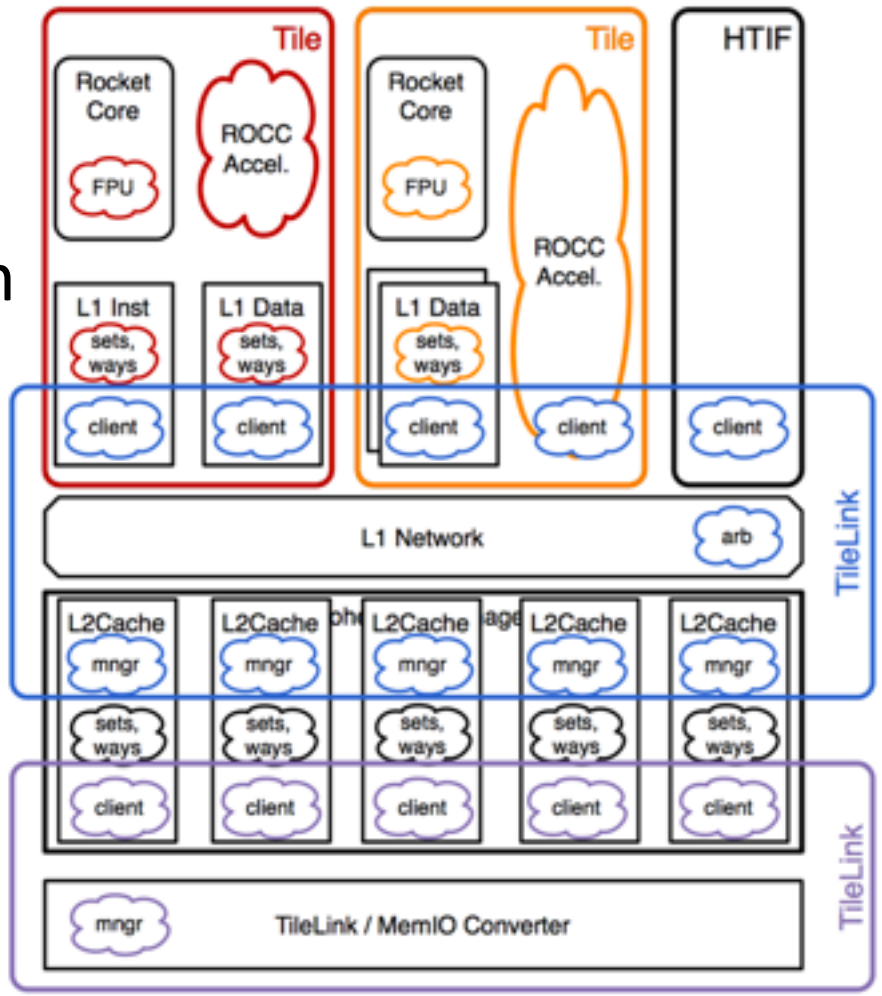
- relaxed memory model
- accrued FP exception flags
- no integer side-effects (e.g., condition codes)
- no cmov or predication
- no implicit register specifiers
  - JAL requires explicit rd
- rs1, rs2, rs3, rd always in same space
  - allows decode, rename to proceed in parallel

The RISC-V ISA

- BOOM supports “M” (mul/div/rem)
  - imul can be either pipelined or unpipelined
- BOOM supports “A”
  - AMOs+LR/SC
- BOOM supports “FD”
  - single, double-precision floating point
  - IEEE 754-2008 compliant FPU
  - SP, DP FMA with hw support for subnormals
- **RV64G**

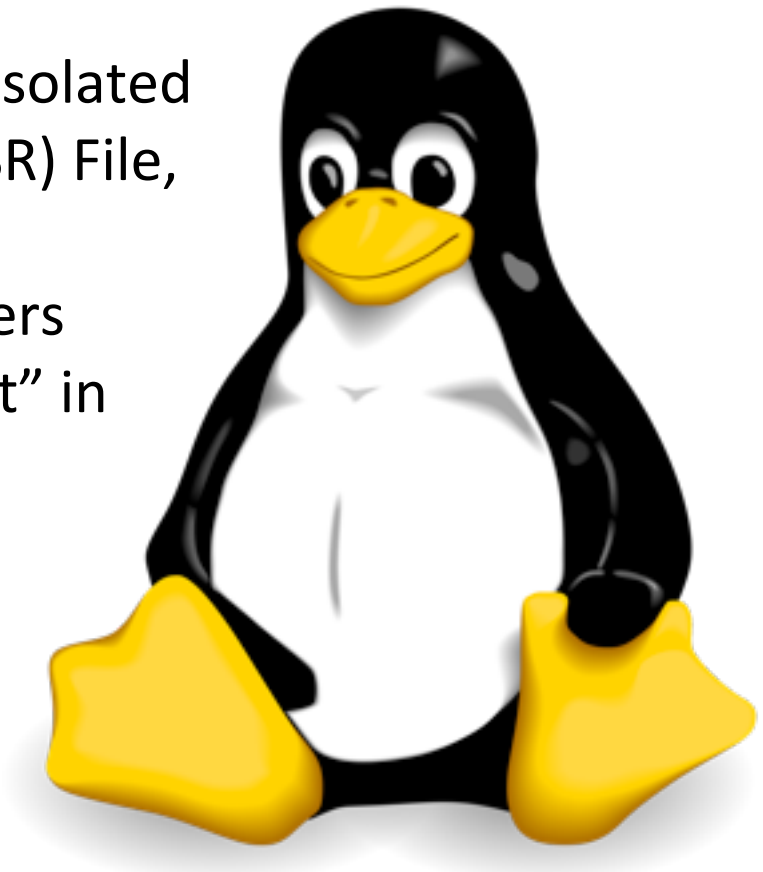
Rocket-Chip SoC Generator

- open-source
- taped out **10** times by Berkeley
- runs at **1.6 GHz** in IBM 45nm
- makes for a great library of processor components!



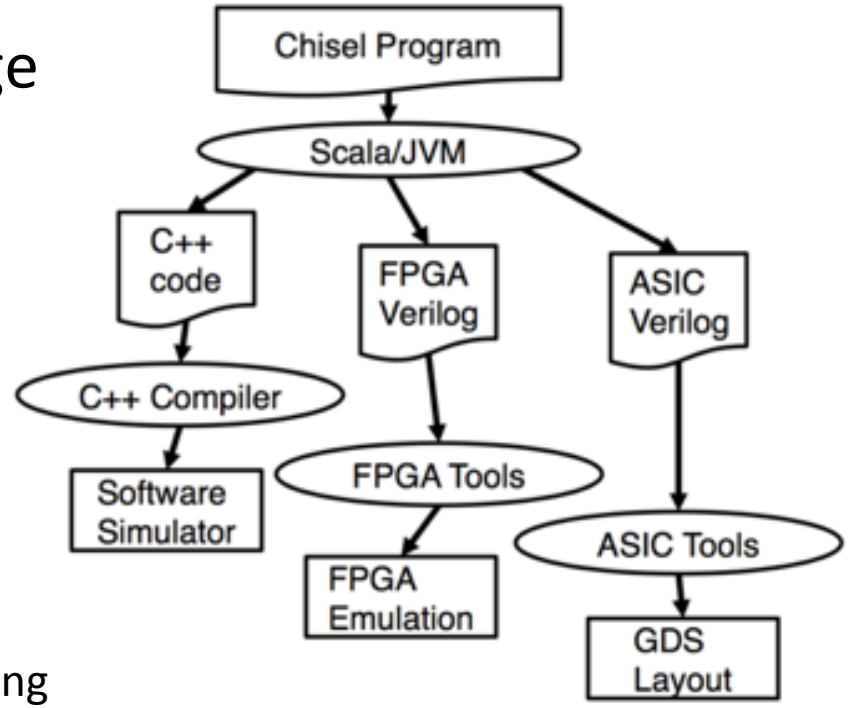
Supports Privileged ISA ("S"), Virtual Memory

