# OpenHW Group

## Proven Processor IP

## Industrial-Grade Verification for Open-Source RISC-V Cores

Mike Thompson

mike@openhwgroup.org

# Topics of this Presentation

- Vision for verification of open-source RISC-V related IP
- OpenHW Group's CORE-V-VERIF project.
- CORE-V-VERIF: OpenHW's single-core Verification Environment.
- Opportunities to apply FORCE-RISCV:
  - Core-v-verif
  - Vector Processor (CVVEC)
  - Multi-core verification environments

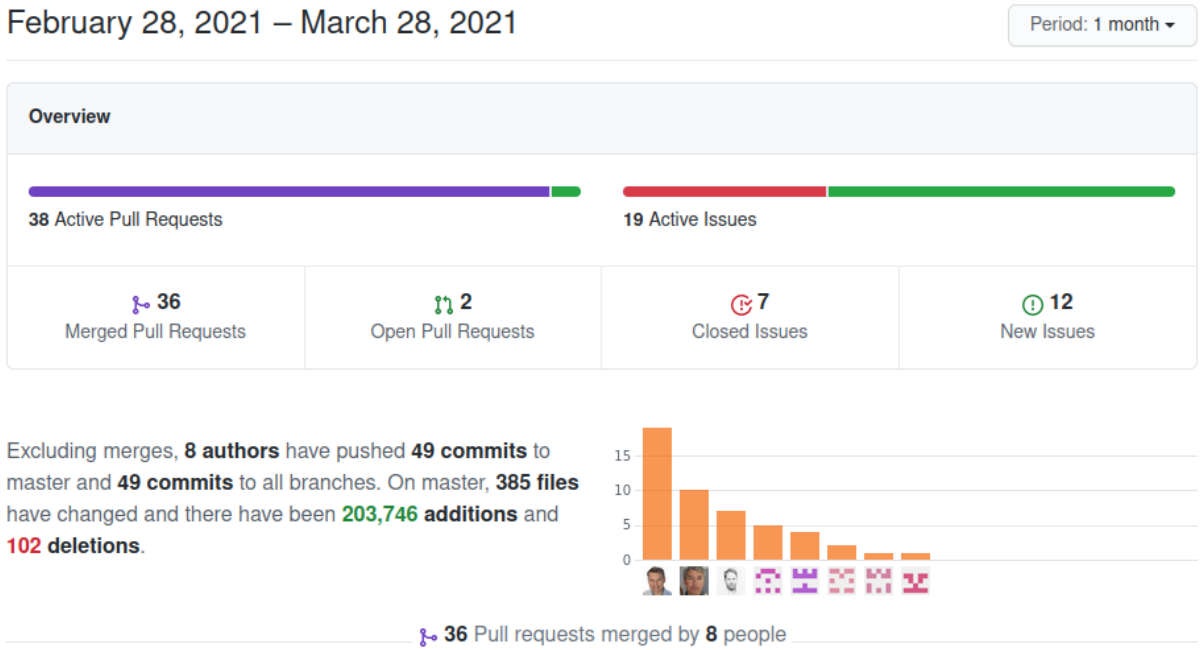
# Vision Statement

- The OpenHW Group will architect, implement and deploy verification environments capable of supporting “industrial-grade” verification of open-source CORE-V cores and related IP.

- This vision will be implemented in a step-wise fashion:
  - Step 1: a re-usable UVM verification environment for single RISC-V cores.
  - Step 2: a methodology for verification of CORE-V related IP such as co-processors, memory controllers, etc.
  - Step 3: a methodology for verification of multicore platforms based on CORE-V and related IP.

