Overview

- RISC-V Overview
- Why People Use RISC-V
- RT-Thread & RISC-V
- Portability
- Embedded & Real Time ISA Extensions
- SW ecosystem
The definition of open computing is RISC-V

RISC-V is the most prolific and open Instruction Set Architecture in history

- RISC-V is inevitable
- RISC-V enables the best processors
- RISC-V is rapidly building the strongest ecosystem
10s of Billions of RISC-V cores deployed for profit!
Open Source HW or Open Standard?

- We are officially an Open Standard HW ISA Architecture
- We are an Open Standard that works heavily with Open Source upstream projects (LINUX, GCC, LLVM, etc.)
- We don’t do reference implementations
- Our work product are specifications with support from Golden Models and Basic Tests
Unlike Open Source Software ...

- Proprietary Custom Extensions are Encouraged and Welcome
- Products don’t just include the ISA, they can do custom implementations/extensions of/to the ISA
- Copyleft is a non-sequitur
- No restrictions on how the specification can be used
  - Only restrictions on branding
Why RISC-V?

- Flexibility
- Cost
- Ecosystem
- Our position in History
- EDA Renaissance
- Pride of Ownership
SoC ISA Balkanization

- Applications processor (usually ARM)
- Graphics processors
- Image processors
- Radio DSPs
- Audio DSPs
- Security processors
- Power-management processor
- > dozen ISAs on some SoCs – each with unique software stack

Why?
- Apps ISA too big&inflexible, poor base for other cores
- Apps ISA proprietary, cannot be used by others’ IP
- IP bought from different places, each proprietary ISA
- Engineers build home-grown ISA cores (don’t do this!)
RISC-V SOC Opportunities

- Move all cores on SoC to RISC-V-based cores
  - Applications processors
  - Graphics processors
  - Image processors
  - AI/ML accelerators
  - Radio DSPs
  - Audio DSPs
  - Security processors
  - Power-management processors
Fragmentation versus Diversity

**Fragmentation:**
Same thing done different ways

**Diversity:**
Solving different problems
How is RISC-V Avoiding Fragmentation?

Two powerful forces keep fragmentation at bay:

- **Users**: No one wants a repeat of vendor lock-in.

- **Software**: No one, not even nation state, can afford their own software stack. Upstream open-source projects only accept frozen/ratified Foundation standards.
Portability

- Unified common standard and a robust Software Ecosystem
- A robust economy around systems and software is reliant on portability
- Application & Runtime Software
  - Profiles
  - API (e.g. POSIX)
- Operation Systems
  - Platforms
  - Supervisor Execution Environment (SEE)
  - Profiles
Profiles

- Generational groups of instructions that work together and present a unified target for the software ecosystem
- Always include a base (a base are a group of state, instructions and behaviors fundamental to being RISC-V)
- Optionally include one or more extensions (like bases, extensions are a group of state, instructions, and behaviors). Extensions may be mandatory, optional, or non-profile options (n/a)
- Major and minor releases. We use Major releases to be targets for the SW ecosystem. Minor releases are checkpoints
- Initial implementations of profiles are likely available from members 12-48 months after a profile is ratified
- Current profiles (RVA) all targeted at RICHOS, General Purpose Multi-User computing. More to come.
- Profiles can live and be used forever. The implementers decide on adoption and lifetime
Managing Diversity

**Raw extensions**
- Base + standard extensions + custom extensions
- Full suite of options available for experimentation and specialized uses
- Massive combinatorial space of options

**ISA Profiles**
- Packages of ISA extensions for given domain
- Initial set: RVI20 (basic), RVA20/22/23 (application processor)
- Factor out common ISA combinations for use in platform standards

**Platform standards**
- Hardware/software standards for platforms (much more than just ISA)
- Initial focus OS-A platform for Unix-like OS (includes IOMMU, AIA, etc)
Profiles

