Architectural Test Task Group Call – Minutes

Thur, 13May2021 8am Pacific → Daylight ← Time

See slide 6 for agenda

Antitrust Policy Notice

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/

SIG Charter

The Architectural Compatibility Test SIG is an umbrella group that will provide guidance, strategy and oversight for the development of tests used to help find incompatibilities with the RISC-V Architecture as a step in the Architectural Compatibility self-certification process

The group will:

- Guide Development of:
  - Architectural tests for RISC-V implementations covering ratified and in-flight specifications for
    - Architectural versions, standard extensions, and implementation options.
  - Tools and infrastructure to help identify architectural incompatibilities in implementations
- Work with LSM and Chairs for resources to get the above work done.
- Mentor or arrange for mentoring for the resources to get the above work done

Adminstrative Pointers

- Chair – Allen Baum allen.baum@esperantotech.com Co-chair – Bill McSpadden bill.mcspadden@seagate.com
- SIG Email sig-arch-test@lists.riscv.org
  - Notetakers: please send emails to allen.baum@esperantotech.com
- Meetings -Bi-monthly at 8am Pacific time on 2nd/4th Thursdays.
  - See https://docs.google.com/spreadsheets/d/1L15_gHl5b2ApkcHVtpZyl4s_A7sgSrNN zoom link
- Documents, calendar, roster, etc. in
  - https://sites.google.com/a/riscv.org/risc-v-staff/home/tech-groups-cal
    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/doc/	tests	https://github.com/riscv/riscv-arch-test/
https://riscof.readthedocs.io/en/latest/index.html	riscof	https://gitlab.com/incoresemi/riscof/
https://riscv-isac.readthedocs.io/	ISA coverage	https://github.com/riscv_isac
https://riscv-ctg.readthedocs.io/	Test Gen.	https://github.com/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/
https://github.com/riscv-admin-docs/architecture-test/	minutes, charter	

- 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	annotated unpriv spec→	release/riscv-spec-sail-draft.pdf
release/riscv-spec-sail-draft.pdf	← annotated source	annotated priv spec→	release/riscv-privileged-sail-draft.pdf

  - https://us02web.zoom.us/rec/share/-XIYazzhIBbQoiZdarCfebdjxjDWiVhf-LxnuVrliN4Bc30yf17ztKkKDU4Og54b.fArPPqnuR-NiXpQU Tutorial
    Access Passcode: tHAR#5$V

Meeting Agenda

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

I. Updates, Status, Progress:
   - I. Still looking for CI/Testing chair, Simulator SIG chair
   - II. ACT policy ready for public review
   - III. Chairs: B,V,Zk and Priv1.12 are top priorities
   - IV. Priv tests will use OS boot for test until ACTs are available.

II. Next steps and Ongoing maintenance
   1. Discussion: testing methodology for SIGs/TGs needing external stimulus/observability "ports".
   2. Discussion: other steps for Migration to Framework v.3.0 (riscof). (blocking items):
      - a) (Sail/Spike model updates, pipecleaning, N people have run it, testing all the “fixed in riscof” issues
      - b) Review Pipecleaner tests:What do we need to do to exercise capabilities for Priv Mode tests
   3. Maintenance updates to V2 to enable future tests
      - a) update RVTEST_SIGUPD to keep automatically adjust base/hidden offset when offset>2K,
      - b) Enable use of Sail model results as the assertion value
      - c) add assertion macros for FP, DP, Vreg to arch_test.h and test_format spec
      - d) add trap handlers for S, VS modes
   4. Tests for non-deterministic result (see attached discussion in email)
      - a) Provide a reference RTL test fixture (as opposed to SW functional model). See. JIRA CSC-6
      - b) Define hooks for concurrency tests
   5. Specific Compliance Policy/Process Gaps:
      - a) Identify Tool providers, e.g. coverage model, test generation for new features/extensions
      - b) Flesh out test development order & identify resources (e.g. Priv,FDD or F,Priv,D..., JIRA CSC-3,5

Discussion

Passdowns, Status: see agenda (slide 6)

See sim_backplane_presentation.pdfmalso attached)
Slide2: Agenda: the Problem, bbase layer proposal, higher abstractions
Slide3: problem: getting dissimilar simuators to talk to each other, or apply stimulus & synch them
Slide 4/5: Simulator Backplane (interrupt controller input, trace output
Slide 7: an IPC model with IPC interfaces
Slide 8: Example RTL interfaces
Slide 9: Signal definitions used by simulators (2,4, 8 value logic) need to be reconciled.
Slide 10: Sample code (with triggers, interrupts, etc)
Slide 11: Backplane pseudocode, and why relaxation is important
Connector: tie all the interfaces together (YAML definition)
Relaxation is tricky
Example: interrupt generation – doesn't solve non-determinsim
Slide 12: Test pseudocode
Imperas - the requirement for something to stimulate events
Test use macros to signal framework right now
SG - Simulator can use signaling or messaging layer
Imperas - :Is this for DV or arch? –
SG - need ref model for sail trace
?Robtnexus: shared mem for performance
Currently using files rather than live messages
Need to support non-cycle accurate – can't be for DV

SG - speaking about interrupts:
· Idea was previously implemented at old company . Problems to solve:
· Get simulator to talk to each other?
· Same Test stimulus used for multiple models.
· Test cases: Interrupts and E-trace.
Slide 13: How is a level triggered interrupt cleared after having the stimulus?
Model needs to interact with stimulus.
Slide 14: sample SDI interface in C, verilog connections needed between extern interrupt stimulator to core model, clic model, core model. These should be modular so we can exchange with models or not.
Slide 15,16 : abstractions: converting RV messages to a send/rcv API
Slide 17-19: applying it to E-trace spec
o S/W decoder/encoder (e.g. in C)
o Core model generates e-trace to encoder to decoder, gets instruction stream.
  o Behavior might change during runtime due to config changes.

