# State of Chisel

## Chisel 3.0, FIRRTL, and Beyond

Chisel Community Conference China 2019 (CCC19 China)

Presented by Edward Wang
Slides by Adam Izraelevitz

A long time ago, in a laboratory (not) far, far away....

## HDL Wars: A New Hope

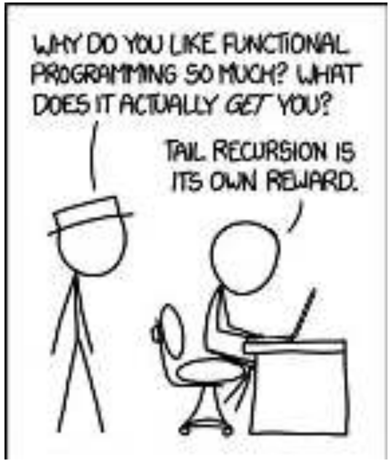
- In 2010, the HW industry was in a difficult place
  - Dennard Scaling had ended
  - Moore’s Law was on its way out
- Demand for computation kept growing
  - New applications
  - More performance for less power/area
  - Only option was to design more complicated, specialized hardware
- Designing hardware was so difficult
  - Billions of transistors
  - Design complexity
- Meanwhile, in a small lab in Berkeley, a solution was brewing….

# Hardware Generators: *Type-Safe Meta-Programming for RTL*

Instead of writing multiple RTL instances, write one instance generator that uses meta-programming for powerful parameterization

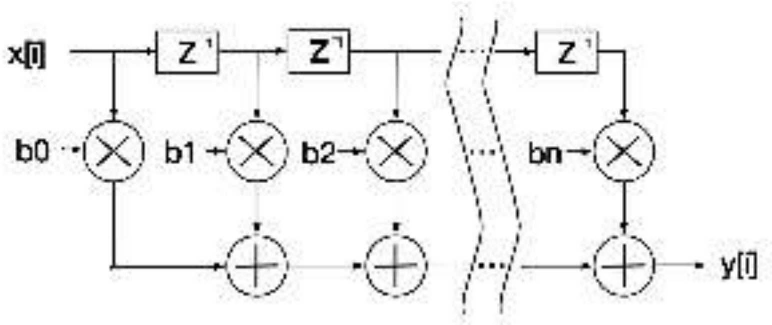
Design Reuse

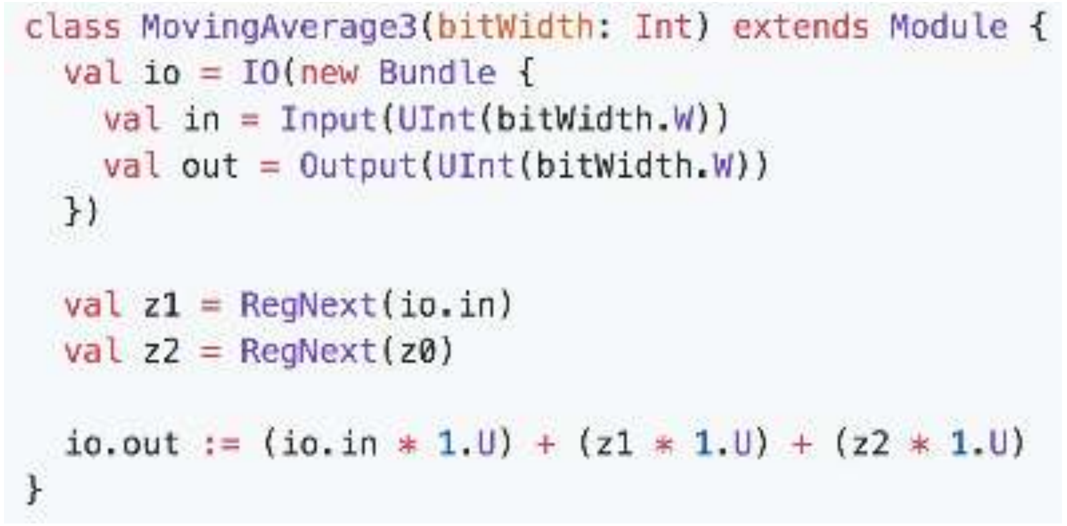
Type-Safety


WHY DO YOU LIKE FUNCTIONAL PROGRAMMING SO MUCH? WHAT DOES IT ACTUALLY *GET* YOU?
TAIL RECURSION IS ITS OWN REWARD.

Powerful Language Features

## No Loss of Expressibility: “Verilog-like” Chisel





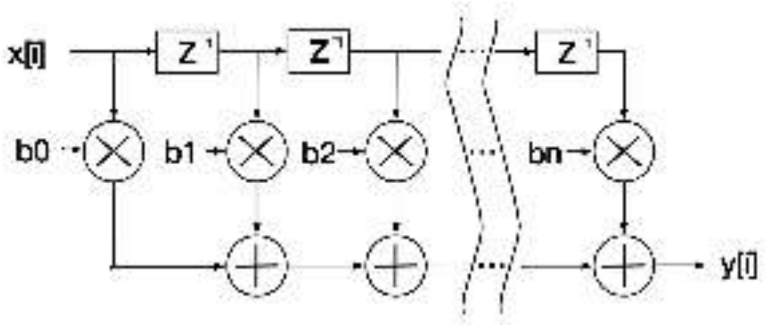
```
class MovingAverage3(bitWidth: Int) extends Module {
  val io = IO(new Bundle {
    val in = Input(UInt(bitWidth.W))
    val out = Output(UInt(bitWidth.W))
  })

  val z1 = RegNext(io.in)
  val z2 = RegNext(z0)

  io.out := (io.in * 1.U) + (z1 * 1.U) + (z2 * 1.U)
}
```

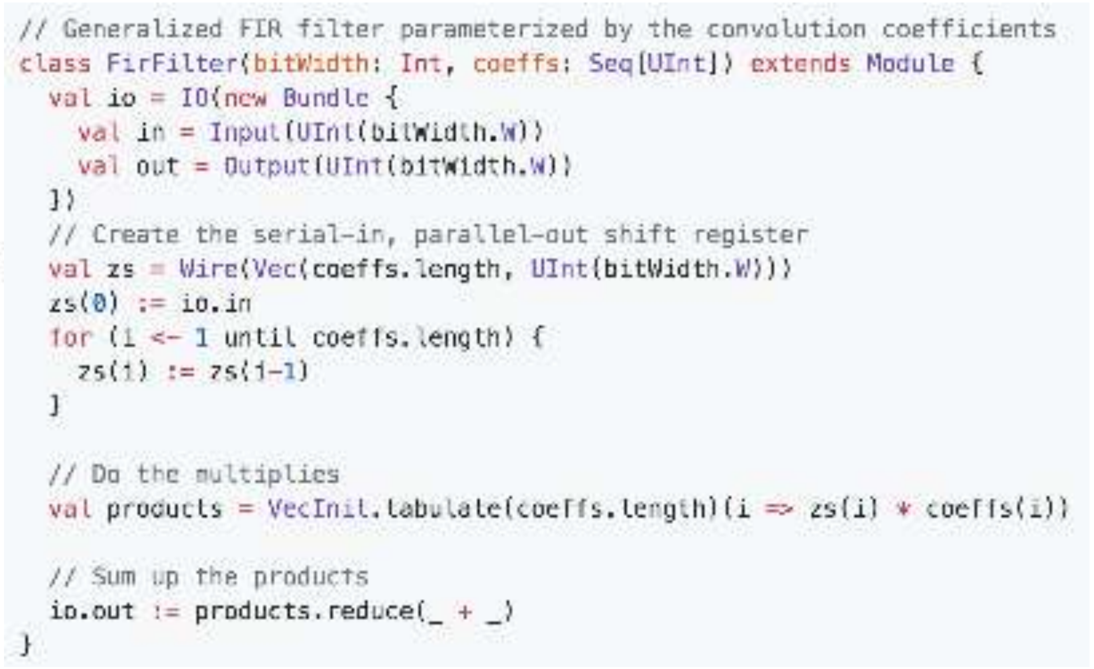
FIR (Finite Impulse Response) Filter
- 3-point moving average

What about >3 points?
What about weighted averages?

