Open Source Hardware Verification

A survey and suggestions for future work

Ben Marshall

University of Bristol Computer Science Department

Outline

- Introduction & Motivation
- A very brief history of commercial EDA & Verification
- Where open source verification is now
- Current challenges for open source verification
- Opportunities for contribution
- Questions & Discussion

Introduction & Motivation

Who am I?

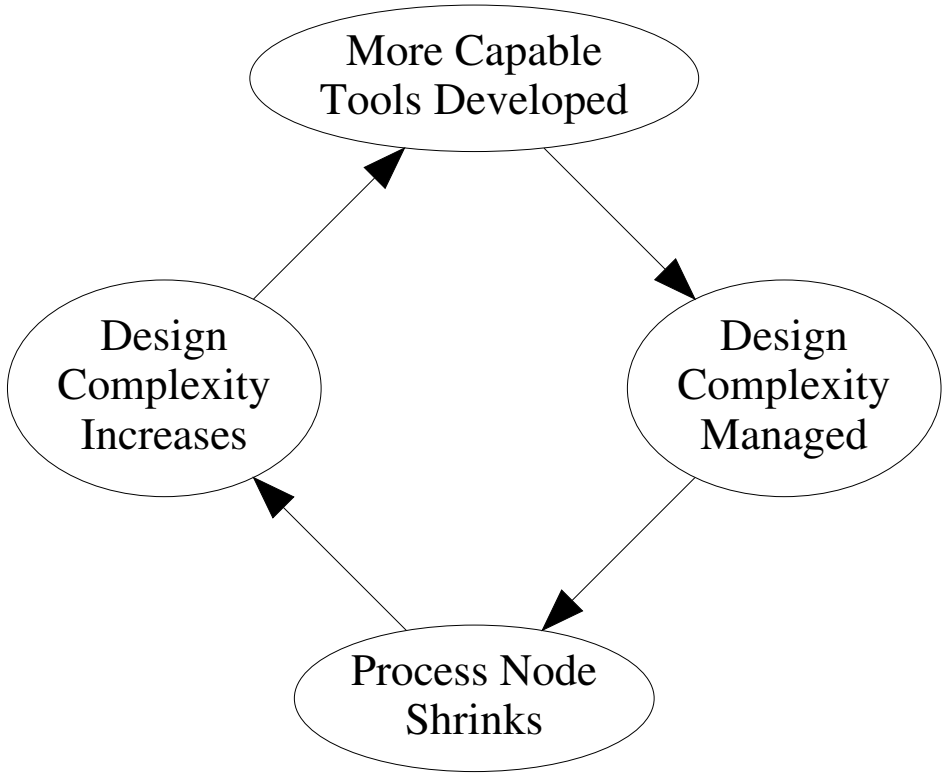
- Background in commercial CPU design and verification.
- Currently working in academia on a cryptographic instruction set extension for RISC-V.
  - XCrypto: `https://github.com/scarv/xcrypto`

Motivations for this talk:

- We spent lots of time looking for existing designs we could build on.
- It became very hard to find out how (if at all) a project or component had been verified.
- We really wanted to invest in and contribute too open source hardware designs, but didn't know which ones to trust.

A Brief History of Commercial EDA

- Fundamentally, EDA tooling development is driven by the need / ability to realise larger and more complex designs.
- Developments in verification techniques are driven by the need to manage the growing complexity of designs.



