embeddedworld2019
Exhibition&Conference
... it's a smarter world

Compliance Methodology and Initial Results for RISC-V ISA Implementations

Lee Moore, Simon Davidmann and Larry Lapides
Imperas Software Ltd.
Oxford, United Kingdom
moore@imperas.com

***Abstract*— For most instruction set architectures (ISAs), compliance to the ISA specification is a given. Since all the SoC designers license the RTL from a single source, of course the RTL complies with the ISA specification. Similarly, the processor IP vendors produce a tool chain to support their ISAs, so those will certainly comply with the ISA specification. Single source does at least provide for consistency, and compliance is not an issue, so software ecosystems flourish.**

With the new, open standard RISC-V ISA, the compliance situation is different. There is no single IP vendor, and complicating the issue is that many implementations will exploit the capability with the open ISA to add custom instructions or other optimizations.

Compliance testing therefore has become mission-critical for the RISC-V ecosystem. For other ISAs compliance testing has been done by the processor IP vendor, and as a result methodologies and tools for compliance testing have been kept internal, and are not readily available to the industry. This paper introduces the methodology being developed for compliance testing of RISC-V products.

The technical issues of determining compliance with the RISC-V ISA are introduced and discussed. These issues include providing a framework for development of additional tests, the development of the tests themselves and reference models. Further issues include how to enable users to target the tests at the particular combination of the RISC-V specification subsets that is being used. The questions of completeness and specification coverage are discussed. Use cases are examined, including testing compliance on various proprietary RTL designs, open source RTL designs, FPGAs, SoCs, ISS models and software tools, with issues experienced being explained.

Keywords—RISC-V, processor, compliance, ISS, testing, simulation

I. INTRODUCTION

The new RISC-V Instruction Set Architecture (ISA) [1] has a lot of momentum behind it, with a growing community now at over 150 members of the RISC-V Foundation. However, momentum is not success. To be successful, RISC-V processors need to be implemented and utilized in SoCs, and those SoCs need to be used in systems. One of the keys to ISA success, as shown by all other ISAs, is a strong ecosystem around the ISA, meaning a robust group of companies providing software tools, software IP, operating systems, etc.

For adoption in systems, and for the ecosystem to develop, devices need to comply with the ISA specification. Without compliance, there is fragmentation. Without compliance, the ecosystem cannot leverage its investment in RISC-V. Without compliance, systems companies have minimal choices in the hardware they use, and have to develop ecosystem products themselves, which is not an area of expertise typically for these companies.

Compliance to the specification is critical to ISA success, and yet compliance testing is a thankless task. It is exciting to add custom instructions to the processor. However, it is not nearly as exciting to tell the processor designers that they made a major mistake and are not compliant to the specification. Also, there are no best known methods for compliance in the industry, since previously all compliance testing was done internally by the processor IP vendor that owned the ISA.

Another issue comes from the open nature of the RISC-V community. In the community, the mode of operation is to have member companies contribute to the various working groups, including the compliance working group. However, the companies in the best position to contribute – the processor IP companies – do not want to or, from a practical perspective, cannot contribute to the compliance testing. This is because their compliance test suites are likely intertwined with their verification test suites. The verification test suites include information on how to test their processor implementations, so exposing those tests would expose proprietary information and differentiating features of their products.

In this paper, compliance is first discussed: what it is, and what it is not. This is followed by a review of the publicly available RISC-V compliance tests and framework, including discussions of how those tests have been developed, the quality and completeness of those tests and descriptions of key elements of the compliance framework. Results of using the compliance tests on RTL and on various silicon implementations and abstract models is presented.

II. What Is Compliance Testing

Compliance means that the device is working within the envelope of the specification. Put another way, compliance testing is a testing technique for validating whether or not the system being developed meets the prescribed standards.

Compliance testing is not design verification. Compliance testing is looking for missing registers, modes, instructions; not for bugs in RTL implementations.

Compliance tests have to be written in such a way that compliance (or non-compliance) is observable in a test signature. The signatures are published so that the user does not have to run a reference model and can compare the results of their target runs to the reference signature.

III. RISC-V Compliance Status

A. Compliance Overview

The RISC-V Foundation initiated a Compliance working group in June 2017. The first contributions, for testing a RV32I processor configuration, were donated in January 2018. There has been progress, however, as compliance is a thankless task, it always seems that there could be more progress. The compliance test suite is therefore a "work in progress".

There are two primary components to the compliance technology: the test suites themselves, and the framework for writing and using those test suites.

Each test suite focuses on a feature set of the RISC-V envelope. The initial focus has been on instructions and the user mode specification, i.e. RV32I, RV32IM, RV32IMC, RV64I, etc. Since the RISC-V platform specifications have not yet stabilized, there has been no publicly available work on compliance tests for the privilege specifications.

The framework includes make, bash and other scripts used to encapsulate compiler tools, linkers, simulators and targets as the Devices Under Test (DUTs). The framework also includes a simulator, essentially an Instruction Set Simulator (ISS), which serves as an example target and also generates reference signatures. The framework enables a user to run each test, have the target produce signatures and compare those signatures to saved golden reference signatures.

The current status of the test suites and the framework are freely available on a GitHub repository [2].

B. Test Suite Status

Currently there are twelve test suites checked into the GitHub repository. The status of some of the test suites is shown in Table 1.

A good question to ask is how to measure test suite quality. Some type of code coverage would seem a good starting point. However, code coverage as usually conceived for RTL or software verification is not applicable, as it is most often connected to the microarchitectural implementation. Imperas provides a code coverage tool as part of its commercial products that provide model coverage [3], however, this does not show how much of the specification is covered.

