

OpenHW Coding Challenge Report

Date: 7.02.2026

Author: DRY

(Note: if you remove the images, the report is 1 page as required).

Environment setup

1. Windows 11 & WSL Fedora 42 Linux.
2. RISCv compiler: CV_SW_TOOLCHAIN=/opt/riscv/riscv-embecosm-embedded-rocky8-20250309/
3. Python: 313 didn't work, setup venv312 to fulfill requirements setup
4. Fix C++ code to remove warnings in /home/dry/projects/riscv/proc_cert/core-v-verif/cv32e40p/tb/core/tb_top_verilator.cpp, else wouldn't build due to c++ -Werror=missing-field-initializers]
c++ compiler version: gcc version 15.2.1 20260123 (Red Hat 15.2.1-7) (GCC)
5. -march issue when building test prog bsp:

```
make -C simulation_results/hello-world/0/test_program/bsp \
  VPATH=/home/dry/projects/riscv/proc_cert/core-v-verif/cv32e40p/bsp \
  RISCv=/opt/riscv/riscv-embecosm-embedded-rocky8-20250309/ \
  RISCv_PREFIX=/opt/riscv/riscv-embecosm-embedded-rocky8-20250309/bin/riscv32-unknown-elf- \
  RISCv_EXE_PREFIX=/opt/riscv/riscv-embecosm-embedded-rocky8-20250309/bin/riscv32-unknown-elf- \
  RISCv_MARCH=rv32imc \
  RISCv_compiler=/opt/riscv/riscv-embecosm-embedded-rocky8-20250309/bin/riscv32-unknown-elf-gcc \
  RISCv_FLAGS="" \
  all
make[3]: Entering directory '/home/dry/projects/riscv/proc_cert/core-v-verif/cv32e40p/sim/core/simulation_results/hello-world/0/test_program/bsp'
/opt/riscv/riscv-embecosm-embedded-rocky8-20250309/bin/riscv32-unknown-elf-gcc -Os -g -static -mabi=ilp32 -march= -Wall -pedantic -c /home/dry/projects/riscv/proc_cert/core-v-verif/cv32e40p/bsp/crt0.S -o crt0.o
riscv32-unknown-elf-gcc: error: missing argument to '-march'
```

..RISCv_MARCH did not propagate ..?

Fix: export CV_SW_MARCH=rv32imc_zicsr_zifencei

Validation

Why RTL code is not part of CV32E40P of the repository

With reference to [core-v-verif quick start](#), the reasons are as follows:

*The RTL implementation of the CV32E40P core is not included in this repository because CORE-V-VERIF is designed to be a **generic, reusable verification environment** that can be applied to multiple CORE-V processor implementations and revisions without duplicating RTL sources. Instead of embedding a specific core, the testbench accesses the CV32E40P RTL by **referencing it as an external dependency**, typically via a separate checkout of the core repository or a user-provided RTL path specified through build configuration variables. During simulation, the verification environment compiles and elaborates the externally supplied RTL together with the CORE-V-VERIF testbench, allowing the same verification infrastructure to be reused across different core versions, implementations, and simulators while keeping the verification logic cleanly decoupled from the design under test.*

(I acknowledge that I have elaborated with AI on this to get a clear and spell-checked summarized paragraph, but from software or engineering point of view this is how I see the reasons why.)

Screenshot with PASSED status

```
*****
Running with Verilator: logfile in simulation_results/hello-world/hello-world.log
*****
mkdir -p simulation_results/hello-world/0/test_program
simulation_results/hello-world/verilator_executable \
  \
  "+firmware=../tests/programs/custom/hello-world/hello-world.hex" \
  | tee simulation_results/hello-world/0/test_program/hello-world.log
scopesDump:
  SCOPE 0x77ccb8a0ea00: TOP.tb_top_verilator
  SCOPE 0x77ccb8a0ea40: TOP.tb_top_verilator.cv32e40p_tb_wrapper.i.ram_i
  SCOPE 0x77ccb8a0ea80: TOP.tb_top_verilator.cv32e40p_tb_wrapper.i.ram_i.dp_ram_i
  DPI-EXPORT 0x426690: read_byte
  DPI-EXPORT 0x4266a0: write_byte
[tb_top_verilator] finished dumping memory
HELLO WORLD!!!
This is the OpenHW Group CV32E40P CORE-V processor core.
CV32E40P is a RISC-V ISA compliant core with the following attributes:
mvendorid = 0x002
marchid = 0x4
mimpid = 0x0
misa = 0x40001104
XLEN is 32-bits
Supported Instructions Extensions: MLC
TOP.tb_top_verilator @ 128430: EXIT SUCCESS
- /home/dry/projects/riscv/proc_cert/core-v-verif/cv32e40p/tb/core/tb_top_verilator.sv:83: Verilog $finish
make[1]: Leaving directory '/home/dry/projects/riscv/proc_cert/core-v-verif/cv32e40p/sim/core'
```

