Compliance Task Group Call – Minutes

Thur, 19Nov2020 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 ← will be changing to new git repo
  - https://drive.google.com/drive/folders/1DemKMAD3D0Ka1MeESRoVCJipSrwiUlEs (lifecycle in "policies/supporting docs" folder, gaps in "planning" folder, compliance specific in "compliance folder")
- Git repositories ←docs riscv → tools
  - https://github.com/riscv/riscv-compliance/tree/master/docs/ https://github.com/riscv/riscv-compliance/
  - https://riscof.readthedocs.io/en/latest/index.html riscof https://gitlab.com/incoresemi/riscof/
  - https://riscv-isac.readthedocs.io/ ISA coverage https://gitlab.com/incoresemi/riscv-compliance/riscv_isac
  - https://riscv-ctg.readthedocs.io/ Compat. Test Gen. https://gitlab.com/incoresemi/riscv-compliance/riscv_ctg
  - https://github.com/riscv/riscv-config/tree/master/docs YAML, WARL config https://github.com/riscv/riscv-config/
  - https://github.com/rems-project/sail-riscv/tree/master/doc Sail formal model https://github.com/rems-project/sail-riscv
- JIRA: https://jira.riscv.org/projects/CSC/issues/CSC-1?filter=allopenissues
- Sail annotated ISA spec: in https://github.com/rems-project/riscv-isa-manual/blob/sail/
  - README.SAIL how to annotate release/riscv-spec-sail-draft.pdf annotated unpriv spec
  - release/riscv-spec-sail-draft.pdf annotated source release/riscv-privileged-sail-draft.pdf annotated priv spec

Meeting Agenda

0. **Looking for more admins, maintainers for riscv-compliance git repo !!**

I. Updates, Status, Progress
   I. Last meetings for 2020: Dec 3, Dec 17 (both off-cycle), then Jan 14

II. Continued Discussion:
   1. RFQ tests – experience and changes
   2. Merging new Base ISA RFQ tests: remaining issues, notifying users

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 13)
        - (e.g. ~~Sail model updates,~~ pipecleaning, N people have run it, testing all the "fixed in riscof" 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

Incore RFQ Progress: Updates from last week

- new coverpoints added for mul/div. SLT[U], Bxx (roughly doubled # of coverpoints
- Inspire PR merged (adds .section.text.init before rvtest_entry_point)
  Inspire - it worked for me, code got put in the right place.
- renamed header fields to avoid word "compliance"
- mtval relocation fixed (special case when a test stores directly into signature region)
- monolithic coverage file split into separate files to be more manageable
- Migration guide written
  -- aliases created, generate warning if old name used
  -- goal: moving from v0.1 to v0.2 of the framework with minimal changes

Inspire: Recommendation: do 1 of 2 things:
1) branch master and change name(s) from compliance to arch OR…..
2) create a new git repository with proper name, "architecture"

Chair - #2 out of scope for this stage
Inspire - if we do it now, at this inflection point, it will be named correctly.
need to tag master; tag should reflect what's current & what's pulled in
there have been 18-20 commits since Incore did the dev branch pull

QC - are there still bug fixes being done on the master?
Inspire - there have been 18 commits, i don't know what they are
Chair - probably model support files & moving riscvOVPsim
Inspire - avoid "device tree" term; has different meaning in most contexts
Is there another set of eyes doing the port?
CoChair - yes, I'm doing the port as well
Incore - I've haven't checked example target; will do so
Incore - RVMODEL_IO_CHECK macro only used in ovpsim to check x0=0.
don't know how/why it's used.
Inspire - IO_CHECK is not used in old version of tests. Should be removed.

AR: Chair: add to minutes "Does anybody use RVMODEL_IO_CHECK?"