- boots Linux!
- just released Privileged ISA v1.7
- instant to update
  - Privileged ISA nearly entirely isolated to Control/Status Register (CSR) File, TLBs
  - updated git submodule pointers
  - changed "tohost" to "mtohost" in one line



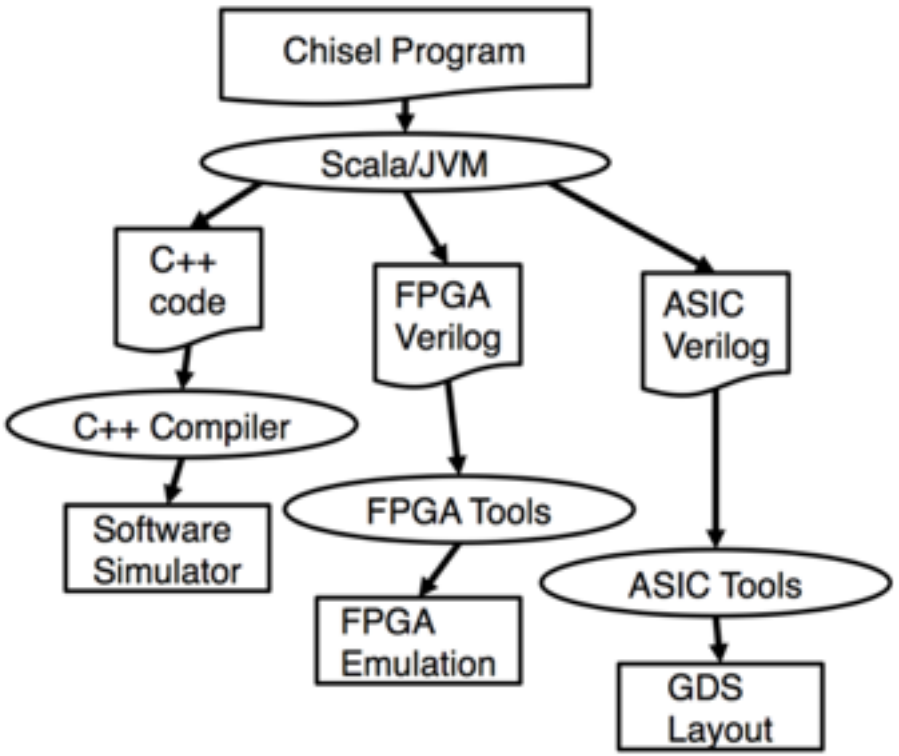
Chisel

- Hardware Construction Language embedded in **Scala**
- ***not*** a high-level synthesis language
- hardware module is a data structure in Scala
- Full power of Scala for writing generators
  - object-oriented programming
    - factory objects, traits, overloading
  - functional programming
    - high-order funs, anonymous funcs, currying
- generated C++ simulator is **1:1** copy of Verilog designs

