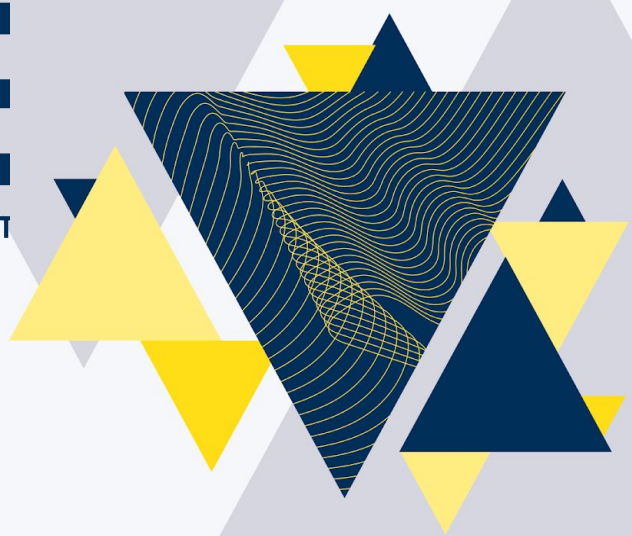
RISE

RISC-V Software Ecosystem

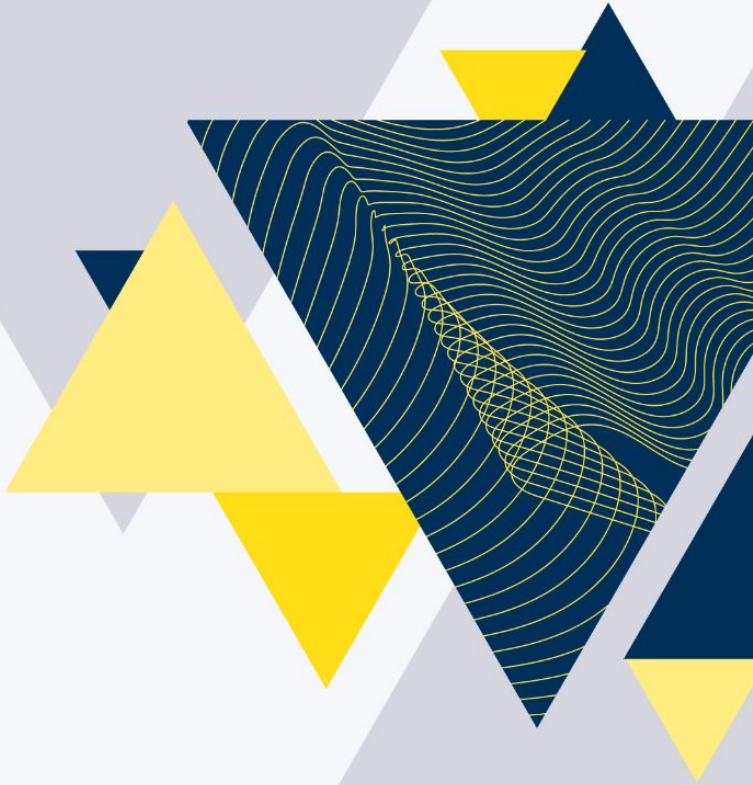


End of 2024
Ecosystem Update

Agenda

- RISE High Level Overview
- RISE Workgroup 2024 Achievements
- Software Ecosystem Collaborations
- RISE Partnership with RISC-V International (Andrea Gallo)
- Q&A

RISE Overview



RISE (RISC-V Software Ecosystem) Objectives

Improve RISC-V platform quality to match other architectures

Advance the RISC-V software ecosystem

How: Working Upstream, Transparently

Coordinate with existing communities and share resources

Open to project submissions from anyone

Prioritize projects based on community input

RISE Members

Premier Members

Google
intel
Qualcomm
MEDIATEK
ANDES TECHNOLOGY
達摩院 DAMO ACADEMY
Red Hat
Rivos
SiFive
VENTANA
SAMSUNG
NVIDIA
Imagination

