RISE

RISC-V Software Ecosystem

Fourth International Workshop on RISC-V for HPC: ISC ‘24

Advancing commercial software ecosystem readiness

May 2024

Dr. Oliver Perks

Accelerating the RISC-V Software Ecosystem

What is RISE?

An industry consortium under the Linux Foundation
Focus on software ecosystem readiness for RISC-V in Open Source

RISE Vs. RISC-V International (RVI)

RVI focuses on hardware standards
RISE is for practical development of software

RISE in not specifically HPC/AI/ML

New architectures need a lot of system software
Foundation to build specific stacks upon

RISE Members

Premier Members

General Members

RISE

Additional Information: https://riseproject.dev/

How RISE Works

Member Resources

2 FTE Commitment
- Or increased fee

Participate in working groups
- Vendor agnostic
- Shared responsibility
- Open + collaborative
- Prevent duplicated effort

Access to build farms + CI

Funded Projects

RFP to pay for development

Open calls for topics
- Not limited to members

Open to anyone
- Industry + Academia

Work upstreamed
- Whole community benefits

RISE Investment in the Ecosystem

Working directly with upstream open source projects to improve RISC-V software ecosystem.

WG	Funded Projects
System Libraries	**Multimedia Enablement, libjpeg-Turbo**
Toolchain	**Rust Tier-1 Support for RV64 Linux;** GCC Improvements
Dev & Infra/Toolchain	**LLVM CI Improvements**
Simulators/ Emulators	**TCG Translation for RVV Instructions**
Language Runtimes	**Accelerate Go Runtimes**
Debug and Profiling	OpenOCD Upstreaming

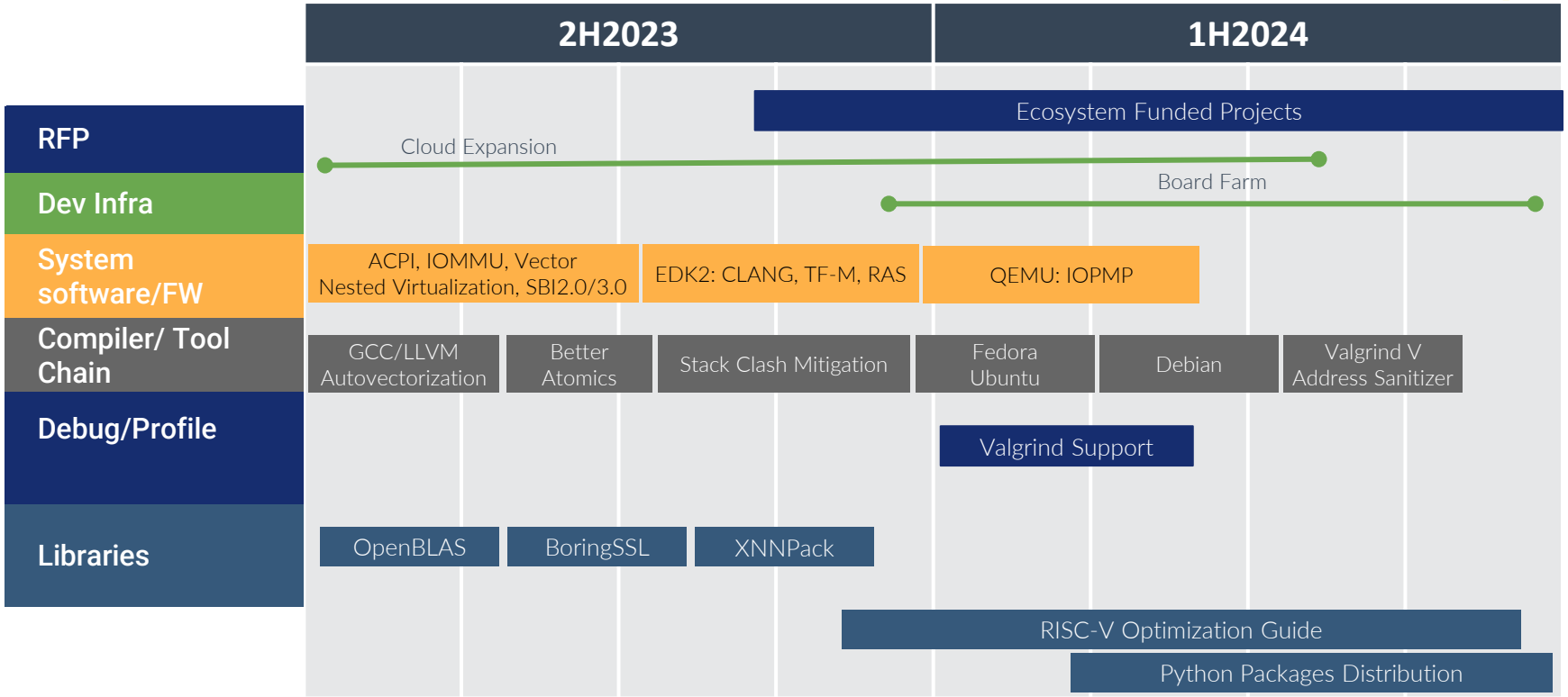
* Bolded are projects already in flight with contracts in place.