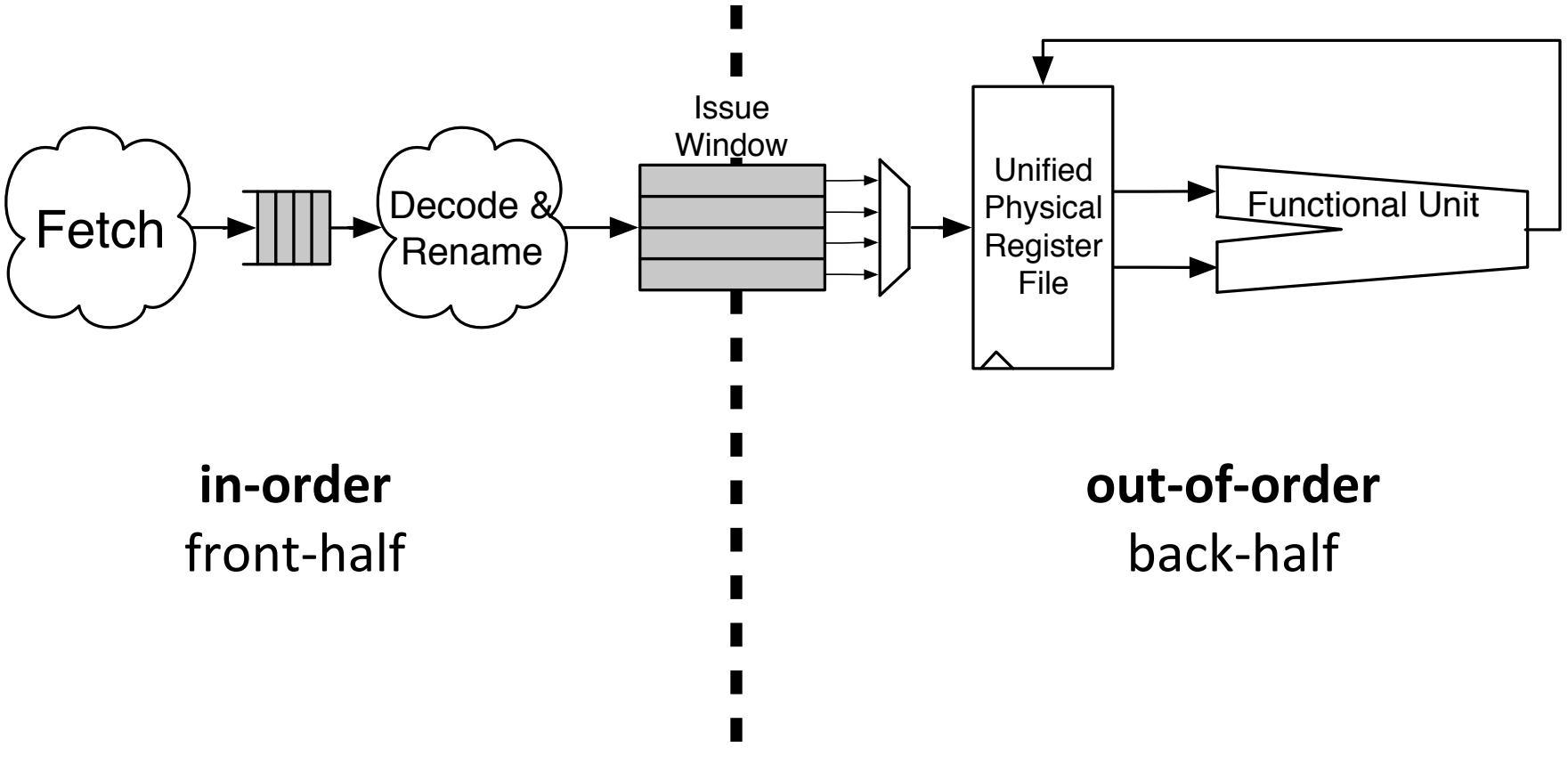


Chisel Hardware Construction Language

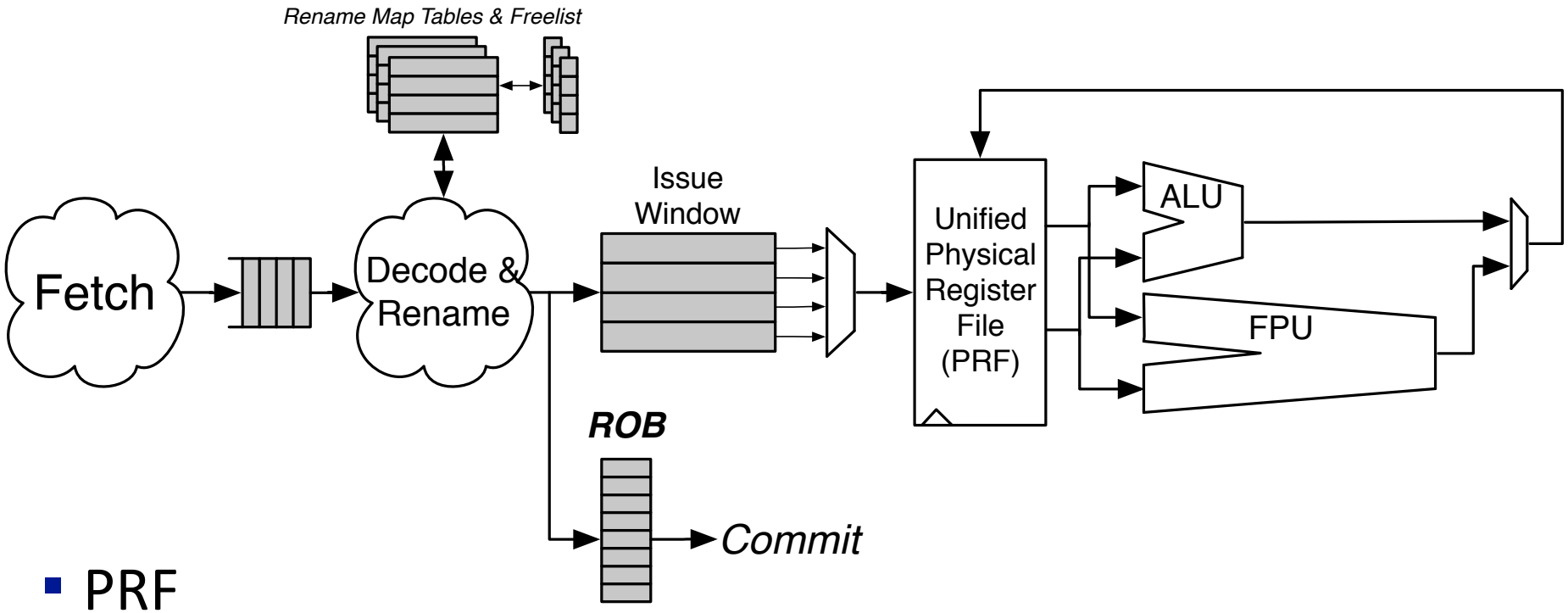
- object-oriented, functional programming
- powerful for writing hw generators
- 12 days (+1092 loc) to add SP,DP floating point
- 9 days (+900 loc) to go from no VM to booting Linux



BOOM

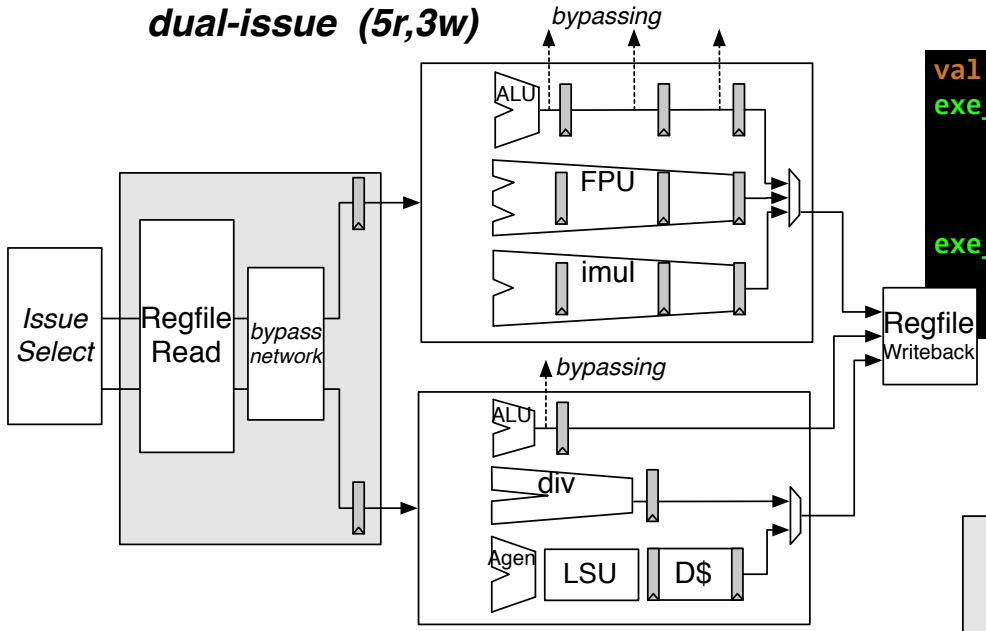

Fetch
Decode &
Rename
Issue
Window
Unified
Physical
Register
File
Functional Unit
in-order
front-half
out-of-order
back-half

