PhD Research Proposal:
"Automatic Emulation From Formal Architecture Descriptions"

Ferdia McKeogh

1 Introduction

Current instruction set simulators (ISS) require manually implementing each new instruction set architecture (ISA). This process is tedious, time consuming and error-prone. The Sail[1] language has been used to describe the semantics of several ISAs, including Armv8, RISC-V, and user-mode x86, for the purposes of formal verification. This proposal will improve on the state of the art by creating a compiler that uses these formal descriptions to produce performant ISSs, automatically. Additionally it will support live-reloading in order to propagate changes in the Sail ISA description to the emulated system without restarting. This would allow for highly rapid, interactive testing of new instructions.

This, by itself, would only emulate the ISA; emulating a full system requires combining multiple cores alongside memory management units, peripherals such as I^2C, SPI, serial ports, timers, GPIO (with interrupts), and security hardware. Therefore a language for describing the wider system to more accurately simulate more complex systems will also be designed. This language will let developers describe at a high level how components should be connected in the system. These components will be collected in a shared registry in order to create an ecosystem of composable units. The software interfaces for components will be specified so that developers may also write their own components to contribute to the ecosystem. Generating new emulated platforms from the description must be convenient and responsive in order to maximise impact on the development experience.

Finally, debugging infrastructure will be required. Firstly to debug and trace the ISS itself including standard tooling for stepping through instruction execution and setting breakpoints, but also viewing live information on the state of cache and compilation status without negatively impacting emulation performance. Hardware accelerated tracing is substantially faster than software tracing, but only exists for when the host and target ISAs are the same; cross-ISA hardware accelerated tracing will be investigated as part of this research.

2 Motivation

The primary motivation of this research is to improve tooling and enable faster development in mixed hardware and software teams. Both hardware and software teams can use the same emulated platform as a reference instead of requiring constant communication on any changes in the hardware or its requirements. Software teams will have access to a fast and accurate emulated environment to build and test software on and hardware teams will be able to make rapid changes from instructions in the ISA to the number and type of cores and their peripherals.

This research will also improve both software and hardware testing. For software, this allows for testing at much larger scales than previously possible. For example fuzz-testing is a useful methodology for large

distributed systems but fault injection on millions of physical devices would be difficult to perform, but very possible on millions of emulated instances. These tests would be more valuable than higher level simulations as they will all run real, unmodified firmware. For hardware testing once it has been physically manufactured the emulated platform can be used to assist in the validation: programs can be run both on the real hardware and the emulated platform and their behaviour or outputs compared.

This will be useful for embedded developers enabling the writing of software before hardware is available. Typically embedded software development requires expensive SEGGER[3] debuggers connected to the hardware (the maintenance of which places additional burden on hardware teams). Different developers may have different hardware versions and any modifications must be manually applied to each unit. By contrast, an emulated platform could have changes applied near instantly, shared across all developers to prevent inconsistencies, all while not requiring any physical hardware or tooling.

3 Challenges

Sail does not distinguish the semantics from the syntax of an instruction, although does describe both, therefore novel research will be required to determine how these two can be separated during compilation after parsing. The syntax and semantics are then used to generate C++ for the parsing and execution of a given instruction in the ISS.

Supporting hot reloading is key to the development experience, but will be a complex problem to solve. Introducing new instructions dynamically during execution will require careful management of different emulator components, such as flushing caches of JIT compiled sections or modifying the instruction parser.

Keeping overheads low to maximise scalability will be crucial to achieving the goal of using firmware extracted, unmodified, from off-the-shelf fitness tracking wristbands to successfully run one billion virtual instances of the fitness trackers.

4 Related Work

Captive[4] is a hardware-accelerated, cross-architecture hypervisor. GenSim[2] is a system for generating ISA simulators and works in conjuntion with Captive. The first section of work will involve creating a Sail frontend for GenSim, as well as modifications to both GenSim and Captive to support hot reloading.

References

[1] Alasdair Armstrong et al. “ISA Semantics for ARMv8-A, RISC-V, and CHERI-MIPS”. In: *Proc. 46th ACM SIGPLAN Symposium on Principles of Programming Languages*. Proc. ACM Program. Lang. 3, POPL, Article 71. Jan. 2019. DOI: 10.1145/3290384.

[2] *GenSim*. URL: https://gensim.org/home (visited on 05/14/2022).

[3] *J-Link Debug Probes by SEGGER – the Embedded Experts*. URL: https://www.segger.com/products/debug-probes/j-link/ (visited on 05/14/2022).

[4] Tom Spink, Harry Wagstaff, and Björn Franke. “Hardware-Accelerated Cross-Architecture Full-System Virtualization”. In: *ACM Transactions on Architecture and Code Optimization* 13.4 (Oct. 2016), 36:1–36:25. ISSN: 1544-3566. DOI: 10.1145/2996798. URL: https://doi.org/10.1145/2996798 (visited on 05/14/2022).