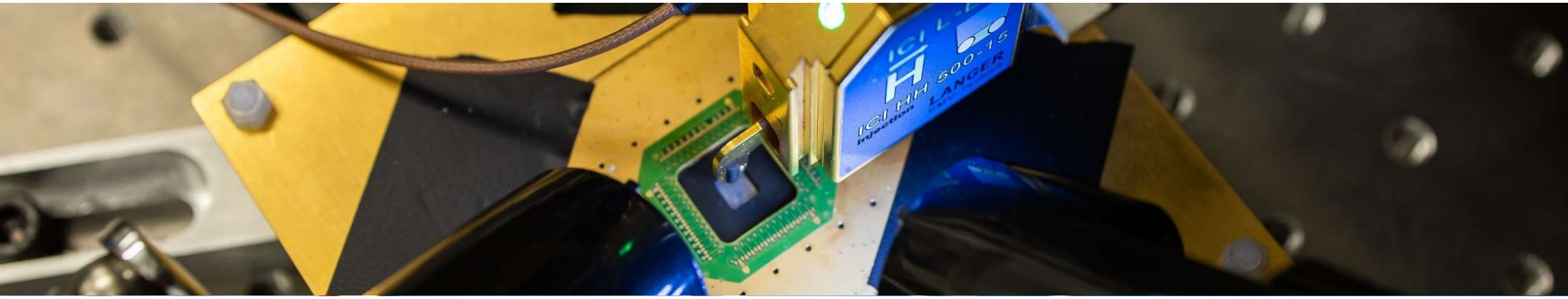


VE-HEP: **H**ardening the value chain through open source, trustworthy **E**DA tools and **P**rocessors

Formal Verification as a Means to Trustworthiness

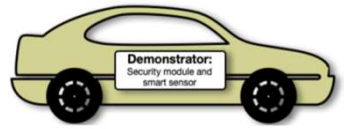
Milan Funck, Tim Henkes, Norbert Herfurth, Christoph Lüth, Steffen Reith, Arnd Weber

10.03.2022

VE-HEP: Hardening the value chain through open source, trustworthy EDA tools and processors

- Project Goal: **Reliability** and **Trustworthiness** by
  - Open Source hardware (RISC-V)
  - Open Source tool chain (OpenRoad)
  - Formal verification

- Demonstrator:
  - Hardware Security Module
  - Based on VexRiscV
- Covers the whole value chain:



… to IHP fab.

… via OpenRoad …

From SpinalHDL …

Partners:

Associated:

Grant no.
ME1ZEUS012/16KIS1342

Verifying a RISC-V CPU

Functional Verification needs

- Specification of desired system behaviour
- Abstract model of the system

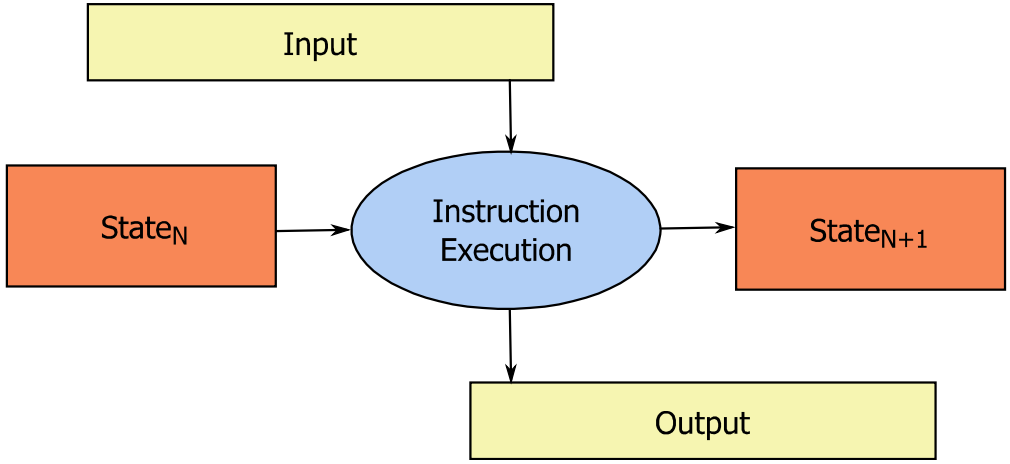
Riscv-Formal specifies behaviour (partly)

