Compliance Task Group Call – Minutes

Weds, Nov 20, 2019 8am Pacific →Daylight← Time

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
  - https://riscof.readthedocs.io/en/latest/ riscof
  - https://riscv-config.readthedocs.io/en/latest/ config: YAML and WARL spec
- 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)
• Lee Moore	(Imperas)
• Bill Mcspadden	(Seagate)
• S Pawan Kumar	(IIT Madras)
• Ilja Stepanov	(Syntacore)
• Neel Gala	(IIT Madras)
• Paul Donahue	(SiFive)
• Premysl Vaclavik	(Codasip)
• Sergey Vasiliev	(Syntacore)
• Simon Davidmann	(Imperas)
• Robert Norton-Wright	(Cambridge U?)

Meeting Agenda (in order of Priority)

1. Pull request 65 –
2. TestFormatSpec

Discussion

- 3.6 Change “framework” to “test environment” or “test target” – add wording that extends test target to include signature generation – a collection assembling and link and executing, preprocessing (e.g. linker) scripts
- 3.8 Target is what .S goes into, and how it produces signature
  - 3.8 Define Target shell – collection of things around the target (assembler linker, linker script, memory initialization and extraction)
- 3.9 Much discussion of whether its possible to annotate tests with conditions, especially if auto generated.
  - If there are thousands of tests, this may not be practical, unless annotation is automated. The unprivileged architecture has simple conditions, and can likely be annotated with simple scripts, often based on the test directory name (which would change 4.1.1 below)
- 3.10: framework doesn’t insert into signatures, but into the report
- 3.10 Remove “master engine”
- 4.1.1 Directory structure is not functional but organizational
- Remove 4.1.1… maybe 4.1.2 also – notify compiler toolchain group
- Remove 4.1.3 emulated ups
  - Only testing HW compliance – need to stuart to demo how div was done at micros
- Removing 4.1.4 To Be Discussed items, which discuss 4.1.1,.4.1.2, and 4.1.3, since they’ve been removed

- Removing 4.3.1 To Be Discussed items as this has been answered, and replacing it with “To be Added ‘ section,
- 4.3.2.3,.4 – ensure that sigbase and sigupd are not required, but encouraged to help manual tests writing
- Fix examples / rules to use updated version of RVTEST_xxx macros

Decisions & Action Items

Decisions

- Remove test taxonomy, binary tests, and emulated op paragraph
- Make other sec 3.6, 3.8, and 3.10 changes and restructuring

Action Items

(Allen) make changes to spec as per discussion

(Allen) notify SW group that it will be responsible for binary tests (i.e. testing that their tools work properly)

(Stuart) Present how Microchip tested SW div compliance

(All) review section 4.3 and beyond

Future Discussions

1. TestFormatSpec sections
2. Coverage metric
3. ~~Email discussion#5 nonconforming extension / emulated op support.~~ Removed until platform spec call for it
4. WARL definition

Backup for previous discussions

Previous: RISCOF Status

Targets and framework (incl. tests) are now in separate repositories.

RISCOF: https://gitlab.com/incoresemi/riscof . YAML based framework, test-pool and relevant docs.
RISCOF-Targets: https://gitlab.com/incoresemi/riscof-plugins plugins for various targets

Differences from current github riscv-compliance and riscof:

- Test preamble significantly minimized. (see here)
- Preamble can add custom boot-code/libraries, etc through RVMODEL_BOOT macro
- No reference signature: obly comparison between golden and DUT models
- RISCOF generates 2 separate elfs for golden DUT models.
  - differ only in MODEL specific features : boot-code, termination sequence, etc. Tests not modified for either of the models.
- RISCOF allows parallel execution of the leveraging pmake (a.k.a make -j<jobs
- RISCOF uses the new directory structure
- RISCOF automatically generates and maintains a YAML database of all the tests

Merging RISCOF to riscv-compliance on github:

- RISCOF now is a complete python framework available as a pypi project: here.

  Rrepo gets shipped as a complete package, installed via **riscof pypi** as a command line tool
  cloning/updating the repo not nee whenand building tool.
- Should only require adding the current docs (readme, license, testpec-format) to RISCOF docs folder
- riscof-plugins/targets will continue to remain separate from RISCOF
- Sifive is missing (issues with verilator compilation)
- ibex, ri5cy are missing– only added recently.