Where Open Source EDA Fits In

Design Description

Verilog / VHDL / Chisel / MyHDL / SpinalHDL / Clash / PyGears

Simulation

Icarus / GHDL / nvc / Verilator / Treadle

Synthesis

Yosys / ABC

FPGA Toolchains

Project IceStorm / Project X-Ray

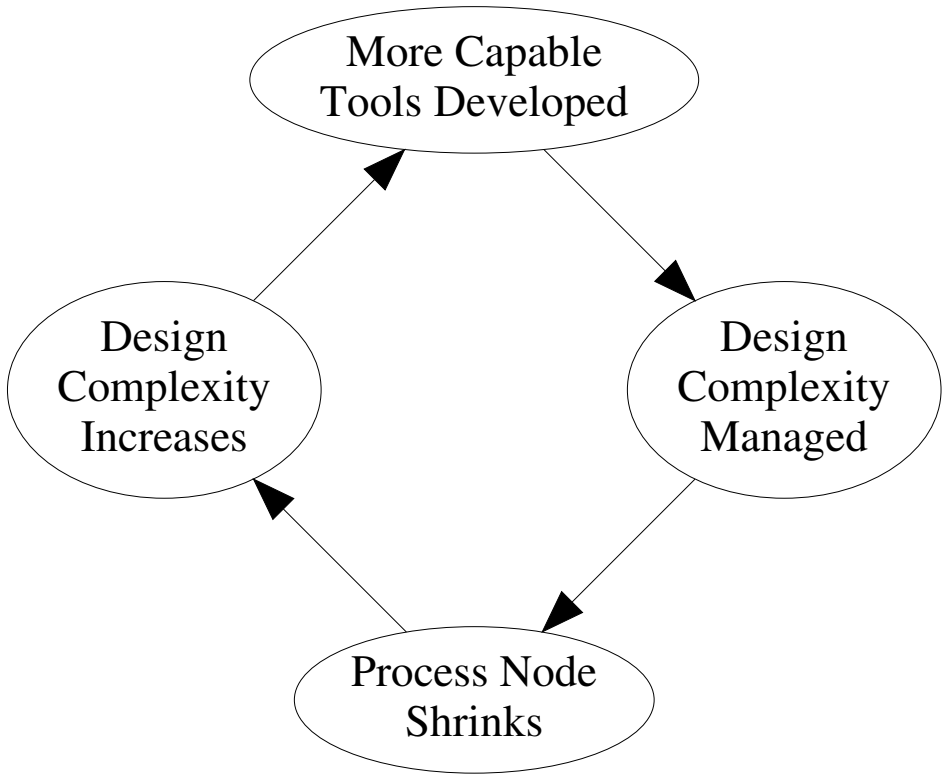
Place & Route

NextPNR / Qflow / Verilog2Routing / Arachne-pnr

- See: `https://github.com/drom/awesome-hdl` for more.

Where Open Source EDA Fits In

- Open Source *Design* tools for hardware are fantastic and getting better all the time.
- We are at the point where it's worth using open source tools to build bigger, more complex designs.
- Open Source designs are being physically manufactured
- More and more designs are appearing on OpenCores / GitHub



Argument: Verification is being neglected

Three Points:

1. Functional verification is a second class citizen in open source hardware.
2. Quantitative evidence of verification effort for open source hardware designs is hard to find and demonstrate.
3. We have *almost* everything we need to do a good job verifying open source hardware with open source tools.

1. Verification is Treated as Second Class

- Only 2.8% of designs on OpenCores describe themselves as test and verification related.
- It's easier to find designs than it is to find re-usable verification IP.
- The motivations sometimes aren't there to make verification seem worthwhile.
- Design is seen as the more interesting problem?

2. Evidence of Verification

Evidenced Verification Means:

- A statement of what verification effort as been attempted.
- A coverage number, representing the degree of design functionality stressed by the verification.
  - Slightly different for formal verification methods.

Why is this useful

Being able to answer questions like:

- Have I seen instruction X followed by instruction Y?
- Have I seen instruction X raise all of the exceptions it can correctly?
- Does my bus master function correctly under stall conditions?

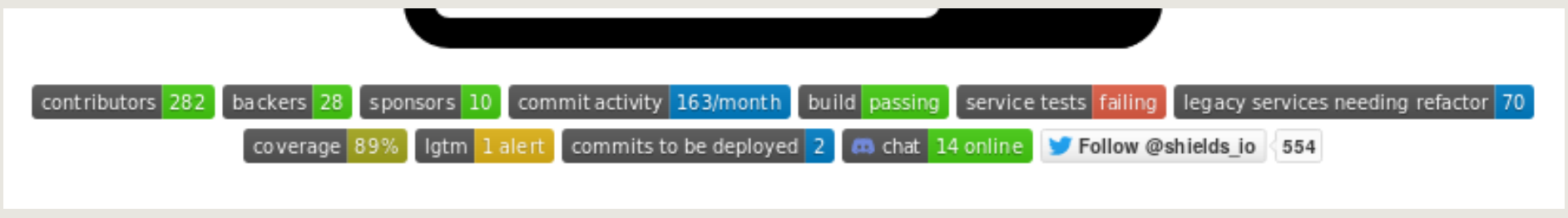
It lets people know where they can help out the most.

2. Evidence of Verification

But...

- There is no consistent way of showing this across existing projects.
- Many projects simply don't state what has been verified.
- Absence of evidence is evidence of absence.