General Members

MICROCHIP
Canonical
ISCAS 中国科学院软件研究所 Institute of Software Chinese Academy of Sciences
AKEANA
BOSC 北京开源芯片研究院 BEIJING INSTITUTE OF OPEN SOURCE CHIP
AMD
SPACEMIT 进迭时空
tenstorrent
ByteDance

RISE Governance Organization Chart

Governing Board

Amber Huffman
Chair

Barna Ibrahim
Vice Chair

Larry Wikelius
Treasurer

Technical Steering

Nathan Egge

Andrew de los Reyes

Program Management

Michelle Martineau

Marketing

Jeffrey Osier-Mixon

Melissa Daniels

Jen Li

RISE

RISE Working Group Leads

Compilers & Toolchains

Jeff Law
Ventana Micro

Debug & Profiling

Xaio Wang
Intel

Language Runtimes

Ludovic Henry
Rivos

Developer infrastructure

Paul Walmsley
SiFive

Distro & Integration

Brian Harrington
RedHat

Kernel

Anup Patel
Ventana Micro

Security Software

Robin Randhawa
SiFive

Firmware

Sunil V L
Ventana Micro

Simulator/Emulators

Daniel Barboza
Ventana Micro

System Libraries

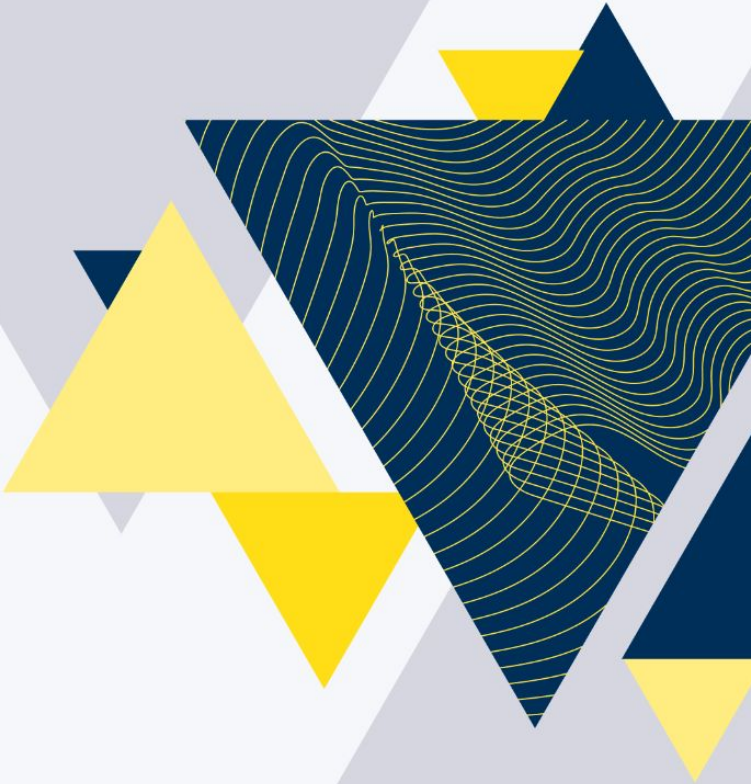
Nathan Egge
Google

RISE Visit our RISE Confluence for Working Group Details

RISE Working Groups

Compilers & Toolchains	LLVM, GCC, GLIBC
System Libraries	FFmpeg, OpenBLAS, OneDAL, XNNpack
Kernel & Virtualization	Linux, Android
Language Runtimes	Python, Java/OpenJDK, Go, Javascript, WebAssembly, Rust
Linux Distro Integration	Ubuntu, Debian, RedHat, Fedora, Alpine
Debug & Profiling Tools	Performance Profiles, DynamoRIO, Valgrind
Simulator/Emulators	QEMU, SPIKE
Software Security	Secure Root-of-Trust, Confidential Compute
System Firmware	UEFI, U-Boot, Coreboot, TF-M
Developer Infrastructure	Build Farm, Board Farm, Developer Tools