<table>
<thead>
<tr>
<th>Bases</th>
<th>Cobalt</th>
<th>Copper</th>
<th>Future</th>
</tr>
</thead>
<tbody>
<tr>
<td>RV32I</td>
<td>RVA20</td>
<td>RVA23</td>
<td>More profile types:</td>
</tr>
<tr>
<td>RV64I</td>
<td>RVA22</td>
<td>RV64I</td>
<td>RVB, RVM</td>
</tr>
<tr>
<td>RV32I</td>
<td>RVA20U64</td>
<td>RV64I</td>
<td>RV128</td>
</tr>
<tr>
<td>RV64I</td>
<td>RVA22U64</td>
<td></td>
<td>Matrix Ops</td>
</tr>
<tr>
<td></td>
<td>RVA22S64</td>
<td></td>
<td>SPMP/IOPMP</td>
</tr>
<tr>
<td></td>
<td>RV64I</td>
<td></td>
<td>CFI</td>
</tr>
<tr>
<td></td>
<td>RV64I</td>
<td></td>
<td>CHERI</td>
</tr>
<tr>
<td></td>
<td>RV64I</td>
<td></td>
<td>GPU</td>
</tr>
</tbody>
</table>

- **Bases**: RV32I, RV64I
- **Cobalt**:
  - On years released, this only has a Mandatory Base
  - All other compatible ratified extensions are optional
  - Mul/Div
  - Atomics
  - Compressed
  - Float
  - Double
  - Priv 1.11
  - MemRegions
  - Fences
  - VirtualMem
  - Vector
  - Bitmanip
  - Scalar Crypto
  - BFloat16
  - FP16
  - Prv 1.12
  - Hypervisor
  - Cache
- **Copper**:
  - Android Features
  - Vector Crypto
  - PtrMasking
  - Zcompressed
  - Priv 1.13
- **Future**:
  - More profile types: RVB, RVM, RV128, Matrix Ops, SPMP/IOPMP, CFI, CHERI, GPU
  - 48/64 bit instructions

- Only a subset of extensions are listed above and it is not an exhaustive list
- Some extensions may be optional or non-profile in one profile and be mandatory in another
The platform definition consists of items with a ✔.
ISA Extensions Targeted at Embedded and Real Time

- Extensions: Fast Interrupts, Compressed Instructions (plus Zc*), FP in integer registers, Multiply without divide, Vector for embedded, S mode timer access, S mode w/o MMU, Wait on reservation set, Pause Hint, Physical Memory Protection, Scalar Crypto

- Profile Families: RVI, RVB, RVM

- Bases: RV32I, RV32E, RV64E
What We Have Done This Year

● **Ratified ISA Specifications**
  ○ Profiles, Code Size Reduction

● **Ratified Fast Tracks**
  ○ Counters, Total Store Ordering, RV32E/RV64E, Non Temporal Hints

● **Documentation**
  ○ Unpriv in Asciidoc, Asciidoc Priv draft

● **New Task Groups & Special Interest Groups**
  ○ Debug, Trace, and Performance Monitoring (DTPM) TG, Graphics SIG, RISC-V Common Software Interface (RVM-CSI) SIG, Vector (SIMD) SIG, Control Transfer Records TG
What’s Coming Soon?

● Profiles
  ○ RVA23, RVI23, RVB23, RVM23, BOD committee on comprehensive ACTs & Certification

● Platforms
  ○ Portability for Systems Software (PRS, BRS, Platform Security, SOC HW, Profiles)

● ISA
  ○ Advanced Interrupts, QOS Register Interface, Control Transfer Records, Debug, Fast Interrupts, I/D Synch, Priv 1.13, Shadow Stacks/Landing Pads, Vector Crypto, Vector FP16

● ISA Fast Tracks
  ○ CAS, BFloat16, Conditional Ops, Counter Mode Filtering, HW PTE A/D, Maybe Ops, Additional Scalar FP, Counter Delegation,

● Non-ISA
  ○ IOMMU, Confidential VM extension, IOPMP, Nexus Trace,

● Documentation
  ○ Full time RVI DOC Architect/Writer approved
  ○ BOD committee on dev experience (content & UX)
  ○ Glossary, Navigation
Software Ecosystem

This is our number one priority
RISC-V will have the best ecosystem

- Largest number of players
- All cores in system become RISC-V
- Software wants to run on the best hardware

- Hardware and software have been co-evolving rapidly
- Long-running silicon and core developments bearing fruit now
- As advantages and future become clearer, greater motivation to move to RISC-V
- Positive feedback on software ecosystem growth
RISC-V ecosystem maturing rapidly