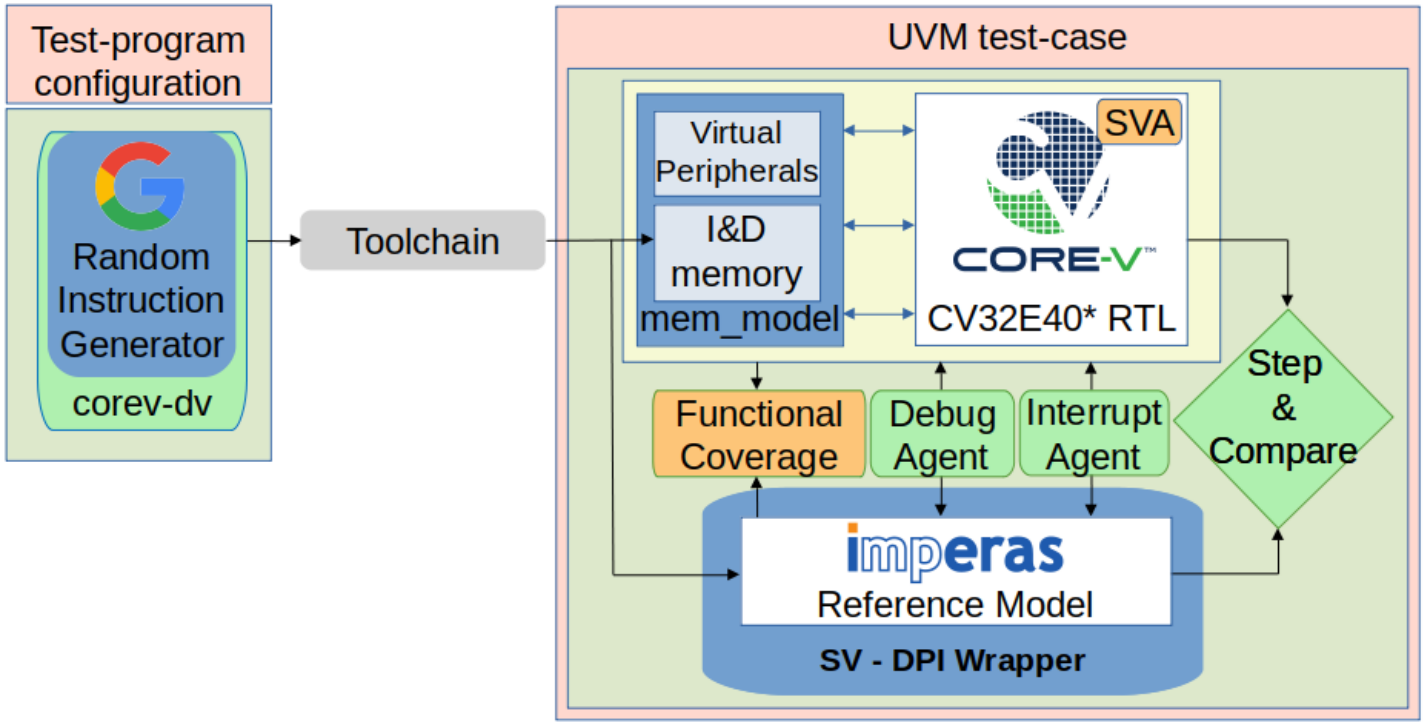
# What is core-v-verif?

- It is “Step 1” of the vision: it is home for the testbenches, test-programs, script-ware etc. for the CORE-V family of RISC-V cores.
  - Note that the cores themselves are maintained in separate repositories.
- CORE-V-VERIF is a GitHub Repository:
  - Very active over the past year.
  - > 25 Contributors.
  - > 525 issues (remember this is “just” for verification).
  - 36 merged PRs from 8 Contributors in March, 2021.
  - **New Contributors always welcome.**



# CORE-V-VERIF UVM Environment

- Proven environment for complete verification of single-core DUT:
  - Successfully deployed on CV32E40P
  - Currently supporting CV32E40X, CVA6.
  - Will be deployed for CV32E20 and CV32E41P.