We want a *generic* FIR filter!

## Massive Increase in Parameterizability: “Software-like” Chisel



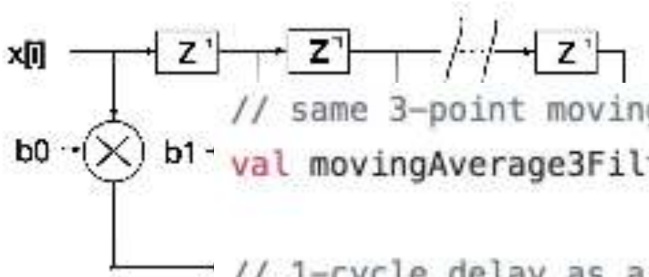


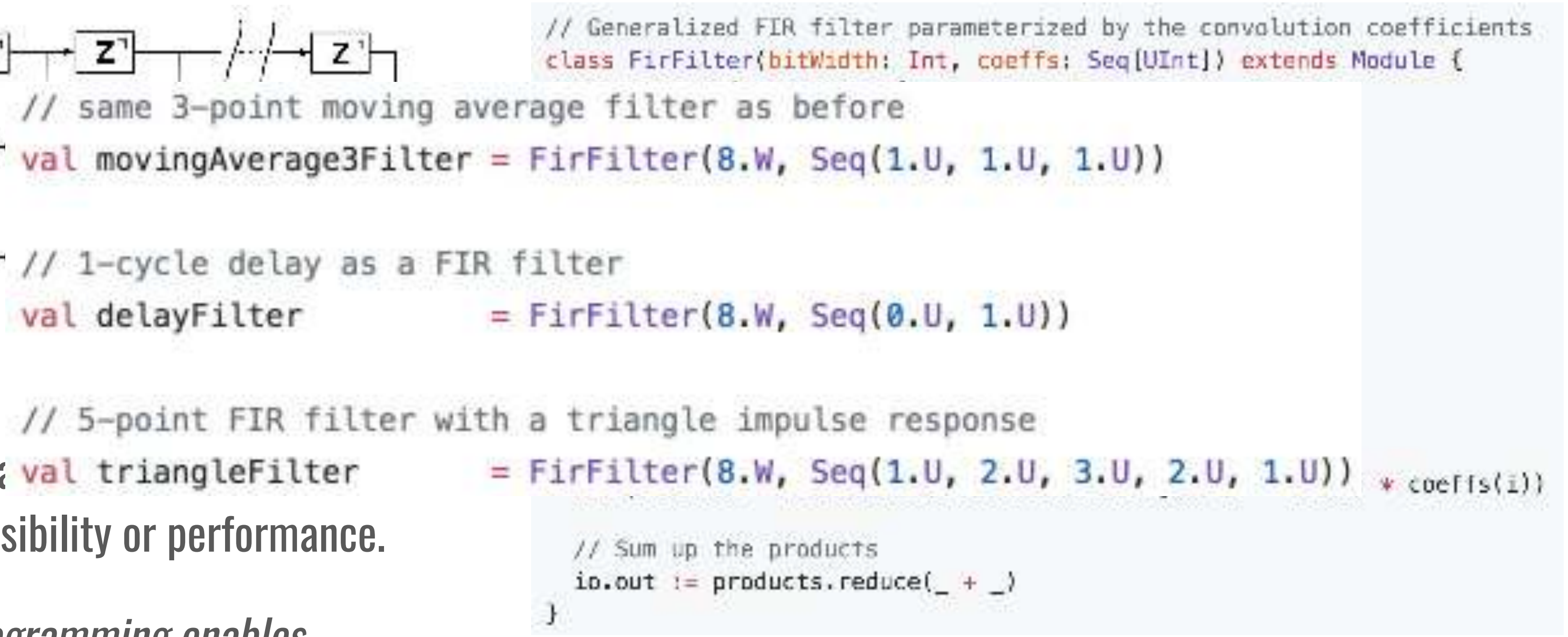
```
// Generalized FIR filter parameterized by the convolution coefficients
class FirFilter(bitWidth: Int, coeffs: Seq[UInt]) extends Module {
  val io = IO(new Bundle {
    val in = Input(UInt(bitWidth.W))
    val out = Output(UInt(bitWidth.W))
  })
  // Create the serial-in, parallel-out shift register
  val zs = Wire(Vec(coeffs.length, UInt(bitWidth.W)))
  zs(0) := io.in
  for (i <- 1 until coeffs.length) {
    zs(i) := zs(i-1)
  }

  // Do the multiplies
  val products = VecInit.tabulate(coeffs.length)(i => zs(i) * coeffs(i))

  // Sum up the products
  io.out := products.reduce(_ + _)
}
```

FIR Filter - Parameterized by bitwidth and *coeffients* with no loss of expressibility or performance.

*Meta-programming enables powerful parameterization.*

# Massive Increase in Parameterizability: “Software-like” Chisel





```
// Generalized FIR filter parameterized by the convolution coefficients
class FirFilter(bitWidth: Int, coeffs: Seq[UInt]) extends Module {
```

```
// same 3-point moving average filter as before
val movingAverage3Filter = FirFilter(8.W, Seq(1.U, 1.U, 1.U))

// 1-cycle delay as a FIR filter
val delayFilter          = FirFilter(8.W, Seq(0.U, 1.U))

// 5-point FIR filter with a triangle impulse response
val triangleFilter       = FirFilter(8.W, Seq(1.U, 2.U, 3.U, 2.U, 1.U))
```

```
 * coeffs(i))

  // Sum up the products
  io.out := products.reduce(_ + _)
}
```

FIR Filter
bitwidth a
of expressibility or performance.