Abstract model:

- CPU state consists of all registers

Correct behaviour as function

$$State_{N+1} = FS(State_N, Input)$$
$$Output = FO\ (State_N, Input)$$



Our Approach: Symbolic Model Checking

Extension of the Riscv-Formal interface

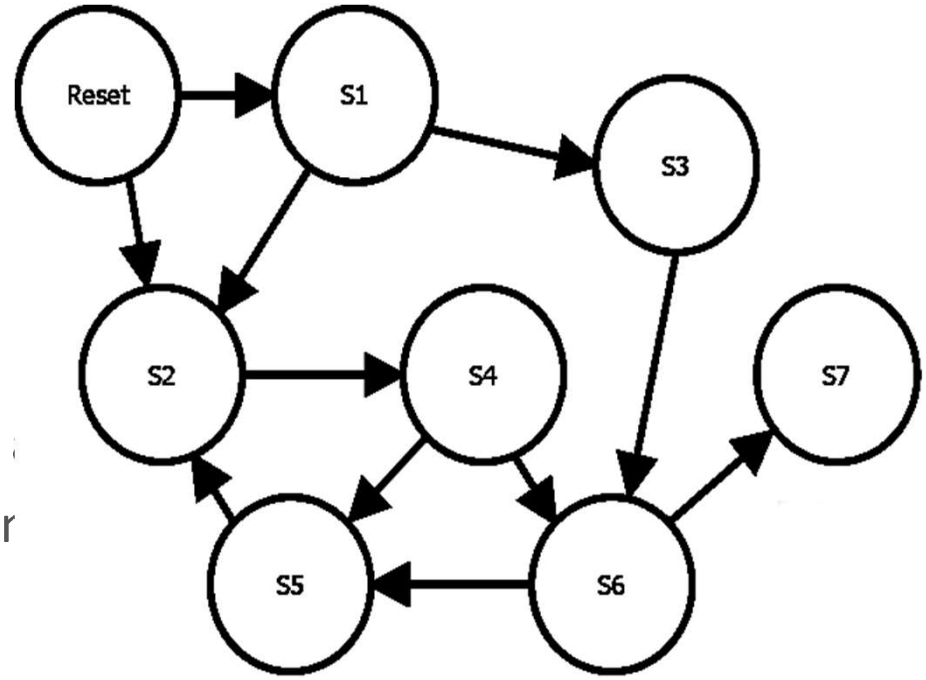
- full access to the state (Registers, CSRs, PC)