RISE Workgroup 2024 Achievements



Libraries and Middleware

Language Runtimes	Developer Infrastructure	System Libraries
• **Java** versions 17, 21, 22, 23, and 24 are now available on RISC-V. Better support for RVA23 extensions • **Python** Packaging and Distributing foundational Python packages (Numpy, Scipy, cmake, ninja, etc.). Engaged with AlmaLinux, for packaging support upstream, funding with RP011 • **Go** - Funded project to accelerate Go runtime in RISC-V in RP001. Landed improvements for Bitmanip, Vector, Vector crypto extensions (sha256, sha512) • **Javascript -** Added support to V8 and SpiderMonkey • **Rust** - Engaged with RISC-V community to bring to Tier 1 level support, funded in RP004 • Published RISC-V Optimization Guide	**Build Farm** - 600% growth since Nov 2023 in machine cycles dedicated to RISC-V • Seven active CI projects: Linux Kernel, gcc pre-commit, gcc post-commit, glibc pre-commit, gcc fuzzing, LLVM fuzzing, Python modules • New RISC-V projects served: LLVM, Python, glibc, and OpenJDK • Funded additional LLVM CI / CD in RP006 • Expanded support to RISC-V GCC and Linux kernel communities • Discovered ~100 LLVM, gcc bugs with toolchain fuzzers on RISE infrastructure	• Added RISC-V Vector optimizations to system libraries: bionic, dav1d, zlib-ng, XNNPACK • Identified toolchain issues blocking RISC-V, fixed bugs upstream or filed open issues • Published best practice guide on how to integrate RISC-V and RVV SIMD into existing library including CI/CD testing with QEMU • **RP005** QEMU TCG improvements: As of 9.2 qemu now 2x-3x faster RVV loads and stores! • Linux Developer Images for the Canaan K230 and Banana PI BPI-F3 with up-to-date tooling to support upstream RISC-V optimization • RISC-V Multimedia Instruction Wishlist - Several new extension proposed now in RVI fast track processes: zip/unzip, dotproduct, vector abs-diff
What's Next: • **Java, Go -** Ongoing improvements for better support of extensions for RVA23 • **Python** - Upstream packaging tools support RISC-V • **WebAssembly -** Support for WASM SIMD • **Kubernetes -** Upstream support for RISC-V	**What's Next:** • Build Farm: Enable more projects as needed • Board Farm: Enable hardware testing for RISE Build Farm projects • Automate Gentoo-based developer tools image for community use, to contain recent builds of RISC-V toolchains, kernel, and other key tools	**What's Next:** • Performance testing framework for RISC-V code: auto-vectorization, intrinsics, or SIMD • Collaborate with RVI devboard SIG to bring Gentoo Developer Images to more platforms • Continue to focus on high-profile, in-demand projects based on member priorities

Developer Tools

Simulators/Emulators	Linux Distro Integration	Debug & Profiling
QEMU ● ACPI support for PLIC ● ACPI SRAT/SLIT (NUMA) ● SMBIOS support for RISC-V ● RVA22 Profile Support ● RISC-V IOMMU support (PCI device)	**The Distro Integration** group meetings are public and have discussions about software needs common to all major Linux distros, Android, and etc. Recent discussions: ● Ensuring a consistent boot flow ● Responsibility for code updates below the OS	**Perf Tools** ● Event discovery via json (for Supervisor Counter Delegation ext), development done, upstreaming WIP ● Support for Control Transfer Records extension development done. Upstreaming WIP **Dynamic binary Instrumentation** ● DynamoRIO basic RVV support (code/decode) is upstreamed ● DynamoRIO dcachesim and drcov clients support get upstreamed **Tracing** ● BFP JIT optimization with Zba extension. (Upstreamed)
What's next: ● RVA23 support ● Further IOMMU enhancements ● P extension support ● World Guard support and others	**What's next:** ● Secure boot signing keys	**What's next:** **Debugging:** ● GDB support for vector register dumping, FP16, BF16, CFI, pointer masking ● FW/Kernel/Debugger support for HW breakpoint/watchpoints **Linux Perf:** ● Drive community patch review for CTR, PMU Counter delegation **DynamoRIO:** RVV support **Valgrind:** Cleanup, fencei, NaN-box checking, B/V extension support **eBPF:** Support to probe mem watchpoint

