Compliance Task Group Call – Agenda

Weds, Sep 11, 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)
- Lee Moore (Imperas)
- Bill Mcspadden (Seagate)
- Jacob Chang (Sifive)
- Paul Donahue (Ventana)
- S Pawan Kumar (IIT Madras)
- Neel Gala (IIT Madras)

Meeting Agenda (in order of Priority)

1. Pull requests
2. Status: RISCOF status, Google riscv_dv
3. WARL syntax,{see slide 6,7}
   https://github.com/neelgala/riscv-config/tree/master/warl_proposal
4. Email discussion#5 nonconforming extension support {see slide 8}
   – continue discussion
   - Reference model support for YAML

RISCV-CONFIG WARL Syntax

WARL: {optional items in curly braces}

- dependency_fields: [CSR::fld1 {, CSR::fldN}*]
  - 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:

{*[dependency_value] ->*} field-name[bit#HI : bit#LO] in [legal-values]

*{ & field-name[bit#*HIn*:bit#LOn] in [legal-values] }**

or

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

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 [0x4000001:0x3FFFFFF] -> unchanged”           # predefined value  if write value is in another 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 [0x4000001:0x3FFFFFF] -> **unchanged**

Previous 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 all 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. YAML update (see slides 8-18)

Discussion

1. Pull requests – #64 n ot checked, but affects nothing else
2. Status: RISCOF status, Google riscv_dv
   - RISCOF running R32IMAC, w/ new macros, can choose models, all targets ported except latest
   - Will create a branch with same structure as current compliance repository for testing
     - Need to have a runthrough of riscof flow after creating branch
   - Templates for CSR schema YAML arein a riscv-config branch, but checks not yet complete
   - Each target can load their own preamble, riscv_test.h (inlcudes setup, trap vector code, etc)
   - Google risc_dv Inst stream generator, has issues with tests, configs used elsewhere, ibex
     - Preamble updated to account for no smode, w/o mtvec (so does riscof)
   - Google risc_dv likes and is using WARL syntax
3. WARL syntax,{see slide 6,7}
   https://github.com/neelgala/riscv-config/tree/master/warl_proposal
4. Email discussion#5 nonconforming extension support {see slide 8}
   d) Is already implemented in riscof
   a) preferred in the absence of platform models (and is highest priority, in any case)

Conclusions & Action Items

Decisions

- Will continue with option a) (support only conforming ratified extensions) until otherwise required

Action Items

- Allen will try again check with formal modelling group to see if they can (easily) define trapping behaviour.
- The riscof repository will be put in its initial form aina riscv-compliance branch for testing

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 previous discussions

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.