OpenHW Group

Proven Processor IP

Program Management Update for CORE-V Verification

Mike Thompson

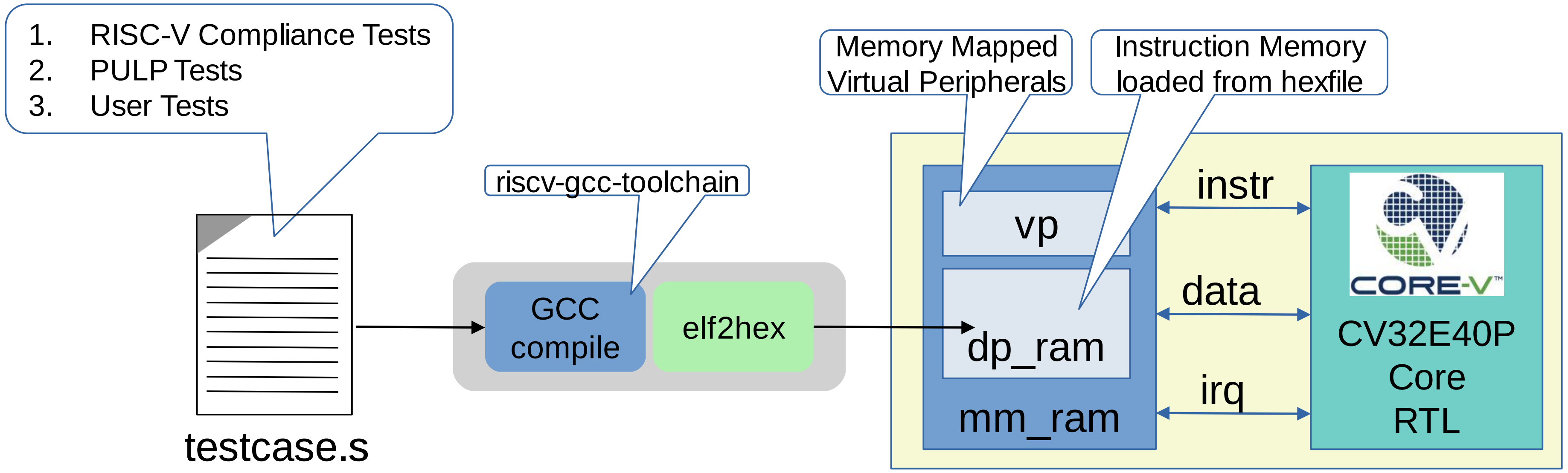
mike@openhwgroup.org

18 February 2020

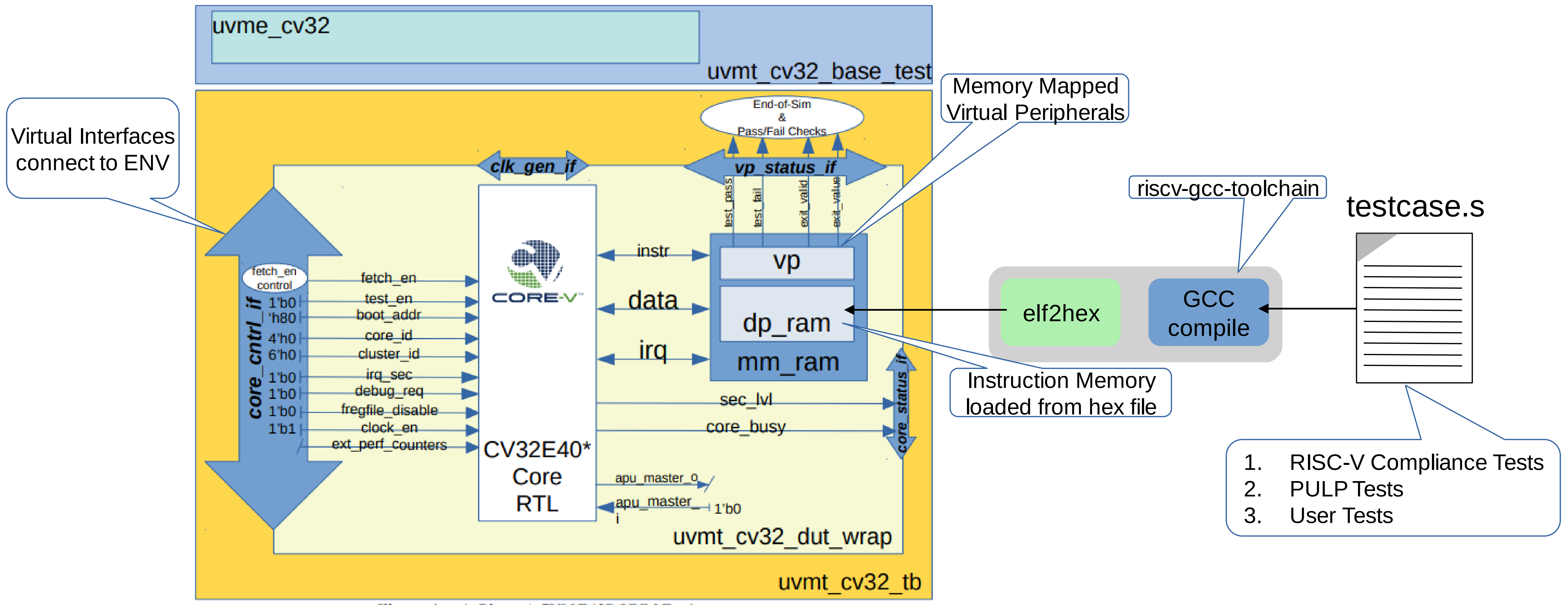
Objectives

- Quick review and update of CORE-V verification (CV32E40P).
- Communicate status:
  - CV32E40P RTL has been moved from PULP to OpenHW
  - Vplans are (almost) as complete as they can be (waiting for User Manual)
  - “Core” testbench and UVM (uvmt-cv32) verification environment in place (gated by #208)
  - Workflow, including task management and continuous integration in place
- Obtain a staffing update:
  - I need to understand each Member Company’s contribution in terms of staffing:
    - How many verification engineers
    - Contribution level (days per week)
    - Skills
  - Is everyone on GitHub and MatterMost?