*Meta-programming enables powerful parameterization.*

## Hired Jonathan Bachrach (2010)

Chisel: Constructing Hardware in a Scala Embedded Language

[illegible]

ABSTRACT

[illegible]

Keywords

[illegible]

1. INTRODUCTION

[illegible]

## By the End of ParLab (2013), We Learned....

+

=

12 Grad Students

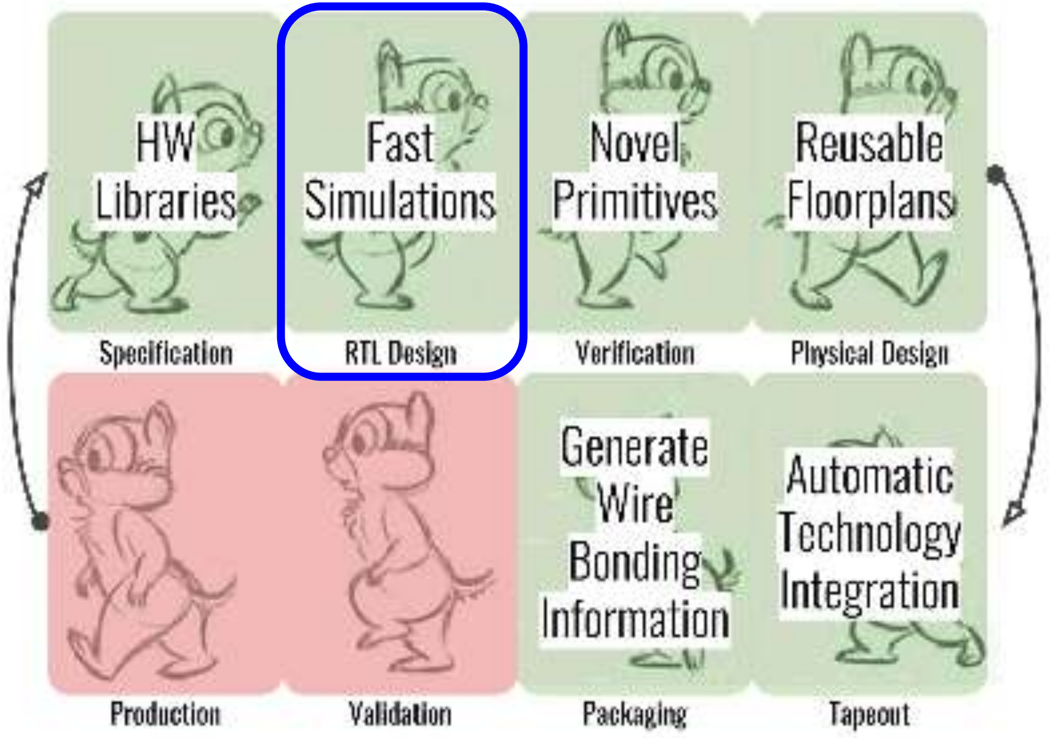
Chisel

Chips

And so, in late 2013, the problem of designing hardware was forever solved.

## Just kidding.

# Improving RTL Design ≠ Solving The Hardware Design Loop



* from “How to Draw Chip and Dale booklet”. (Walt Disney, 1955)

## What had to change?

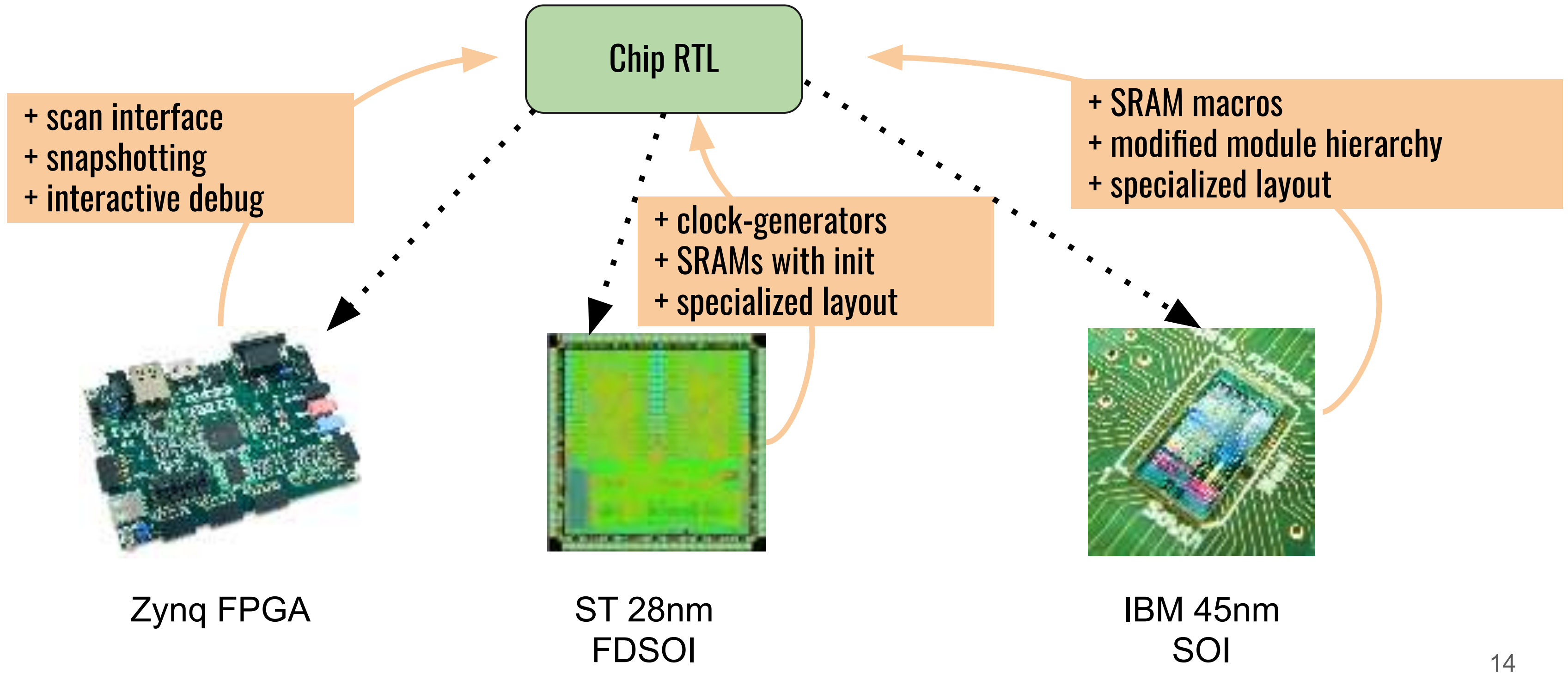
Hardware Design Ecosystem

CODE WRITTEN IN HASKELL IS GUARANTEED TO HAVE NO SIDE EFFECTS.
...BECAUSE NO ONE WILL EVER RUN IT?

