Compliance Task Group Call – Minutes

Thur, 15Oct2020 8am Pacific →Daylight← Time

See slide 6 for agenda

Antitrust Policy Notice

RISC-V International meetings involve participation by industry competitors, and it is the intention of RISC-V International to conduct all its activities in accordance with applicable antitrust and competition laws. It is therefore extremely important that attendees adhere to meeting agendas, and be aware of, and not participate in, any activities that are prohibited under applicable US state, federal or foreign antitrust and competition laws.

Examples of types of actions that are prohibited at RISC-V International meetings and in connection with RISC-V International activities are described in the RISC-V International Regulations Article 7 available here: https://riscv.org/regulations/

If you have questions about these matters, please contact your company counsel.

RISC-V International Code of Conduct

RISC-V is a free and open ISA enabling a new era of processor innovation through open standard collaboration. Born in academia and research, RISC-V ISA delivers a new level of free, extensible software and hardware freedom on architecture, paving the way for the next 50 years of computing design and innovation.

We are a transparent, collaborative community where all are welcomed, and all members are encouraged to participate.

We as members, contributors, and leaders pledge to make participation in our community a harassment-free experience for everyone.

https://riscv.org/risc-v-international-community-code-of-conduct/

Charter

The Compliance Task Group will

- Develop? compliance tests for RISC-V implementations, taking into account approved specifications for:
  - Architectural versions (e.g. RV32I, RV32E, RV64I, RV128I)
  - Standard Extensions (H,S,U,A,B,C,D,F,J,K,M,N,P,Q,T,V,N), Priv Mode ← change to only ratified spec as of this date
  - All spec'ed implementation options
    - (incl. MHSU modes, optional CSRs, optional CSR bits)
- Develop a method for selecting and configuring appropriate tests for a RISC-V implementation, taking into account:
  - Platform profile and Execution Environment (EE)
  - Implemented architecture, extensions, and options
- Develop a framework to apply the appropriate tests to an implementation and verify that it meets the standard
  - test result signature stored in memory will be compared to a golden model result signature

Adminstrative Pointers

- Chair – Allen Baum allen.baum@esperantotech.com
- Co-chair – Bill McSpadden bill.mcspadden@seagate.com
- TG Email tech-compliance@lists.riscv.org
  - Notetakers: please send emails to allen.baum@esperantotech.com
- Meetings -Bi-monthly at 8am Pacific time on 2nd/4th Wednesdays.
  - See https://docs.google.com/spreadsheets/d/1L15_gHl5b2ApkcHVtpZyl4s_A7sgSrNN zoom link
- Documents, calendar, roster, etc. in
  - https://lists.riscv.org/tech-compliance/ see /documents & /calendars subdirectories
  - https://riscof.readthedocs.io/en/latest/ riscof
  - https://riscv-config.readthedocs.io/en/latest/ config: YAML and WARL spec
  - https://drive.google.com/drive/folders/1DemKMAD3D0Ka1MeESRoVCJipSrwiUlEs (lifecycle in "policies/supporting docs" folder, gaps in "planning" folder, compliance specific in "compliance folder")
- Git repositories
  - https://github.com/riscv/riscv-compliance/
  - https://gitlab.com/incoresemi/riscof (riscof framework)
  - https://github.com/riscv/riscv-config/
  - https://github.com/rems-project/sail-riscv/ (Sail formal model)
- JIRA: https://jira.riscv.org/projects/CSC/issues/CSC-1?filter=allopenissues

Meeting Agenda

0. **The profile TG is looking for a co-chair from the Compliance TG. Please email me if you have any interest or want to nominate someone**

I. Updates, Status, Progress
   1. Test Format spec changes: start, end, macros, including LA, LI

II. Continued Discussion: Merging new Base ISA tests:
   1. Removing the requirement for assertions
      - Situations where they are useful vs. better coverage rules.
   2. The meaning of a “data propagation report”
   3. Responsibility for model coverage: which models must be tested prior to merge?