"CORE" Testbench



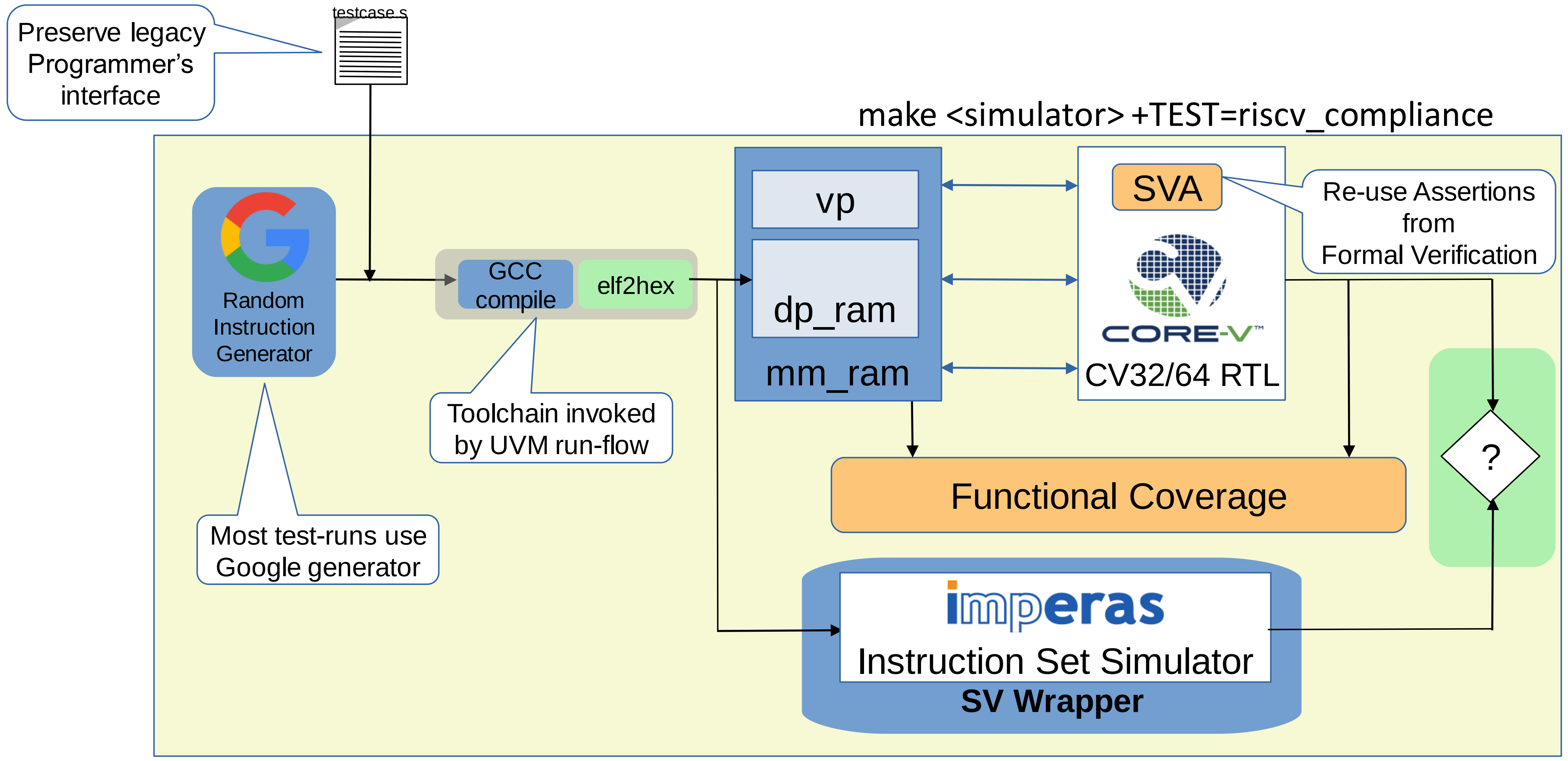
"UVMT_CV32" Verification Environment



Status of CORE and UVMT_CV32 as of 2020-02-18

- CORE testbench:
  - Integrated with CV32E40P
  - Compile is gated by GitHub Issue #208 (https://github.com/openhwgroup/cv32e40p/issues/208)
  - With work-around applied CORE TB + CV32E40P RTL can run all inherited RISCY testcases (two known failures)
- UVMT_CV32 environment:
  - Integrated with CV32E40P
  - With work-around for Issue #208 applied able to compile and run
    - "Smoke-test" : compile, reset, a few clocks, check-for-X and terminate;
    - All inherited RISCY testcases