Base layer API Proposal :
- Process based , Connect w/ Unix mechanisms. Shared memory/Sockets had good perf.
- Simulator A (model). Process has API to get/set signals and step. All simulators build a simulator dependent ifc; they require their own interface to talk to simulator backplane.
- Simulator backplane: gathers/broadcast/resolves the signals it receives.
- Have all processes synchronized via a global timestep . All processes can be synchronous. General timestamp with different clocks can be figured out.
- Cycle with backplane in middle.
  - Similar to cadence tools to single step simulator itself. Start sim and single step. Similar to verdi but more general.
  - Proposing as open source effort.
- Instruction accurate vs cycle accurate. What is the definition of instruction accurate? Interrupts/other asynch events make this confusing (inherently nondeterministic!).
- For each interface, different stimulus types & data required for stimulus must be defined.
  - Signal def: What kind of type should it be? weak high, weak low. Different simulators support different types. Need to consider all of them. Inout types need to also be considered
  - Argue later about the format (YAML?)

Use case Scenario:
- Ex:: a write to a register, sdi_ are all api functions. Simulator advances upon sdi_cycle call
- Backplane must read & parse connection file, use it to make the connections between each simulator, gather, resolve, and relax the signals. All simulators must quiesce through relaxing. Multiple drivers of a signal relax to avoid an infinite loop caused by contention.
- Random interrupts generation example in slides. C example uses SDI + asm file w/irq handler which needs to do the write_reg to clear the interrupt. This does not solve the checking (verification) problem. This is purely for stimulus and clearing stimulus.

Chair - Framework std interrupt handler has model specific macro hooks already in place for this. Should be with both plic and clic compatible.
- Needs a mechanism to VPI For verilog (VPI has?), Sail,/Spike/Ovpsim . (more detail needed?)

Other:
- Could write different abstract layers. "Signaling layer", "Messaging Layer".
- The spec is defined in messages. Defining w/ messages aligns w/ spec vs. physical better.

Q: Is this required for basic riscv spec testing?
SG: yes. If there is SAIL model for e-trace, it needs to be attached to decoder/encoder and should be end to end flow. Need reference models for e-trace/nexus for full spec testing.
- Nexus trace group does validation from trace, uses files, should work in live system. Model require cycle accuracy. C models may not have that. So support for non-cycle accurate simulators too. What is the accuracy of Nexus test? It is instruction accurate with timestamp. Only for single hart flow. Non determinism could be an issue.

?? - see also https://github.com/riscv-verification/RVVI/blob/main/docs/rvvi.md
work being done under the auspices of open HW. Something to potentially build on.

Decisions & Action Items

Decisions

Outstanding Action Items

NEW

Old

Chair : write up ACT testing ,reporting requirement policy <done>
see for risc-v members > policies > … >Architectural Compatibility Test

Chair : document target process for removing target environment files from riscv-compliance repo into a target repo and contact all model maintainers to inform them of the process and timeline. <restarted>

Chair: more brainstorming on handling nondeterminism, concurrency <discussion started on retrofitting riscof for concurrency>

Chair: need clarity on tool source/version report.

Inspire: add support for QEMU target <?>

Incore: Try YAML version of SAIL to see if it works <ongoing>

SH: write up coverage taxonomy

Pull/Issue Status

Issue#	Date	submitter	title	status	comments
#4	03-Jul-2018	Kasanovic	Section 2.3 Target Environment	Fixed in riscof	Will be closed in V3
#22	24-Nov-18	brouhaha	I-MISALIGN_LDST-01 assumes misaligned data access will trap	^	HW misalign support not configurable
#40	4-Feb-19	debs-sifive	Usage of tohost/fromhost should be removed	|	now
#142	17-Nov-20	subhajit26	Not able to run compliance test for rv32E device and RV32E ISA	RV32E only	Not RV32EC or RV32EM
#146-9	01-Dec-20	Imperas	Test I EBREAK,ECALL, MISALIGN_JMP/LDST, OpenHW	v	HW misalign support not configurable
#107	22-Apr-20	jeremybennett	Clang/LLVM doesn’t support all CSRs used in compliance test suite	under discussion	-can we add an alias?
#115	06-jun-20	adchd	How to support on-board execution?	under discussion	
pull#129	31-jul-20	nmeum	sail-riscv-ocaml: Disable RVC extension on all devices not using it	In process	Who can review this?
pull#184	15-apr-21	dansmathers	Updating http reference for constr	In process	Approved, needs merge
#119	17-jun-20	allenjbaum	Missing RV32i/RV64i test: Fence	Test has been written	Close when RFQ test is merged
#188	26-Apr-21	neelgala	Updates required in K_unratified tests to be compatible with current RISCOF		
#189	26-Apr-21	neelgala	Proposal to enhance the RVTEST_ISA macro		
#190	26-Apr-21	neelgala	The 16-byte signature boundary issue		