# SystemVerilog Simulators

- CORE-V-VERIF **UVM** Verification verification environment runs on all major commercial SystemVerilog simulators:





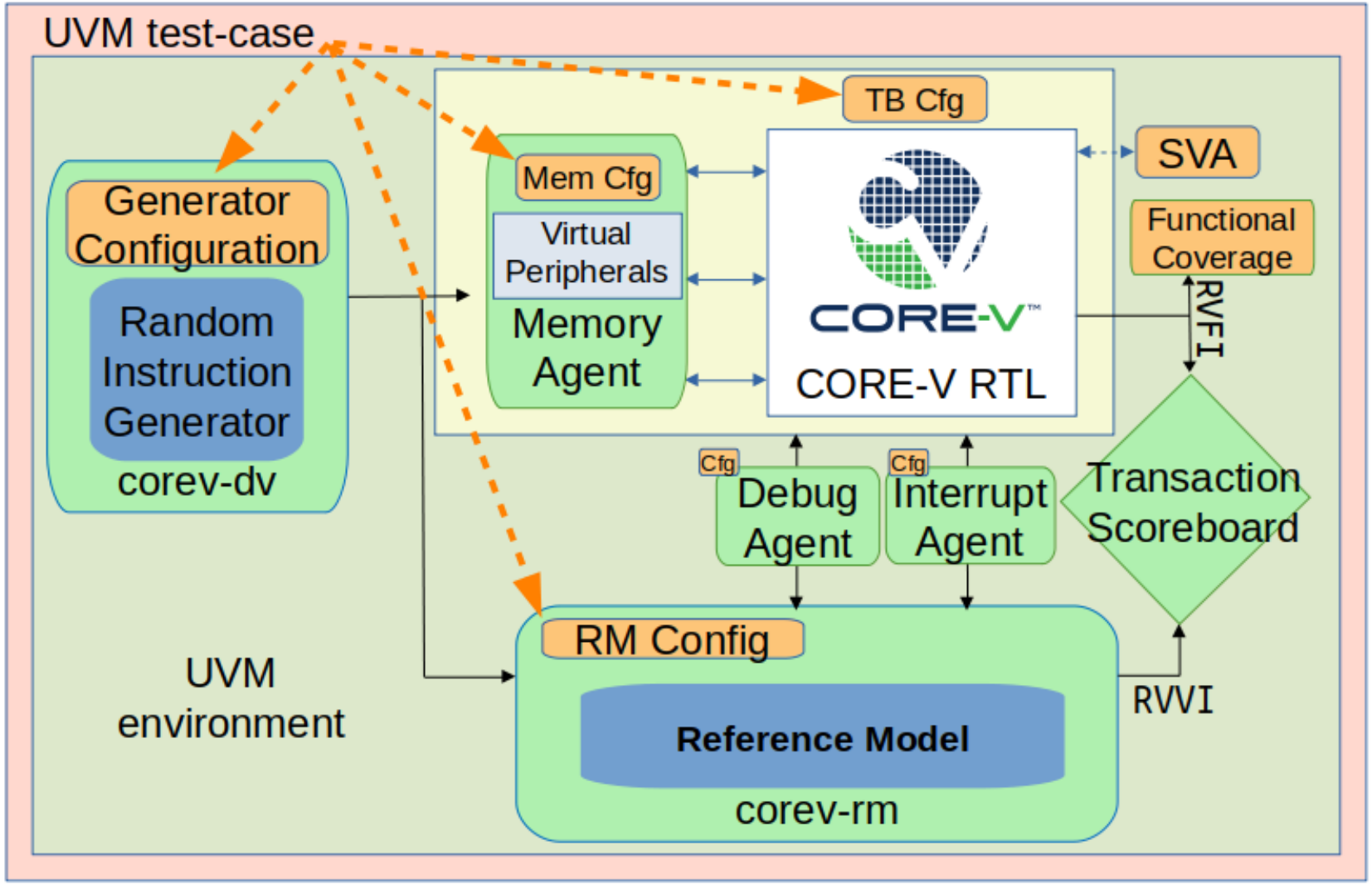
# Improvements to CORE-V-VERIF (1 of 2)

- Both Core RTL and RM presents non-standard, “core-specific” instruction trace interfaces to the environment:
  - Integrating a new core and/or RM involves a lot of re-work.
- Fixed Reference Model:
  - Changing the RM is a “forklift” operation that impacts step-and-compare function and functional coverage collection.
- ISA Functional coverage data sourced by Reference Model:
  - Should be sourced by RVFI to allow for operation independent of RM.
- Support cost of step-and-Compare functionality is high:
  - Requires throttling of the core clock (always confuses the Designers!).
  - Changes to cycle-timing behavior of core often changes cycle-timing behavior of core’s tracer interface - this makes the environment ‘brittle’.
  - Not future proof: it’s unclear if step-and-compare can support out-of-order cores.