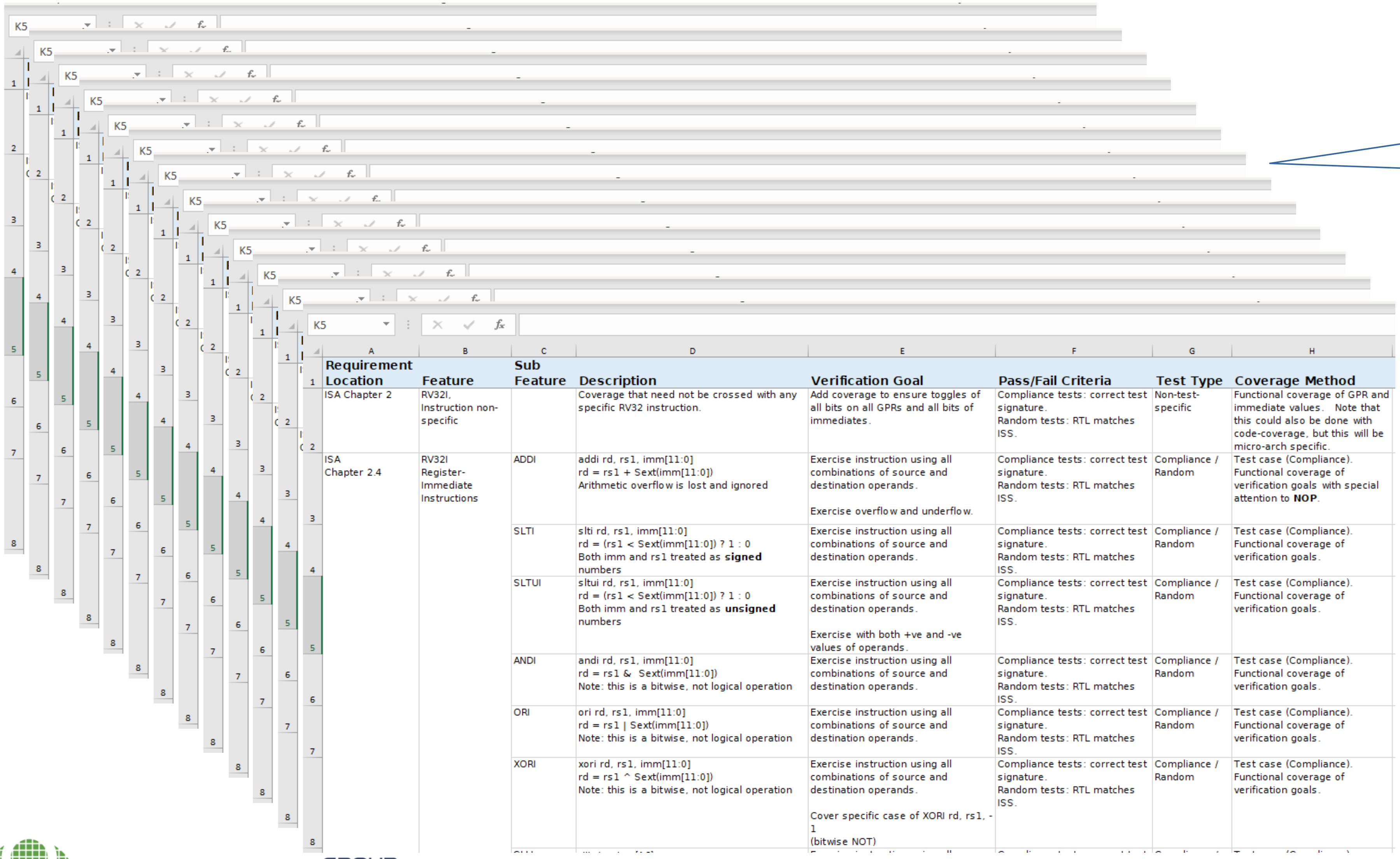
Planned UVM Environment for CV32


Preserve legacy Programmer's interface
testcase.s
make <simulator> +TEST=riscv_compliance
Random Instruction Generator
Most test-runs use Google generator
GCC compile
elf2hex
Toolchain invoked by UVM run-flow
vp
dp_ram
mm_ram
SVA
CV32/64 RTL
Re-use Assertions from Formal Verification
Functional Coverage
?
imperas
Instruction Set Simulator
SV Wrapper

Future Plans

- Should we continue to maintain the CORE testbench?
  - Pros:
    - Fast, simple testbench
    - Works with Verilator
  - Cons:
    - Additional overhead
    - Cannot achieve full coverage
- UVMT_CV32 environment:
  - Envisioned as the primary verification environment for CV32E40P and CV32E40
  - Re-use opportunities for CV64A
- CV32E40* TILE environment:
  - Highly desirable.
  - Gated by specification of the CV32E40* tile and engineering staffing.

