State of the Union

Krste Asanovic
UC Berkeley, RISC-V Foundation, & SiFive Inc.
krste@berkeley.edu

RISC-V Summit
San Jose Convention Center, CA, USA
December 10, 2019

RISC-V Origin Story

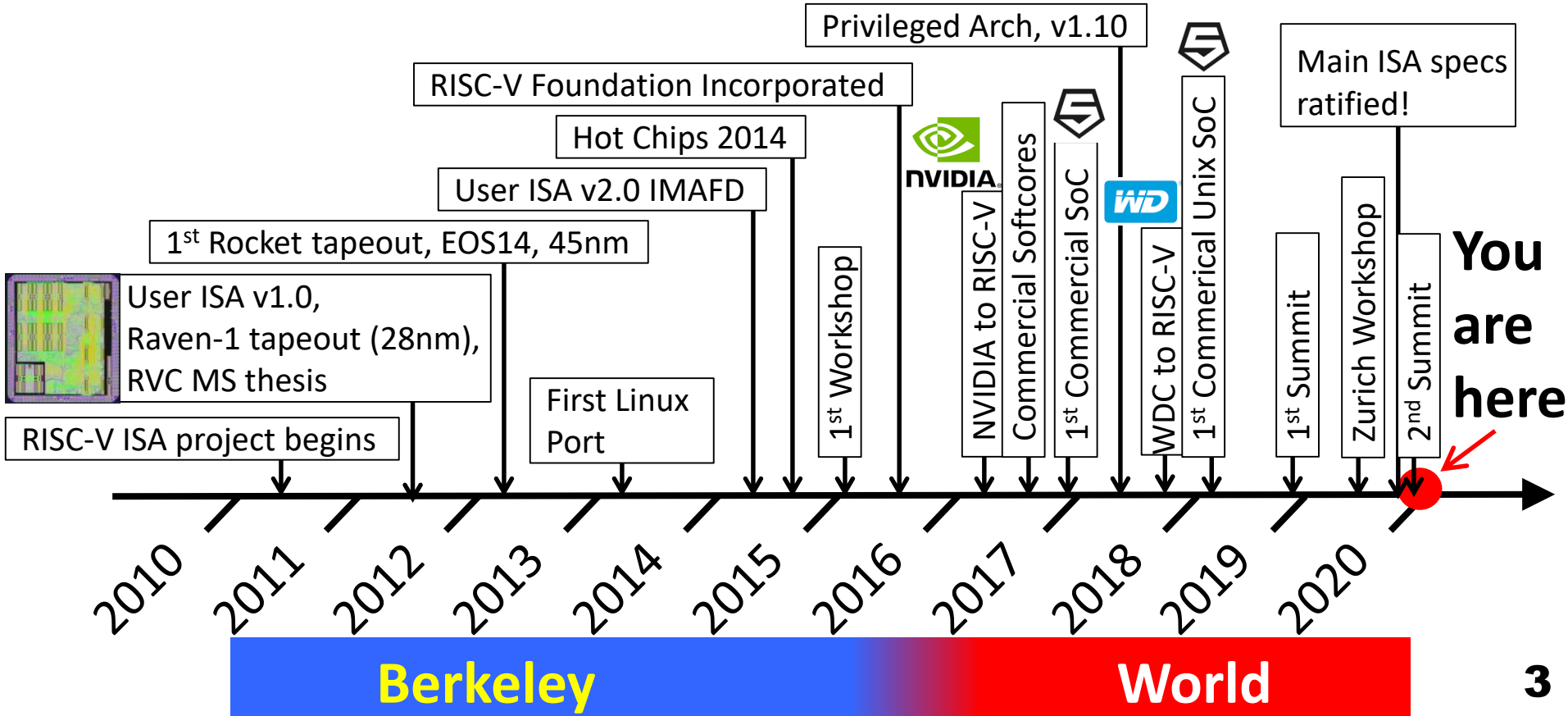
For Berkeley Par Lab project 2010, we needed an ISA:

- simple to implement
- efficient
- extensible
- no constraints on sharing with our work with others

No existing ISA met needs, so created RISC-V *(Andrew Waterman, Yunsup Lee, David Patterson, Krste Asanovic)*

- Turns out, many others wanted same things
- From initial roll out in 2014, rapid uptake everywhere

RISC-V Timeline



What's Different about RISC-V?