BOOM



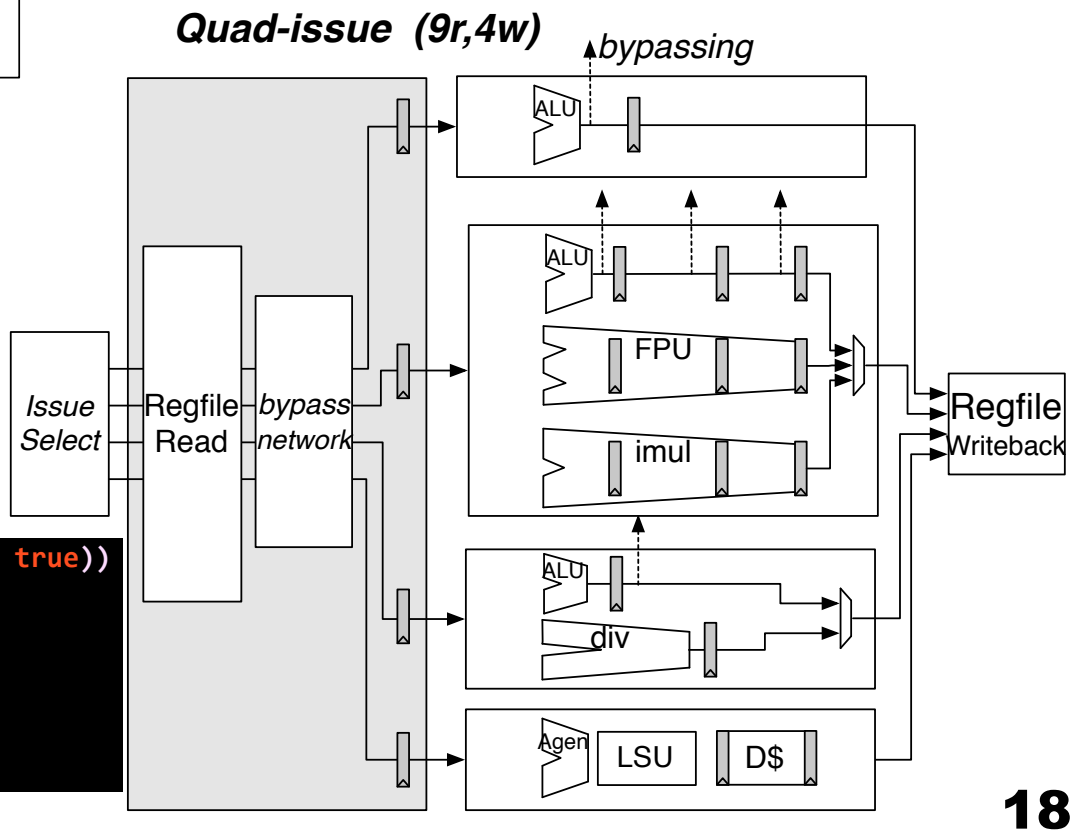
- PRF
  - explicit renaming
  - holds speculative and committed data
  - holds both x-regs, f-regs
- Unified Issue Window
  - holds all instructions
- split ROB/issue window design

Parameterized Superscalar




```
val exe_units = ArrayBuffer[ExecutionUnit]()
exe_units += Module(new ALUExeUnit(is_branch_unit    = true
                                  , has_fpu          = true
                                  , has_mul          = true
                                  ))
exe_units += Module(new ALUMemExeUnit(fp_mem_support = true
                                  , has_div          = true
                                  ))
```

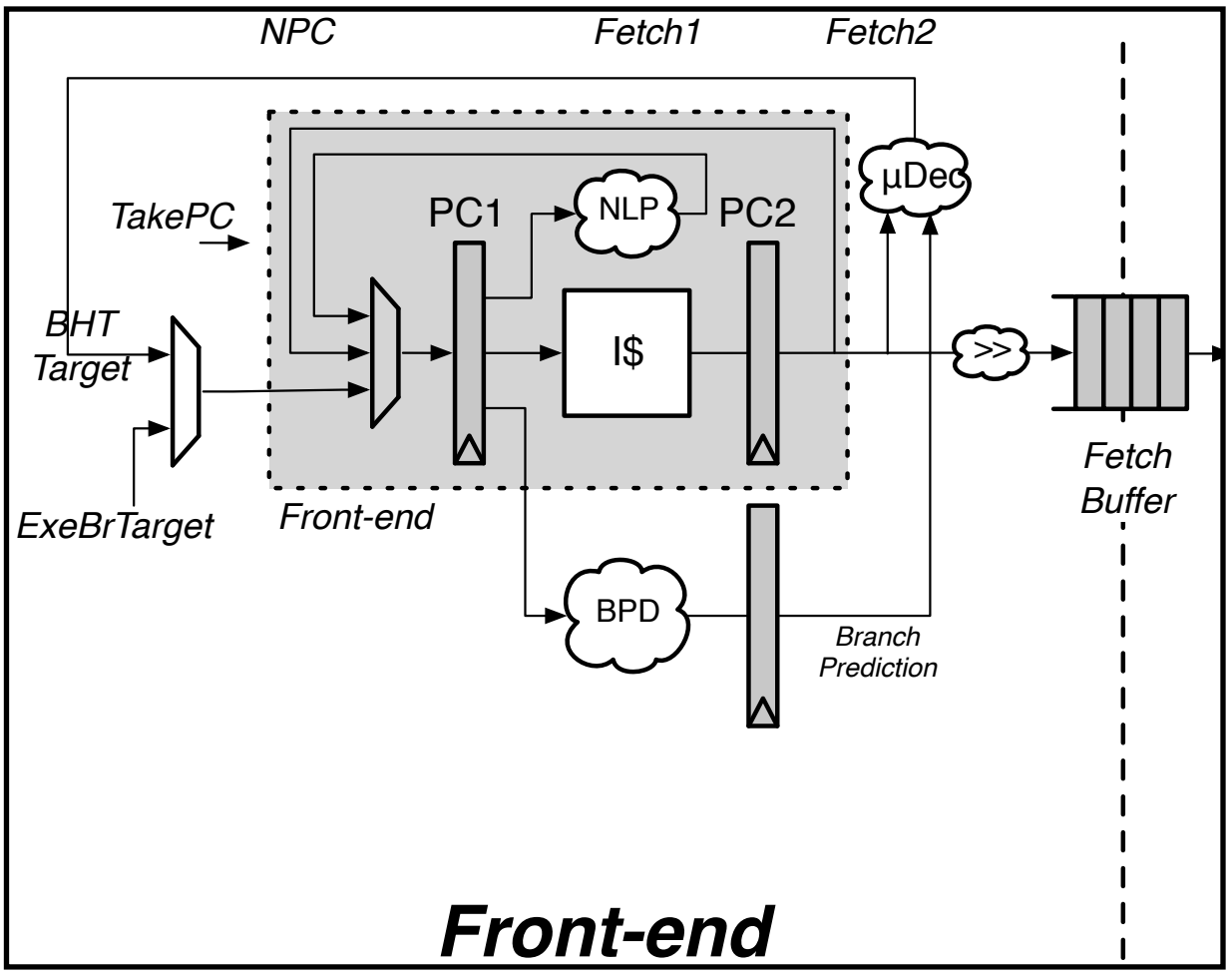
OR




```
exe_units += Module(new ALUExeUnit(is_branch_unit = true))
exe_units += Module(new ALUExeUnit(has_fpu = true
                                 , has_mul = true
                                 ))
exe_units += Module(new ALUExeUnit(has_div = true))
exe_units += Module(new MemExeUnit())
```

Full Branch Speculation Support

- next-line predictor (NLP)
  - BTB, BHT, RAS
  - combinational
- backing predictor (BPD)
  - global history predictor
  - SRAM (1 r/w port)