Workflow: Verification Plans

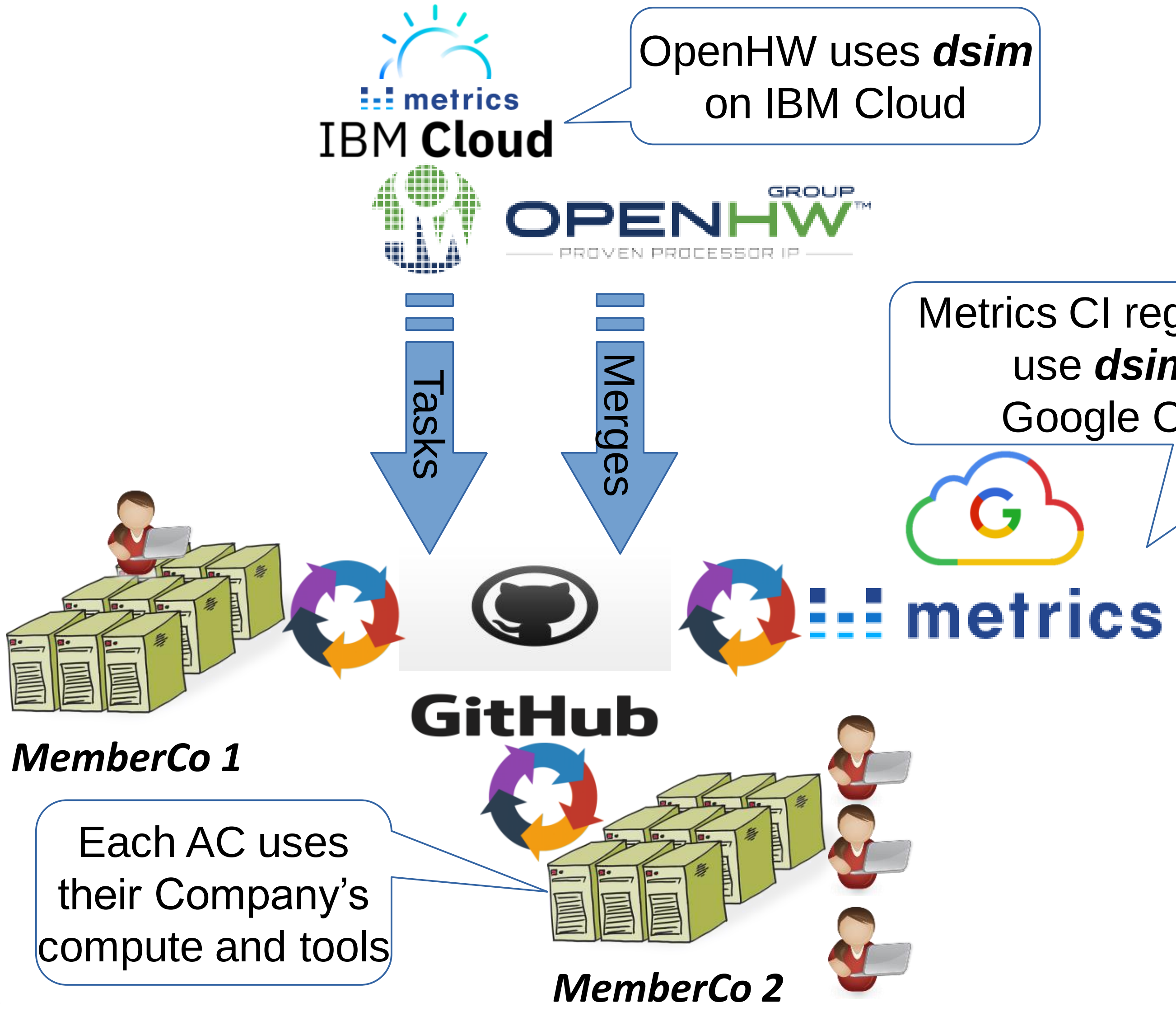


Requirement Location	Feature	Sub Feature	Description	Verification Goal	Pass/Fail Criteria	Test Type	Coverage Method
ISA Chapter 2	RV32I, Instruction non-specific		Coverage that need not be crossed with any specific RV32 instruction.	Add coverage to ensure toggles of all bits on all GPRs and all bits of immediates.	Compliance tests: correct test signature. Random tests: RTL matches ISS.	Non-test-specific	Functional coverage of GPR and immediate values. Note that this could also be done with code-coverage, but this will be micro-arch specific.
ISA Chapter 2.4	RV32I Register-Immediate Instructions	ADDI	addi rd, rs1, imm[11:0] rd = rs1 + Sext(imm[11:0]) Arithmetic overflow is lost and ignored	Exercise instruction using all combinations of source and destination operands. Exercise overflow and underflow.	Compliance tests: correct test signature. Random tests: RTL matches ISS.	Compliance / Random	Test case (Compliance). Functional coverage of verification goals with special attention to **NOP**.
		SLTI	slti rd, rs1, imm[11:0] rd = (rs1 < Sext(imm[11:0]) ? 1 : 0 Both imm and rs1 treated as **signed** numbers	Exercise instruction using all combinations of source and destination operands.	Compliance tests: correct test signature. Random tests: RTL matches ISS.	Compliance / Random	Test case (Compliance). Functional coverage of verification goals.
		SLTUI	sltui rd, rs1, imm[11:0] rd = (rs1 < Sext(imm[11:0]) ? 1 : 0 Both imm and rs1 treated as **unsigned** numbers	Exercise instruction using all combinations of source and destination operands. Exercise with both +ve and -ve values of operands.	Compliance tests: correct test signature. Random tests: RTL matches ISS.	Compliance / Random	Test case (Compliance). Functional coverage of verification goals.
		ANDI	andi rd, rs1, imm[11:0] rd = rs1 & Sext(imm[11:0]) Note: this is a bitwise, not logical operation	Exercise instruction using all combinations of source and destination operands.	Compliance tests: correct test signature. Random tests: RTL matches ISS.	Compliance / Random	Test case (Compliance). Functional coverage of verification goals.
		ORI	ori rd, rs1, imm[11:0] rd = rs1 | Sext(imm[11:0]) Note: this is a bitwise, not logical operation	Exercise instruction using all combinations of source and destination operands.	Compliance tests: correct test signature. Random tests: RTL matches ISS.	Compliance / Random	Test case (Compliance). Functional coverage of verification goals.
		XORI	xori rd, rs1, imm[11:0] rd = rs1 ^ Sext(imm[11:0]) Note: this is a bitwise, not logical operation	Exercise instruction using all combinations of source and destination operands. Cover specific case of XORI rd, rs1, -1 (bitwise NOT)	Compliance tests: correct test signature. Random tests: RTL matches ISS.	Compliance / Random	Test case (Compliance). Functional coverage of verification goals.

Hierarchically organized by Feature.