- Consider relatively advanced RISC-V software ecosystem despite very little development hardware to date
- New dev boards/products appearing en masse in next year or two
- E.g., Google announced Android support for RISC-V, Dec 2022
Debian Port Progress

SiFive shipped
Unmatched board
Accelerating the Rich RISC-V Ecosystem

<table>
<thead>
<tr>
<th>Applications</th>
<th>Security</th>
<th>HPC</th>
<th>AI / ML</th>
<th>Cloud - Data Center - Storage - Networking</th>
<th>Consumer - IoT - Automotive</th>
<th>Mobile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libraries</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Infrastructure</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Runtimes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hypervisor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Boot</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Design Tools + Compilers

Ecosystem Standards, upstream / support common tools
ISA Extensions, Profiles, Platforms
Security model, IOMMU, Industry SIGs
Architecture Tests
SAIL Golden Model
Attributes: Debuggable, Secure, Performant, Reliable/Serviceable/Diagnosable

Implementation
RTL
DV
Design & Microarchitecture
Silicon
Training
Research
Soft IP
Academia
Services
Android

● Google Support

● Google Requirements
  ○ Engagement with RISC-V
  ○ Likely in 2024
  ○ Much is underway

● Lifetime can be 10 years
# RT-Thread on RISC-V

<table>
<thead>
<tr>
<th>RV32</th>
<th>RV64</th>
<th>RV-Core</th>
</tr>
</thead>
<tbody>
<tr>
<td>HiFive1</td>
<td>K210</td>
<td>bumblebee</td>
</tr>
<tr>
<td>RV32M1_VEGA</td>
<td>Allwinner D1</td>
<td>QingKeV3</td>
</tr>
<tr>
<td>HPM6750</td>
<td>Allwinner D1S</td>
<td>QingKeV4</td>
</tr>
<tr>
<td>GD32VF103</td>
<td>QEMU/RISCV64</td>
<td>E310</td>
</tr>
<tr>
<td>AB32VG1</td>
<td>VIRT</td>
<td>cv32e40p</td>
</tr>
<tr>
<td>CH32V103</td>
<td></td>
<td>Andes D45</td>
</tr>
<tr>
<td>CH32V208</td>
<td></td>
<td>SMART-EVB for T-Head</td>
</tr>
<tr>
<td>CH32V307</td>
<td></td>
<td>CPU E9xx Series</td>
</tr>
<tr>
<td>CH569</td>
<td></td>
<td>PicoRV32</td>
</tr>
<tr>
<td>BL808</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESP32C3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Software Ecosystem Resources

- **Foundational Software Status** for each extension
- Draft *spreadsheet* of software on RISC-V status (140+ being tracked)
  - Hired RVI Software Ecosystem Director. First task is a comprehensive one stop clearinghouse of RISC-V commercial and open source software status
- RISC-V Ecosystem *Landscape*
- RISC-V *Exchange* (100s)
- A new Linux Foundation Project named RISE to accelerate open source software development on RISC-V
# Profiles with Key Ecosystem Status