RISE Members Contribution

Coordination and collaboration among the RISE members is across an array of software areas to deliver high performance implementations for RISC-V.

Compilers & Toolchains	Autovec, shadow stacks, benchmarking, large code model
System Libraries	Boringssl, libflac, libdav1d
Kernel & Virtualization	CI farm
Language Runtimes	RISC-V Optimization Guide Published; Python Packages Distribution Java, Go, V8 and Node.js improvements, esp CI and ISA support;
Debug & Profiling Tools	Valgrind support
Simulator/Emulators	IOPMP in QEMU
System Software/FW	UEFI, ACPI, TianoCore, EDK2
Dev & Infra	Linux Kernel CI, GCC CI, OpenJDK CI, GCC Fuzz CI

Accomplishments 2024



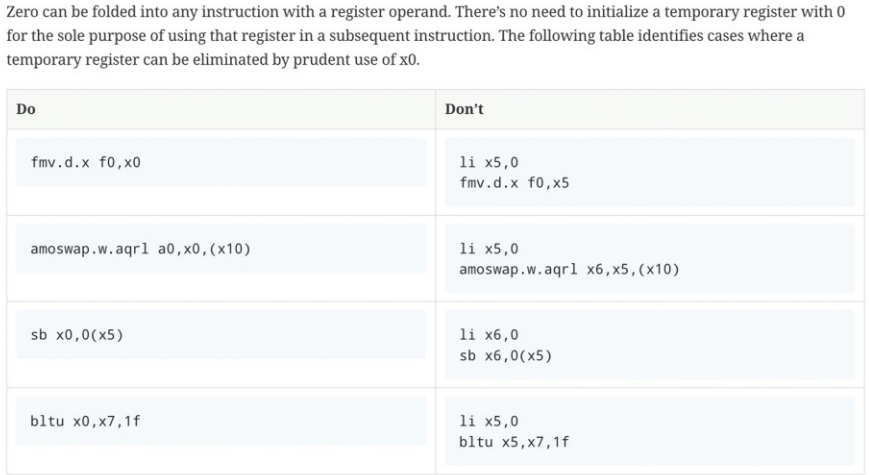
RISE

RISC-V Optimization Guide

https://riscv-optimization-guide.riseproject.dev/

Vendor agnostic porting and optimization guide
- Does not cover CPU specific microarchitecture

Best practices for high performance RISC-V cores
- Including assembly code examples



Zero can be folded into any instruction with a register operand. There's no need to initialize a temporary register with 0 for the sole purpose of using that register in a subsequent instruction. The following table identifies cases where a temporary register can be eliminated by prudent use of x0.

Do	Don't
`fmv.d.x f0,x0`	`li x5,0` `fmv.d.x f0,x5`
`amoswap.w.aqrl a0,x0,(x10)`	`li x5,0` `amoswap.w.aqrl x6,x5,(x10)`
`sb x0,0(x5)`	`li x6,0` `sb x6,0(x5)`
`bltu x0,x7,1f`	`li x5,0` `bltu x5,x7,1f`

RISE

RFP006: RISC-V LLVM Testing Improvements

Limited testing by running generated code for RISC-V

- On a native or emulated environment

Improve automated testing of the larger LLVM test suite

- Including testing for projects like LLDB

Use the RISE build farm or self hosted

Essential for the robustness of the whole ecosystem

- But not exactly ‘exciting’ or ‘novel research’

Getting Involved with RISE

Using RISE output	Git repos - https://gitlab.com/riseproject PIP Repos - https://gitlab.com/riseproject/python/wheel_builder Online documentation - https://wiki.riseproject.dev/ Optimization guide - https://riscv-optimization-guide.riseproject.dev
Join RISE	Apply for membership (€40k + 2 FTE) Governing board currently full
Apply for Funding (RFP)	RFP form (QR code)

RFP Form