One Spreadsheet per Feature

Workflow: Task Assignment, Execution and CI

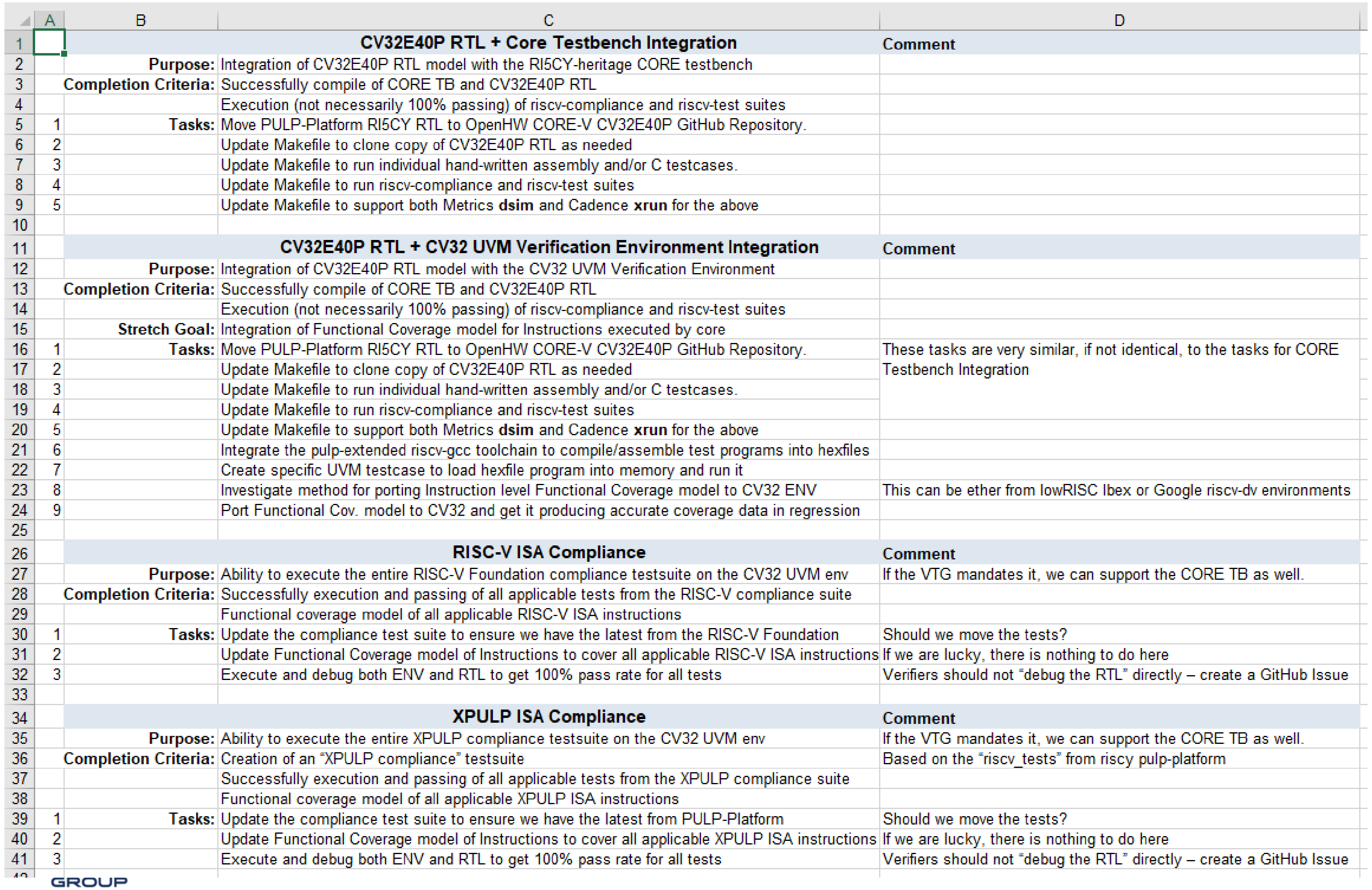


OpenHW uses ***dsim*** on IBM Cloud

Metrics CI regressions use ***dsim*** on Google Cloud

Each AC uses their Company's compute and tools

Workflow: Tasks – we are <u>not</u> using spreadsheets



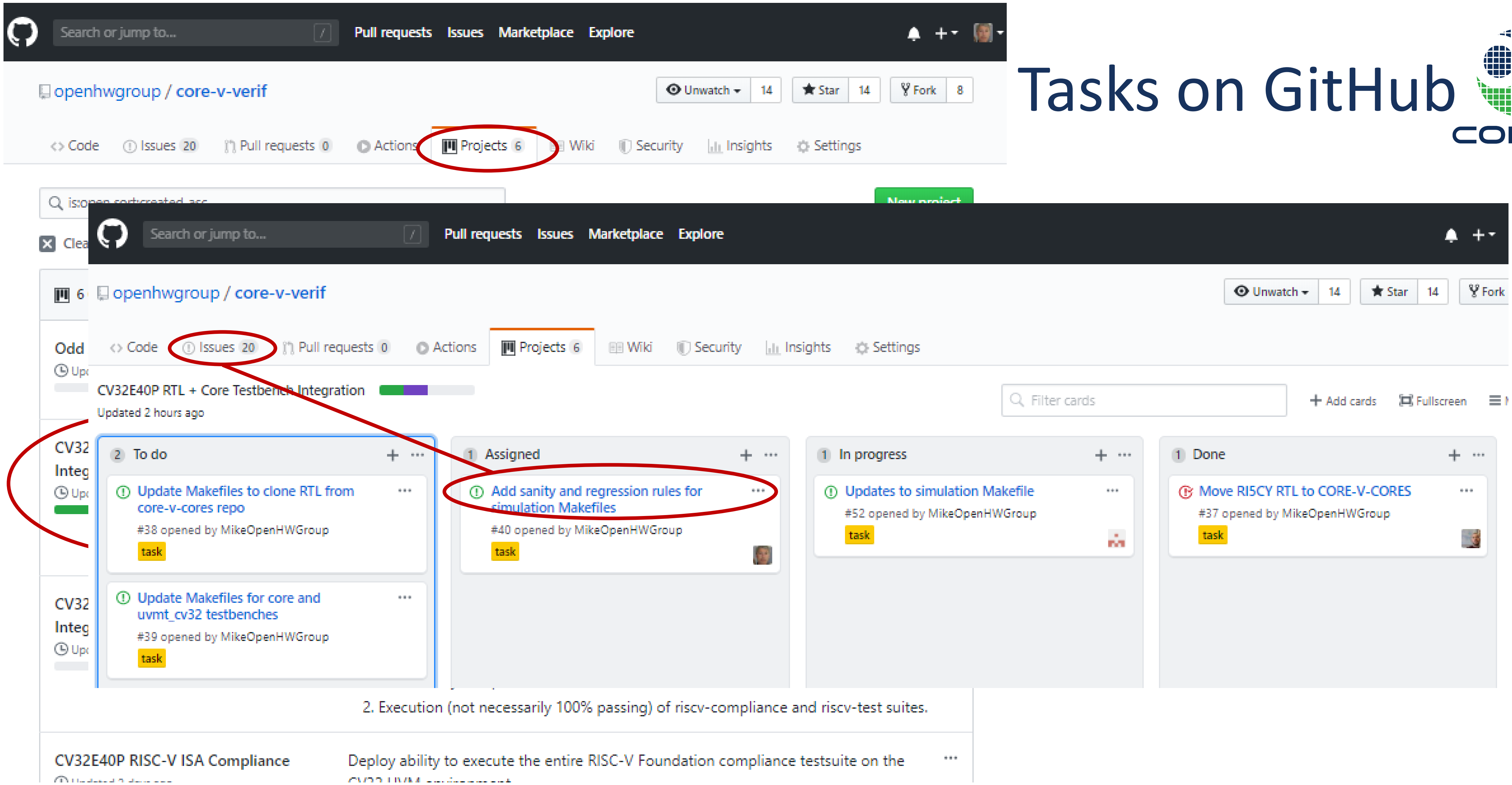
	A	B	C	D
1			**CV32E40P RTL + Core Testbench Integration**	**Comment**
2		**Purpose:**	Integration of CV32E40P RTL model with the RI5CY-heritage CORE testbench	
3		**Completion Criteria:**	Successfully compile of CORE TB and CV32E40P RTL	
4			Execution (not necessarily 100% passing) of riscv-compliance and riscv-test suites	
5	1	**Tasks:**	Move PULP-Platform RI5CY RTL to OpenHW CORE-V CV32E40P GitHub Repository.	
6	2		Update Makefile to clone copy of CV32E40P RTL as needed	
7	3		Update Makefile to run individual hand-written assembly and/or C testcases.	
8	4		Update Makefile to run riscv-compliance and riscv-test suites	
9	5		Update Makefile to support both Metrics **dsim** and Cadence **xrun** for the above	
10				
11			**CV32E40P RTL + CV32 UVM Verification Environment Integration**	**Comment**
12		**Purpose:**	Integration of CV32E40P RTL model with the CV32 UVM Verification Environment	
13		**Completion Criteria:**	Successfully compile of CORE TB and CV32E40P RTL	
14			Execution (not necessarily 100% passing) of riscv-compliance and riscv-test suites	