III. Next steps and Ongoing maintenance
   1. Migration to Framework v.2. video: https://youtu.be/VIW1or1Oubo, slides: https://lists.riscv.org/g/tech-compliance/files/Presentations/TestFormatSpec.pdf
      1. Review Pipecleaner tests:What do we need to do to exercise capabilities for Priv Mode tests
      2. What steps do we need to complete to cut over to V.2 (see slide 10)
      3. (e.g. Sail model updates, pipecleaning, N people have run it, testing all the “fixed in riscof” issues
   2. Develop SAIL maintenance plan
   3. Identify Tool providers, e.g. coverage model, test generation for new features/extensions
   4. Flesh out test development order & identify resources (e.g. Priv,FDD or F,Priv,D..., JIRA CSC-3,5
   5. Provide a reference RTL test fixture (as opposed to SW functional model). See. JIRA CSC-6
   1. Specific Compliance Policy/Process Gaps:
      1. Certifying passing architecture tests: what needs to be in the report? Where does report get sent? (e.g. vendorID/archID)
      2. Can we certify actual HW if only its core RTL has passed architecture tests?
      3. How do we enable configurable & licensed core IP

Discussion

1.1Status: nothing reported
2.1Pseudo-Inst use
Chair: Use (per architecture) fixed sized LA and LI macros (this will change test format spec slightly); Start/end macros also are changed, to initialize GPRs and save final GPR state.

Continued discussion: Removing the requirement for assertions
Imp: assertions help with automatic test generators.We automatically insert assertions that are related to the functionality of the tests. Example: vector. We run into lots of errors when we generate tests. Lots of work for setup. Put in an assertion, unrelated to the DUT. It's about debugging test, not implementation. We want to get legal instructions. They have nothing to do with signatures.
Chair: So assertions are assumptions about the test construction.
<lots of discussion about purpose and use of assertions.>
IMP: Can't see that signatures solve the debug problem. **<Note:** this is debug of test itself, not DUT**>**
Chair: What I was proposing was not the removal of assertions, but the removal of the assertion *requirement*. For RFQ, they're not necessary; no state or setup involved except GPRs
SH: Assertions help prove the quality of the tests. How do we test the tests?
Chair: Currently, we use coverage, based on coverpoints we've defined.
IMP: At Imperas we do the following 3 things, 1) we do golden model code coverage, 2) functional coverage of instructions, 3) enforce data propagates to signature.
Chair: Not sure we can subset sail model for line coverage. We did find bugs in the SAIL model
InCore: mtval and EBREAK
SH: is there a method to submit issues wrt SAIL model?
InCore: submit github issue:
IMP: Where is the issue documented?
InCore: https://github.com/riscv/riscv-isa-manual/issues/600) and https://github.com/rems-project/sail-riscv/issues70
CoChair : If assertions are used, should they be turned off for signature generation?
Chair: yes, not need when running tests, only when developing

The meaning of a "data propagation report"
Chair: Do you have a concise definition of a data propagation report?
Imp: You have to demonstrate that the thing you're trying to cover, propagates to the signature. The report shows that what you cover is seen in the signature. **<Note**: not just what is being stored but test intent, which is derived from the configuration of the test (e.g. not just FADD.S, but FADD.s with the specific dynamic rounding mode in FCSR)**>**
InCore: I think what has been proposed (from Monday meeting) satisfies this requirement for the RFQ.
(2 measures: verify that number of signature stores matches number of signature region changes, and map signature stores to the instructions that generated them.
- Branches, Store and other ops withott a destination register must be special cased
- **<Note**: this applies only to Base ISA tests that have no dependencies on anything besides reg values**>**

Responsibility for model coverage: which models must be tested prior to merge?
Chair: Tests work with SAIL & SPIKE. Do we need to see that a test works with other simulators?
CTO: Who I doing the the remedial work? (graduated TGs with no existing tests)
Who is going to drive this? Does the this TTGgroup do the remedial work or do we get help from the priv/unpriv group? We need to have this conversation with other groups; perhaps in email.
Imp: Look at charter. See what this group is responsible for. We have some that are orphaned
CTO: Exactly. We need to be clear what extensions this group is responsible for.
Imp: Split into 2 lists. One that is for tech-compliance, and then extensions for other groups.
Bristol is sitting in on this group to understand what is needed for Crypto.
SH: What does **Bristol** observe?
Bristol: Daunting. I've bootstrapped some tests to see if SAIL model matches SPIKE. For coverage, it's not terribly difficult. Instructions are very simple. Randomization is not that hard. There are levels for which we do testing. We did complete AES flow first, and then did a per instruction test. Should we then put in algorithm tests?
Chair: If you've tested a complete flow for op development, then only op testing is needed for testing implementations,
Bristol: We have an example of a purely random instruction for RNG. How to test?
Chair: Statistical measurements are the only way; difficult to do w/ SAIL model as a reference
Chair: Similar example: vector reduce; Sail & DUT must perform calculations in the same order