Platform

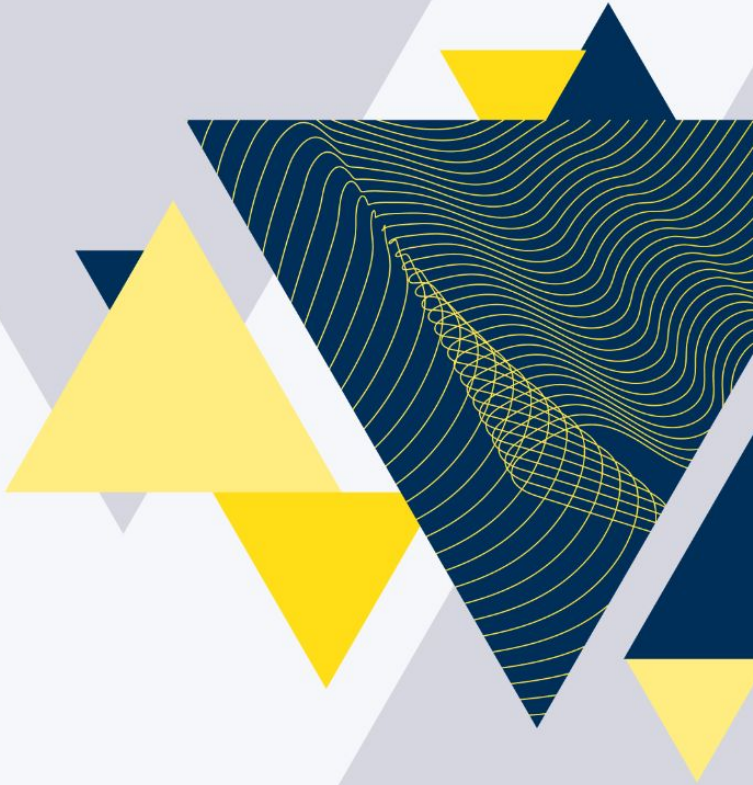
Kernel and Virtualization	System Firmware	Compilers & Toolchains

Linux Kernel (upstreamed)
• AIA drivers with DT and ACPI support
• ACPI - LPI and CPPC support
• IOMMU driver with DT support
• SBI v2.0 - Steal time, System suspend, Debug console, and PMU snapshot
 	
KVM (upstreamed)
• KVM self tests
• KVM unit tests
• AIA in-kernel irqchip with IMSIC guest file support
• KVM host nested acceleration
• Virtualize SBI v2.0 - Steal time, System suspend, Debug console, and PMU support
	• Svpbmt extension support on EDK2.
	• Coreboot support for Sifive Unmatched.
	• Native/hosted debug support in opensbi.
	• Domain context switch support in opensbi.
	• librpmi reference implementation
	• RPMI/MPXY support in OpenSBI
	• PoC / demo ready for
	  • DynamicTablesPkg for RISC-V in EDK2;
	  • StandaloneMMPkg for RISC-V in EDK2;
	  • OP-TEE for RISC-V
		GCC
		• If-conversion and Zicond support (gcc-14)
		• Basic RVV Autovectorization support (gcc-14)
		• Optimization of CRC using table lookups or clmul (gcc-15)
		• Stack clash protection (gcc-15)
		• Ceil/Round improvements, Zfa support (gcc-15)
		• Constant Synthesis Improvements (gcc-15)
		• Performant autovectorization support (gcc-15)
		