<table>
<thead>
<tr>
<th>Extension/Base Name - Best Guess</th>
<th>Ratification Package Name</th>
<th>Description (what this does, in English)</th>
<th>IC</th>
<th>Extensions Included (subsets)</th>
<th>Implies (and Transitive)</th>
<th>Incompatible (and Transitive)</th>
<th>Ratified Year (or Expected) or Future</th>
<th>RVA20 (64 only)</th>
<th>RVA22 (64 only)</th>
<th>RVA23 (64 only)</th>
<th>MAJOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>original</td>
<td>Atomics</td>
<td>unpriv</td>
<td>Zaame, Zalrsc</td>
<td>y</td>
<td>2019</td>
<td>o m m m m m m m m m m m</td>
<td>n n</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>original</td>
<td>Compressed instructions</td>
<td>priv</td>
<td></td>
<td>y</td>
<td>2019</td>
<td>o m m m m m m m m m m m</td>
<td>y y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>original</td>
<td>Floating point, double-precision (implies F)</td>
<td>unpriv</td>
<td>F</td>
<td>y</td>
<td>2019</td>
<td>o m m m m m m m m m m m</td>
<td>y y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>original</td>
<td>Floating point, single-precision</td>
<td>unpriv</td>
<td>Zicsr</td>
<td>y</td>
<td>2019</td>
<td>o m m m m m m m m m m m</td>
<td>y y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>H</td>
<td>Hypervisor</td>
<td>unpriv</td>
<td></td>
<td>y</td>
<td>2021</td>
<td>n n n n n o o o o o</td>
<td>n n</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>original</td>
<td>Multiply/divide</td>
<td>priv</td>
<td></td>
<td>y</td>
<td>2019</td>
<td>o m m m m m m m m m m m</td>
<td>y y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>User level interrupts</td>
<td>unpriv</td>
<td></td>
<td>n</td>
<td>future</td>
<td>n n n n n n n n n n</td>
<td>n n</td>
<td>n n n n n n n n n</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q</td>
<td>original</td>
<td>Floating point, quad-precision</td>
<td>priv</td>
<td></td>
<td>y</td>
<td>2019</td>
<td>n n n n n n n n n n n n</td>
<td>n n</td>
<td>n n n n n n n n n</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RV32E</td>
<td>RV32I</td>
<td>Integer base for RV32</td>
<td>unpriv</td>
<td></td>
<td>y</td>
<td>2023</td>
<td>n n n n n n n n n n</td>
<td>n n</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RV32I</td>
<td>RV32I</td>
<td>Integer base for RV32</td>
<td>unpriv</td>
<td></td>
<td>y</td>
<td>2019</td>
<td>m n n n n n n n n</td>
<td>n n</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RV64E</td>
<td>RV64I</td>
<td>Integer base for RV64</td>
<td>unpriv</td>
<td></td>
<td>y</td>
<td>2023</td>
<td>n n n n n n n n n</td>
<td>n n</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RV64I</td>
<td>RV64I</td>
<td>Integer base for RV64</td>
<td>unpriv</td>
<td></td>
<td>y</td>
<td>2019</td>
<td>m m m m m m m m m</td>
<td>n n</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

RISC-V®

[Only if H for any hpmcounter that is not hardware supported.]

**MAJOR**

- **m**: mandatory
- **M**: m if H is implemented
- **o**: optional
- **n**: non-profile options
- **p**: part of an optional extension but not an optional extension itself