The software world *loves* badges... Maybe we could too?



3. We have (almost) everything we need!

Coverage Collection

- Verilator supports line and toggle coverage.
- Meta-HDLs can get line coverage for free from their parent language.
- Functional coverage is a bit more work, but possible.

Verification Libraries

- UVVM / OSVVM / VUnit / Cocotb

Formal Tools

- SymbiYosys / Z3 and friends

`github.com/ben-marshall/awesome-open-hardware-verification`

An open source verification wishlist

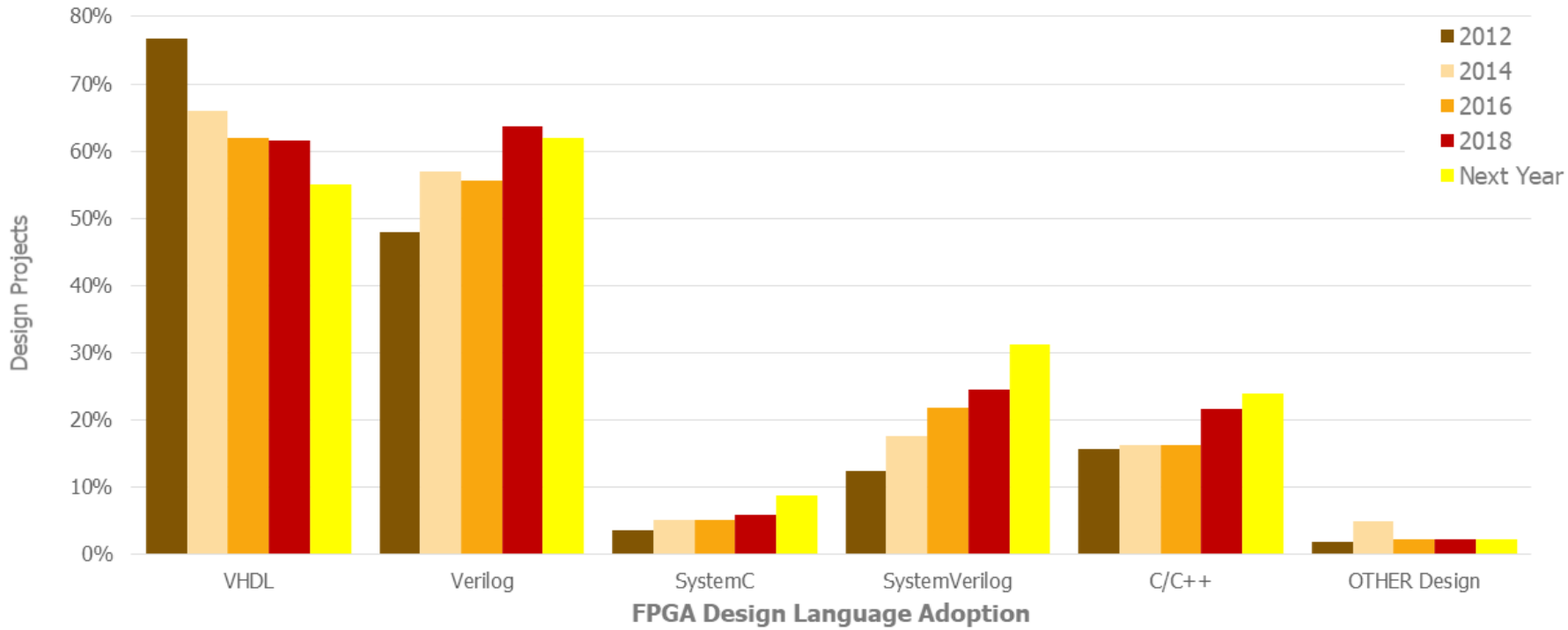
- Re-usable behaviour specifications
  - Use the same spec to generate testbenches / formal properties for different languages / implementations.
- Verification Libraries for Meta-HDLs
  - Functional Coverage Collection
  - Stimulus Generation
  - Re-usable verification IP
- A verification mindset
- Companies: If you are open-sourcing a design, why not include the verification infrastructure as well?