15		**Stretch Goal:**	Integration of Functional Coverage model for Instructions executed by core	
16	1	**Tasks:**	Move PULP-Platform RI5CY RTL to OpenHW CORE-V CV32E40P GitHub Repository.	These tasks are very similar, if not identical, to the tasks for CORE
17	2		Update Makefile to clone copy of CV32E40P RTL as needed	Testbench Integration
18	3		Update Makefile to run individual hand-written assembly and/or C testcases.	
19	4		Update Makefile to run riscv-compliance and riscv-test suites	
20	5		Update Makefile to support both Metrics **dsim** and Cadence **xrun** for the above	
21	6		Integrate the pulp-extended riscv-gcc toolchain to compile/assemble test programs into hexfiles	
22	7		Create specific UVM testcase to load hexfile program into memory and run it	
23	8		Investigate method for porting Instruction level Functional Coverage model to CV32 ENV	This can be ether from lowRISC Ibex or Google riscv-dv environments
24	9		Port Functional Cov. model to CV32 and get it producing accurate coverage data in regression	
25				
26			**RISC-V ISA Compliance**	**Comment**
27		**Purpose:**	Ability to execute the entire RISC-V Foundation compliance testsuite on the CV32 UVM env	If the VTG mandates it, we can support the CORE TB as well.
28		**Completion Criteria:**	Successfully execution and passing of all applicable tests from the RISC-V compliance suite	