# Improvements to CORE-V-VERIF (2 of 2)

- Only supports a single generator: riscv-dv
  - OpenHW does not have direct influence over future direction of riscv-dv.
- Generator is not a UVM component controllable by UVM run-flow:
  - Must be run in a standalone ‘simulation’ before running the core environment.
  - Control knobs for randomization are static.
- Disjoint control of configuration and stimulus:
  - Lack single-point-of-control of test-program, testbench configuration and UVM testcase.
- Static Memory Model:
  - Fixed Address range.
  - No support for PMA or Virtual Memory.
- Most Assertions are embedded in the RTL:
  - Makes them invisible to the DV plan.

# The Goal: a universal UVM environment for CORE-V cores



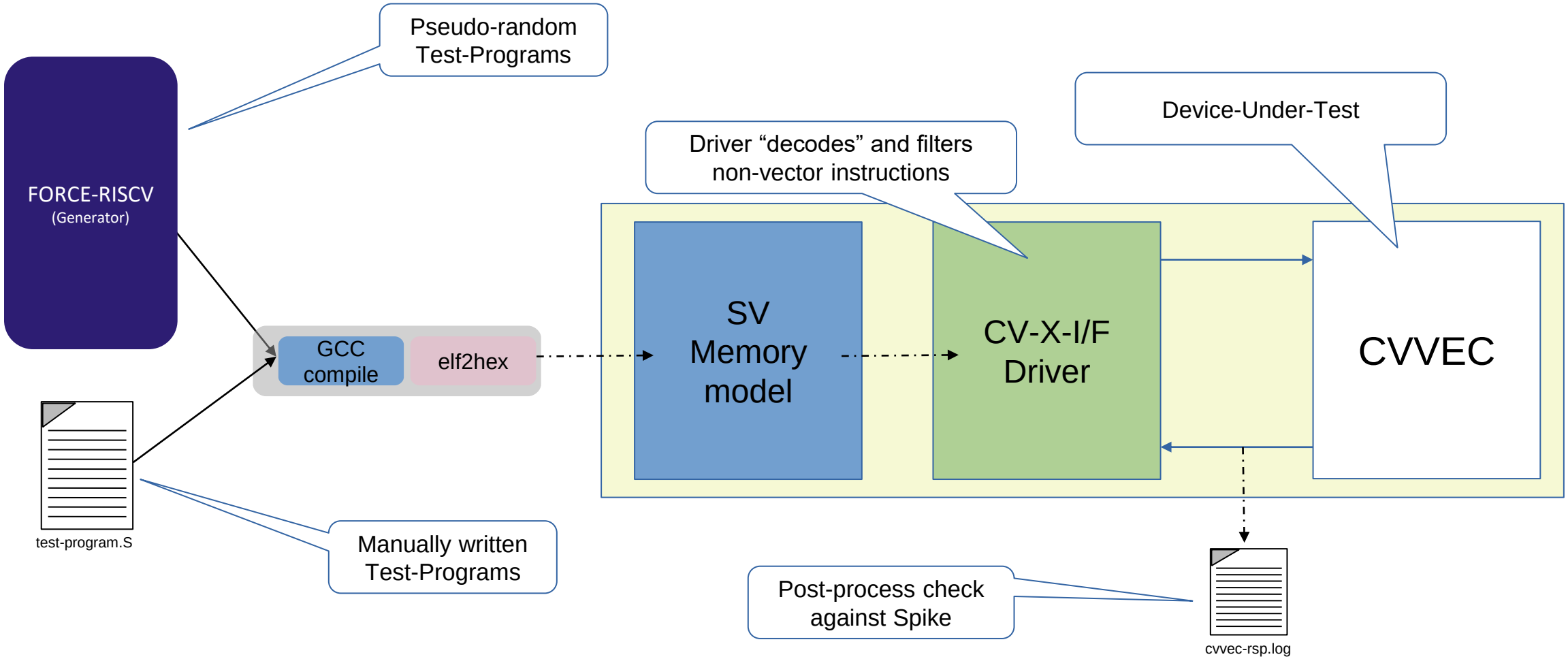
# Using FORCE-RISCV in CORE-V-VERIF

- Deploying FORCE-RISCV to CORE-V-VERIF should be considered if one or more of the following are true:
  - Google suspends support for SystemVerilog implementation of RISCV-DV.