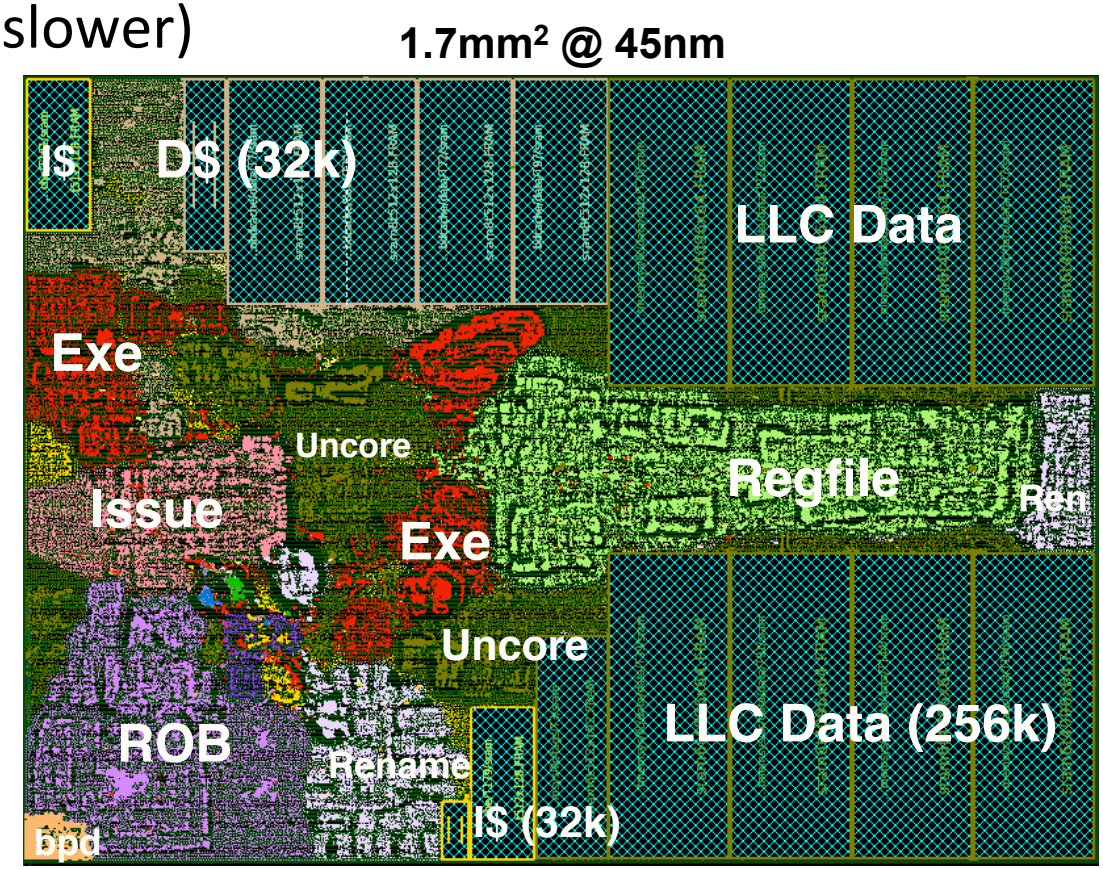


Load/Store Unit

- load/store queue with store ordering
  - loads execute fully out-of-order wrt stores, other loads
    - store-data forwarded to loads as required
- non-blocking data cache

Synthesizable

- Runs on FPGA
  - (Zynq zedboard and Zynq zc706)
- **2GHz** (30 FO4) in TSMC 45nm
  - speed of logic (SRAM is slower)



2-wide BOOM layout.

preliminary results

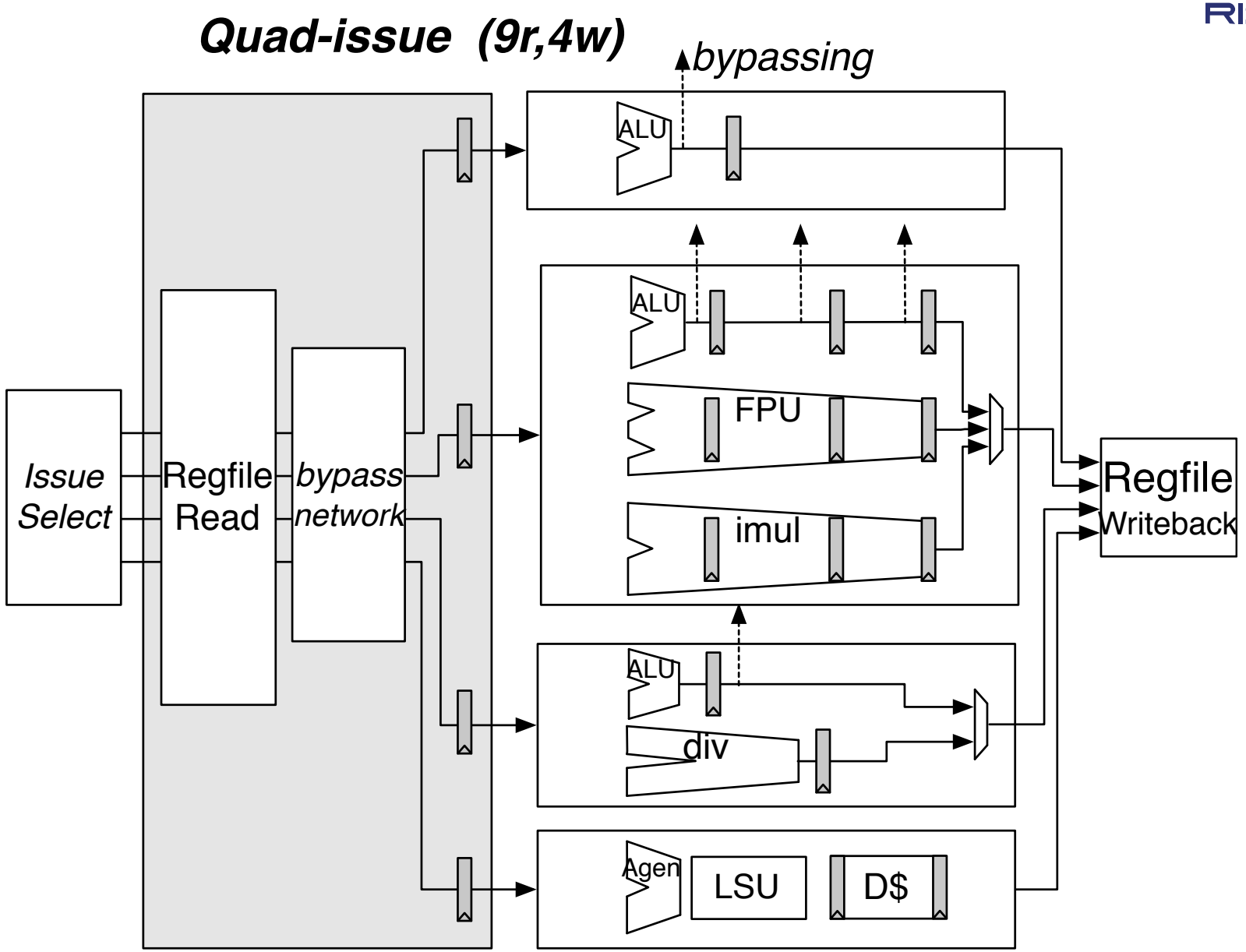
Benefits of using Chisel

- **~9,000 loc** in BOOM github repo
- additional ~**11,500** loc instantiated from other libraries
  - **~5,000 loc** from Rocket core repository
    - functional units, caches, PTWs, etc.
  - **~4,500 loc** from uncore
    - coherence hubs, L2 caches, networks, host/target interfaces
  - **~2000 loc** from hardfloat
    - floating point hard units

Feature Summary

Feature	BOOM
ISA	RISC-V (RV64G)
Synthesizable	√
FPGA	√
Parameterized	√
floating point	√
AMOs+LR/SC	√
caches	√
VM	√
Boots Linux	√
Multi-core	√
lines of code	**9k + 11k**

That's BOOM!

