Risc-V Compliance Task Group August 15, 2018 meeting minutes

Allen	(Esperanto) [Chair]	Flemming Anderson	(Galois)
SimonD	(Imperas)	Ilja Stepanov	(Syntacore)
Rishiyur Nikhil (	Bluespec)	Ken Dockser	(Qualcomm)
Radek Hajek	(Codasip)	Jeremy Ralph	(Xtreme-EDA)
Martin Palkovic	(Codasip)	Ramprasad Chandrasekaran	(Xtreme-EDA)
Ben Selfridge	(Galois)	Prashanth Mundkur	(individual)
Joe Kiniry	(Galois)		

Presentation: Rishiyur Nikhil (Bluespec) of RISCV Formal Working Group

- introduced his RISCV formal feature set using set of slides
  - https://github.com/rsnikhil/Experimental_RISCV_Feature_Model
- golden reference models are universal, e.g. spike, formal specs, but also golden simulators like Imperas
  - but need to constrain the golden reference model so it behaves the same as the target
  - feature set documents all implementor choices from the riscv.org specification
- current experimental prototype - as strawman domain specific language and tool that uses it
  - a DSN and a tool that uses it (See the pdf for more info), defining:
    Features: modes, options (even address maps)
    Constraints, and preconditions
    Basic functions used to compose constraints & preconditions
    Separate file for non-standard features
- Showed demo code... - all avail from the githib repo including the yaml and python files.
  - data from Allen's spreadsheet and Simon's Imperas files

Q: What precisely does “not relevant” mean?
Defaults: declared, so output has complete set

Todo:
1. IS the list complete?
2. add defaults
3. Apply to formal model

<u>Joe Kiniry (Galois) [principal scientist]</u>
Complex systems have dials, levers to configure - so feature modeling is important
Galois has experimented with tools that allow you to play with feature models
    feels need to use things like SAT solvers for these feature models
    e.g. building Linux Kernel by hand difficult, uses a SAT solver to check Linux feature model for consistency..

Galois is building a RISC-V ISA feature model spec into their model using Haskell.
    Capture as much as possible in the type system.

Compliance - feature model can define something to be tested in detail
    - can auto generate testbenches
    - did the java virtual machine ISA like this
so feature model can be used to generate tools to use the formal models.

Also - generation of a sufficiently comprehensive test suite built from these models
    - easy to get low hanging fruit
    - Galois is focusing on security not on compliance

-these techniques don’t generate long test sequences (eg 25+ instructions)

- normally small test sequences - should be OK for RISC-V.
- RISC-V ISA does have corner cases which most likely might become difficult...

Galois have worked with Ian Clarke (Edinburgh and people in Cambridge (ARM , x86 work).

- tests can be added by statically reasoning about code binaries, and adding new tests

-More than just compliance testing: full verification testing of the designs.

Joe wants to build a DSL on the front end (Galois has a lot of experience in DSL)

- Needs to work on his program manager to get funding...

Allen: interesting, foundation is aware; ideally, we want test generation tools.

Ben (Galois): have run the coverage analysis tool on the compliance suite

it will tell you how well each bit of every source operand has been used.

Simon (Imperas): We have also done this

Ben & Simon will compare notes ; Imperas may add to hand-coded Compliance suite.

Joe: SSITh project

Product line engineering

Alloy : language to express these

Largest :l inux kernel SMT model behind the scenes

Bens work embeds features into type system GRIFT model

Embodied into the model itself

Can generate test that cover entire model (done for java)

Test gen from formal model has low hanging fruit

But long tests are very difficult, >20 inst

Usually pipelining and OOO take long traces to cover

Complement with formal model of the implementation

GRIFT is open source

See the results:tool output exercising of every bit of sources

Add coverage type?