  - RISCV-DV fails to adequately support the DV plan required by a core.
  - Futurewei agrees to support new features to ease integration of FORCE-RISCV into UVM environments.
  - CORE-V-VERIF is called upon to verify ann instruction set not currently supported by RISCV-DV.
    - The most obvious example of this is the RISC-V Vector ISA.

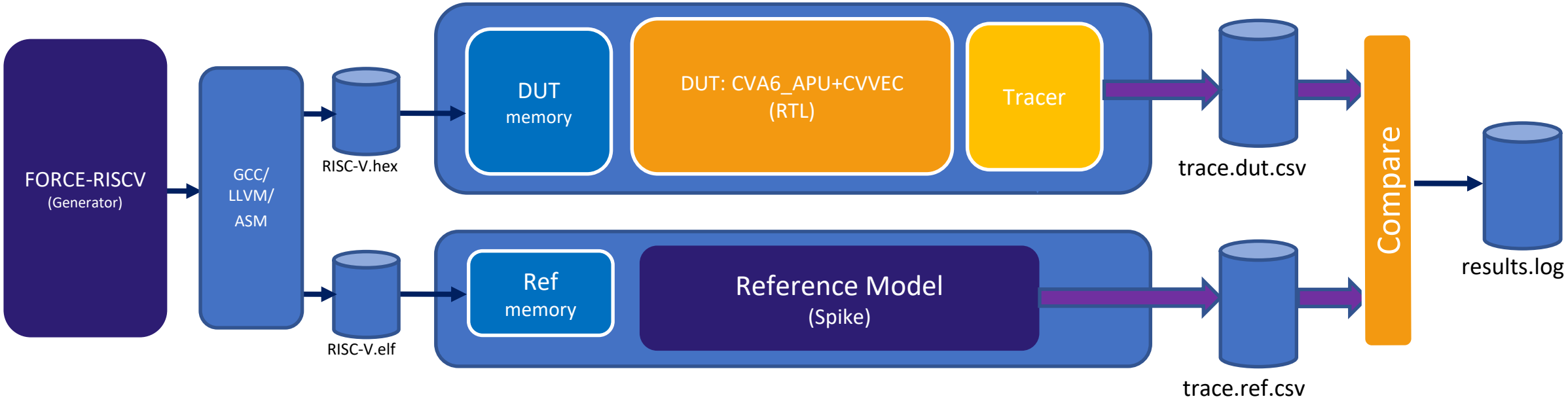
# Using FORCE-RISCV for CVVEC

- CVVEC is a new project that is not well supported by CORE-V-VERIF.
- OpenHW is currently in discussions with CVVEC team to determine best verification strategy:
  - Need to recognize that CVVEC is currently an academic, not industrial, project. A heavy-weight UVM environment and industrial verification flow may overburden the project.
  - Wish to minimize “throwaway work”.
- The following slides are concepts for using FORCE-RISCV as the generator for CVVEC:
  - These have not yet been reviewed or agreed upon.