		LLVM
		• Zfa support (llvm-18)
		• Constant synthesis improvements (llvm-19)
		• Improved code generation for large VLEN uarchs (llvm-19)
		• Shadow Stacks (llvm-19)
		• Shrink Wrapping (llvm-20)
		• Stack clash protection (llvm-20)

What's next: • KVM irqbypass support • IOMMU ACPI RIMT support • SBI v3.0 support in Kernel and KVM • CoVE support Host & Guest • CBQRI & Ssqosid in Kernel and KVM	**What's next:** • Librpmi improvements • OP-TEE • Adherence to BRS requirements in EDK2	What's Next: • Vector mem* and str* in glibc • Shrink-wrapping (LLVM) • Permutation improvements (GCC) • Permutation elimination (GCC, LLVM) • xz vectorization (GCC, LLVM)

RISE

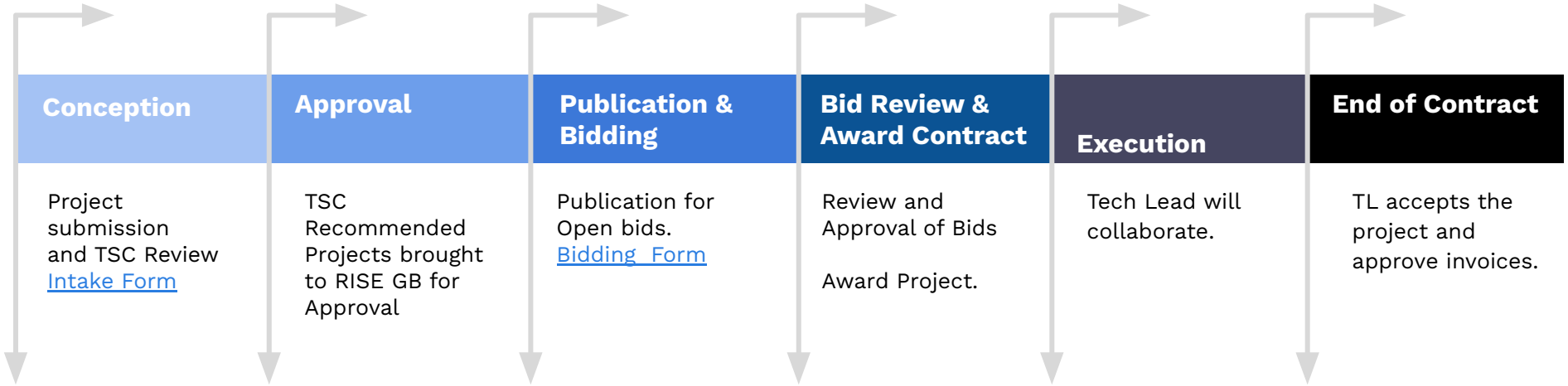
Software Ecosystem Collaborations



RISE RFPs (Request for Proposal)

Goal: Strengthen the RISC-V software ecosystem by supporting upstream projects and addressing priorities through an open RFP process.

Transparent Process



RISE

RFP Highlighted Results

- **Go** - Enhanced Go Runtime performance with RISC-V Bitmanip, Vector, Vector Crypto extension support, optimizing critical libraries and resolving unaligned memory access issues.
- **Rust -** The Rust RISC-V Linux target now meets Tier-1 requirements without host tools, passing all compiler tests on qemu-system-riscv64. With maintainers in place, it is on track toward full Tier-1 support as silicon improves.
- **QEMU TCG -** Enhance QEMU performance for vector (V) and crypto (Zvk) extensions with efficient TCGOps, boosting OSS development via faster emulation and CI/CD. Achieved 2x faster memory operations and halved AOSP boot time.
- **OpenOCD -** RISC-V OpenOCD upstreaming. Structured patch integration, thorough testing, and repository archival to streamline future development and distribution.
- **LLVM CI** for RISC-V, Leveraged QEMU-based testing to support profiles and optimized build configurations, delivering faster feedback and enhancing reliability and efficiency in RISC-V toolchain development.
- **Compiler Spec optimization -** Two projects aimed at improving LLVM and GCC for RISC-V, addressing performance gaps versus AArch64 SPEC2017 benchmarks. This led to reduced inefficiencies and closed architectural gaps.