- ***Simple***
  - Far smaller than other commercial ISAs
- ***Clean-slate design***
  - Clear separation between user and privileged ISA
  - Avoids µarchitecture or technology-dependent features
- A ***modular*** ISA designed for ***extensibility/specialization***
  - Small standard base ISA, with multiple standard extensions
  - Sparse and variable-length instruction encoding for vast opcode space
- ***Stable***
  - Base and standard extensions are frozen
  - Additions via optional extensions, not new versions
- ***Community designed***
  - With leading industry/academic experts and software developers

RISC-V Ecosystem

Software

Open-source software:
Gcc, binutils, glibc, Linux, BSD, LLVM, QEMU, FreeRTOS, ZephyrOS, LiteOS, SylixOS, …

Commercial software:
Lauterbach, Segger, IAR, Micrium, ExpressLogic, Ashling, AntMicro, Imperas, UltraSoC …

RISC-V Foundation

ISA specification | Golden Model | Compliance

Hardware

Open-source cores:
Rocket, BOOM, RI5CY, Ariane, PicoRV32, Piccolo, SCR1, Shakti, Serv, Swerv, Hummingbird, …

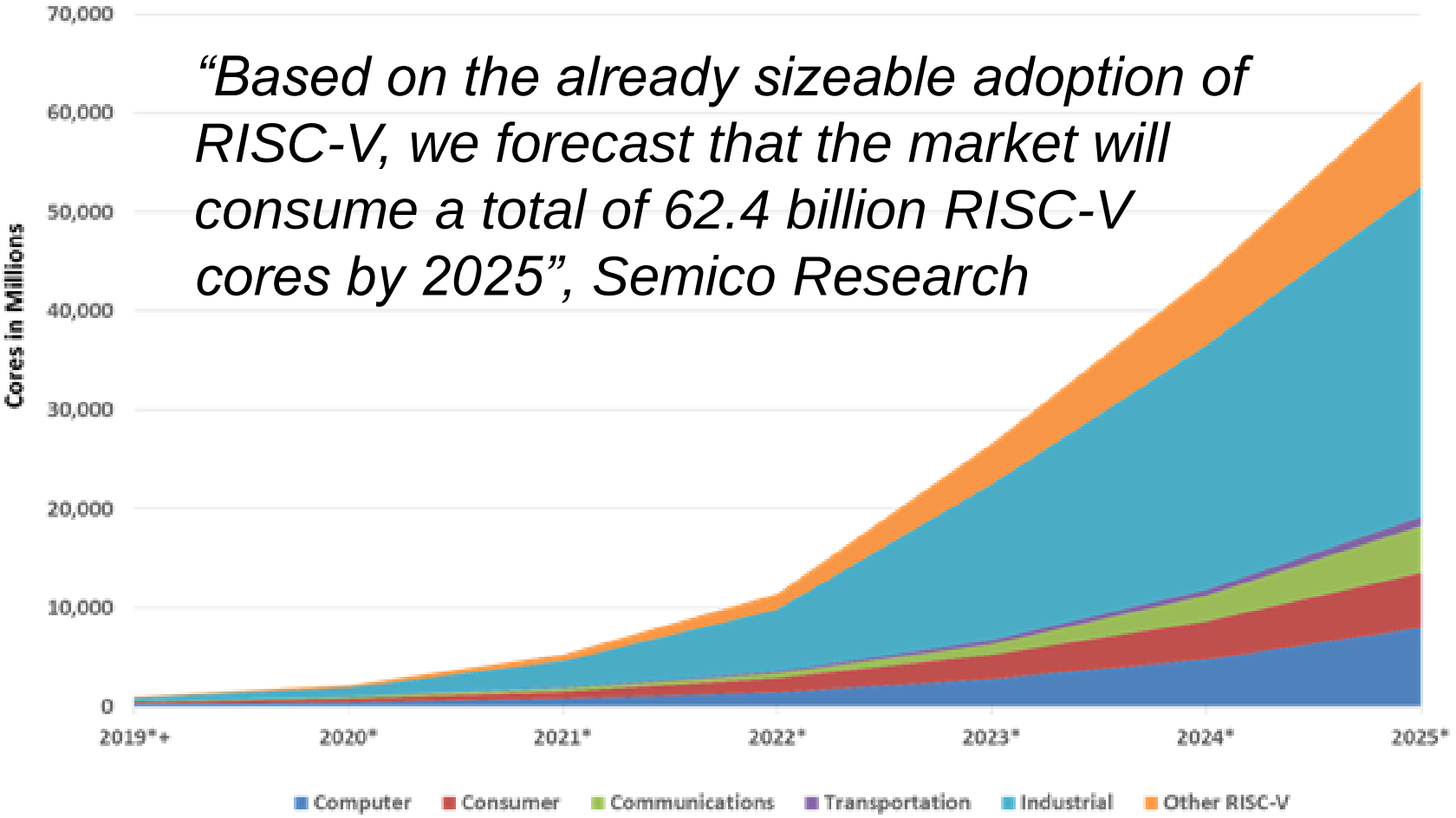
Commercial core providers:
Alibaba, Andes, Bluespec, Cloudbear, Codasip, Cortus, InCore, Nuclei, SiFive, Syntacore, …