Chair: Are we in agreement that to merge, tests much run in SAIL, SPIKE and Qemu?
CTO: Need to run on real hardware for ratification.
Chair: But that would be ratification of the implementation.
CTO: No. Need to make sure the implementation can run architectural tests.
<<Clarification: this applies only to non- "remedial" tests.>>

(no other disagreements. Other model specific files in the repo will be move to a subrepo? And the model owners will be responsible for ensuring that their models pass merged tests.

Test merging Process:
- Submit a pull request
- Upon decision to merge (based on defined criteria, see slide 11) , authors will
- Send out a message to other model maintainers to test/file issues
- After 2 weeks without issues being filed, tsets will be merged into repo.

Decisions & Action Items

Decisions

- We will remove the requirement for assertions in tests. If appropriate to verify that tests are properly generated, they may be included, but will be disabled by default in testing implementations
- We will add our own fixed length versions of LA and LI and use only them inside tests (note that this doesn't apply to code outside tests, including trap handlers, setup and teardown code)
- Tests will be eligible for merging if they pass on Spike, QEMU, Sail models and a HW model (for non-remedial tests), and meet other requirements (see slide 11)
- After becoming eligible for merging, tests will be put into a pull request, maintainers of other models (taken from riscv.org/exchange/cores-soc and models known to have architectural test supporting files), and after (2 weeks) without issues being raised, they will be merged

Outstanding Action Items- New

Allen/Mark: discuss requirement for HW implementation before merge

Allen/Mark: indientify responsibilities and resources for "remedial" tests

Allen: discuss with repo admins and maintainers where staging area is, and if model specific files will be moving (now or forv .2 framework)

Allen: get contact info for all model maintainers and inform them of new merging process

Neel/Allen: update test format spec with macro, entry, exit changes

Old

[everybody]: comment on base ISA cover points:

https://github.com/incoresemi/riscv-compliance/tree/dev/coverage

(this is needed to complete the TG's responsibilities for the RFQ)

Imperas: make pull request for updated assertion macro

Stuart: write up coverage taxonomy

Everybody: read policy docs, send gaps in compliance (e.g. formal model support, possible mismatch between config TG and riscv-config) and priority to cto@riscv.org

Previous Action Items / Progress Update

- SH will add file regarding coverage - no progress….in progress
- Imperas / Incore: ensure headers, macros, dir structure match newest spec, assertions are not inline – waiting for assertion macro update, Imperas pull request
  - delete this if assertions are no longer a requirement?
- Chair coordinate with Riscof to determine pipecleaning exercise - to be reviewed in TG
- Chair to communicate with TSC about reorganization comments - waiting TSC feedback
- Configuration Structure TG vs. Riscv-Config: - discussions underway – see https://github.com/riscv/configuration-structure/ and new profile group. See also https://sites.google.com/a/riscv.org/risc-v-staff/home/ > information > current organization.pptx

Note: initials are company abbreviations

Architectural Test Rationale – Intent and Limits

RISC-V Architectural Tests are an evolving set of tests that are created to help ensure that SW written for a given RISC-V Profile will run on all implementations that comply with that profile.

These tests also help ensure that the implementer has both understood and implemented the specification.

The RISC-V Architectural Tests test suite is a minimal filter. Passing the tests and having the results approved by RISC-V International is a prerequisite to licensing the RISC-V trademarks in connection with the design.

Passing the RISC-V Architectural Tests does ***not*** mean that the design complies with the RISC-V Architecture. These are only a basic set of tests.

The RISC-V Architectural Tests are ***not*** a substitute for rigorous design verification; it is the responsibility of the implementer to deploy extensive testing.

