# Compliance Task Group Call – Minutes

Weds, Jan22 2019  8am Pacific →Standard← Time

See slides 9,10 for discussions and action items

# 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 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
  - 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)

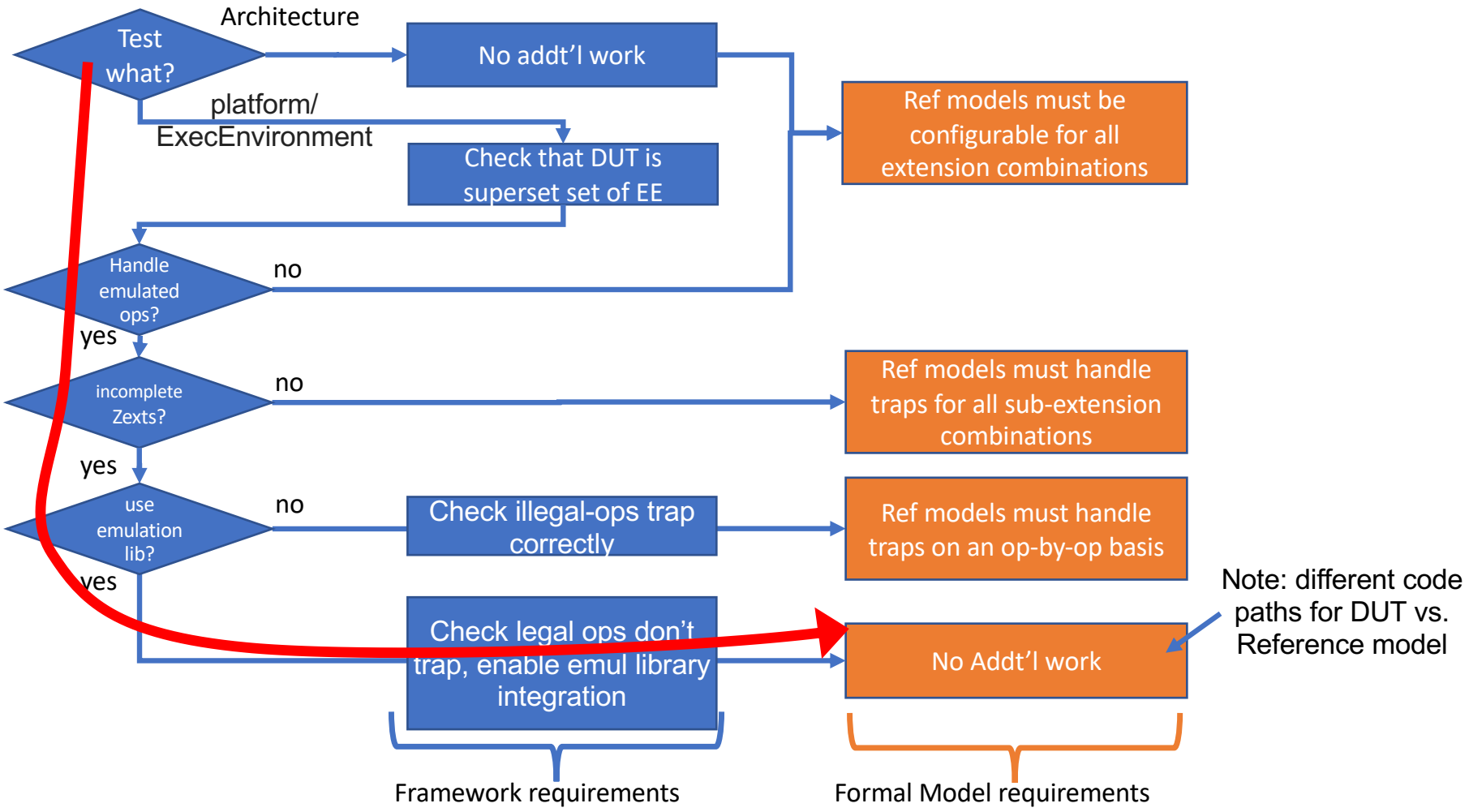
# Attendees

- Allen Baum (Esperanto)