Dump wave forms

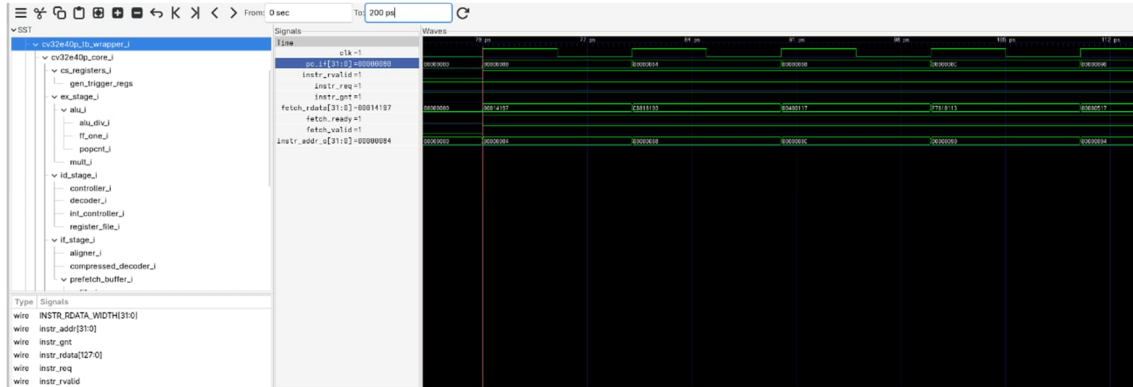
I do not have any commercial tool like DSIM, and the provided Makefile rule to generate *.fst assumes DSIM exists which is a commercial utility. However, we can generate *.vcd and then either load that or convert that file to *.fst:

1. make veri-test TEST=hello-world WAVES=1
2. gtkwave verilator_tb.vcd; or : vcd2fst verilator_tb.vcd verilator_tb.fst then gtkwave verilator_tb.fst

ISSUE: seems generated *.fst is corrupt, so that vcd2fst does not work locally on my system ...

*.vcd gets loaded ok.

3. Highlight op of instruction fetch interface on gtkwave:



Strech goal

(was is 'Stretch' or 'Scratch'? required test is for a _scratch_ register...)

The test case code is in the pull request (see user "**dry-75**", branch "**challenge-dry**"). Example from test run:

make veri-test TEST=csr_mscratch32

```
Verilator Report: Verilator 5.034 2025-02-24 rev fedora-5.034
- Verilator: Built from 0.000 MB sources in 0 modules, into 0.000 MB in 0 C++ files needing 0.000 MB
- Verilator: Walltime 0.010 s (elab=0.000, cvt=0.000, bld=0.000); cpu 0.005 s on 1 threads; allocated 17.793 MB
make -C cobj_dir -f Vtb_top_verilator.mk
make[1]: Entering directory '/home/dry/projects/riscv/proc_cert/core-v-verif/cv32e40p/sim/core/cobj_dir'
g++ -tb_top_verilator.o verilator.o verilator_dpi.o verilator_threads.o Vtb_top_verilator.o -pthread
make[1]: Leaving directory '/home/dry/projects/riscv/proc_cert/core-v-verif/cv32e40p/sim/core/cobj_dir'
mkdir -p simulation_results
mkdir -p simulation_results/csr_mscratch32
mv cobj_dir/Vtb_top_verilator simulation_results/csr_mscratch32/verilator_executable
*****
* Running with Verilator: logfile in simulation_results/csr_mscratch32/csr_mscratch32.log
*****
mkdir -p simulation_results/csr_mscratch32/0/test_program
simulation_results/csr_mscratch32/verilator_executable \
  "+firmware=../tests/programs/custom/csr_mscratch32/csr_mscratch32.hex" \
  | tee simulation_results/csr_mscratch32/0/test_program/csr_mscratch32.log
scopesdump:
SCOPE 0x7e9b6d20ea00: TOP.tb_top_verilator
SCOPE 0x7e9b6d20ea40: TOP.tb_top_verilator.cv32e40p.tb_wrapper.i.ram.i
SCOPE 0x7e9b6d20ea80: TOP.tb_top_verilator.cv32e40p.tb_wrapper.i.ram.i.dp.ram.i
DPI-EXPORT 0x429b00: read_byte
DPI-EXPORT 0x429b00: write_byte
[tb_top_verilator] finished dumping memory
csr_mscratch32: verify full 32-bit RW of mscratch (CSR 0x340)
csr_mscratch32: PASS
TOP.tb_top_verilator @ 48460: EXIT SUCCESS
- /home/dry/projects/riscv/proc_cert/core-v-verif/cv32e40p/tb/core/tb_top_verilator.sv:83: Verilog $finish
(../vcs13) \
```

See comments in C file – csr_mscratch32.c

Submission summary

1. All specific version of tools used are noted under Environment setup
2. friction point(s):
 - a. Setup: the core-v-verif did not build out-the-box per quick start guide, against master branch (see above) – easy fix
 - b. Issue with python version – noted above
 - c. GtkWave: interface might be somewhat limited, I cannot increase/zoom on waves to increase their magnitude, and make overlaid values on graph larger
3. Screenshots are all provided above.