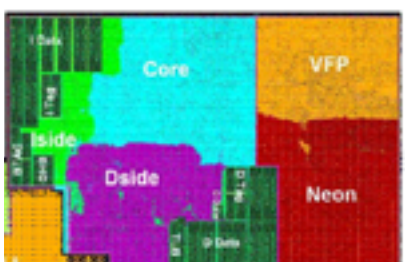

Quad-issue (9r,4w)
bypassing
ALU
ALU
FPU
imul
ALU
div
Agen
LSU
D$
Issue Select
Regfile Read
bypass network
Regfile Writeback

Comparison against ARM

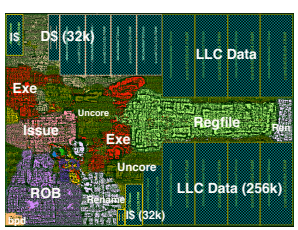
Category	ARM Cortex-A9	RISC-V BOOM-2w
ISA	32-bit ARM v7	64-bit RISC-V v2 (RV64G)
Architecture	2 wide, 3+1 issue Out-of-Order 8-stage	2 wide, 3 issue Out-of-Order 6-stage
Performance	**3.59** CoreMarks/MHz	**3.91** CoreMarks/MHz
Process	TSMC 40GPLUS	TSMC 40GPLUS
Area with 32K caches	~2.5 mm^2	~1.00 mm^2
Area efficiency	**1.4** CoreMarks/MHz/mm^2	**3.9** CoreMarks/MHz/mm^2
Frequency	1.4 GHz	1.5 GHz

+9%!

note:
not to scale

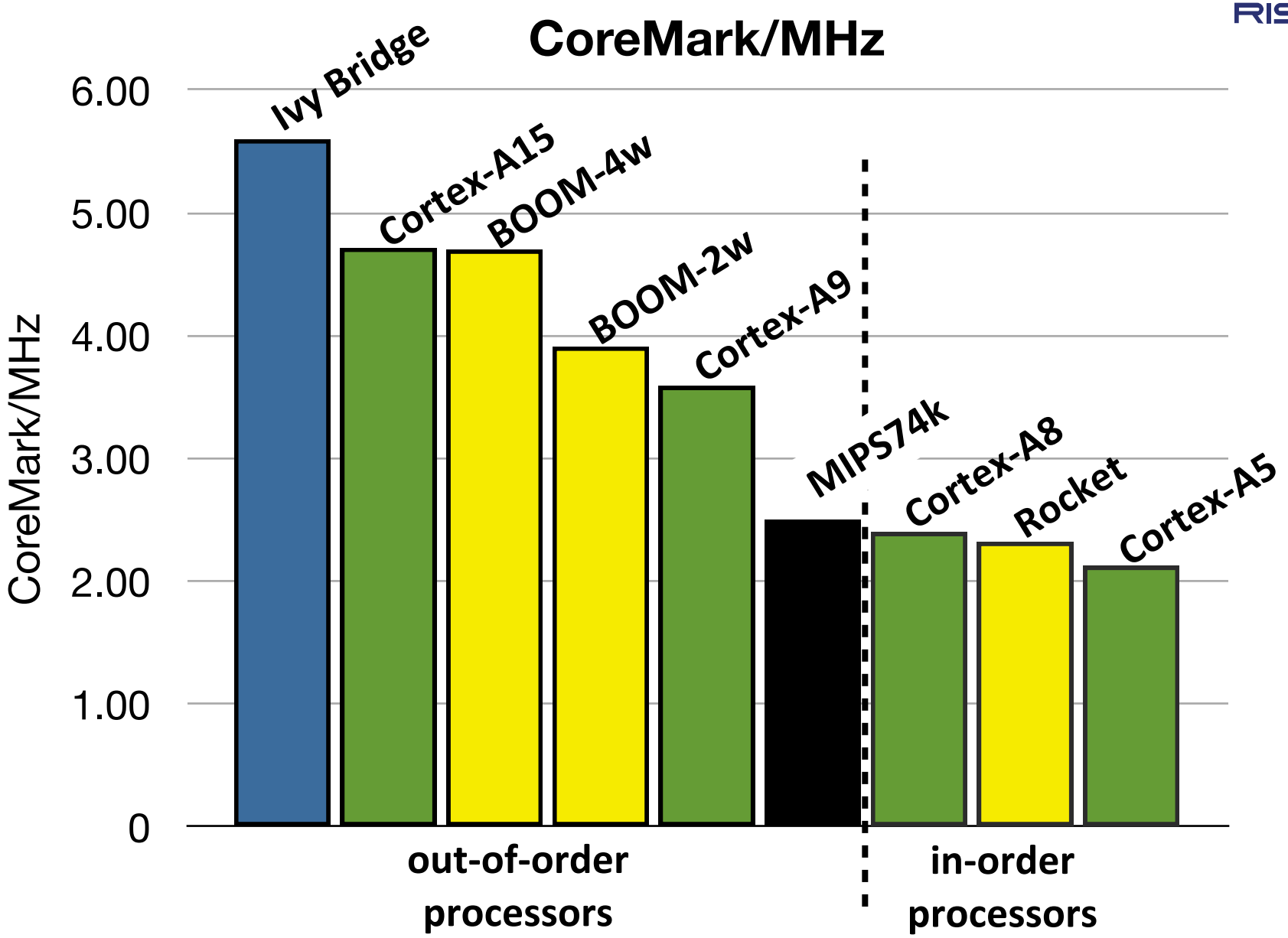


preliminary results



2-wide BOOM layout.