A fault simulation coverage tool developed by Imperas does provide instruction decode coverage analysis. This tool explores the decodes of the instructions and mutates the legal bits and detects that there is a test that stimulates and observes each bit. This tool is an add-on the ISS target simulator, and so runs quite fast, and provides other analysis including data coverage. In addition, this tool can be used to measure coverage of custom instructions.

To show how this can be used, the RV32I test suite can be analyzed. This test suite has 54 hand-coded tests, each of which averages 150 instructions. Decode coverage ranges from 0% on some instructions, such as fence instructions, to 100% for the majority of the instructions. Generating the decode coverage data went quickly, as the Fault Simulation Coverage Tool ran 478,390 simulations in 308 seconds.

C. ISS for Compliance Framework

Writing the tests is a significant task. The test developer needs to have detailed knowledge of the operation of the instructions, and needs to have tools for analyzing and debugging the tests. An instruction accurate simulator with appropriate tools has been used by Imperas for writing tests. Any simulator used for this purpose needs to fully implement the specification, needs full configurability to all options of the specification, needs tracing capabilities to observe all resources affected by all instructions, needs access to comprehensive debug, and needs to be able to be encapsulated and controlled in an external environment.

A block diagram of the use model for the Imperas riscvOVPsim simulator used for test development is shown in Fig. 1. This simulator and associated RISC-V processor model is available free, with no license keys or license management, from the RISC-V compliance suite GitHub repository[2]. The configurable RISC-V model, which supports the 2.2, 2.3, 1.10 and 1.11 versions of the RISC-V Foundation specification, is built using the Open Virtual Platforms (OVP) [4] VMI APIs for processor model development. This model is available both as binary and as source code, distributed under the Apache 2.0 open source license.

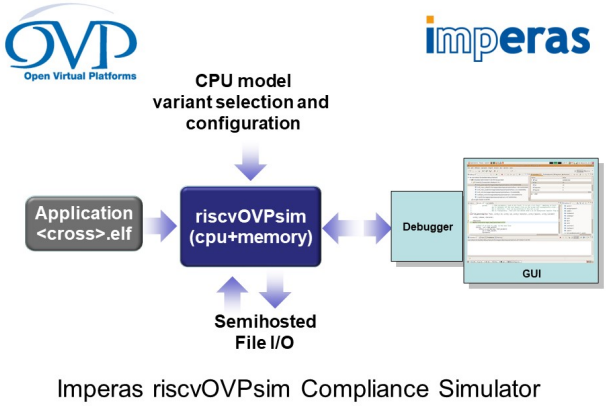

Fig. 1. riscvOVPsim simulator for compliance test development.

The riscvOVPsim simulator is an instruction accurate simulator using a Just-In-Time (JIT) simulation engine. Simulation performance is over 1,000 million instructions per second. The simulator connects to a GDB debugger, has the required trace capabilities, can generate the required compliance signatures and can be encapsulated by the Imperas Fault Simulation Coverage Tool for test analysis.

IV. Compliance Results

The compliance test suites have been used to test compliance of proprietary RTL, open source RTL, FPGAs, SoCs and other simulators. The process is to load the test .elf file, run the simulation, write the signature and compare the signature to the golden reference signature. Some work had to be done to enable RTL simulators and actual hardware to be used as the DUT.

With every DUT there were areas of non-compliance. Typical issues included missing registers, missing instructions, floating point mode change issues and PMP implementation issues. The floating-point mode change issue is more a specification issue, as it is likely the specification will be changed to allow multiple implementation options. Obviously, the missing registers and instructions are a problem. PMP issues are also critical, since incorrect implementation of PMP could result in a security hole for the device.

V. Recommendations for Usage

The compliance tests are available for anyone to use, for checking RTL, silicon or simulators. Here are the basic steps for usage:

1. Clone or fork the compliance tools from the RISC-V compliance GitHub repository
2. Encapsulate the DUT target in the cloned version of the test suite
3. Get the initial RV32I test suite running and generating signatures
4. Compare those signatures to the golden references
5. Add other test suites as available and appropriate for the target DUT
6. Run compliance checking whenever the DUT is modified (should be added to automated regression tests)
7. Update the clone/fork when there are changes to the RISC-V compliance GitHub repository

VI. Summary

Compliance is critical for RISC-V success, however, it is still a work in progress. There has been significant progress in the last year, and now there are test suites available for compliance checking, as well as a framework to enable RISC-V users to run the test suites, compare to golden reference signatures and develop additional test suites.

While there is ongoing work on adding compliance test suites to support the A, F and D user mode instructions, work needs to start on compliance test suites for privilege modes and platforms. Also, new RISC-V ISA subsets, such as vector instructions, bit manipulation and DSP should not be ratified as part of the specification until their compliance suites are being developed.

Acknowledgment

Thanks to all the voluntary members of the RISC-V Foundations' Compliance working group for their efforts.

References

[1] The RISC-V ISA specification is available here: https://riscv.org/specifications/.

[2] https://github.com/riscv/riscv-compliance/

[3] Imperas product web page: http://www.imperas.com/products

[4] Open Virtual Platforms (OVP) Library: http://www.ovpworld.org/library/wikka.php?wakka=Library

TABLE I. Test Suite Status.

Test Suite	Development History	Number of Tests	Status
RV32I	Originally developed by Codasip; updated by Imperas to improve coverage	54	Tests use correct style/macros; excellent coverage of most instructions (no coverage of fence, scall, sbreak, pseudo and csr instructions)
RV32IM	Developed by Imperas	7	Tests use correct style/macros; excellent coverage of most instructions
RV32IMC	Developed by Imperas	24	Tests use correct style/macros
RV64I	Developed by Imperas	8	Tests use correct style/macros
RV64IM	Developed by Imperas	3	Tests use correct style/macros
RV32UI	From /github/riscv-tests (academic origin; ported by Imperas)	?	Poor coverage