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, YAML
  - https://riscv-config.readthedocs.io/en/latest/

Draft Test Coverage Proposal (unpriv)

Classes of things we want to test for

- Decode
  - Immediate – test all bits in either polarity will affect output
  - Register specifiers – test that changing any bit will affect output, ensure all regs are tested
  - Variations – test values of opcodes suffixes that have any string after a "." in its opcode
- Register combinations
  - Destructive (dest = either src) and non-destructive
  - Non-updating (i.e., targeting X0), or non-supplying (X0 as an input)
  - All registers (or immediate bit) should be used per instruction *category*
- Special and exception cases
  - Explicitly defined (e.g. shifts>=XLEN & RD=X0
  - Implicitly defined – corner cases
    - Maximal and minimal inputs, or creating maximal outputs
    - Inputs that special case outputs (mostly FP cases, also. shiftamt>=XLEN)
    - Outputs crossing value boundary (e.g. address cross word/page/superpage/VA boundary, FP crossing exponent boundary)

proposed
coverage & categories

Arith[I],	*W1/0,crys*
Logical[I],	*W1/0*
Shift[I],	*W1/0/msk,+*
Auipc,Lui,	
Ld,St,	*W1/0, bndXing*
Br,	*W1/0, bndXing*
Jmp ,	*W1/0, bndXing*
Ebreak/ Ecall	

W1/0= walking 1/0
BndXing=: boundary crossing

This works for 32i base ops – what do we need to add for priv modes? Mem model? Sequential Dependencies? Other extensions?

Need a review of existing (non-RISC-V) compliance specs

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.

RISCV-CONFIG WARL Syntax

WARL: {optional items in curly braces}

- dependency_fields: [list] – use this when legal/illegal values depend on other fields (in list)
- legal: [<warl-string>{,<warl-string>*}]
- wr_illegal: [<warl-string>{,<warl-string>*}] -> update_mode

where <warl-string> is either ”&” separated list of rangehi:rangelo lists

{[dependency_value] ->} field-name1[bit#hi:bit#lo] in [legal-range-list]

*{ & field-name2[bit#hi:bit#lo] in [legal-range] }**

or ”&” separated list of bitmasks

{[dependency_value] ->} field-name1[bit#hi:bit#lo] bitmask [mask, fixval]

{ & field-name2[bit#hi:bit#lo] bitmask [mask, fixval] }*

(can’t mix ranges and bitmasks)

RISCV-CONFIG WARL Example1

```
#         When base of mtvec depends on the mode field.
WARL:
  dependency_fields: [mtvec::mode]
  legal:
    - "[0] -> base[29:0] in [0x20000000, 0x20004000]"                    # can take only 2 fixed values when mode==0.
    - "[1] -> base[29:6] in [0x00000:0xF00000] & base[5:0] in [0x00]" # 256 byte aligned when mode==1
  wr_illegal:
    - "[0] -> unchanged"
    - "[1] wr_val in [0x2000000:0x4000000] -> 0x2000000"             # predefined value if write value is in this range
    - "[1] wr_val in [0x4000001:0x3FFFFFFF] -> unchanged"            # predefined value  if write value is this range

#        When base of mtvec depends on the mode field. Using bitmask instead of range
WARL:
  dependency_fields: [mtvec::mode]
  legal:
    - "[0] -> base[29:0] in          [0x20000000, 0x20004000]"          # can take only 2 fixed values when mode==0.
    - "[1] -> base[29:0] bitmask [0x3FFFFFC0,  0x00000000]"          # 256 byte aligned when mode==1
  wr_illegal:
    - "[0] -> unchanged"                                              # no illegal for bitmask defined legal strings.

"
```

RISCV-CONFIG WARL Example2

```
# no dependencies. Mode field of mtvec can take only 2 legal values using range-descriptor
WARL:
 dependency_fields:
 legal:
  - "mode[1:0] in [0x0:0x1]                              # Range of 0 to 1 (inclusive)"
 wr_illegal:                                             # default to 0 if not a legal value
  - "0x00"

# no dependencies. using single-value-descriptors
WARL:
 dependency_fields:
 legal:
  - "mode[1:0] in [0x0,0x1]                              # also Range of 0 to 1 (inclusive)"
 wr_illegal:
  - "0x00”
```

- "[1] wr_val in [0x2000000:0x4000000] -> 0x2000000 & wr_val in [0x4000001:0x3FFFFFFF] -> **unchanged**