Industry Comparisons



preliminary results

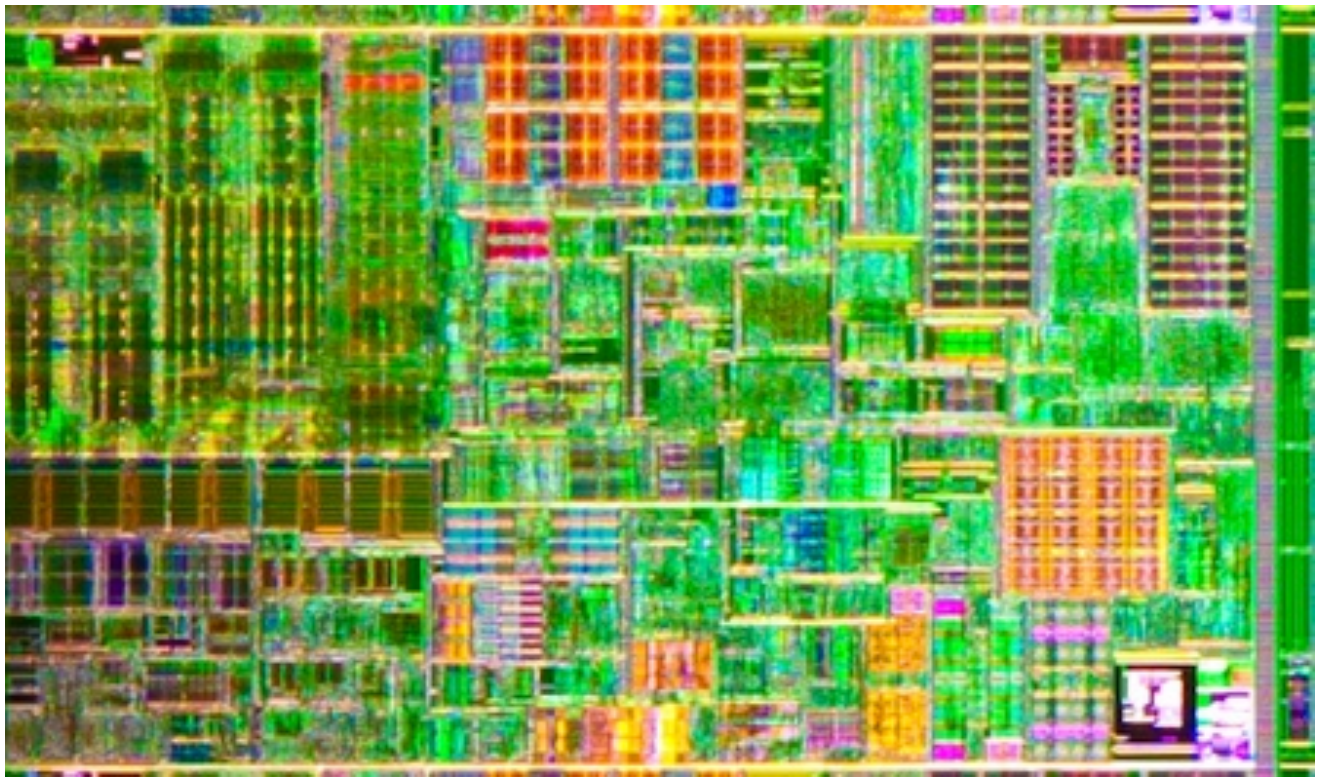
Industry Comparisons

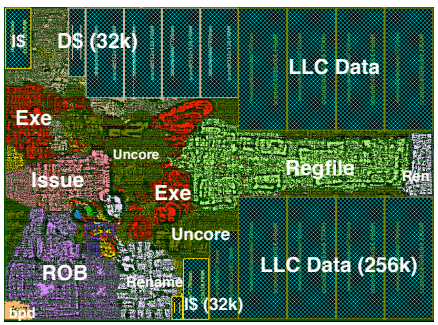
Processor	Core Area	CoreMark/ MHz	Freq (MHz)	IPC
Intel Xeon E5 2668 (Ivy)	~12 mm^2@22nm	5.60	3,300	1.96
ARM Cortex-A15	2.8 mm^2@28nm	4.72	2,116	1.50
BOOM-4wide	1.1 mm^2@45nm	4.70	1,000	1.50
BOOM-2wide	0.8 mm^2@45nm	3.91	1,500	1.26
ARM Cortex-A9	2.5 mm^2@40nm	3.59	1,400	1.27
MIPS 74K	2.5 mm^2@65nm	2.50	1,600	-
Rocket (RV64G)	0.5 mm^2@45nm	2.32	1,500	0.76
ARM Cortex-A5	0.5 mm^2@40nm	2.13	-	-

48x

preliminary results

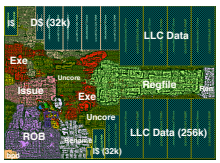
Ivy Bridge Tile Comparison





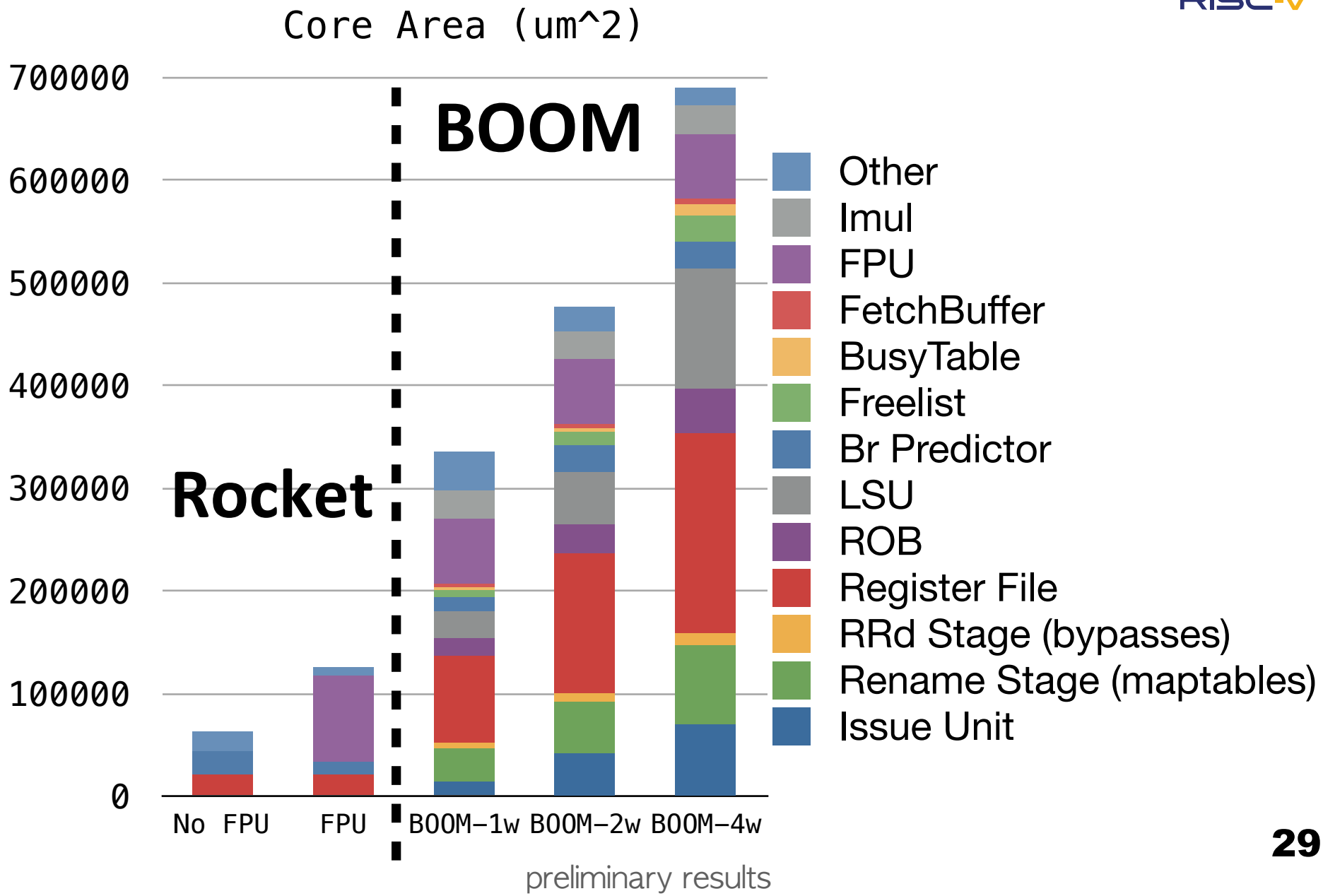
BOOM-2w Chip
(32kb/32kB + 256kB caches)
1.7mm² @ 45nm