29			Functional coverage model of all applicable RISC-V ISA instructions	
30	1	**Tasks:**	Update the compliance test suite to ensure we have the latest from the RISC-V Foundation	Should we move the tests?
31	2		Update Functional Coverage model of Instructions to cover all applicable RISC-V ISA instructions	If we are lucky, there is nothing to do here
32	3		Execute and debug both ENV and RTL to get 100% pass rate for all tests	Verifiers should not "debug the RTL" directly – create a GitHub Issue
33				
34			**XPULP ISA Compliance**	**Comment**
35		**Purpose:**	Ability to execute the entire XPULP compliance testsuite on the CV32 UVM env	If the VTG mandates it, we can support the CORE TB as well.
36		**Completion Criteria:**	Creation of an "XPULP compliance" testsuite	Based on the "riscv_tests" from riscy pulp-platform
37			Successfully execution and passing of all applicable tests from the XPULP compliance suite	
38			Functional coverage model of all applicable XPULP ISA instructions	
39	1	**Tasks:**	Update the compliance test suite to ensure we have the latest from PULP-Platform	Should we move the tests?
40	2		Update Functional Coverage model of Instructions to cover all applicable XPULP ISA instructions	If we are lucky, there is nothing to do here
41	3		Execute and debug both ENV and RTL to get 100% pass rate for all tests	Verifiers should not "debug the RTL" directly – create a GitHub Issue