Inhouse cores:
Nvidia, WD, +others

Why is RISC-V so popular?

- Engineers sometimes "*don't see forest for the trees*"
- The movement is ***not*** happening because some benchmark ran 10% faster, or some implementation was 30% lower power
- The movement ***is*** happening because ***new business model*** changes everything
  - Pick ISA first, then pick vendor or build own core
  - Add your own extension without getting permission
- Implementation features/PPA will follow
  - Whatever is broken/missing in RISC-V will get fixed

“Based on the already sizeable adoption of RISC-V, we forecast that the market will consume a total of 62.4 billion RISC-V cores by 2025”, Semico Research



Source: Semico Research Corp.

Our Modest RISC-V Project Goal

Become the industry-standard ISA for all computing devices

This is happening!

Far faster, more domains, than anyone predicted

Demand at every performance level (low to ludicrous)

Demand for all features of all other ISAs

Plenty of work to do, hardware and software!

2019: The ISA Foundation is in Place!

- RV32IMAFDQC , RV64IMAFDQC ratified
  - +Zifencei, +Zicsr
- Privileged Arch 1.11 ratified
- Base memory model RVWMO ratified
- Debug spec 0.13 ratified
- SAIL formal spec adopted as golden model
- Compliance Suite v0.1 imminent

- Substantial upstream software support

Growing RISC-V

- The base ISA and initial standard extensions were intended to be simple, efficient, and extensible
- Never thought this would be the end of RISC-V
- A large variety of extensions being pursued, likely have 1000s of instructions in RISC-V eventually
- But will always be possible to build a core and use upstream tools using only RV32I integer base.