**RVI32** is only applicable to RVM and RVI profiles.
# Commercial & Open Source SW

<table>
<thead>
<tr>
<th>Project Name</th>
<th>Source Code Location</th>
<th>mark's target</th>
<th>mark's categories</th>
<th>Project Area</th>
<th>Has it started work on RISC-V?</th>
<th>RISC-V Support Stage (the baseline is the x86_64 support when possible; supported means it has the base support for RISC-V, works, but there is still more work to be done; fully supported means out-of-box support for RISC-V)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LibreOffice</td>
<td><a href="https://git.libreoffice.org/cons/">https://git.libreoffice.org/cons/</a></td>
<td>user</td>
<td>app</td>
<td>Office Suite</td>
<td>1</td>
<td>supported</td>
</tr>
<tr>
<td>MediaWiki</td>
<td><a href="https://github.com/mediawiki/mediawiki">https://github.com/mediawiki/mediawiki</a></td>
<td>it</td>
<td>application</td>
<td>Collaboration</td>
<td>1</td>
<td>supported</td>
</tr>
<tr>
<td>WordPress</td>
<td><a href="https://core.trac.wordpress.org/browser">https://core.trac.wordpress.org/browser</a></td>
<td>application</td>
<td>Collaboration/Content Management</td>
<td>1</td>
<td>supported</td>
<td></td>
</tr>
<tr>
<td>gcc</td>
<td><a href="https://github.com/golang/go">https://github.com/golang/go</a></td>
<td>developer</td>
<td>compiler</td>
<td>Toolchain</td>
<td>1</td>
<td>supported</td>
</tr>
<tr>
<td>GNU C</td>
<td><a href="https://github.com/go-go/gnu">https://github.com/go-go/gnu</a></td>
<td>developer</td>
<td>compiler</td>
<td>Toolchain</td>
<td>1</td>
<td>supported</td>
</tr>
<tr>
<td>GO</td>
<td><a href="https://github.com/golang/go">https://github.com/golang/go</a></td>
<td>developer</td>
<td>compiler</td>
<td>Toolchain</td>
<td>1</td>
<td>supported</td>
</tr>
<tr>
<td>Green Hills</td>
<td>?</td>
<td>developer</td>
<td>compiler</td>
<td>Operating System</td>
<td>1</td>
<td>supported</td>
</tr>
<tr>
<td>LLVM</td>
<td><a href="https://github.com/llvm/llvm-project">https://github.com/llvm/llvm-project</a></td>
<td>general development</td>
<td>compiler</td>
<td>Compiler</td>
<td>1</td>
<td>supported</td>
</tr>
<tr>
<td>Rust</td>
<td><a href="https://github.com/rust-lang/rust">https://github.com/rust-lang/rust</a></td>
<td>compiler</td>
<td>Language</td>
<td>1</td>
<td>supported</td>
<td></td>
</tr>
<tr>
<td>MariaDB</td>
<td><a href="https://github.com/MariaDB/server">https://github.com/MariaDB/server</a></td>
<td>infrastructure</td>
<td>database</td>
<td>Database</td>
<td>1</td>
<td>supported</td>
</tr>
<tr>
<td>memcached</td>
<td><a href="https://github.com/memcached/memcached">https://github.com/memcached/memcached</a></td>
<td>infrastructure</td>
<td>database</td>
<td>Storage</td>
<td>1</td>
<td>supported</td>
</tr>
<tr>
<td>MongoDB</td>
<td>?</td>
<td>database</td>
<td>Database</td>
<td>1</td>
<td>supported</td>
<td></td>
</tr>
<tr>
<td>MySQL</td>
<td><a href="https://github.com/mysql">https://github.com/mysql</a></td>
<td>database</td>
<td>Database</td>
<td>1</td>
<td>supported</td>
<td></td>
</tr>
<tr>
<td>Redis</td>
<td><a href="https://github.com/redis/redis">https://github.com/redis/redis</a></td>
<td>database</td>
<td>Database</td>
<td>1</td>
<td>supported</td>
<td></td>
</tr>
<tr>
<td>cryptosapi</td>
<td>?</td>
<td>security</td>
<td>library</td>
<td>1</td>
<td>supported</td>
<td></td>
</tr>
<tr>
<td>MSS</td>
<td><a href="https://github.com/Microsoft/MSS">https://github.com/Microsoft/MSS</a></td>
<td>Windows</td>
<td>database</td>
<td>1</td>
<td>supported</td>
<td></td>
</tr>
<tr>
<td>RISC-V</td>
<td>?</td>
<td>security</td>
<td>library</td>
<td>1</td>
<td>supported</td>
<td></td>
</tr>
</tbody>
</table>

[Image of RISC-V logo]
Software Landscape
RISC-V Exchange

The RISC-V Exchange hosts the hardware, software, services, and learning offerings in the RISC-V community. Browse the list or search for an offering below.

Search Exchange... Hardware Cores Software Services Learning

Software
Software Type
- Accelerated Libraries
- Accelerated Libraries, Linux, macOS
- Application Infrastructure
- Application Infrastructure, Simulators
- Bootloaders
- BSD Distro
- C Compilers and Libraries
- C compilers and libraries, Compilers and runtimes for other languages
- Cloud infrastructure
- Configuration
- Connectivity management
- Course materials
- Debugging
- Hypervisors
- Hypervisors, Emulation
- Hypervisors, Emulation, Debugging
- Hypervisors, Emulation, Debugging, Course materials
- Hypervisors, Emulation, Debugging, Course materials, Hypervisors

CREATOR Simulator

Organization: UCBM
CREATOR: dataCic and generic assembly programming simulator
Software Type: Simulators

emmtrix Parallel Studio

Organization: emmtrix Technologies GmbH
emmtrix Parallel Studio allows the parallelization and code generation for different processors. With the support for the RISC-V architecture it provides capabilities to optimize the performance and generate C code for the CPU cores as well as for the vector extensions.
Verification

- DV is done by implementers and DV providers
  - See risc.org/exchange for a list of providers
- Implementations can be wildly different
- Open source implementations include full RTL and DV
- Compatibility is determined by passing basic architecture tests (supplied by RISC-V and depends on SAIL formal model for golden results) and self-attestation for OSs, DV, etc.
Join RISC-V

Change the World!
Questions