Stable and User-Friendly API

External Collaborations

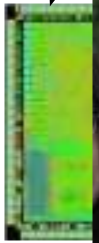
# Platform-Specific or Application-Specific RTL Changes



+ interactive debug

specialized layout

Zynq FPGA

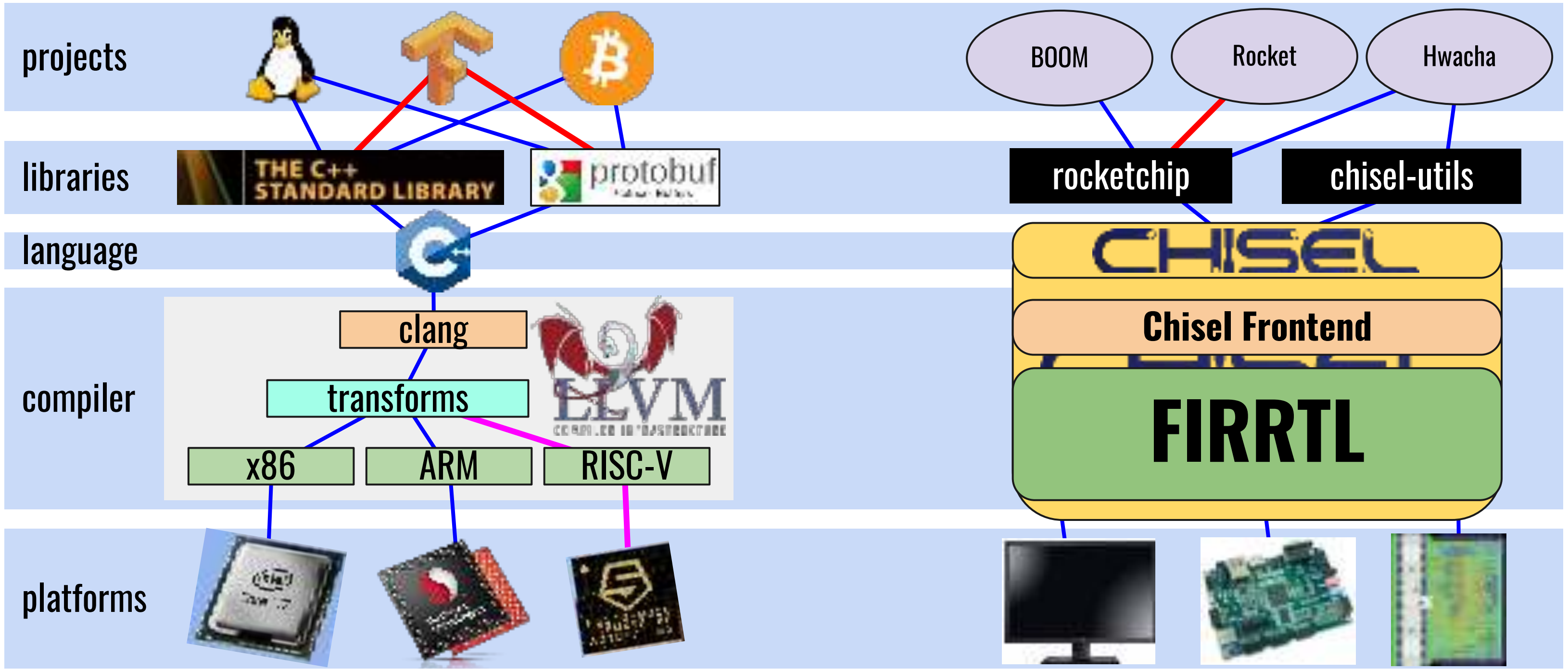


ST

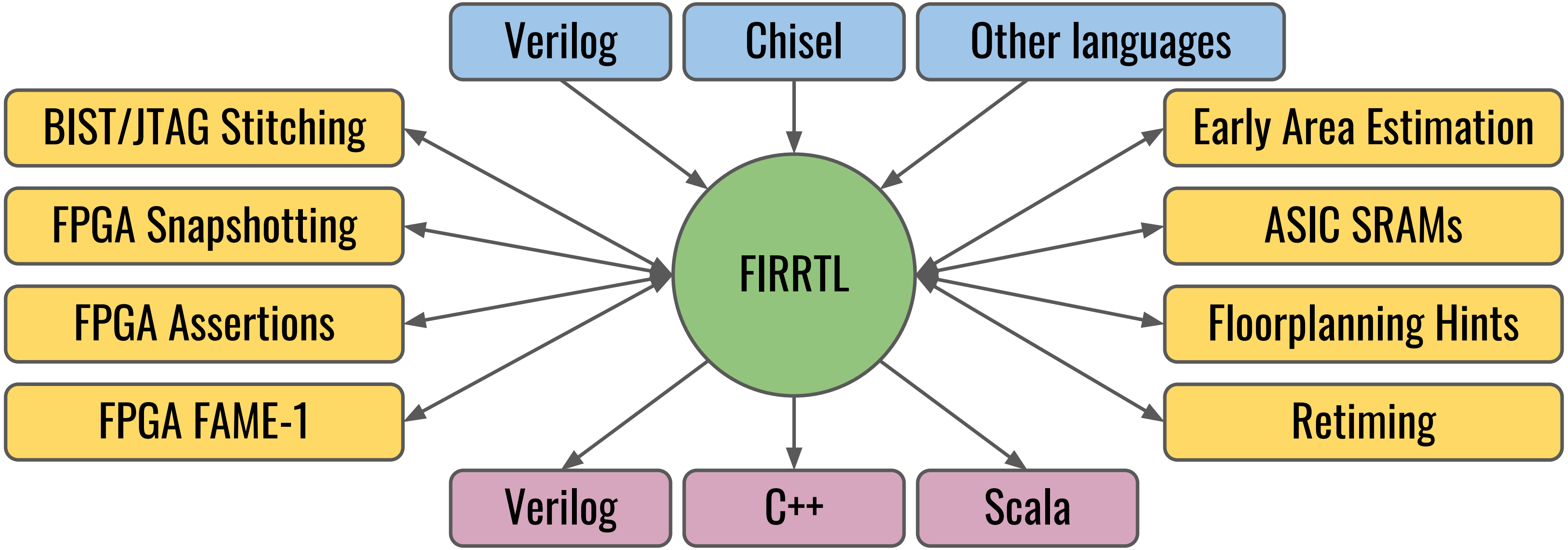
FDSOI

IBM 45nm
SOI

## Realization: We need a software stack, but for hardware



## For an ecosystem, we needed a hardware compiler infrastructure



## Developing, Porting, Code Reviews, Testing, and so forth

Spring 2015
Summer 2015
Fall 2015
Winter 2015
Spring 2016
Summer 2016
Fall 2016
Winter 2016
Spring 2017
Summer 2017
Fall 2017

- Designed FIRRTL Compiler
- Designed Chisel 3 Frontend
- Ported RocketChip
- Ported Chisel Testers
- Added Fixed-Point Type
- Added Analog Type
- Added multi-clock
- Released Chisel 3.0.0
- Released FIRRTL 1.0.0

Andrew Waterman <waterman@cs.berkeley.edu>
Yunsup Lee <yunsup@sifive.com>
Adam Izraelevitz <izraelevitz@google.com>
jackkoenig <jack.koenig3@eecs.berkeley.edu>

## What had to change?

Hardware Design Ecosystem

CODE WRITTEN IN HASKELL IS GUARANTEED TO HAVE NO SIDE EFFECTS.
...BECAUSE NO ONE WILL EVER RUN IT?