Learning from Industry

- **Remember:** Industry has been through all of this before.
- Use the lessons from industry to decide where to invest in tools.
  - Methodologies, best practices etc.
- *Wilson Research Group Functional Verification Studies: 2012-2018*
  - Extremely good resource for looking at how commercial verification has changed over time.
  - Verification is not a solved problem: **As of 2018, 84% of surveyed FPGA design projects suffered a non-trivial bug escape into production**

Learning from Industry

FPGA: Design Language Adoption Next Twelve Months



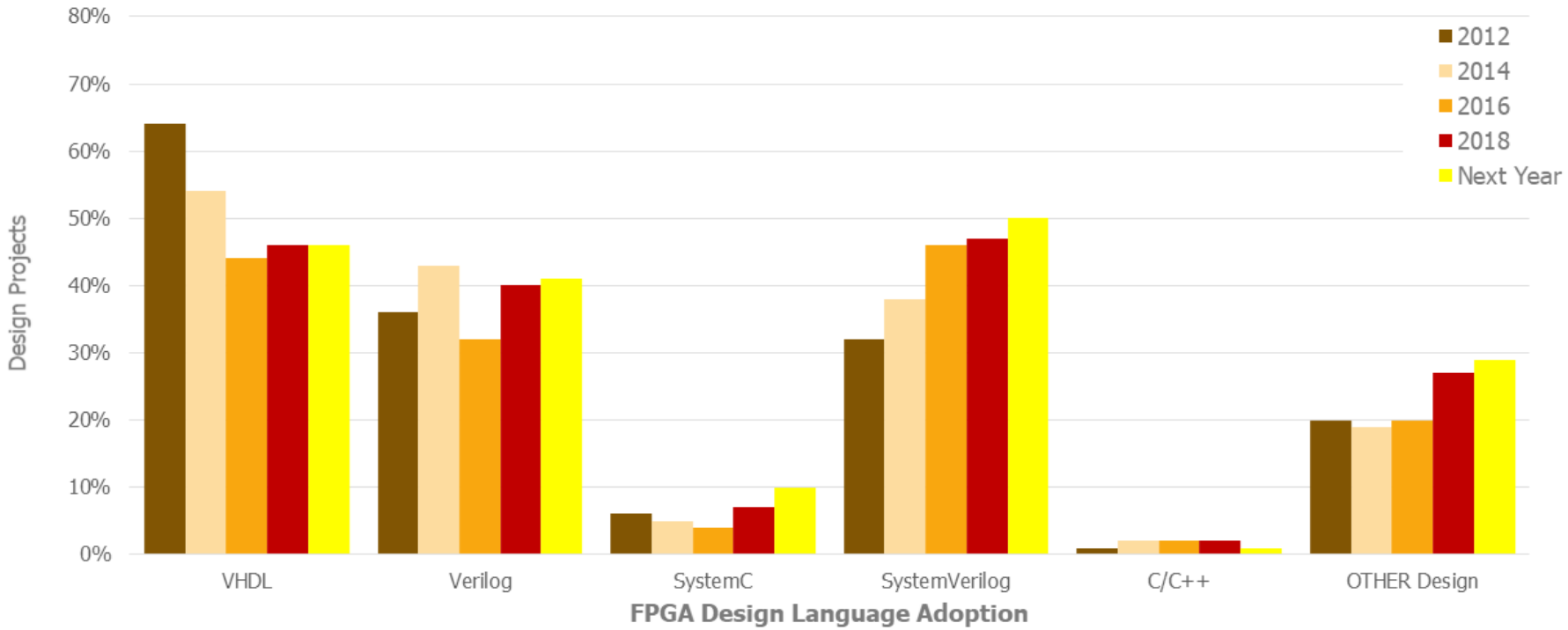
Source: Wilson Research Group and Mentor, A Siemens Business, 2018 Functional Verification Study

* Multiple answers possible

Mentor A Siemens Business

Learning from Industry

FPGA: Verification Language Adoption



Source: Wilson Research Group and Mentor, A Siemens Business, 2018 Functional Verification Study

* Multiple answers possible

Mentor A Siemens Business

Miscellaneous Talking Points

- Verifying security claims
- Human factors
  - More software oriented people moving into hardware development.
  - Agile development practices more suited to verification than to design?
  - Different perspectives on *product* development and testing.
- Commercial hardware development team structures look very different to open source governance / development structures.

Conclusions

- Open source hardware designs should be proud to show off how well verified they are.
- Developing re-usable verification infrastructure is an extremely worthwhile contribution to open source hardware development.
- There is already plenty of material from industry on how to do a good job of verification.
  - Guides on doing the same thing with open source tools are another valuable contribution.
- We should remember that verification is *as important* as design. We can't afford to neglect it.