Notes from the 22Mar meeting:
The TG chair is the mgr of the group - not the person saying what is to be done.
For references, see compliance_TG_22Mar18_agend-rev2 on the workroup site

Discussions:

What are we testing #1
- what is the dividing line between compliance and validation/verification?
  - fuzzy line between verification and compliance
  - full verification needs white box - cant be done...
- not testing all possible source/ encodings and all possible input operands
  - compliance is to spec...
  - primarily testing that Input state → output state is correct
  - don't test all exhaustive - just one per format
- check corner cases
  - e.g. floating point: check each inst. can signal all possible legal exceptions
  - e.g. for add – don't need to check every combination of registers
    - and for each instruction type
  - e.g. check all bits can be used in immed format, bits appear in right place
  - e.g. test that all bits of all source operand encodings do the right thing
    - e.g. immediates, reg numbers)

(See slide 3)
What are we testing #2

- We are testing compliance of implementation - not just HW
- another way to look at it: compliance tests if provider's hardware runs user's SW
  - hardware is not just a microprocessor core!
  - tests functionality, not performance
  - not our job to say the product is a good product, just that it works

Q: is this hw compliance or sw compliance ?
A: compliance is about system, not care about whether it is implemented in hw or sw

macros have been defined to configure tests to run only what you are claim to be testing
- provider has to implement those macros
- IEEE FP is an example of HW vs SW or both. (e.g.
- vendors must use macros set up environment
- Vendors need to provide code to support unimplemented but required HW
  - (e.g. subnormal FP support, or Divide, or unaligned support)

The User ISA spec has only one option – misaligned
The Privileged spec has many more
Platform definitions define constraints on implementation options

(**slide 4)**
TG history- 2 approaches to testing for compliance
- self test: test decides if its result
- signature: store results of tests and validate against golden model

The direction that has been decided upon is the signature approach.

Note that a signature does not imply a single value;
it can be a number of values stored at some specific offset in some region of memory

Each test will be documented with its purpose/goals, pointers to corresponding spec, description of the test, and the code itself

There is a lot of info about what tests are and do in the documents section on the workgroup site

Test suites will be modular, with a set of tests for each ISA subset

(Start of) slide 7 discussion:
Q: How do we (formally) describe platform?
e.g. what are valid physical addrs, options implemented/required

Q; can compliance test use other section of architecture spec?
A: compliance tests should expect only things that are legal at that level of the spec tests should not require any other features to be implemented

Q: where does golden reference come from? (Slide 7, bullet 1)
can a machine generated model come from the haskell model?
A: haskell model is work in progress - still a lot to do,
including mem model and complete set of options implemented (or not)

Q; can a formal model handle all config options –
A: no answer yet

Q: how much of the configuration options will be available in the haskell model?
A: spec has lots of choices for each piece
instruction compliance - very few choices
in priv level - platform profile will limit these

Q: is there a min machine mode?- ... - big discussion....
A krste says its just instructions...
simon/stuart says need full machine mode with all required M-mode CSRs
(someone mentioned SW models for education as an ISA only example)
krste - ISA tests and platform test are separate
feels needs for profile tests as well as ISA test
ISA tests can run in different modes (M/S/U) for different platform profiles
wants rv23i tests to run in linux under M, S, and U modes...