Targets migrated/ported

Codasip, SifFive formal are proprietary, no access, can't be tested
Grift ported, continues to fail because config doesn't match,
Chair – owner contacted
Ibex, r5cy, rocket: migrated, can't test because commercial simulators or undocumented build steps
riscOVPsim, Spike, Sail(C+Ocaml) ported, migration versions available
OpenHW: model is unavailable, can't be tests
<**AR Chair**: find model owners, remove support files if can't be resolved>

Inspire,CoChair – Targets other than Sail, Spike should be removed. adds confusion to repo
Inspire - what about QEMU? any support?
Chair – unaware of any
Inspire – will add QEMU support; have already done the work. <AR>
< QEMU support, use cases: not for arch. testing - intent is to ensure that QEMU itself passes the arch tests so it can be used for its real purpose, to test that SW using new features and extensions work as intended>

Inspire - Remove ovpsim binary from repo history?it's large, no longer used
***<AR Imperas** - remove ovpsim binary from old the git repo commits? >*

PR #144 is the RFQ update to the comnpliance rep

Resource Update
- we have an offer to maintain SAIL from a volunteer group in China
- SAIL code will need to move under tech-compliance git repo
- SAIL language/compiler will reside at Cambridge
- we have an offer to develop FP tests & coverage models from IIT Madras

Chair – **CoChair** how is your port going?
CoChair – model works; working on git methodology for target mgmt.

RFQ updates

- Docs added/updated:
  MIGRATION.adoc: how old framework targets can work with changes made as part of this PR
  README.adoc updated to avoid the word "compliance" and updated new target porting section
  TestFormatSpec updated to avoid the word “compliance” and updated macro and signature definitions
  compliance_xxx.h files renamed to avoid the word “compliance”: arch_test.h & model_test.h
- Added example_target directory to host dummy starting points for porting targets. (provided by Marc Karasek).
- Migrated/ported existing targets (except codasip and sifive-formal) to the new framework changes.
- riscv-test-env/p/riscv_test.h changes
  macros renamed to avoid conflicts with the new framework macros (adds “_OLD” to RVTEST_[CODE/DATA]_[BEGIN/END])
  re-structured RVTEST_DATA_BEGIN_OLD/END to ensure all target specific data are after the signature.
- added new file riscv-test-suite/env/arch_test.h which contains the macros used by the new set of tests & symlinked in the riscv-test-env directory. The arch_test.h also includes aliases for the old macros.
- Moved encoding.h to riscv-test-suite/env so it can’t be modified by the targets, symlinked for backwards compat.
- Directory structure changes
  riscv-test-suite directory structure changed to matchTestFormatSpec document.
  root-level Makefile.include created to decouple the Makefile and target specific settings.
  Added riscv-test-stats directory for coverage and data propagation reports
- New tests added for RV32IMC and RV64IMC extensions.
- Added riscv-target and Makefile.include to the .gitignore file to stop tracking target specific changes.
- Added special targets for compile(build), simulate(run) and verify in the Makefiles of each test-suite.
- Fixed signature relocation for cases where signature section is not at fixed offset from data section
- Coverpoint YAML split into separate files

Decisions & Action Items

Decisions

Add new label to beginning of of RVTEST_CODE_BEGIN

Add QEMU Targets to repo

Move SAIL repo into riscv github repo

AR: Remove RVMODEL_IO_CHECK? Unless it can be justified

Outstanding Action Item

NEW

Inspire: Put dummy link script/model inclusion script into repo<new>

Everyone: report if RVTEST_IO_CHECK macro is used

Inspire: add support for QEMU target

Chair: get SAIL repo moved into a riscv repo

Old

Chair : get contact info for all model maintainers and inform them removal of model specific support code from the repo <started>

QC: extract bits of FAQ as guidelines for test writing <?>
Incore: Try YAML version of SAIL to see if it works <not done>

[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

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.

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, directory structure)
- use only files that are part of the defined support files in the repository
- run in framework
- run in SAIL and not fail any tests
- generate a valid signature using SAIL (that can be saved & compared with another DUT/sim)
- Report that test results propagate to signature
- 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)
- must be commented in 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 more 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
- 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	^	
#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	Fixed in RISCOF	
#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.	v	
#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
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