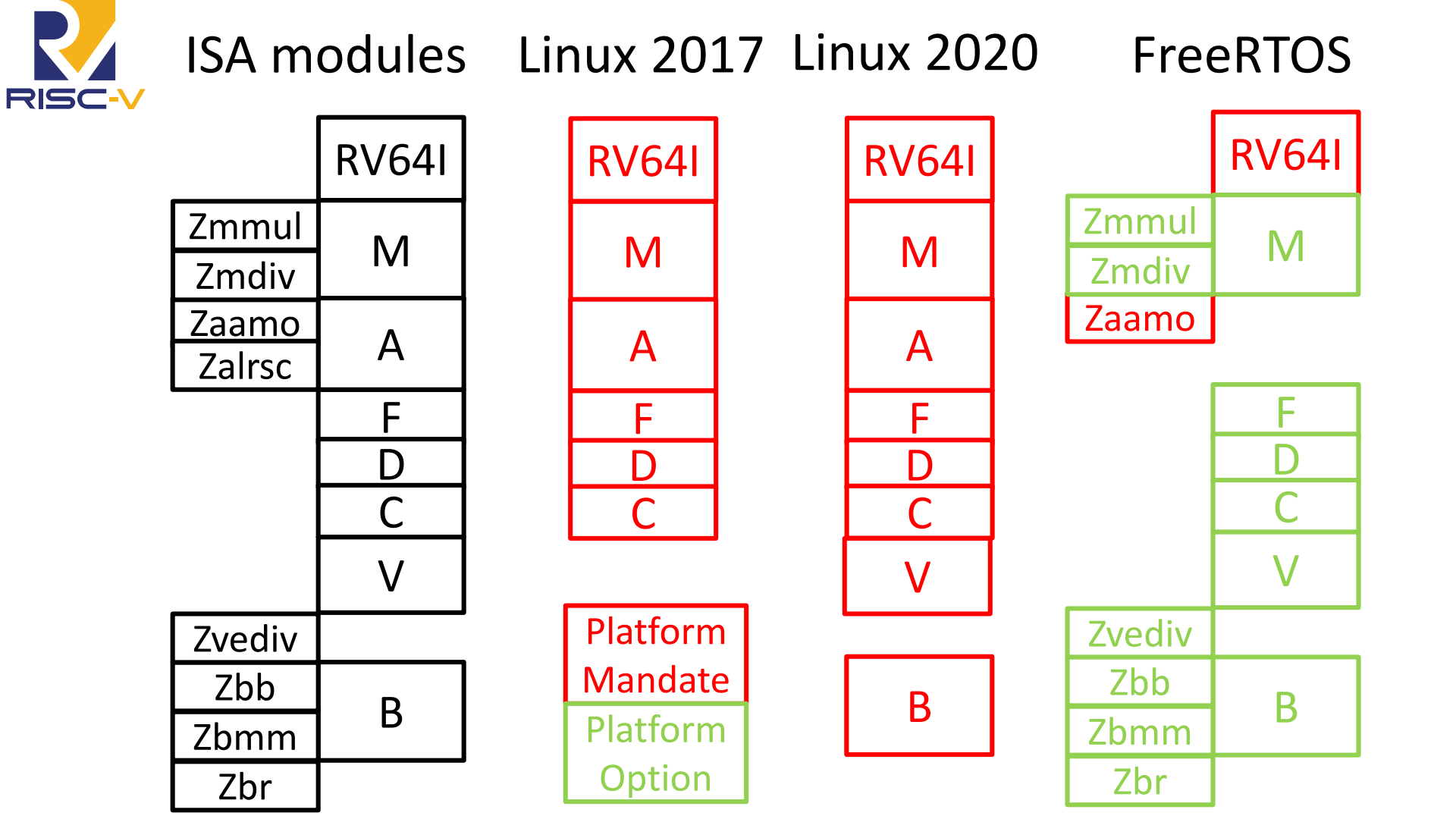
Fragmentation versus Diversity

Fragmentation:
Same thing done different ways

Diversity:
Solving different problems

How we structure RISC-V standards

- ISA specs
  - Modular ISA, each module eventually ratified and frozen
  - Each module can contain options
  - Modules designed to combine in system without conflicts
  - Custom ISA space separated from standard ISA space
  - *ISA specs do not constrain how the modules are used*
- Platform specs
  - Driven by software ecosystem (e..g, FreeRTOS, server)
  - Provides constraints on choice of modules and options so software is straightforward to port

