

# Compliance Task Group Call – Minutes

Thur, 29Oct2020 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

# Administrative Pointers

- Chair – Allen Baum [allen.baum@esperantotech.com](mailto:allen.baum@esperantotech.com)
- Co-chair – Bill McSpadden [bill.mcspadden@seagate.com](mailto:bill.mcspadden@seagate.com)
- TG Email [tech-compliance@lists.riscv.org](mailto:tech-compliance@lists.riscv.org)
  - Notetakers: please send emails to [allen.baum@esperantotech.com](mailto:allen.baum@esperantotech.com)
- Meetings -Bi-monthly at 8am Pacific time on 2<sup>nd</sup>/4<sup>th</sup> Wednesdays.
  - See [https://docs.google.com/spreadsheets/d/1L15\\_gHI5b2ApkcHVtpZyl4s\\_A7sgSrNN](https://docs.google.com/spreadsheets/d/1L15_gHI5b2ApkcHVtpZyl4s_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/1DemKMAD3D0Ka1MeESRoVCJipSrwiUIEs>  
(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/rem-s-project/sail-riscv/> (Sail formal model)
- JIRA: <https://jira.riscv.org/projects/CSC/issues/CSC-1?filter=allopenissues>

# Meeting Agenda

- 0. **Looking for more admins, maintainers for riscv-compliance git repo !!**
- I. Updates, Status, Progress
- II. Continued Discussion:
  - 1. **Merging new Base ISA tests Experience running new tests on custom model**
  - 2. Data propagation report
- 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>
    - What steps do we need to complete to cut over to V.2 (see slide 12)
      - (e.g. ~~Sail model updates~~, pipecleaning, N people have run it, testing all the “fixed in riscov” issues
      - Review Pipecleaner tests: What do we need to do to exercise capabilities for Priv Mode tests
  - 2. Moving all model proposal to remove all model specific support code out of riscv-compliance repo
  - 3. Specific Compliance Policy/Process Gaps:
    - 1. Develop SAIL maintenance plan
    - 2. Certifying passing architecture tests: what needs to be in the report? Where does report get sent? (e.g. vendorID/archID)
    - 3. Can we certify actual HW if only its core RTL has passed architecture tests?
    - 4. How do we enable configurable & licensed core IP
    - 5. Identify Tool providers, e.g. coverage model, test generation for new features/extensions
    - 6. Flesh out test development order & identify resources (e.g. Priv,FDD or F,Priv,D..., JIRA CSC-3,5
    - 7. Provide a reference RTL test fixture (as opposed to SW functional model). See. JIRA CSC-6

# Discussion

**Inspire** has fixed XORI test. How to merge? <make a pull request, **now done and merged**>

**Co-chair:** has run tests on one of his core implementations for RFQ test - (old tests run on all cores)

1. There are 4 repos: riscv-tests (old AW tests, not compliance, official riscv-compliance, Incore riscv-compliance branch, and Incore riscv-compliance/dev branch. **Only the last should be tested**
2. Different tests use different toolchains, e.g. Centos6, Centos7 for bit manip. -- ditto for LLVM. RFQ dev branch: shell command "command" is only in Centos7, not centos6 <**needs fix**>?
3. Need a git methodology to ensure that customer-specific "risc-target/<device>" environment files are not propagated to the main repo.  
*This is a general issue not specific to the new tests, but applies equally to the current tests. Suggestion was to use git submodules, but that has drawbacks (e.g. synching). TBD*
4. Debug of signature failures is painful. Mapping mismatch to actual source code takes some work. All tests have at least one signature mismatch on one of our cores. Having examined one of the failing tests (slli-01), it appears the signature compare failure occurs at the end of the test where register file state is dumped to the signature (all 32 registers). In looking at the objdump of the source file, I see that the 'auipc' instruction is used. Since I changed the .ld file to relocate the start address, any reference in the signature derived from a .text address (as in the auipc instruction) will yield differences to the reference signature. (This is exactly what Simon pointed out in his email.)  
*The cause is a late addition that dumped register state into the signature outside the test. This was added in order to have a general noop test (but I didn't think it through...)*  
**FIXED** - That capability has been disabled as a default
5. Can't comment yet on data propagation to signature. There is a method of registers/memory initialization to ensure that they get overwritten. Macros seem to be showing that the signature increments and new values get added.
6. Number of generated signatures match the reference output file. That's good. All tests pass a great many number of signatures.
7. Seagate's environment does signature checking in the SV/UVM check\_phase(). This means checking is done at the end of the test, not on a signature by signature basis  
*This is specific to that environment, and is not generally applicable. Partly this is business/usage model difference between cores designed & used in house, vs designed & sold.*  
<< **FIXED:** scripting changes have been made to enable separation of compiling, simulating and verifying for other use cases>

8. Having problems getting all tests compiled. So far only the I extension gets compiled correctly. This is probably operator error. Make var, RISC\_V\_DEVICE, is not a list. However, the description from Makefile.include reads:  
# set the RISC\_V\_DEVICE environment to extension you want to compile, simulate and verify.  
# Leave this blank if you want to iterate through all the supported extensions of the target  
export RISC\_V\_DEVICE ?=  
which implies that RISC\_V\_DEVICE is a list.  
*Operator/doc error: Needs more commentary to give examples and commentary?*
9. Ran into the max immediate offset for a JAL instruction. Test will not link/load as is. Problem is with the location of the signature area and the .text section overlapping.  
*Ironically, this is because our coverage has improved – we are checking max jump offset values. This is a problem that we will need to solve for the future, not just here. The alternatives are:*
  - a. Live with it
  - b. Break test up into different buckets, alter make script to run only those below a certain size
  - c. Add a variable that can configure tests to compile only tests below specific sizes.*In each of these cases, the test report needs to be clear that certain tests were deliberately excluded, and that an exception is being asked for.*
10. Compiled stuff goes into <root>/work - but only one work dir, so can't do multiple architectures without replication. *This is not specific to the new tests. Can be set with cmd line arg.*
11. I have not done any work on the RISCOF repo. It appears that Ben Marshall of Bristol University has started looking at RISCOF (for the cryptography working group).