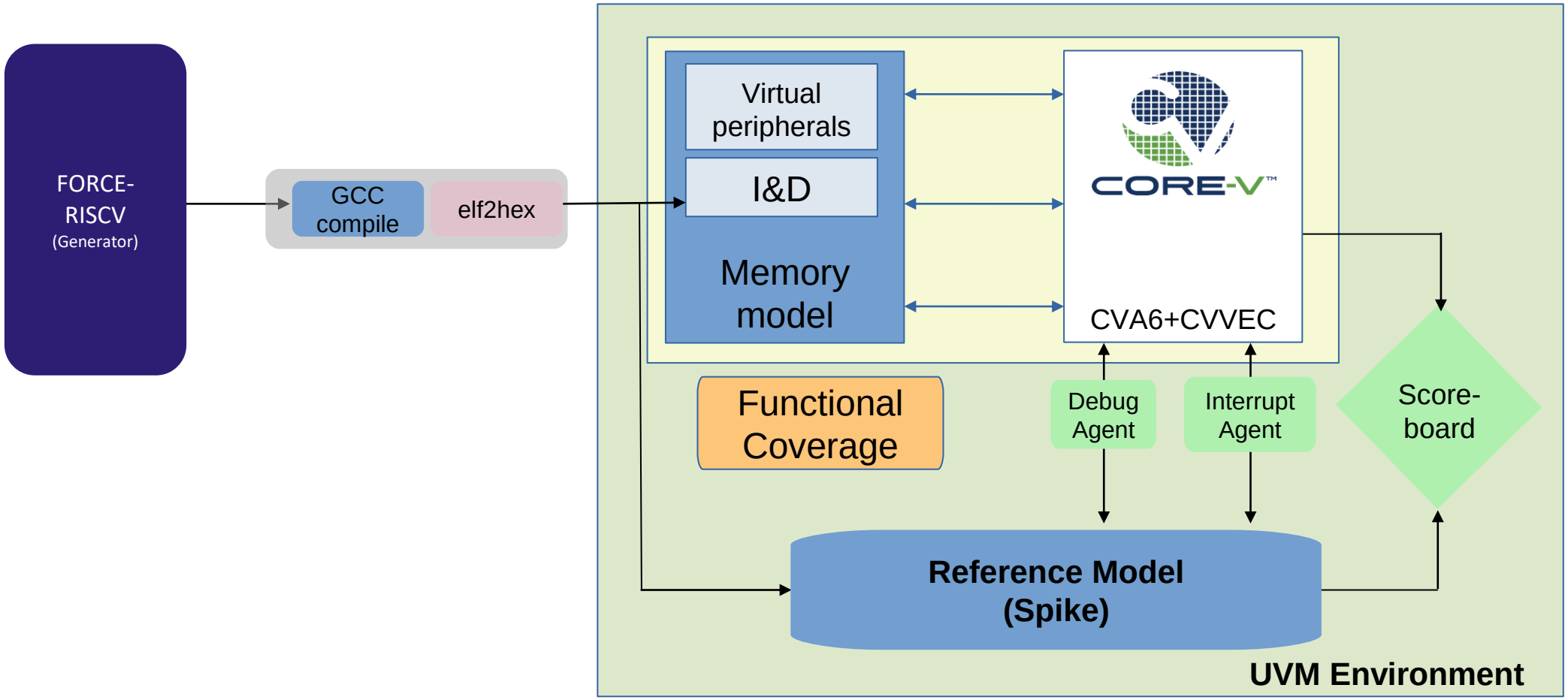
# Concept 1: Stand-alone CVVEC testbench



# Concept 2a: CVA6+CVVEC testbench



# Concept 2b: UVM Based CVA6+CVVEC testbench



# Multi-core Verification

- Up to 2021-06 OpenHW has not received a Project Concept proprosal for a multi-core project.
- Current thinking is that each multi-core project will be unique, and a universal environment such as CORE-V-VERIF is impractical.
- Futurewei has indicated that FORCE-RISCV has “built-in” support for multi-core verification:
  - We would like to learn more about this.

# Next Steps

- I believe the best near-term opportunity to use FORCE-RISCV on a COREV project is to engage with the CVVEC team.
- Developing a prototype of the “Concept 1” testbench would be a good place to start.
- Direct hands-on contributions by Futurewei engineers will ensure success.

# Thank You!