Compliance Task Group Call – Minutes

Weds, Aug 27, 2019 8am Pacific →Daylight← Time

See slides 6,7 for summary

Charter

The Compliance Task Group will

- Develop a framework for RISC-V tests, taking into account approved specifications for:
  - Architectural versions (e.g. RV32I, RV32E, RV64I, RV128I)
  - Standard Extensions (M,A,F,D,Q,L,C,B,J,T,P,V,N)
  - 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 method 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 – Stuart Hoad stuart.hoad@microchip.com
- TG membership- Sue Leininger sue@riscv.org
  - Send email to her - you must have a lists.riscv.org login
- TG Email tech-compliance@lists.riscv.org
  - Notetakers: please send emails to allen.baum@esperantotech.com
- Meetings -Bi-monthly at 9am Pacific time on 2nd/4th Wednesdays
  - Location is https://zoom.us/j/6213886723
- Documents, calendar, roster, etc. in https://lists.riscv.org/tech-compliance/
  see /documents, /calendars subdirectories
- Git repositories
  - https://github.com/riscv/riscv-compliance/
  - https://github.com/rsnikhil/Experimental_RISCV_Feature_Model
  - https://github.com/rsnikhil/Forvis_RISCV-ISA-Spec
  - https://gitlab.com/incoresemi/riscof (Shakti framework)

Attendees

- Allen Baum (Esperanto)
- Simon Davidmann (Imperas)
- Deborah Soung (SiFive)
- Grant Martin (Cadence)
- Henrik Gustafsson (Qualcomm)
- Paul Donahue (Ventana)
- S Pawan Kumar (IIT Madras)
- Stuart Hoad (Microchip)

Meeting Agenda (in order of Priority)

1. #5 nonconforming extension support
2. #6 filter vs all – need to filter because traps aren't guaranteed?
3. YAML update

Discussion

1. #5 nonconforming extension support
   what is the goal?
   - Check that test target complies with architectural definition?
   - Check that test target complies with Platform spec. / execution environment?

   If the latter, can/should platform allow non-conforming extensions? (e.g. M-extension but with DIV handled as an SW emulation)

   Options discussed: (are there more?)

   a) only check for full extensions, don't support nonconforming extensions even if they meet platform reqs
   b) redefine extensions (e.g. Mul+DivTrap) – (not scalable)
   c) ensure reference model can cause traps on arbitrary (configurable) opcodes
   d) enable framework to load arbitrary emulation routines (requires e)
   e) redefine extensions with modes (e.g Umode M-extension, but not Mmode)
      (requires d, asserts Mmode won't use soft ops)

   Notes: this is a long term goal, not short term, but we need to put the infrastructure in place

2. #6 filter vs run all all tests -need to filter because traps aren't guaranteed?

   "We explicitly do not want to universally mandate trapping behavior on opcodes beyond those for which the implementation asserts compliance - only if we're checking compliance for a platform specification that requires trapping on those opcodes. There are only a few platforms where requiring traps is beneficial." - Krste

3. YAML update (see slides 8-18)

Conclusions & Action Items

Decisions

- Continue nonconforming

Action Items

- Allen will check with formal modelling group to see if they can (easily) define trapping behaviour.

YAML History

- From Nikhil (Formal Working Group chair):
- In RISC-V Compliance Testing: when testing an implementation's compliance with the RISC-V specification, we compare the implementation's execution to the execution of a "Golden Reference Model" (an executable Formal ISA Spec, or a reference simulator). That Golden Reference Model is usually a "universal" simulator that can model any RISC-V system, and needs to be constrained to the specific set of features in the given implementation.
- July 2018 - Experimental YAML file and checker
  - Target to configure the different formal models \
- Through late 2018 and early 2019 discussions in compliance group
  - Configuring simulators, configuring compliance framework, compilers