Stable and User-Friendly API

External Collaborations

## Emphasis on Clarity

### Chisel 2.0.0

```
Reg(UInt(3))
```

1. Register of type UInt, width of 3
2. Register whose next cycle's value is 3
3. Register whose initial value is 3
4. Register no initial value and width of 2

### Chisel 3.0.0

```
Reg(UInt(3.W))
RegNext(3.U)
RegInit(3.U)
Reg(chiselTypeOf(3.U))
```

1. Register of type UInt, width of 3
2. Register whose next cycle's value is 3
3. Register whose initial value is 3
4. Register no initial value and width of 2

## Emphasis on Clarity

### Chisel 2.0.0

```
Reg(UInt(3))
```

1. Register of type UInt, width of 3
2. Register whose next cycle's value is 3
3. Register whose initial value is 3
4. Register no initial value and width of 2

### Chisel 3.0.0

```
Reg(UInt(3.W))
RegNext(3.U)
RegInit(3.U)
Reg(chiselTypeOf(3.U))
```

1. Register of type UInt, width of 3
2. Register whose next cycle's value is 3
3. Register whose initial value is 3
4. Register no initial value and width of 2

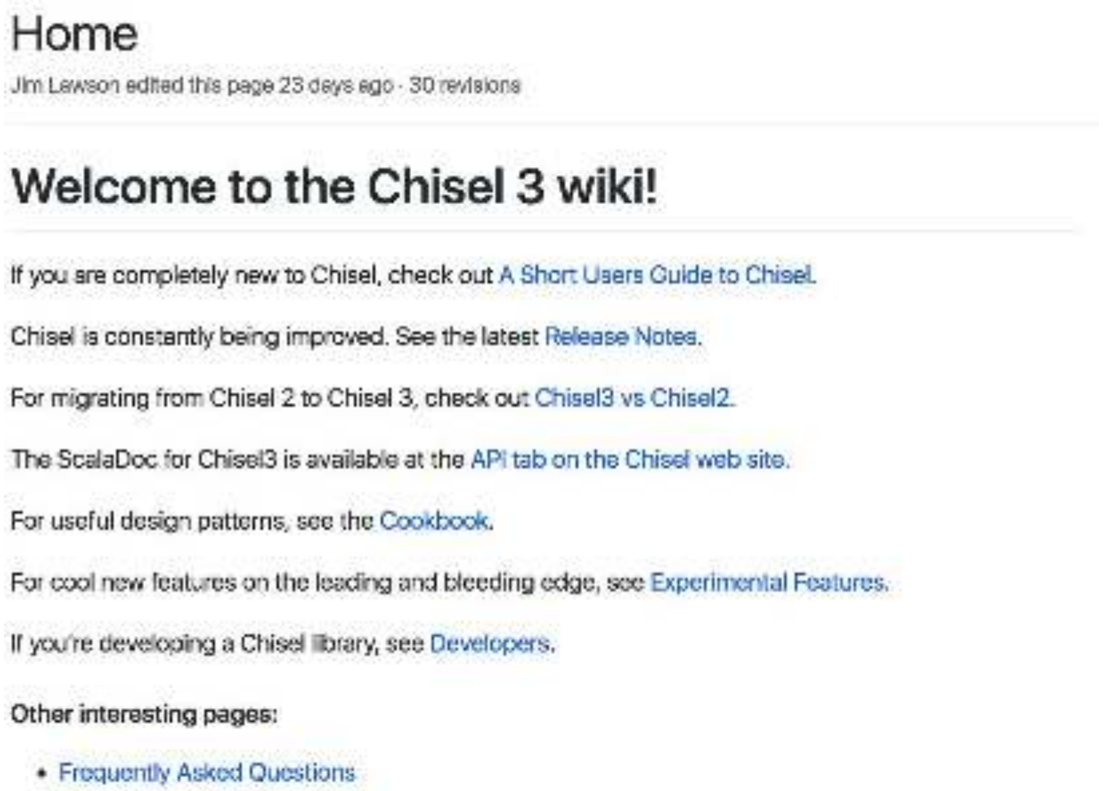
## What had to change?

Hardware Design Ecosystem

CODE WRITTEN IN HASKELL IS GUARANTEED TO HAVE NO SIDE EFFECTS.
...BECAUSE NO ONE WILL EVER RUN IT?
Stable and User-Friendly API

External Collaborations

## Documentation, Documentation, Documentation



Home

Jim Lawson edited this page 23 days ago · 30 revisions

Welcome to the Chisel 3 wiki!

If you are completely new to Chisel, check out A Short Users Guide to Chisel.

Chisel is constantly being improved. See the latest Release Notes.

For migrating from Chisel 2 to Chisel 3, check out Chisel3 vs Chisel2.

The ScalaDoc for Chisel3 is available at the API tab on the Chisel web site.

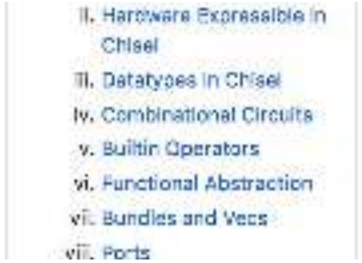
For useful design patterns, see the Cookbook.

For cool new features on the leading and bleeding edge, see Experimental Features.

If you're developing a Chisel library, see Developers.

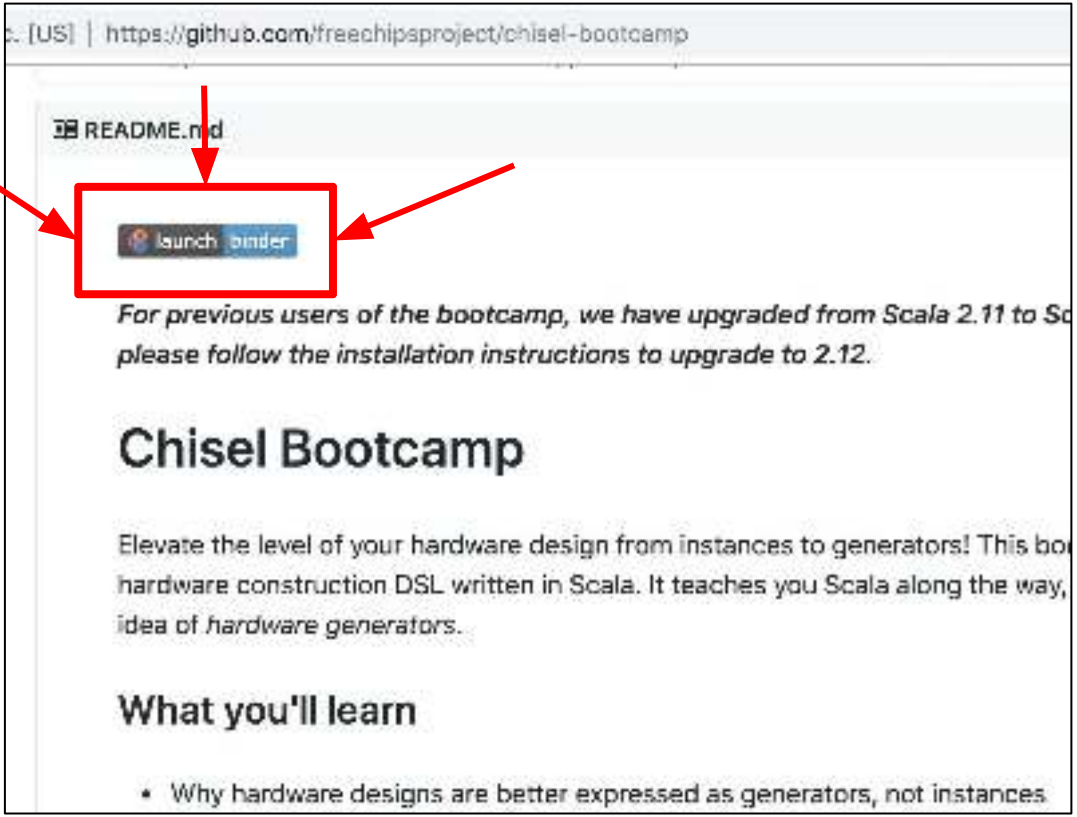
Other interesting pages:

- Frequently Asked Questions



ii. Hardware Expressible in Chisel
iii. Datatypes in Chisel
iv. Combinational Circuits
v. Builtin Operators
vi. Functional Abstraction
vii. Bundles and Vecs
viii. Ports

## Bootcamps and Tutorials



*For previous users of the bootcamp, we have upgraded from Scala 2.11 to Sc*
*please follow the installation instructions to upgrade to 2.12.*

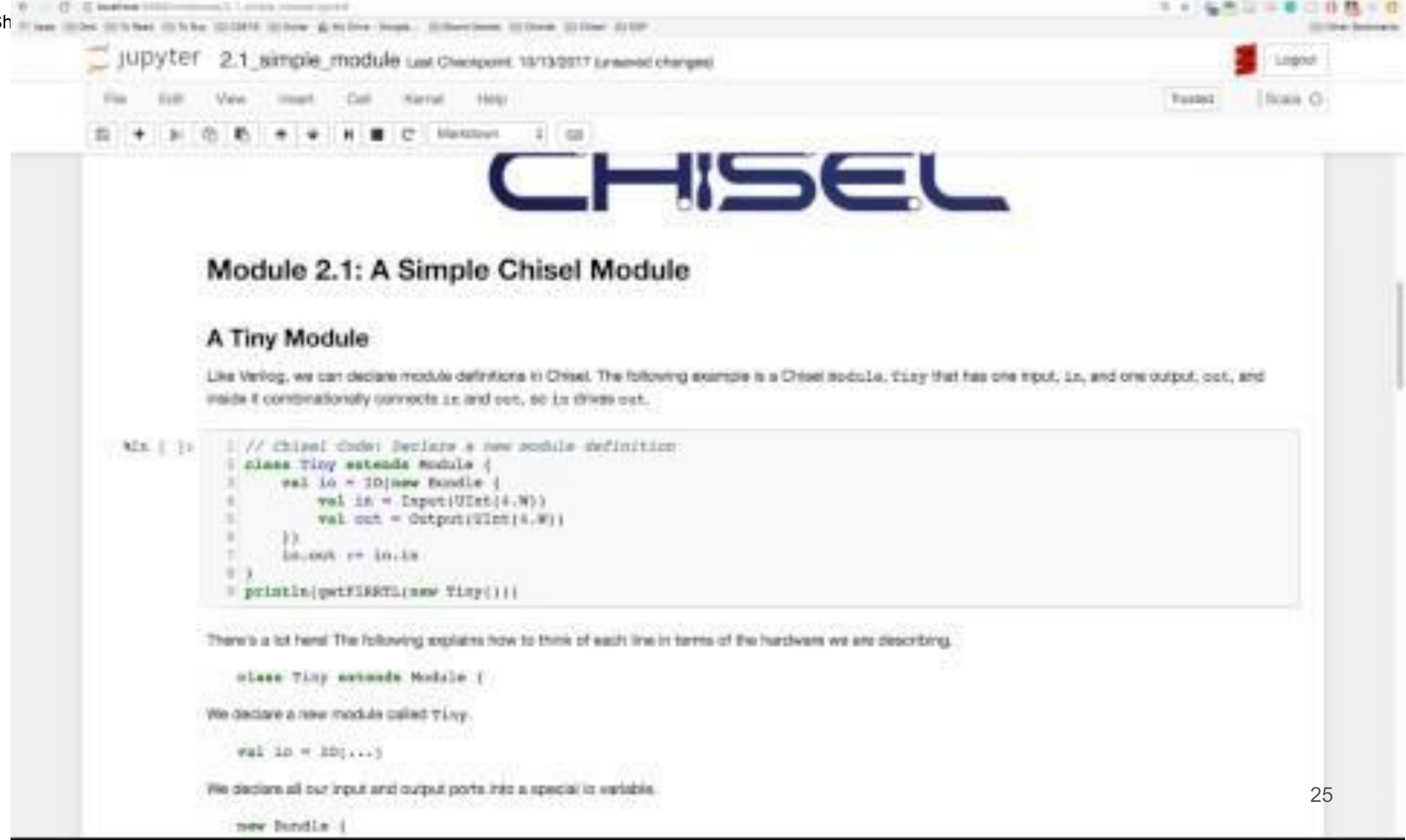
Chisel Bootcamp

Elevate the level of your hardware design from instances to generators! This bo
hardware construction DSL written in Scala. It teaches you Scala along the way,
idea of *hardware generators*.

What you'll learn

- Why hardware designs are better expressed as generators, not instances

## Bootcamps and Tutorials



### Module 2.1: A Simple Chisel Module

#### A Tiny Module

Like Verilog, we can declare module definitions in Chisel. The following example is a Chisel `Module`, `Tiny` that has one input, `in`, and one output, `out`, and inside it combinationally connects `in` and `out`, so `in` drives `out`.

```
In [ ]:
// Chisel Code: Declare a new module definition
class Tiny extends Module {
    val io = IO(new Bundle {
        val in = Input(UInt(4.W))
        val out = Output(UInt(4.W))
    })
    io.out := io.in
}
println(getFIRRTL(new Tiny()))
```

There's a lot here! The following explains how to think of each line in terms of the hardware we are describing.

```
class Tiny extends Module {
```