RISE

RISE Developer Appreciation Program

Purpose: Recognize & support developers driving RISC-V adoption.

Goal: Make RISC-V the standard platform for developers.

Focus on Open Source:

- Contributions to be open-source and publicly available
- Aims to enable RISC-V integration over pure performance optimization

How to Participate:

- Port, test, release project on RISC-V
- Submit via the Developer Appreciation GitHub

RISE
RISC-V Software Ecosystem

Hardware needs Software which needs Hardware

Andrea Gallo

Hardware needs Software which needs Hardware

Hardware Needs More Software

- Software Enablement
  - Cannot do much without!!!
- Reference Workloads
  - Identify gaps and performance bottlenecks in real-life use cases
- Reference Stacks
  - Critical for the Developer Experience
  - Accelerate time-to-value add for device makers, avoid wasting months just to get a baseline up and running
- Certification
  - IP / pre- and on-silicon test suites
  - Platform test suites
  - Models

Software Needs Better Hardware

- Better Performance
  - AI/ML
  - Media codecs
  - Real Time
- Better Security
  - Pointer Masking, Shadow Stack, Landing Pads, Control Flow Integrity
- Standardization
  - Profiles and Platforms to run binary OS distros on compliant devices
  - Avoid unnecessary fragmentation

Golden example: new vector instructions for media

Ongoing optimisation of media codecs*

- dav1d, x264, ffmpeg, XNNPACK, etc.

Identify bottlenecks

- Vector transpose, absolute difference, zero extended move to vector register

Estimate speed-up

- Evaluate the contribution of each operation to the overall performance

Life cycle

- Explain the product use cases and gap vs other ISAs
- Propose a new set of media extensions
- Identify resources and build a plan for the RISC-V TSC to form a new TG

(*) findings reported by Nathan Egge from Google, Punit Agrawal and others from ByteDance
https://lf-rise.atlassian.net/wiki/spaces/HOME/pages/8588516/RISCV64+new+vector+instructions+requirements+for+video+multimedia

Improving ***both*** perf and security!

Performance

Quality-of-Service (QoS) Identifiers, Obviating Memory-Management Instructions after Marking PTEs Valid