Workflow: Tasks – we are using GitHub

- Individual work assignments will be managed as Issues.
- Tasks will be grouped into per-repository Projects.
  - **CORE-V-DOCS Projects** (https://github.com/openhwgroup/core-v-docs/projects)
    - CV32E40P Verification Planning
  - **CORE-V-VERIF Projects** (https://github.com/openhwgroup/core-v-verif/projects)
    - CV32E40P RTL + Core Testbench Integration
    - CV32E40P RTL + UVM Environment Integration
    - CV32E40P RISC-V ISA Compliance
    - CV32E40P XPULP ISA Compliance
    - CV32E40P Formal Verification
    - Odd Jobs
- New projects will be created as the program progresses.
- Tracking will be via per-project Kanban boards (next slide).

Its ironic that I am known as an Agile Heretic.

Tasks on GitHub



openhwgroup / core-v-verif

is:open sort:created-asc

CV32E40P RTL + Core Testbench Integration

Updated 2 hours ago

2 To do	1 Assigned	1 In progress	1 Done
Update Makefiles to clone RTL from core-v-cores repo #38 opened by MikeOpenHWGroup task	Add sanity and regression rules for simulation Makefiles #40 opened by MikeOpenHWGroup task	Updates to simulation Makefile #52 opened by MikeOpenHWGroup task	Move RI5CY RTL to CORE-V-CORES #37 opened by MikeOpenHWGroup task
Update Makefiles for core and uvmt_cv32 testbenches #39 opened by MikeOpenHWGroup task			

2. Execution (not necessarily 100% passing) of riscv-compliance and riscv-test suites.

CV32E40P RISC-V ISA Compliance

Deploy ability to execute the entire RISC-V Foundation compliance testsuite on the

Thank You!