Tools etc., that need configuring reflect user choices from RISC-V spec

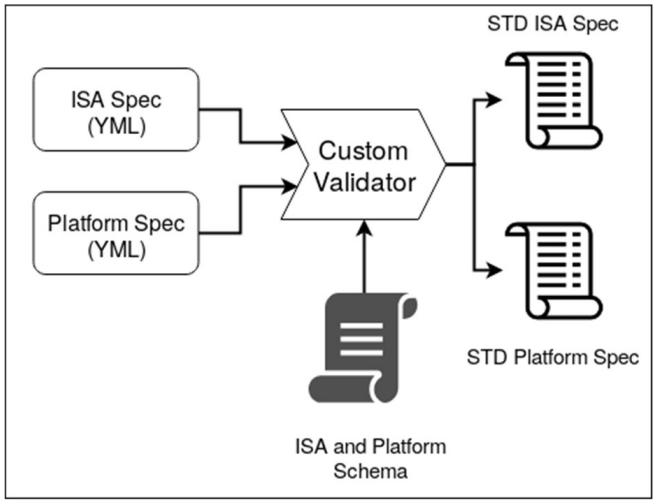
- •Test framework (to choose the appropriate tests)
- Individual tests to select parts of tests (using headers, macros, defines) Toolchains (eg 32bit, compressed, maths, etc…)
- Simulators: riscvOVPsim, spike, …
  - Imperas' riscvOVPsim is free on github, and is a full RISC-V spec (inc. vectors, bitmanip) envelope model/simulator and has over 60 config options
- Test generators, frameworks:
  - riscv-dv from Google Cloud is comprehensive RISC-V instruction stream generator
    - open source
    - SystemVerilog UVM
    - Targeted to be used in OpenHW Group and CHIPS Alliance open hardware verification flows
- Models: Formal Model SAIL, …
- …

YAML for RISC-V target config

- 1H2019 IITMadras (Shakti) IncoreSemi prototyping python test framework
  - Includes YAML config, validation – inspired by Nikhil's early work, Allen Baum added some WARL definitions, and Imperas guided with requirements, reviews, adoption
- 3Q2019
  - extracted YAML config and validator into github.com/riscv/riscv-config
  - Imperas added option to convert YAML into riscvOVPsim config inputs
  - riscv-dv (Google) requirements for CSRs in device YAML being reviewed/iterated
- Currently working on YAML rev2 to include Google riscv-dv requirements