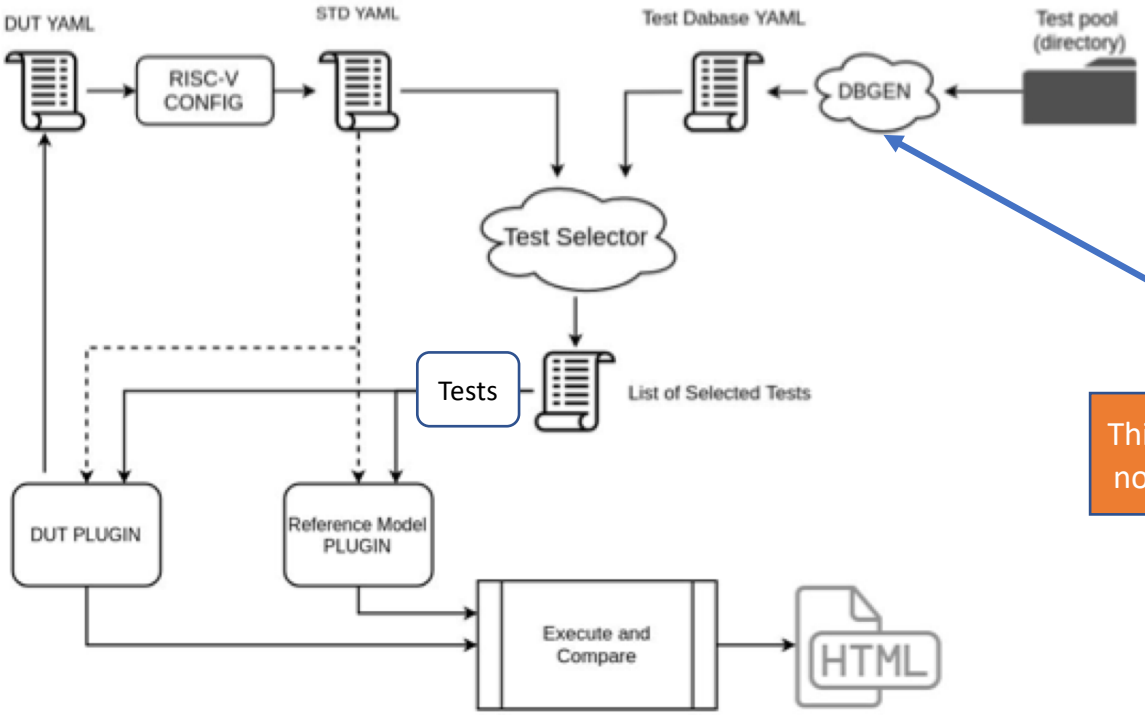
# Meeting Agenda (in order of Priority)

1. Pull request ( #65)
   - Review comments on emulated op handling (slide 6)
   - Review comments on need for test pool reference doc
   - If no substantive comments, will set up vote to ratify test spec 1.2.4
2. RISCOF reviews:
3. Reviewing and Closing issues (currently 23 open)
4. Looking towards the future
   - Getting more repository maintainers
   - Funding to get more tests/tools for tests, better coverage metrics
   - Transitioning to a standing committee – what is needed?
   - Research: using formal models to generate tests?

# Emulated Ops



# Test Pool




This is the test-spool reference doc; not used outside the framework

# Issue Status

| Issue# | date | submitter | title | status |
|---|---|---|---|---|
| #37 | Jan 31 | astimonov | rv32imc C.SWSP test5 writes a word outside the binary | bug - needs fixing |
| #30 | Jan 21 | RolfyYu | The golden results of rv32ui and rv64ui should be differ | bug - needs fixing |
| #08 | Oct 17, 2018 | AnttiLukats | RV32I/I-IO.S bad file name | bug - needs fixing |
| #67 | Sep 25 | rongcuid | RV32I Immediate Operands error | fixed in commit cae8567? |
| #11 | Oct 26, 2018 | neelgala | illegal.S in rv32mi generates supervisor interrupt - might not be supported on all implementations | fixed, close |
| #32 | Jan 25 | debs-sifive | breakpoint.s undesired behavior if trigger doesn't exist? | fixed? |
| #33 | Jan 28 | debs-sifive | rv32si/ma_fetch.S produces a different sig if RVC supported | fixed? |
| #27 | Dec 21, 2018 | jlucnagel | Macros are checking side-effects | fixed? |
| #28 | Dec 21, 2018 | bluewww | I-SB-01 test war hazard (address register) | fixed? |
| #38 | Jan 31 | santhoshvlsi | I-LB-01 test - Load the data into X0 GPR register | not a bug? |
| #42 | Feb 05 | as-sc | Misaligned fetch bit must be excluded for RVC | not a bug? |
| #70 | Oct 08 | towoe | Target error exit condition | not a bug? |
| #63 | Aug 13 | jeremybennett | Global linker script is not appropriate bug | TBD - fixed in riscof? |
| #47 | Feb 16 | aprnath | Machine mode atomic extension tests? | will be fixed in v.2, should currently work |
| #03 | Jul 03, 2018 | kasanovic | 2.4 Processor configuration clarification | will be fixed in V.2 |
| #04 | Jul 03, 2018 | kasanovic | Section 2.3 Target Environment | will be fixed in V.2 |
| #09 | Oct 22, 2018 | neelgala | Setting SATP and PMP should be optional | will be fixed in V.2 |
| #16 | Nov 07, 2018 | neelgala | I-MISALIGN_JMP-01.S assumes C-ext can be turned off | will be fixed in V.2 |
| #22 | Nov 24, 2018 | brouhaha | I-MISALIGN_LDST-01 assumes misaligned data access traps | will be fixed in V.2 |
| #31 | Jan 25 | debs-sifive | I-MISALIGN_JMP-01.S outdated `mbadaddr use ` in handler? | will be fixed in V.2 |
| #40 | Feb 04 | debs-sifive | Usage of tohost/fromhost should be removed | will be fixed in V.2 |
| #45 | Feb 12 | debs-sifive | Reorganization of test suites for code maintainability | will be fixed in V.2 |
| #72 | Oct 25 | vogelpi | Allow for non-word aligned `mtvec` | will be fixed in V.2 |

