Compliance Task Group Call – Agenda

Thur, 08Oct2020 8am Pacific →Daylight← Time

See slide 6 for agenda

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,H,J,M,N,P,T,V,N), Priv Mode
    - 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

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/

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/
- 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

II. Discussion: Merging new Base ISA tests: decisions we being **<u>voting</u>** on (see slide 11)

   1. Which pseudo instructions are allowed?
      - Disallow use of pseudo instructions that might cause different length expansions depending on toolchain
      - Specifically replace LA and LI with fixed length versions (e.g. pad with noops to max length
   2. Remove the requirement for assertions
      - Usefulness is questionable if we are doing signature comparison vs. self checking tests.
   3. Remove the requirement for a specific size of test, require test bench that can adjust to the required size
      - The argument has been that IOT devices may have tiny SRAMs, so tests need to be small
      - But, base ISA JAL as a 1MB range. Are we going to fail an RTL that can execute a base ISA op if the physical implementation has limits?
      - In any case, for the Base ISA tests, limit test sizes to 1.5MB, ensuring we can test the full range of JALs

1. 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
2. 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 I

1.1Status: nothing reported

2.1Pseudo-Inst use

Chair: Getting ready to merge new tests into git repo. Issues:. Using LA and LI Pseudo -insts

IMP: Why don't you want LA?

Chair : They expand to different length sequences. Gives differing results. Want to wrap them to force them into consistent length. Done by a macro.

IMP : So we'll still use pseudo-instruction, just wrapped; Has it been a problem (yet)? We haven't one.

Chair : Has not been a problem yet, but expect to see issues with signature comparison

Chair : <further explanation> Tests have to be completely address independent

IMP : Linker allows you to load test into the same place.

Chair : <examples: if test prologs are different, Las can become different sizes, and relative addresses will be different>

QC: We'll use macros instead of pseudo- insts?

Chair : We will use macros that wrap pseudo- insts to cause them to be fixed length

2.2 Remove requirement for assertions

Chair : No tests need to have assertions

QC: Don't we need this in order to help debug the failure?

SAIL: We're supplying a go/no-go test. We're not interested in debugging the test. Nor is it useful when the user is debugging his implementation failure.

CTO: Assertions useful at tests beginning and end. But concerned about it taking extra time.

Chair : The way the assertion have been used as a self-checking mechanism.

IMP : Out test generator puts in assertions to help debug. Assertions can be turned off for final run, but are essential for debugging a test.

Chair : Can you give me an example?

IMP : < example described > < hate to use vector >The test generator has to know what it's doing.

IMP : It helps improve the quality of the test.

Incore: How do you get the assertion values into the assertion? <test fgenerator has a model>

SAIL: We need to differentiate between assertions in test set up and assertions within the test to help debug and assertions as part of the test (which would become a self-checking test).

IMP: Krste said we don't want self-checking test.

Chair : I'm having trouble seeing the difference between this and self test Is test set-up stuff stored in signature?

SAIL : Why don't we just have an environment (set up) test? If good, move on.

IMP : Set up may be different on a test-by-test basis.

IMP2: We added assertions when we were developing these tests for debugging. It was very hard to debug based on signature failures. Far simpler to debug at point of failure. The tests so far have been trivial. More complicated tests need better debugability. They don't need to be there for production. Only for test development.

Chair : This is all about test acceptance. Test that are so complex that assertions are needed are more DV tests rather than compliance tests.

IMP2 : I don't agree that they are DV tests. DV tests exploit micro-arch Example: remainder tests. Negative remainder.

Chair : We have coverage for that.

Chair : Testing instruction sequencing. Eg - read after write

Chair : Simple tests with lots of setup. Eg - LD/ST, user/supervisor, PMP…

SAIL : Does it matter to this group if assertions are used if we have good coverage?

IMP : 3 things needed for development. 1) run with assertions enabled 2) functional coverage 3) results propagate into the signature. If you do well with # 2 and #3, then you don't need the assertions. I'll buy that.

IMP2 : I agree with that. It's possible to have good coverage but then have rubbish in the signature.

Discussion II

2.3. Size of test

Chair : if we don't have memory footprint, then we can't run tests.
We need to require that test benches can support what the tests require

3 Test Merging

Chair : What do we need to do to merge these tests?
IMP2 : Need to go through the flow. And make sure that they work for all simulators.
Incore: Simulators: Spike & Sail
Chair : What about OvpSim?
Incore : We could add that.
QC : What about opensource RISCV RTL?
Chair : The contract requires only SAIL on Spike. It doesn't say we run on OvpSim. Or on RTL.
QC: We could extend contract.

<<diversion into requirements to accept tests>>

IMP: RFQ requires data propagation into signature.
We need to have "the report" that shows data propagation into signature. Where is it?
Chair : I don't know how that would be done. And not required by contract
<discussion of contractual obligations>
<major sticking point: how to prove data propagation into signature report>
Chair : Correct by construction using macros
IMP : Are we saying that we don't care about results?
Chair : If you can show that you have stored something at every signature offset,
does that show that there is data propagation?
Incore: Overwriting 0xdeadbeef? Does that show propagation? Count number of deadbeefs?
IMP : Confused. Assertions and data propagation is good practice. We're not doing these.
Chair : Disagree. We have proposed a simple data propagation mechanism.
IMP : Counting will not be good enough. Counter example: we couldn' t see register changes. It didn't propagate into signature. Because of bad choice of data values.
IMP2: It was in the presence of a fault
Chair: because cover points were terrible…

.

Chair : <back to contract> We want to accept the tests. This is contractually stated in contract.
QC: We don't want to make them public yet. Can accept them but not merge them.
Chair : This is not what the contract says. Acceptance is signaled by merging
Incore : We are concerned about "not merging" that prevents us from fulfilling future milestones.
QC: if RV and Incore agree, we can accept work without merging and proceed

Decisions & Action Items

Decisions

- We will use LA and LI, provided they are embedded in macros that guarantee a fixed size expansion

- Assertions are permitted but not required in tests. They are one way to ensure that tests are testing the intent of a test

- There will be no specific size limit on tests; test benches will be required to support what a test needs.

- The requirement to merge tests to mark the acceptance of the RFQ tests will be waived for this milestone, and a letter (email) will be sent instead. This still needs to be done for final acceptance in December.

Outstanding Action Items- New

[Chair]: Send email to confirming that the Statement of Work by Incore has been completed for the first <done>

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
- 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