Let the solver engine choose to either

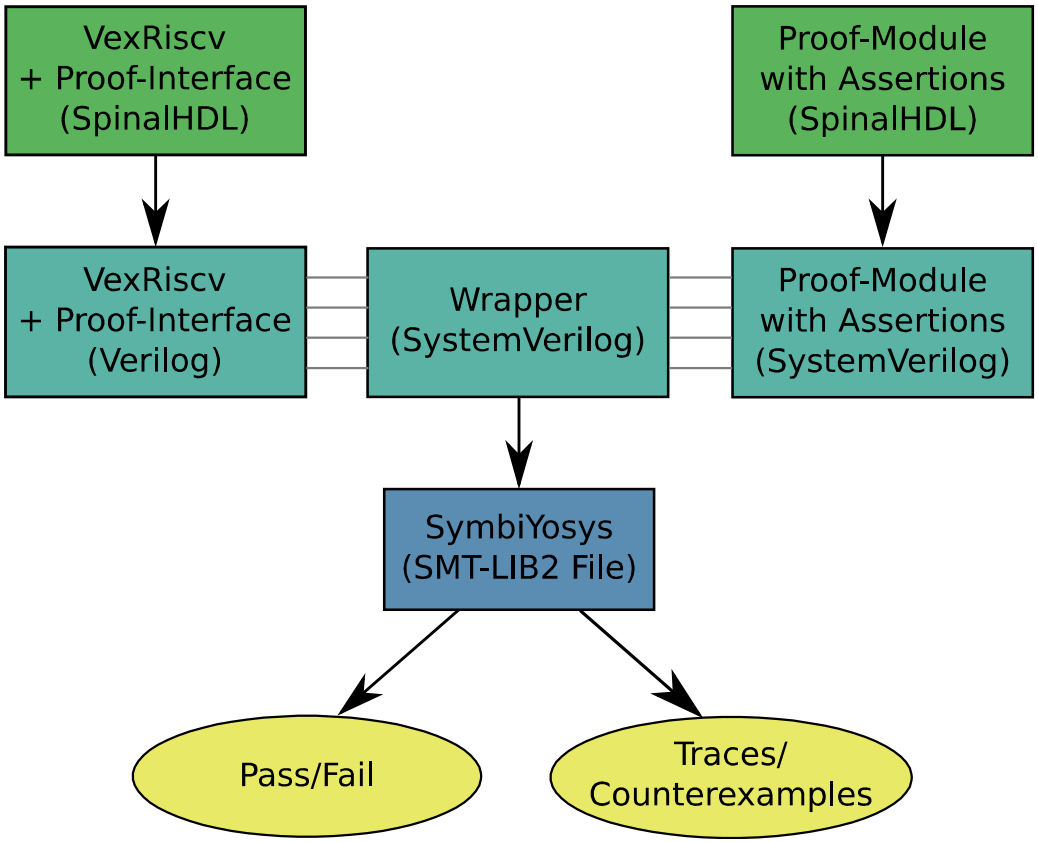
- start from the reset state, or
- start from any other valid state

Symbolic Model Checking:

- Let the solver choose any input.
- Give the solver just enough cycles to execute
- Verify the correctness of the resulting state ar



Tool Chain & Work Flow



Current State of Work

We have verified a simple RISC-V core (µRV32I)

- ~3580 lines of Verilog-Code
- No pipeline
- RISC-V 32I with basic set of 17 CSRs

Results for the current proof:

- Verified/covered correct execution of the 37 RISC-V 32I basis instruction:
  - few simple bugs found
- Verified correct behaviour for any other unknown/illegal instruction:
  - multiple decoder-bugs found
- SpinalHDL-Code: ~ 1250 lines, generated SystemVerilog code: ~1900 lines
- Execution time: ~3,5 minutes

Future Plans and Potential Problems

- Future Plans:
  - Extend verification to VexRiscv -- pipeline is challenging
  - Extend verification for fence-, ecall-, and basis csr-instructions
  - Equivalence check for hardened processor

- When is a State valid?
  - CPU-Abstraction loses other internal registers

- Some insider knowledge is required
  - No longer a black box verification

Trustworthiness by Formal Verification

Concept/Specification/Design:

- Unintentional vulnerabilities:
  - Example in HEP: RISC-V ISA
    - Prove all instructions are implemented correctly
    - Prove further invalid instructions are handled correctly
    - Hardening does not introduce vulnerabilities
- The Importance of Being Formal:
  - allows reasoning about completeness and consistency;
  - specifications are well-defined and unambiguous;
  - reviewing focusses on specification, not implementation.
- Works well with open source (hardware and software)
  - Anybody can run the verification for themselves.

Thank you for your attention!

For more information:

http://hep-alliance.org/

Contact:

Prof. Christoph Lüth
Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI)
christoph.lueth@dfki.de
http://www.informatik.uni-bremen.de/~clueth