Ivy Bridge-EP Tile
(32kB/32kB + 256kB caches)
~12nm @ 22nm

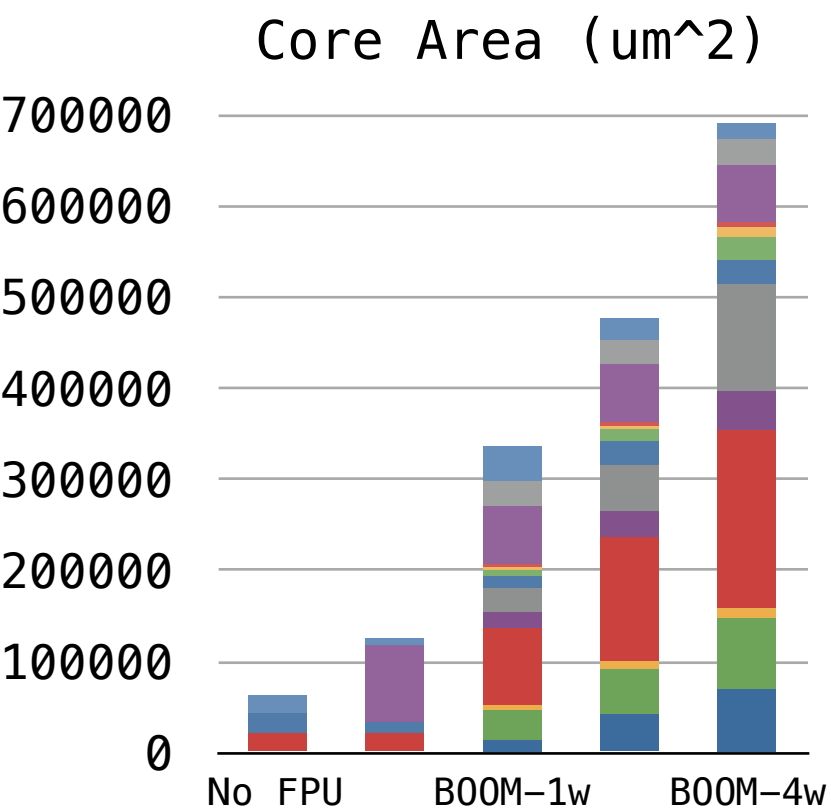


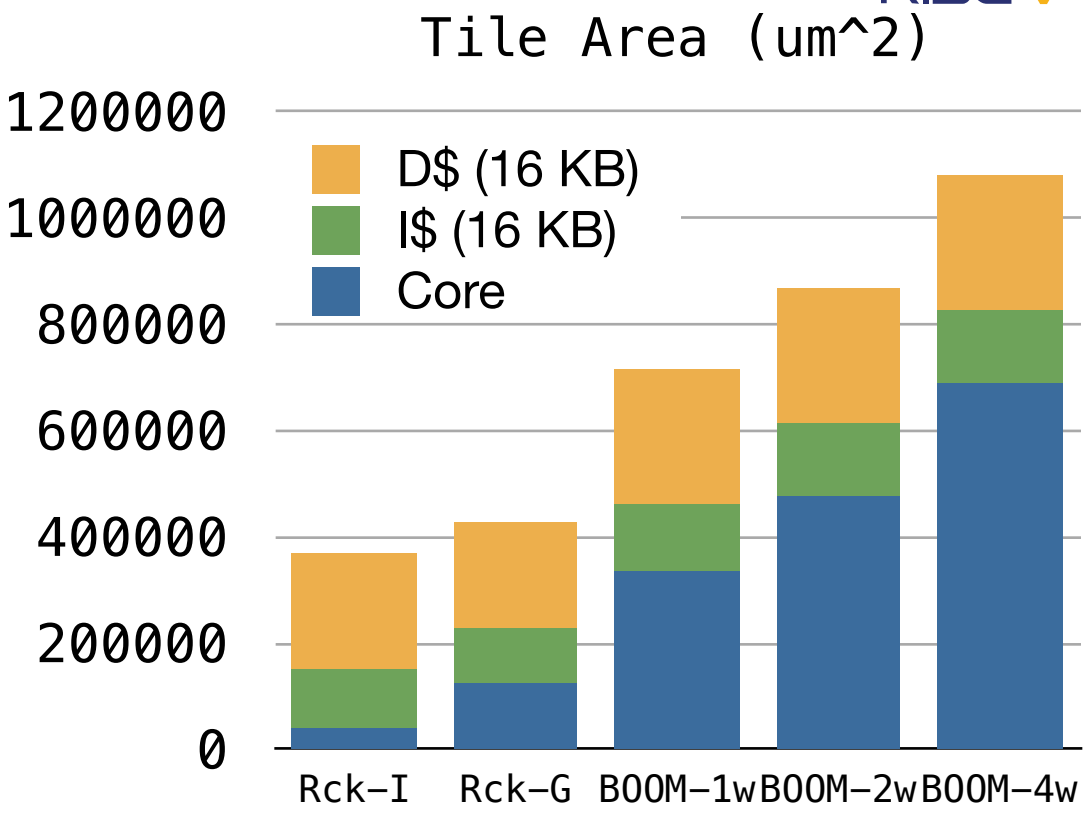
BOOM-2w Chip
scaled to **0.4mm² @ 22nm**
preliminary results

Synthesis Results



Synthesis Results





- Issue Unit
- RRd Stage (bypasses)
- ROB
- Br Predictor
- BusyTable
- FPU
- Other
- Rename Stage (maptables)
- Register File
- LSU
- Freelist
- FetchBuffer
- Imul

preliminary results

Lessons

- RISC-V is a great ISA
  - it gets out of your way
  - the instruction count difference is greater between gcc versions than between ISAs
- code-reuse is great
  - leveraging existing Rocket-chip infrastructure
- Way too much of my time is wasted on corralling benchmarks
  - we should share our efforts
  - https://github.com/ccelio/Speckle/
    - make generating portable SPEC CPU2006 easy
- Debugging is hard
  - good verification tests are more valuable than good RTL
  - use asserts EVERYWHERE
  - use an ISA simulator in parallel with RTL simulation

"Speckle" - a wrapper for SPEC CPU2006

- SPEC is designed to be run natively
  - a pain for cross-compiling, running on a simulator or FPGA
- If you have a copy of CPU2006...
  - modify the provided cfg file
  - Speckle will compile and generate a portable directory of binaries, input files, and input arguments, and a run script
- https://github.com/ccelio/Speckle/

Conclusion

- BOOM supports full RV64G + privileged ISA (VM support)
- Able to boot Linux and run CoreMark, SPECINT, and Dhrystone benchmarks
- BOOM is 9,000 loc and 3 person-years of work

- Future Work
  - bring-up more interesting applications
  - add ROCC interface
  - explore **new** µarch **designs**
  - **tape-out** this fall
  - **open-source** by winter workshop

Questions?

Funding 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 Lab industrial sponsors and affiliates Intel, Google, 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.*