# Discussion

1. Emulation support: Defer until a platform spec exists – just a waste of time now
2. Removing test_pool_referece document (/database) from test format spec – broad agreement

   Related: manually inserting test_case macro conditions is error prone – should be generated automatically

   Automatically generating seems to be unsolved – can't be just based on op being tested because some tests aren't about ops, but more global options (e.g. misalign, VM supported, WARL CSR legal values, etc)

   If a test generator can determine conditions, it can also fill in the macro parameter
3. Riscof review:

   a. Riscof has 3 steps; configuration already split off.. Request is for each steo (dbgen, selection, run) be 3 independent programs
   Enables test developers to do it incrementally.
   Should be possible now (with right command line?)
   Neel will document and show an example

   b. Riscof requires nonstandard tools (actually, requires tools that may be unsupported in some OSes or not included in standard distribution (e.g. pmake not available under CentOS, python 3.7 not standard in Debian long term releases). Not all riscof problem (e.g. riscv-tools requires device-tree-parser which is nonstandard

   c. Too many apt-gets

   d. Required configuration is scattered around

   e. Some hard-coded paths in python scripts

   f. Some commands need to be executed from specific paths to work

5. Potential fixes:
   a. Release a docker image
   b. Rewrite to need more generic support
   c. For initialization and configuration, have a generator script rather manual steps – fill in dialog boxes for required customization
   d. Better documentation of assumed directory structure and/or standardize required directory structure

# Decisions & Action Items

## Decisions

- Defer emulation (and platform) support until we have a platform definition that requires it
- Remove mention of TestPool Reference Doc from TestFormat Spec; it isn't required to write tests

## Action Items

Allen - update test format spec, removing TestPool Reference doc mentions, and move anything related to emulation to a "deferred" section

Simon – will re-review the test format spec (Allen will send out poll for approval if no further issues reported)

Simon – come up with tset condition examples that are difficult to do manually? (Vector spec)

Neel,Pawan
- document how to use riscof as separate pieces
- investigate if possible to release a docker image
- look into a generator script
- remove hard-coded directory references from scripts, establish/document a standard directory structure for the riscof environment

Allen-document required changes for emulation to send to Neel, Pawan

# Next Meeting Agenda (in order of Priority)

# Backup from previous discussions

# State of Compliance

- The de-facto compliance framework is the existing GIT repository
- There have been significant updates to this lately
  - a separate config repository that enables precise description of implemented options, including more precise WARL definition
  - more model support
  - more coverage support
- This will be formalized as a v.1 compliance framework
- V.2 is under review
  - updated test spec format, designed to be easier to write compliance tests (under review)
  - updated framework (riscof) (under review)
    - implements the updated test spec format,
    - specifically enables dynamic signatures, instead of static or self checking
  - formalizes WARL description
    - Limits illegal→legal mappings, making formal models and tests easier to write

# Future Discussions

1. Coverage metric (slide 15-16)
2. WARL definition – (slides 17-20)

   https://riscv-config.readthedocs.io/en/latest/yaml-specs.html#warl-field-restriction-proposal

# 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

# Draft Test Coverage Proposal (more, incl priv)

- Forwarding: result of one op can be used as the source of the very next instruction
  - Need at least a case within and between instruction classes
- Changing non-reg state used by an op, immediately followed by op that uses it, e.g. :
  - changing the rounding mode for an FP op
  - writing into the instruction stream, followed by a fencei affecting the next ifetch
  - changing a page table entry  or PMP entry, or SATP affecting the next access
  - changing xEPC or xSTATUS followed by xRET
  - changing MISA followed by any op enabled or disabled by it
  - changing xTVEC, xDELEG, xIE followed by a trap
  - write once behavior (PMP-lock)
- Ops that change non-reg status, immediately followed by op that tests it, e.g.:
  - FP status after an FP op
  - xSTATUS.FS,XS fields after FP, Vector or other coprocessor op
  - xCAUSE, xEPC, xTVAL, xPP after an interrupt or exception

# 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/

# 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 [0x4000001:0x3FFFFFF] -> 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 [0x4000001:0x3FFFFFF] -> **unchanged**