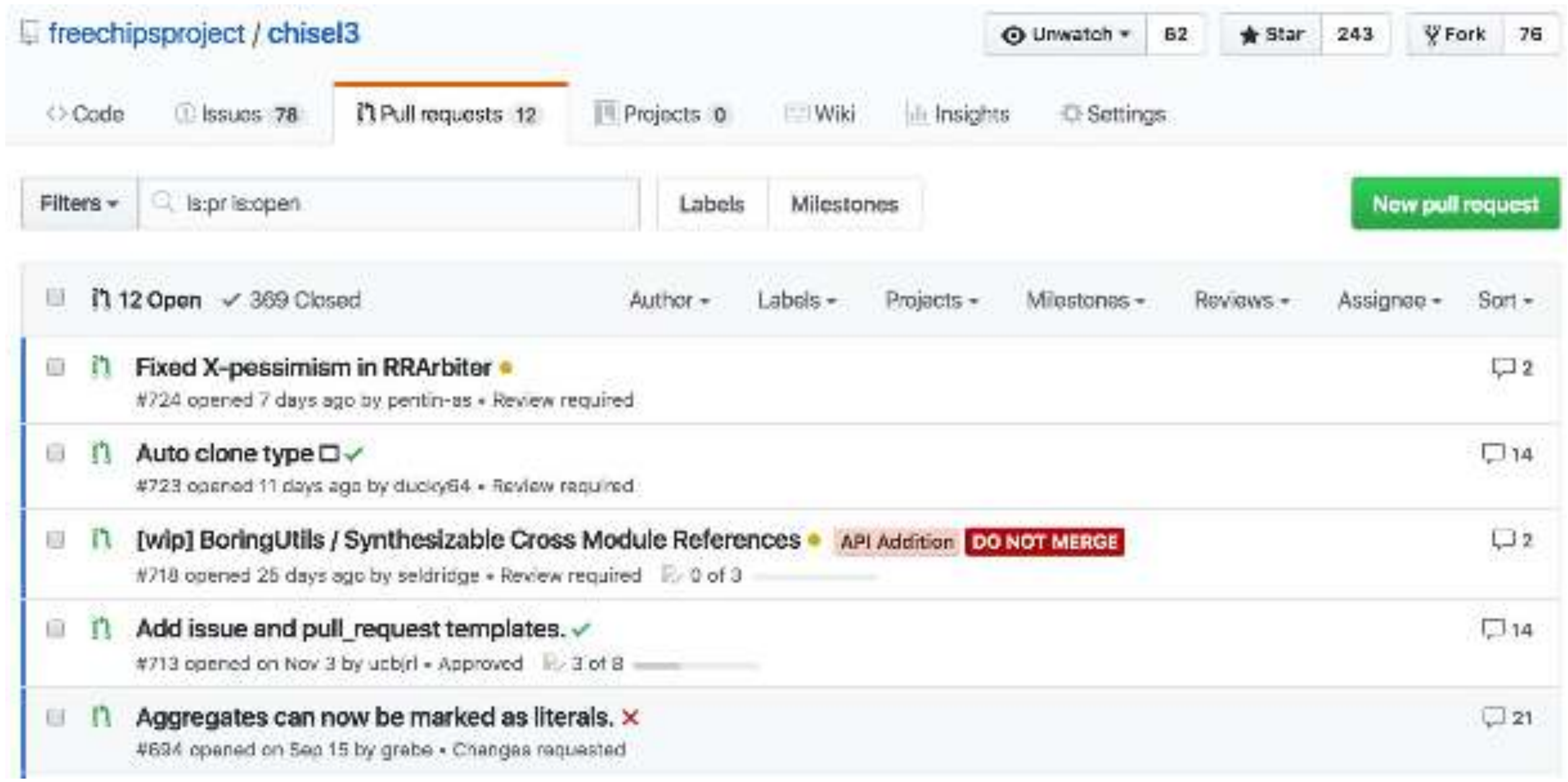
We declare a new module called `Tiny`.

```
val io = IO(...)
```

We declare all our input and output ports into a special io variable.

```
new Bundle {
```

# Open-Development via Github Issues/Pull Requests



freechipsproject / chisel3

Unwatch 62 | Star 243 | Fork 76

Code | Issues 78 | Pull requests 12 | Projects 0 | Wiki | Insights | Settings

Filters | is:pr is:open | Labels | Milestones | New pull request

12 Open ✓ 369 Closed | Author | Labels | Projects | Milestones | Reviews | Assignee | Sort

Fixed X-pessimism in RRArbiter ● — 2
#724 opened 7 days ago by pentin-as • Review required

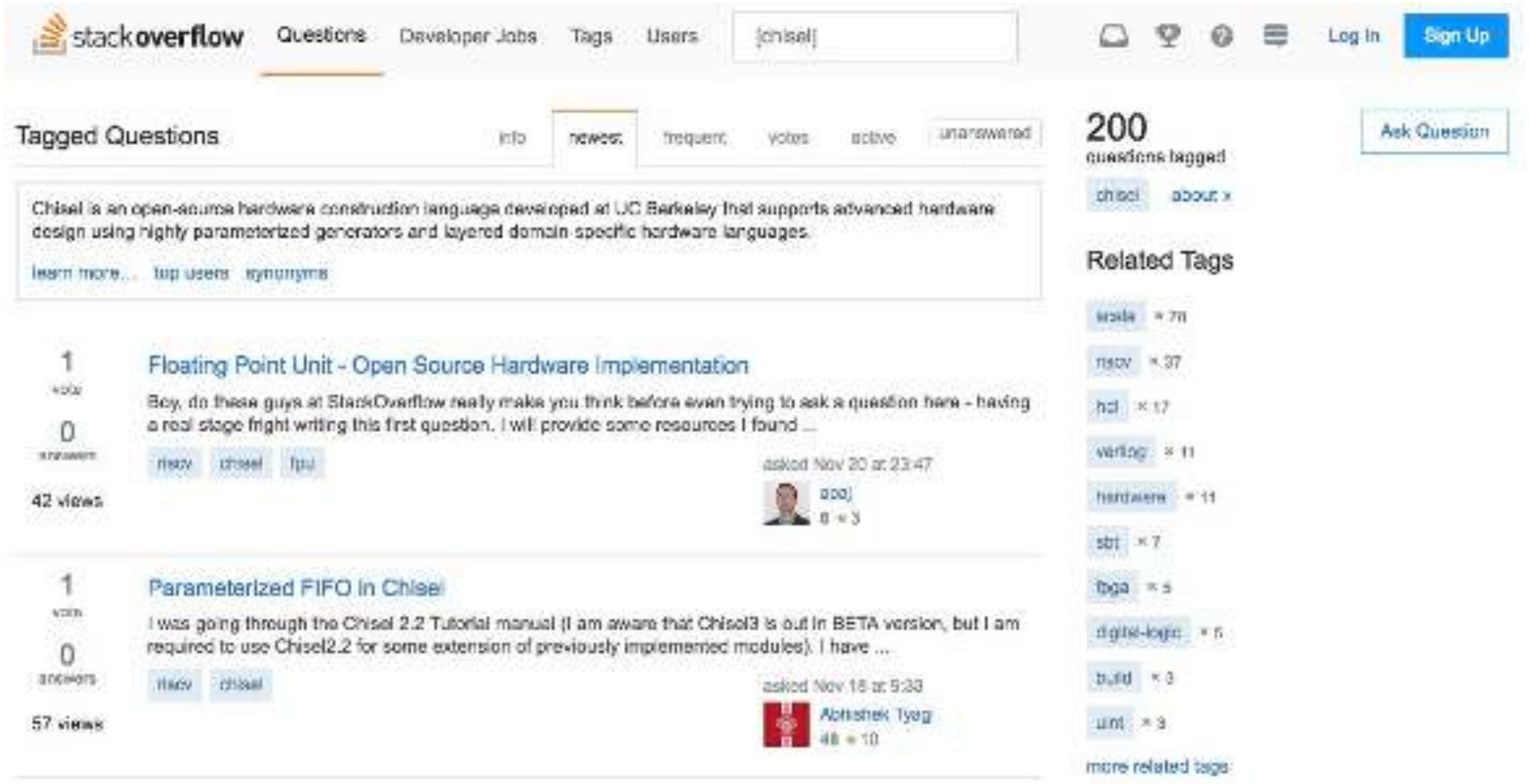
Auto clone type ✓ — 14
#723 opened 11 days ago by ducky64 • Review required