---

Q: Has anyone run on rv64?

*(Not yet, evidently. So far only Imp and Co-chair (outside of Incore) have even tried*

**Inspire:** Text and code is not split out (explanation of where it goes: where it is in the binary, and where it expected to be in memory

*An example linker script in following slide*

*For further discussion: We will be removing riscv-target/<device> directories except for spike, and sail; they will be maintained by the model owners.*

*Should we also remove spike and sail and move the support into their repos, removing directory completely?*

# Linker script to separate data from text sections

Below is a snippet from this file with explanation as to what it does.

```
MEMORY {
sram (rwx) : ORIGIN = 0x0EE90000, LENGTH = 0x4000
maskrom_mem (rx) : ORIGIN = 0x2EFC0000, LENGTH = 0x20000. }

SECTIONS {
.text.init ALIGN((ORIGIN(maskrom_mem) + 0x0), 64) :
    AT(ALIGN((ORIGIN(maskrom_mem) + 0x0), 64)) {
    PROVIDE(_ftext =.);
    *(.text.init)
    PROVIDE(_etext =.); }

.text ALIGN((ADDR(.text.init) + SIZEOF(.text.init)), 64) :
    AT( ALIGN((LOADADDR(.text.init) + SIZEOF(.text.init)), 64)) {
    *(.text)
    }

.tohost ALIGN((ORIGIN(sram)), 64) :
    AT( ALIGN((LOADADDR(.text) + SIZEOF(.text)), 64)) {
    *(.tohost)
    }

.data ALIGN((ADDR(.tohost) + SIZEOF(.tohost)), 64) :
    AT(ALIGN((LOADADDR(.tohost) + SIZEOF(.tohost)), 64)) {
    *(.data)
    }

PROVIDE(_data = ADDR(.data));
PROVIDE(_data_lma = LOADADDR(.data));
PROVIDE(_edata = ADDR(.data) + SIZEOF(.data));
```

Each section has is defined either to start at a region in the MEMORY area via the ORIGIN(). This is the area the code / data expects to be at runtime. So for the .text.init section this address is 0x2EFC0000 and for the .tohost section it is 0x0EE90000.

The section that dictates how the elf and binary derived from the eld is packaged is the AT() keyword. This tells the linker where the load address for a section is. For the elf file this would be where the elf loader would load this section to. It also dictates where the section is physically in the elf file. This is used to build up the ROM image.

In the above .text.init is the first section and expects to be loaded at 0x2EFC0000. The next section .text follows this in the elf image at 0x2EFC0000 + sizeof(.text.init). In the same way each of the next sections follows the previous section.

In the above you can see the next section .tohost has a different runtime 0x0EE90000 address than its load address of LOADADDR(.text) + SIZEOF(.text).

The linker script file is used to setup a runtime address and a load address for each section. Sometimes, in the case of .text and .text.init these would be the same address. In other cases .tohost, .data and .data.strings they are different.

The linker script file can also export variables back into the code to provide the starting and ending address of a given section. Above you see the .data section is start and end execute addresses and its load address are exported as variables \_data, \_edata and \_data\_lma.

In the code you can then use these addresses to copy from the load address to the execute address.

```
la t0, _data_lma; \
la t1, _data; \
la t2, _edata; \
1: \
lw t3, 0(t0); \
sw t3, 0(t1); \
addi t0, t0, 4; \
addi t1, t1, 4; \
bltu t1, t2, 1b;
```

You can specify that all sections get loaded and executed from the same address space using this framework as well. In this case all the load addresses would equal the execute addresses. You would not need the copy code in this case.



# Decisions & Action Items

## Decisions

### Outstanding Action Item

**Chair** : get contact info for all model maintainers and inform them removal of model specific support code from the rep <not done>

#### NEW

**Chair/Incore**: update test format spec with macro, entry, exit changes <done>

**Co-Chair**: submit ticket issue RE: doc updates for splitting compile/run <done>

**QC**: extract bits of FAQ as guidelines for test writing <?>

**Incore**: add text to current github Readme pointing to riscof <done>

**Incore**: Try YAML version of SAIL to see if it works <not 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

SH: 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](mailto:cto@riscv.org)

# 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
- Report that test results propagate to signature
- generate a valid signature using SAIL (that can be saved & compared with another DUT/sim)
- has a clear configuration - i.e. which ISA extension it can be used with
- improves coverage
- use only standard instructions (fixed size per architecture LI, LA allowed)
- use only files that are part of the defined support files in the repository
- must be commented test\_case header

# 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		HW misalign support not configurable
#137	12-oct-20	Ode-jtz	The signature.outputs aren't identical w /references.		
#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	
#137	27-Oct-20	MarcKarasek	xori on 64 bit compiler needs neg number for 32 bit tests	Fixed	Should be resolved

# JIRA Status

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