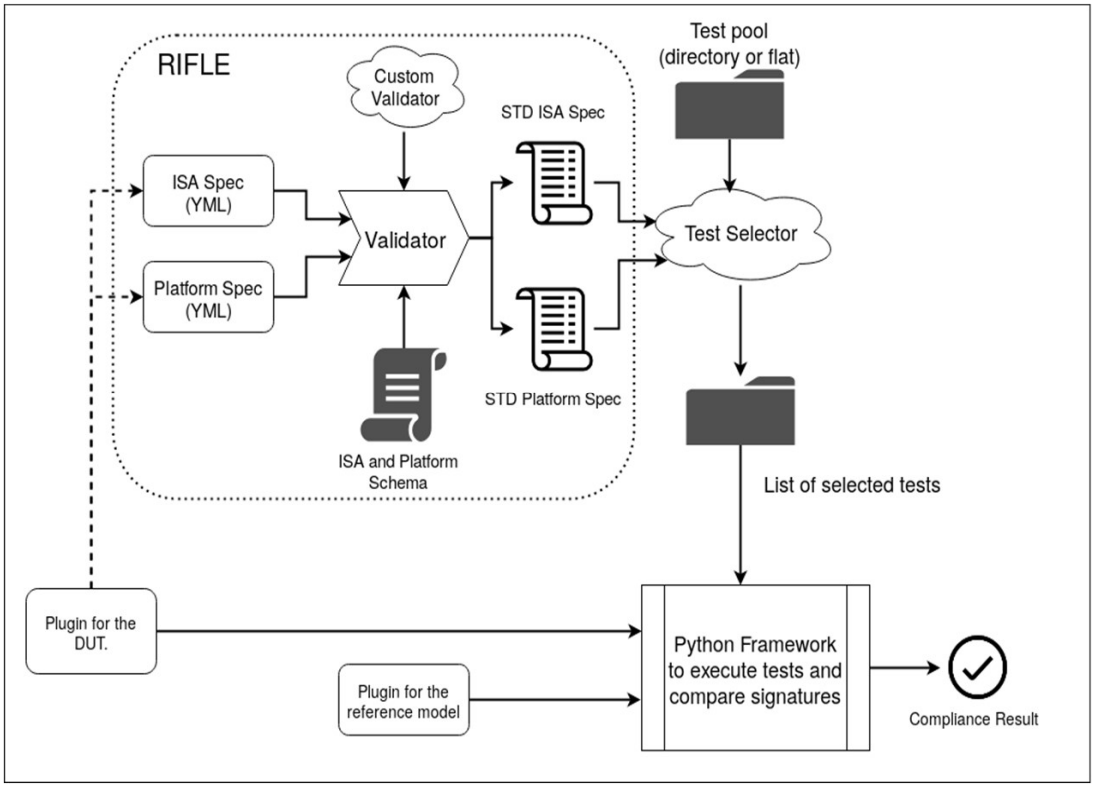
Current status: riscv-config

- https://github.com/riscv/riscv-config



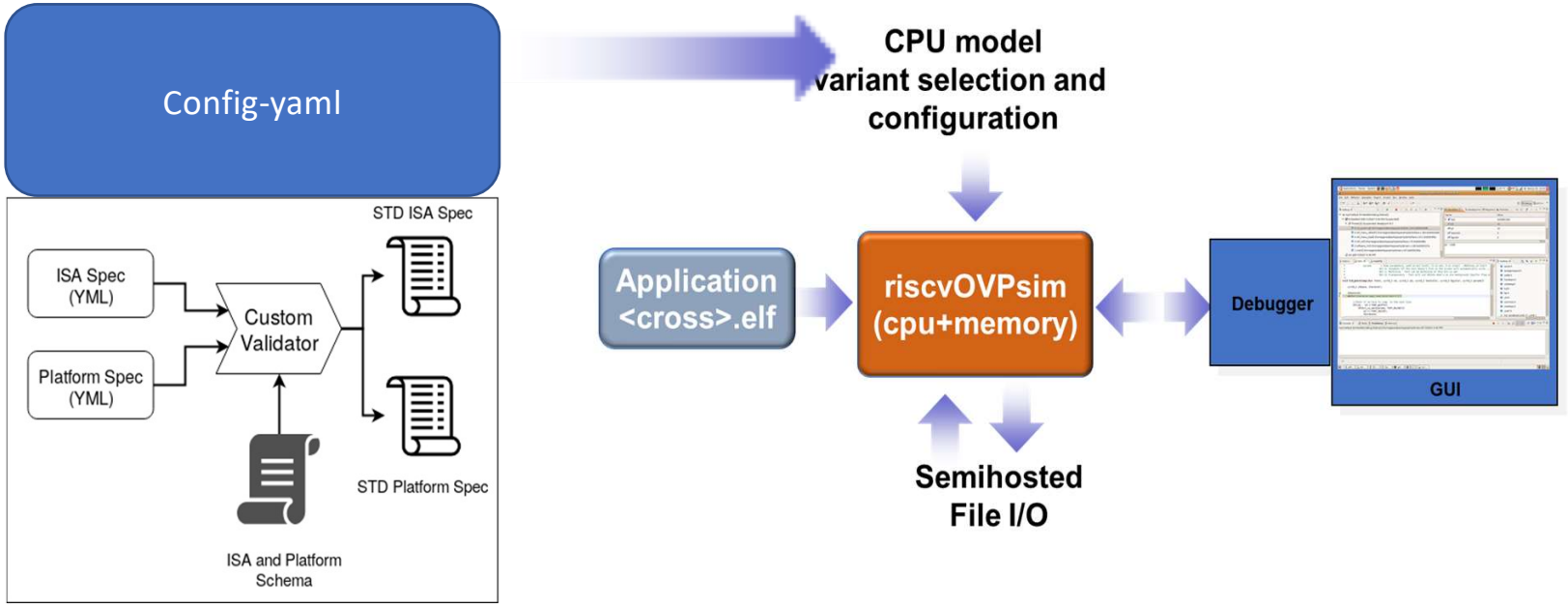
- ISA Spec: This YAML file is to capture the ISA related features implemented
- Platform Spec: This YAML file is to capture the platform specific features implemented
- Validator checks that all inputs are legal and consistent, writes out full checked YAML
- Currently using rev1 of YAML definitions

Current status: riscof (IncoreSemi shakti python test environment)



- Under development: https://gitlab.com/incoresemi/riscof/

Current status: riscvOVPsim



- https://github.com/riscv/riscv-compliance/tree/master/riscv-ovpsim
- https://github.com/riscv/riscv-compliance/tree/master/riscv-ovpsim/config-yaml
- Python program (imports the riscv-config module) and writes out OVPsim config files

Current Status: riscv-dv https://github.com/google/riscv-dv

- Using YAML for
  - ISS config
  - RTL simulator config
  - Testlist config
  - CSR config
- Working on using YAML
  - with compliance group rev2 YAML to configure device target
  - And to configure test generator options

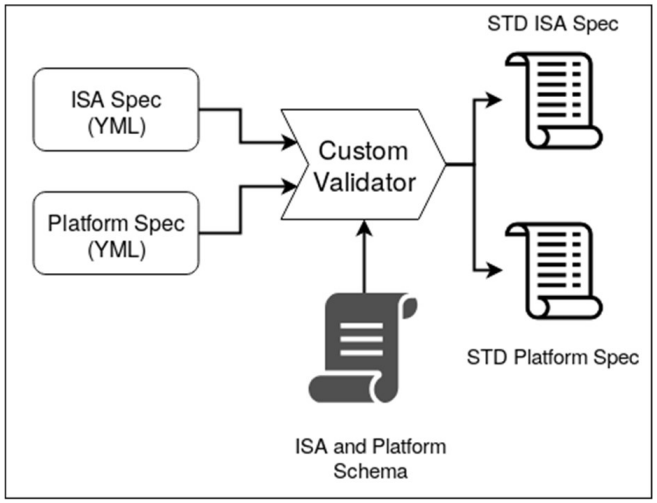
Configuring RISC-V Formal SAIL model

- Had initial discussions with SAIL team & Formal group chair
- Nikhil will look at riscv-config rev2 YAML spec
  - And provide feedback
- Imperas is looking at SAIL model to see what needs configuring...
- Expect progress during September 2019

