



The RISE Project: Advancing RISC-V Software

Ludovic Henry, RISE TSC
Nathan Egge, RISE TSC

RVI Summit EU, June 2026

Collaboration: RISE (RISC-V Software Ecosystem)



Raise

RISC-V platform quality



Ensure

RISC-V is top tier for tools
and libraries



Align

Ecosystem partners efforts for
adoption

How: Working Upstream, Transparently

Open and Transparent Organization

Coordinating contributions

Bringing open source communities together

RISE Members

Premier Members



tenstorrent



General Members



Additional Information: <https://riseproject.dev/>

RISE Software Focus Areas

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

Enablement / Optimization	Compilers & Toolchains: LLVM, GCC, GLIBC
	System Libraries: FFmpeg, OpenBLAS, oneDAL, XNNPACK, oneDNN
	Language Runtimes: Python, Java/OpenJDK, Go, JavaScript/V8, WebAssembly, Rust
Platform	Kernel & Virtualization: Linux, Android
	Linux Distro: Ubuntu, Debian, RedHat, Fedora, RockyLinux, AlmaLinux, Gentoo
	Firmware: UEFI, U-Boot, Coreboot, TF-M
	Security Software: Secure Root-of-Trust, Confidential Compute
Developer Tooling	Simulator & Emulators: QEMU, SPIKE
	Developer Infrastructure: Build Farm, Board Farm, Developer Tools
AI/Machine Learning	PyTorch, TensorFlow, TFLite, Llama.cpp, IREE
MCU Software	RTOS (Zephyr, NuttX, FreeRTOS, etc.), Application APIs (DSP, NN)

RFP Highlighted Results



LLVM SPEC optimization

Reduces execution time by 15% on SPEC CPU® 2017-based benchmark on SpacemiT-X60



Go

Compiler Optimization (RVV, bitmanit), math and crypto intrinsics. [Releases upstream](#) since Go 1.21



Python Packaging

Building packages for commonly used projects
<https://gitlab.com/riseproject/python/>



IREE

Enabling and Optimizing models E2E for RVV



Llama.cpp

Complete support and optimization for RVV with Vector Length Agnostics optimizations



QEMU TCG

Enhanced performance for vector (V) and crypto (Zvk) extensions; faster emulation and CI/CD. Achieved 2x faster memory operations and halved AOSP boot time.



Rust

On track to meet full Tier-1 requirements



OpenOCD

RISC-V support upstreaming



LLVM CI

QEMU-based testing to support profiles and optimized build configurations.



PyTorch

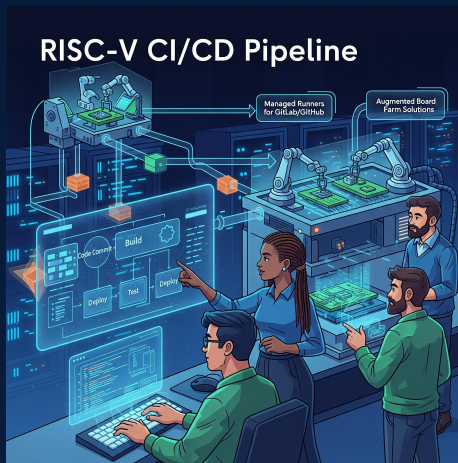
Basic Aten Ops optimization with RVV support

Engaging Developers



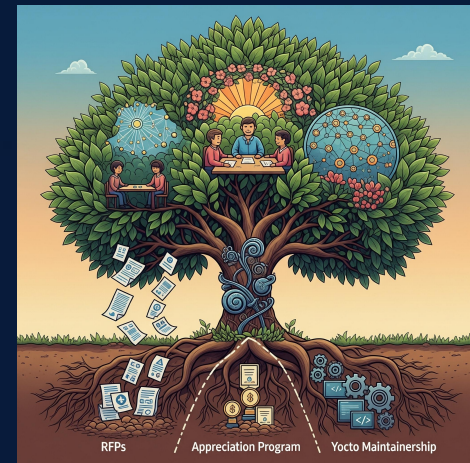
Development Resources

- RISC-V Optimization Guide
- RISE Case Study: Adding RVV 1.0 to dav1d AV1 decoder
- Gentoo Developer Image
- Python packages



Automated testing on CI

- Build farm
- Board farm
- Github/Gitlab runners



Investing in communities

- Dev Appreciation Program
- RFPs
- Yocto Project maintainer

Developer Infrastructure

GitHub Runners

Easy hardware access for any GitHub project

- Install the [GitHub App](#)
- Native RISC-V runners
`runs-on: ubuntu-24.04-riscv`
- 24,000 jobs, 700k minutes, 116 organizations, 264 repositories
- Llama.cpp, PyTorch, Numpy, k0s, alibaba/zvec, and many more...
- Partnership with CNCF
- Scaleway EM-RV1, Spacemit K3, Alibaba A210, MILK-V Jupyter, ...

Build Farm

- Integration to Kernel, GCC, LLVM CI

Board Farm

Connect available HW to SW communities

- Partnering with RVI Lab Ecosystem WG
- Hosting at OSU-OSL, Scaleway
- Starting with donation from Alibaba
- Reach out to host **your** hardware

Developer Appreciation Program

Rewarding Developers who port Software to RISC-V

- 500€ for Small contributions, 3000€ for Large contributions
- File an issue: <https://github.com/rise-dev-appreciation>

Rewarded contributions

- Gem5 - Support for H Extension and SVNAPOT - [#5](#) [#12](#)
- Delve - A Go debugger - [#8](#)
- VOLK 🦀 - Vector-Optimized library used by GNU Radio project - [#11](#)
- syscall_intercept - User-space interception of system calls - [#9](#)
- Lightning - Template based JIT library - [#10](#)
- SIMD Everywhere - Portable SIMD library - [#4](#)
- MAMBO - Dynamic binary instrumentation and modification - [#3](#)

Yocto Project Maintainer

Project Goals

- Enable RISC-V as a fully supported platform for Yocto Project / OpenEmbedded
- Optimize the meta-riscv layer to comply with Yocto Project Compatible guidelines
- Incorporate RISC-V hardware into the Yocto Project Autobuilder
- Maintain a presence on the Yocto Project bug triage team
- Work directly with RISC-V International to set priorities

Special thanks to:



Developer Resources

Project Goals

- Turn-key images to jump-start developers with latest toolchains:
 - gcc-16.1, clang-21.1.8, rust-1.95, glibc-2.43, cmake-4.3.2, etc.
- Bespoke CFLAGS for each board
- Available today for:
 - [Canaan K230](#), [BPI F3 / ROMA II](#), [Orange Pi RV2](#), [K3 CoM260 / Pico ITX](#)
- Talk to us about adding your devboard!
- See the [crossdev-stages](#) project

RISC-V Optimization Guide

- riscv-optimization-guide.riseproject.dev

Python Packaging

- <https://gitlab.com/riseproject/python/>
- 79 Python projects made available on RISC-V, and counting
- Upstreaming is the goal

How RISE is Contributing

Foster Public Open Source Standard Collaboration

Establish Developer Infrastructure

Activate Broader Developer Community

Becoming a member

Request For Proposals (RFPs)

Developer Appreciation Program

**How you can
get involved**

Shared Vision for Future



Building Stronger RISC-V Software Ecosystem Together