To be added to the riscv/riscv-compliance/doc/ directory as “RISC-V Architectural Test Rationale”

Test Acceptance Criteria – second cut

Tests must:

- conform to current standard of test spec (macros, labels)
- run in framework
- run in SAIL and not fail any tests
- ?Demonstrate that test results propagate to signature
- generate a valid signature using SAIL (that can be saved and compared with another dut/sim)
- has a clear configuration - i.e. which ISA extension it can be used with
- have a code, data, and signature memory footprint that is small enough*
- improve coverage
- use only standard instructions ←~~are LA ,LI allowed?~~
- use only files that are part of the defined support files in the repository
- must be commented, both in header ~~and inside test cases~~

** need to define "small", "X" ← will vary by extension, base ISA expected to be <8K.* ***Tests of JAL will max at over 1MB***

Framework Requirements – first cut

The framework must:

- Use the TestFormat spec and macros described therein
  - (which must work - including assertions)
- Choose test cases according to equations that reference the YAML configuration
- Define macro variables that can be used inside tests based on the YAML configuration
- Include the compliance trap handler, & handle its (separate) signature area
- Load, initialize, and run selected tests between two selected models, extract the signatures, compare results, and write out a report file
- Exist in a riscv github repo, with a few than one maintainer.
- Be easy to get running, e.g.:
  - run under a variety of OSes with the minimum number of distro specific tools.
  - Not require sudo privileges
- Maybe: have the ability to measure and report coverage
  - Coverage specification is a separate file
  - Could be a separate app

Pull/Issue Status

Issue#	Date	submitter	title	status	comments
#04	3-Jul-18	kasanovic	Section 2.3 Target Environment	Fixed in RISCOF	
#22	24-Nov-18	brouhaha	I-MISALIGN_LDST-01 assumes misaligned data access will trap		
#40	4-Feb-19	debs-sifive	Usage of tohost/fromhost should be removed		
#45	12-Feb-19	debs-sifive	Reorganization of test suites for code maintainability		
#63	13-Aug-19	jeremybennett	Global linker script is not appropriate		
#78	26-Jan-20	bobbl	RV_COMPLIANCE_HALT must contain SWSIG		
#90	11-Feb-20	towoe	Report target execution error		
#72	26-Oct-19	vogelpi	Allow for non-word aligned `mtvec`	deferred	needs v.2
#105	22-Apr-20	jeremybennett	Non-standard assembler usage	under discussion	Simple fix
#106	22-Apr-20	jeremybennett	Use of pseudo instructions in compliance tests	under discussion	
#107	22-Apr-20	jeremybennett	Clang/LLVM doesn't support all CSRs used in compliance test suite	under discussion	
#108	22-Apr-20	bluewww	RI5CY's `compliance_io.h` fails to compile with clang	under discussion	
#109	06-May-20	Olofk	Swerv fails because parallel make	under discussion	
#115	06-jun-20	adchd	How to support on-board execution?	under discussion	
#116	06-jun-20	simon5656	loss of 64bit test infrastucture	under discussion	
#119	17-jun-20	allenjbaum	Missing RV32i/RV64i test: Fence	Test has been written	Close when test is merged
#125	15-jul-20	ShashankVM	Request to stop hosting closed source code on riscv repo	under discussion	
pull#128	29-jul-20	nmeum	grift: update for new directory structure		Who can review this?
pull#129	31-jul-20	nmeum	sail-riscv-ocaml: Disable RVC extension on all devices not using it		Who can review this?
#132	15-aug-20	davidmlw	Why not just use mepc for mret?	answered	Should be resolved
#135	04-sep-20	MikeOpenHWGroup	Request for a Tag on this Repo	assigned	

JIRA Status

Issue#	Date	submitter	title	status	comments
IT-1	27Aug/20	Allen Baum	Need to modify the description of compliance in https://riscv.org/technical/specifications/	done	
IT-4	01/Sep/20	Allen Baum	Add Jira link to TG home pages	In prog	
CSC-1	20/Aug/20	Ken Dockser	Come up with names for the tests suites that we are creating		1st step done
CSC-2	20/Aug/20	Ken Dockser	Produce concise text to explain the Architecture Tests intent and Limits		Written, needs pull req
CSC-3	20/Aug/20	Ken Dockser	Come up with an internal goal for what we wish to accomplish with the Architectural Tests		Not written
CSC-4	20/Aug/20	Ken Dockser	Develop a roadmap for all the different categories of test suites that will need to be created		Not written
CSC-5	20/Aug/20	Ken Dockser	Develop a roadmap for releases of single-instruction Architecture Tests		Not written
CSC-6	20/Aug/20	Ken Dockser	Develop a reference RTL test fixture that can stimulate and check the CPU under test		Needs more discussion