Capacity and Bandwidth QoS Register Interface

BF16 Extensions

B Standard Extension for Bit Manipulation Instructions

Smcdeleg, Indirect CSR Access

Security

Priv 1.13, Pointer Masking

Double Trap, Resumable Non-Maskable Interrupts

Shadow Stacks and Landing Pads

Zaamo and Zalrsc Extensions, Byte and Halfword Atomic Memory Operations

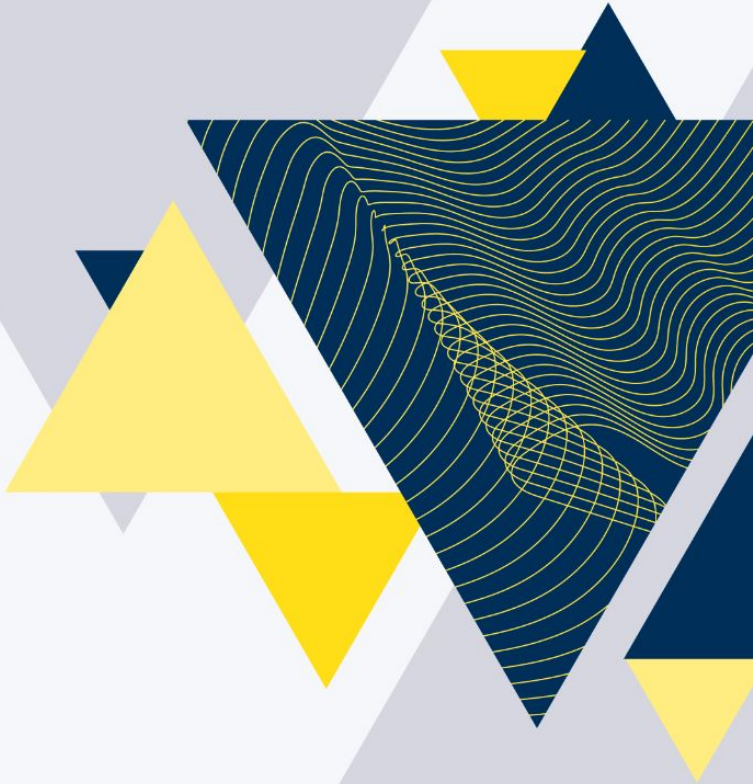
May-Be-Operations

Supervisor Binary Interface Specification

Functionality

RVA23, RVB23, RERI Architecture Specification, Functional Fixed Hardware Specification, E-Trace Encapsulation, N-Trace, Trace Control Interface, Trace Connectors

Q&A



How RISE is Contributing

- Foster Public Open Source Standard Collaboration
- Establish Developer Infrastructure
- Activate Broader Developer Community

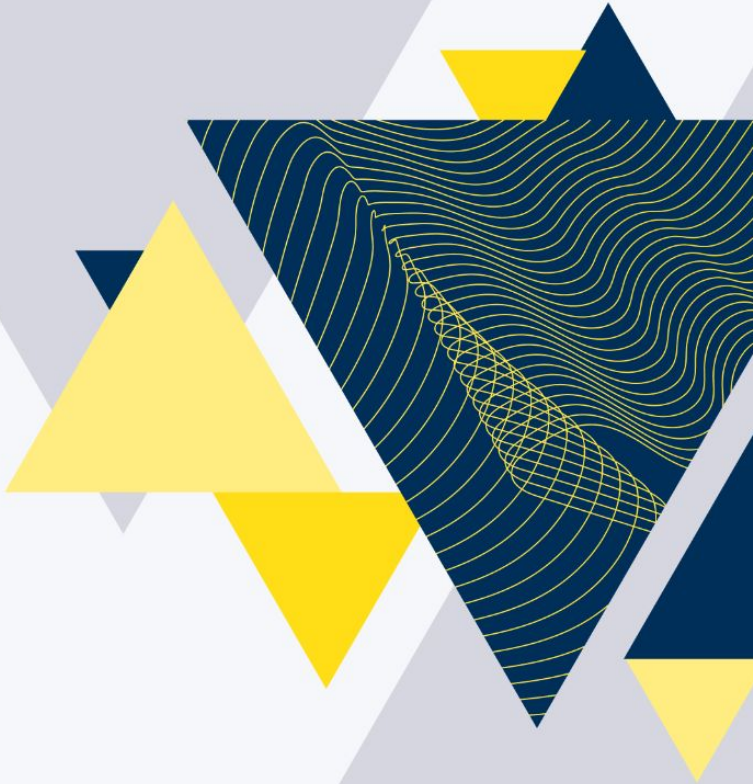
- Becoming a member
- Request For Proposals (RFPs)
- Developers Incentive Program

How you can get involved

https://riseproject.dev/

RISE

Thank You!



RISE Governing Board

Organization	Voting Rep	Alternate Voting
Google	Tim Kilbourn	Lars Bergstrom
Intel	Mark Skarpness	Gary Martz
Qualcomm	Larry Wikelius	
Rivos	Aaron Durbin	Andrew de los Reyes
Ventana	Travis Lanier	Marc Canel
SiFive	John Ronco	Paul Walmsley
Red Hat	Steve Wanless	Jeffrey Osier-Mixon
T-Head (Alibaba)	Dr. Jing Yang	
Andes	Charlie Hong-Men Su	Rich Chuang
Mediatek	Tom Kao	Hunglin Hsu
Samsung	Daniel Park	Dwarkaprasad Dayama
Nvidia	Amit Pabalkar	Vikram Sethi
Imagination	Matthew Bubis	Chris Smith