Current status: compliance framework

- Imperas moving the use of make fragments and variables over to using YAML device config files
  - With appropriate converters as needed
- Imperas developing flow, with pipecleaner device config YAML files & flow for Google/lowRISC/Ibex variants
  - lowRISC 32bit core, 3 variants (I, IMC, EMC) evolved from ETH PULP zero-riscsy/pulpino
    - DrivenbyGoogleCloud
  - Configuring riscvOVPsim as reference
  - Running Ibex as target with Verilator
  - Configuringtoolchains
- Future:
  - When SAIL model is configurable, adopt it as reference in compliance flow

Current status: riscv-config https://github.com/riscv/riscv-config



- ISA Spec: YAML file that captures ISA related features implemented
- Platform Spec: YAML file that captures platform specific features implemented
- Validator checks that all inputs are legal and consistent, writes out full checked YAML
- Currently using rev1 of YAML definitions

RISCV-CONFIG

- Examples & definitions
  - https://github.com/riscv/riscv-config/tree/master/examples
  - https://github.com/riscv/riscv-config/tree/master/riscv_config/schemas
  - https://github.com/riscv/riscv-compliance/tree/master/riscv-ovpsim/config-yaml/examples
- Validator
  - https://github.com/riscv/riscv-config/blob/master/riscv_config/checker.py
- Example integration of converter (OVPsim)
  - https://github.com/riscv/riscv-compliance/tree/master/riscv-ovpsim/config-yaml
- WARL
  - https://riscv-config.readthedocs.io/en/latest/yaml-specs.html
- YAML Rev2 CSR definitions & WARL proposal
  - https://github.com/neelgala/riscv-config/tree/master/warl_proposal

Backup for prievous discussions

License Inconsistencies

Ken Dockser writes

In going through the files on git hub I have found inconsistencies in the licenses specified. Based on riscv-compliance/doc/README.adoc, the intent was to use BSD and Creative Commons.

- The top level license (riscv-compliance/COPYING.BSD) is a 3-clause BSD,
- The riscv-compliance/riscv-test-env/LICENSE specifies a slightly different 3-clause BSD license naming Regents as the copyright holder.
- In riscv-compliance/riscv-test-env/test_macros.h an Apache v2 license is employed. In fact, Apache-v2 shows up in 57 files.

Desire is to use Apache? (BSD regents may eventually be replaced. RISCOF uses BSD 3-clause

Test Spec

- Proposed Structure is 2-level: <arch>_<modes>/<feature(s)>
  - <arch> are rv64i, rv32i, rv32e
  - <modes> are M, MS, MU, MSU: modes that the test will run in
    - Always starts in M at least, so always present ←still disagreements
  - <feature(s)> are
    - lettered extension [A | B | C | M ...] or subextension [Zam | ...]
    - more general names when tests cross extensions (e.g. Priv, Interrupt, VM, Integer).
    - Exact syntax/names for cross-extension subdirectories has not been ennumerated.
    - Tests that can /should be run in multiple modes replicate the subdirectory
- New Standard Macros
  - RVTEST_CASE(CaseName,CondStr, [DocTmp, DocString])
    - Test ases must be inside `#ifdef TEST_CASE_<CaseName>,#endif` pairs
  - RVTEST_SIGBASE(BaseReg,Val)
  - RVTEST_SIGUPD(BaseReg, Reg, Value)
  - RVTEST_M2S, M2U, S2U, U2M, U2S, S2M (?) -- TBD

Foundation Expectations

- Objective: publish compliance test 1.0 and finish the public review **before** the RISC-V summit in Dec. Shorter term is pre-1.0 by EO Q3
- Scope: publish tests and expected results run from the executable RISC-V formal specs -- make sure that all formal specs agree with each other
  - (Note: this approach will not work for priv spec)
- Minimal acceptance criteria is RV32Imc and RV64Imc
- Allen will focus on driving the task group to make this happen
- Nikhil will be tasked to ask all formal spec groups to commit their executable model support in the riscv-compliance repository
- Silviu and Yunsup will make the {compliance manager} CFP happen. They just need to understand what help is needed.