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	done	
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	done	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

BACKUP

Test Acceptance Criteria

Tests merged into the ACT test_suite repo must :

- conform to the current format spec (macros, labels, directory structure)
  - including framework-readable configurations - i.e. which ISA extension it will be tested with (using Test_Case macro parameter equations) for each test case
- use only files that are part of the defined support files in the repository, including standard trap handlers
  - TBD: how to install test specific (not model specific) handlers
- Be able to be loaded, initialized, run, signal completion, and have signature results extracted from memory by a/the framework
- run using the SAIL model and not fail any tests
- generate signature values either
  - directly from an instruction result (that can be saved & compared with DUT/sim)
  - by comparing an instruction result with a configuration-independent value range embedded in the test code (e.g. saving above, below, within)
  - by comparing an instruction result with a configuration-independent list of values (e.g saving matches or mismatched)
    - (it can be useful to also return a histogram of value indices that matched)
- Store each signature value into a unique memory location in a signature region that is
  - delimited by standard macros embedded in the test which can be communicated to the test framework
  - pre-initialized to values that are guaranteed not to be produced by a test
- have defined coverage goals in a machine readable form that can be mechanically verified
- improve coverage (compared to existing tests) as measured and reported by a coverage tool (e.g. ISAC)
- use only standard instructions (and fixed size per architecture macros, e.g. LI, LA are allowed)
- be commented in test_case header (ideally listing coverpoint covered)

Tests that are otherwise accepted, but depend on tools or simulators that have not be upstreamed must be put into a <Ext-Name_unratified>/ directory instead of <Ext-Name>/

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 to be used inside tests based on the YAML configuration
- Include the compliance trap handler(s), & handle its (separate) signature area(s)
- 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 for test generation
  - Coverage specification is a separate file
  - Could be a separate app

Non-determinism in Architectural Tests

The RV architecture defines optional and model/µarch defined behavior.
This implication: there are tests that have multiple correct answers. E.g.:

- Misaligned accesses: can be handled in HW, by "invisible" traps w/ either misaligned or illegal access causes, and do it differently for the same op accessing the same address at different times (e.g. if the 2nd half was in the TLB or not)
- Unordered Vector Reduce ops: (different results depending on ordering & cancellation)
- Tests involving concurrency will have different results depending on microarchitectural state, speculation, or timing between concurrent threads (e.g. modifying page table entry without fencing)

From the point of view of ACTs, there are 2 (& sometimes more) legal answers. The golden model only generates one. Possible mechanisms to test include:

- Modify (if necessary) & configure reference model to generate each legal result, run it with each config, & accept either result from the DUT (e.g. misalign or un-fenced PTE modification)
- Provide specific handlers for optional traps
- Use self-testing tests(compare with list or range of allowed outcomes from litmus tests)
- Avoid tests that can generate non-deterministic results
- Ultimately: develop new frameworks that can handle concurrency along with reference models that can generate all legal outcomes
- It is the responsibility of the TG that develops an extension to develop the strategy for testing features and extensions that can have nondeterministic results

TGs under the SIG

- IF you’re creating work product, you should be a TG
- If changing requirements, plans ABIs, etc
  - Test plan==SOW
- The Architectural Compatibility Test Task Group will define and maintain specifications for
  - test formats
  - test-benches and frameworks needed for
    - privilege testing privilege testing,
    - Concurrency/ Memory model testing
    - Asynchronous event testing (interrupts)
    - Nondeterministic tests
  - ISA test coverage goals
  - test tools (e.g. coverage, generators)
- The Architectural Compatibility Test Task Group will maintain the appropriate GitHub:
  - tests for the individual ISA extensions
  - issues related to the tests
  - the operation and issues related to the framework
- The Architectural Compatibility Test Task Group will
  - work with the different privilege and un-privilege ISA extension Task Groups
    - to help them write test plans/specs for the ISA tests
    - to help them work with the sub-contractors (IITMadras, RIOS, CAS, etc) to deliver the tests
  - assess quality of delivered tests and be maintainer for the test GitHub

Meeting Conventions

- We don’t solve problems or detailed topics in most meetings unless specified in the agenda because we don’t often have enough time to do so and it is more efficient to do so offline and/or in email. We identify items and send folks off to do the work and come back with solutions or proposals.
- If some policy, org, extension, etc. can be doing things in a better way, help us make it better. Do not change or not abide by the item unilaterly. Instead let's work together to make it better.
- Please conduct meetings that accommodates the virtual and broad geographical nature of our teams. This includes meeting times, repeating questions before you answer, at appropriate times polling attendees, guide people to interact in a way that has attendees taking turns speaking, ...