[wip] BoringUtils / Synthesizable Cross Module References ● API Addition DO NOT MERGE — 2
#718 opened 25 days ago by seldridge • Review required 0 of 3

Add issue and pull_request templates. ✓ — 14
#713 opened on Nov 3 by ucbjrl • Approved 3 of 8

Aggregates can now be marked as literals. ✗ — 21
#634 opened on Sep 15 by grebe • Changes requested

## Stack Overflow!



Tagged Questions

info newest frequent votes active unanswered

Chisel is an open-source hardware construction language developed at UC Berkeley that supports advanced hardware design using highly parameterized generators and layered domain-specific hardware languages.

learn more… top users synonyms

1 vote 0 answers 42 views

Floating Point Unit - Open Source Hardware Implementation

Boy, do these guys at StackOverflow really make you think before even trying to ask a question here - having a real stage fright writing this first question. I will provide some resources I found ...

riscv chisel fpu

asked Nov 20 at 23:47 aooj 8 ● 3

1 vote 0 answers 57 views

Parameterized FIFO in Chisel

I was going through the Chisel 2.2 Tutorial manual (I am aware that Chisel3 is out in BETA version, but I am required to use Chisel2.2 for some extension of previously implemented modules). I have ...

riscv chisel

asked Nov 18 at 5:33 Abhishek Tyag 48 ● 10

200 questions tagged

chisel about »

Ask Question

Related Tags

scala × 78
riscv × 37
hdl × 17
verilog × 11
hardware × 11
sbt × 7
fpga × 5
digital-logic × 5
build × 3
uint × 3

more related tags

## Academic Impact: 338 citations (as of 2019)

Chisel: Constructing Hardware in a Scala Embedded Language

[illegible]

Berkeley
UNIVERSITY OF CALIFORNIA

IBM

Stanford University

Microsoft Research

Cornell University

DTU Technical University of Denmark

UNIVERSITY OF CAMBRIDGE

TEXAS TECH UNIVERSITY

Massachusetts Institute of Technology

CARNEGIE MELLON UNIVERSITY

NTNU Norwegian University of Science and Technology

Nara Institute of Science and Technology NAIST

BERKELEY LAB

UNIVERSITY of WASHINGTON

BOSTON UNIVERSITY

## Active Users (That We Know Of)

SiFive

Stanford University

DRAPER

Berkeley Architecture Research

BERKELEY LAB

antmicro

intel

Esperanto TECHNOLOGIES

DTU Technical University of Denmark

Massachusetts Institute of Technology

Yale

BAE SYSTEMS

IBM

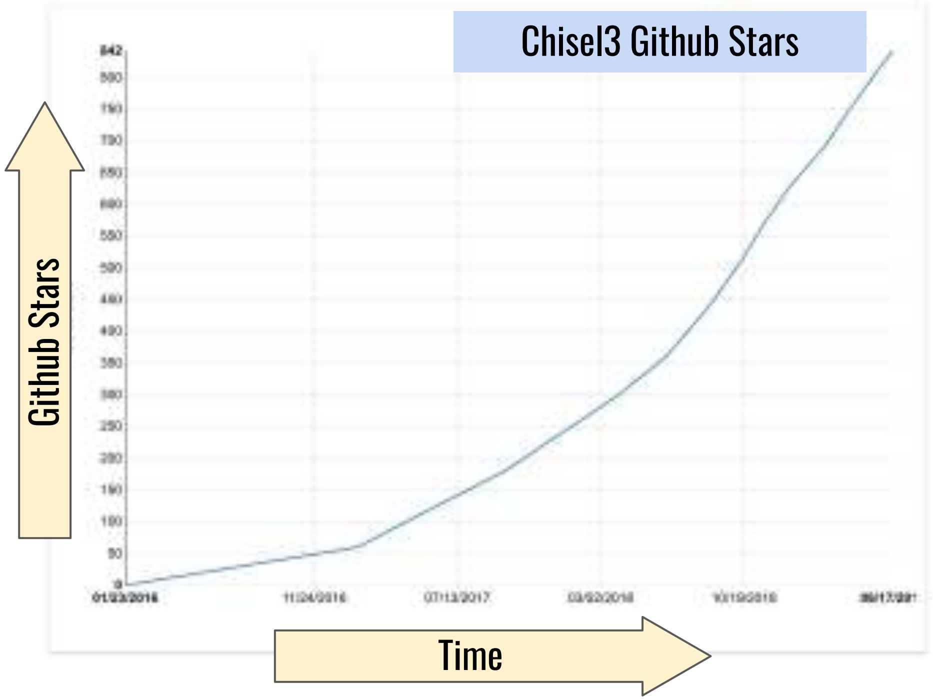
lowRISC

BWRC

Google

## Looking Forward

# Where is Chisel headed?



## Future Areas of Growth

### Diversify and Strengthen Community

- In-Person Meetups
- Open Dev Meetings
- Commercial Collaboration
- Enable Passionate Individuals

### Develop Needed Features and Provide Timely Support

- Standardize Verification Infra
- Generating Hardware Collateral
- Preserve Backwards Compatibility
- Advanced Circuit Transformations

### Formalize Governance for Ecosystem Stability

- CHIPS Alliance Partnership
- Constitution and Member Rights
- Communicate Direction/Roadmap
- Closer Commercial Partnerships

## Future Areas of Growth

### Diversify and Strengthen Community

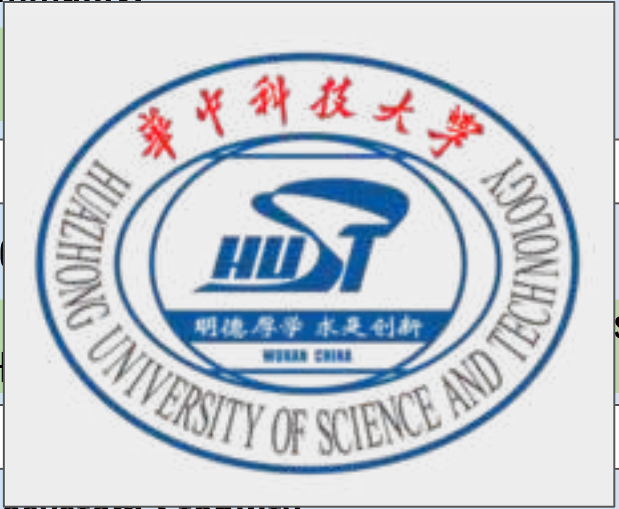
In-Person Meetups

### Develop Needed Features an

Standardize Verification Infra

H

s



### Formalize Governance for Ecosystem Stability

CHIPS Alliance Partnership

Constitution and Member Rights

Communicate Direction/Roadmap

Closer Commercial Partnerships

## Excited? Intrigued? Skeptical?

SiFive China

乐鑫信息科技
ESPRESSIF SYSTEMS

优矽科技
UC TECH IP

重德智能

## Shoutout to Chiselers out there! (And many more!!)