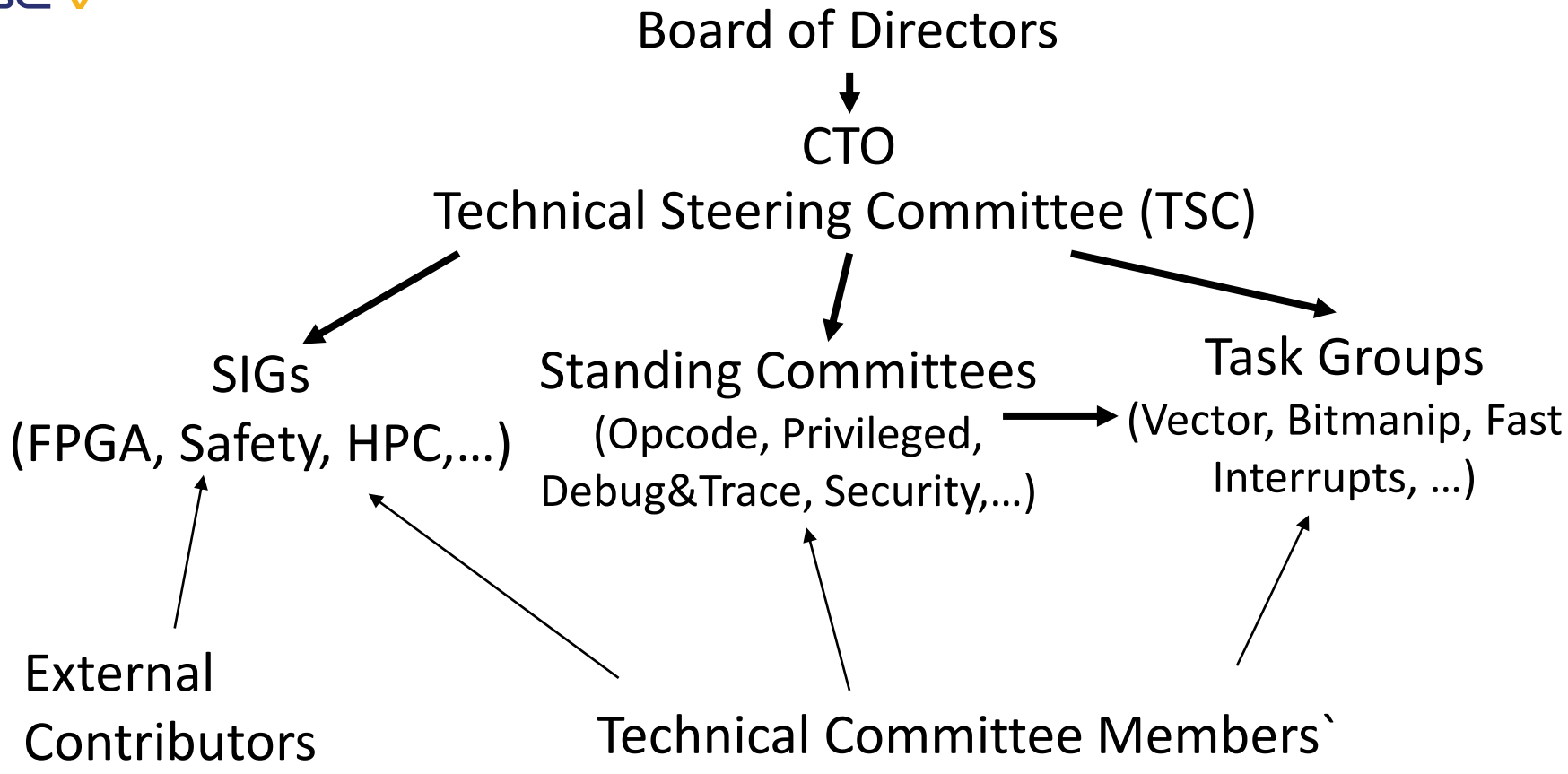

ISA modules
Linux 2017
Linux 2020
FreeRTOS
RV64I
Zmmul
Zmdiv
M
Zaamo
Zalrsc
A
F
D
C
V
Zvediv
Zbb
Zbmm
Zbr
B
RV64I
M
A
F
D
C
Platform Mandate
Platform Option
RV64I
M
A
F
D
C
V
B
RV64I
Zmmul
Zmdiv
M
Zaamo
F
D
C
V
Zvediv
Zbb
Zbmm
Zbr
B

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.

Scaling the RISC-V Standards Org



Special Interest Groups

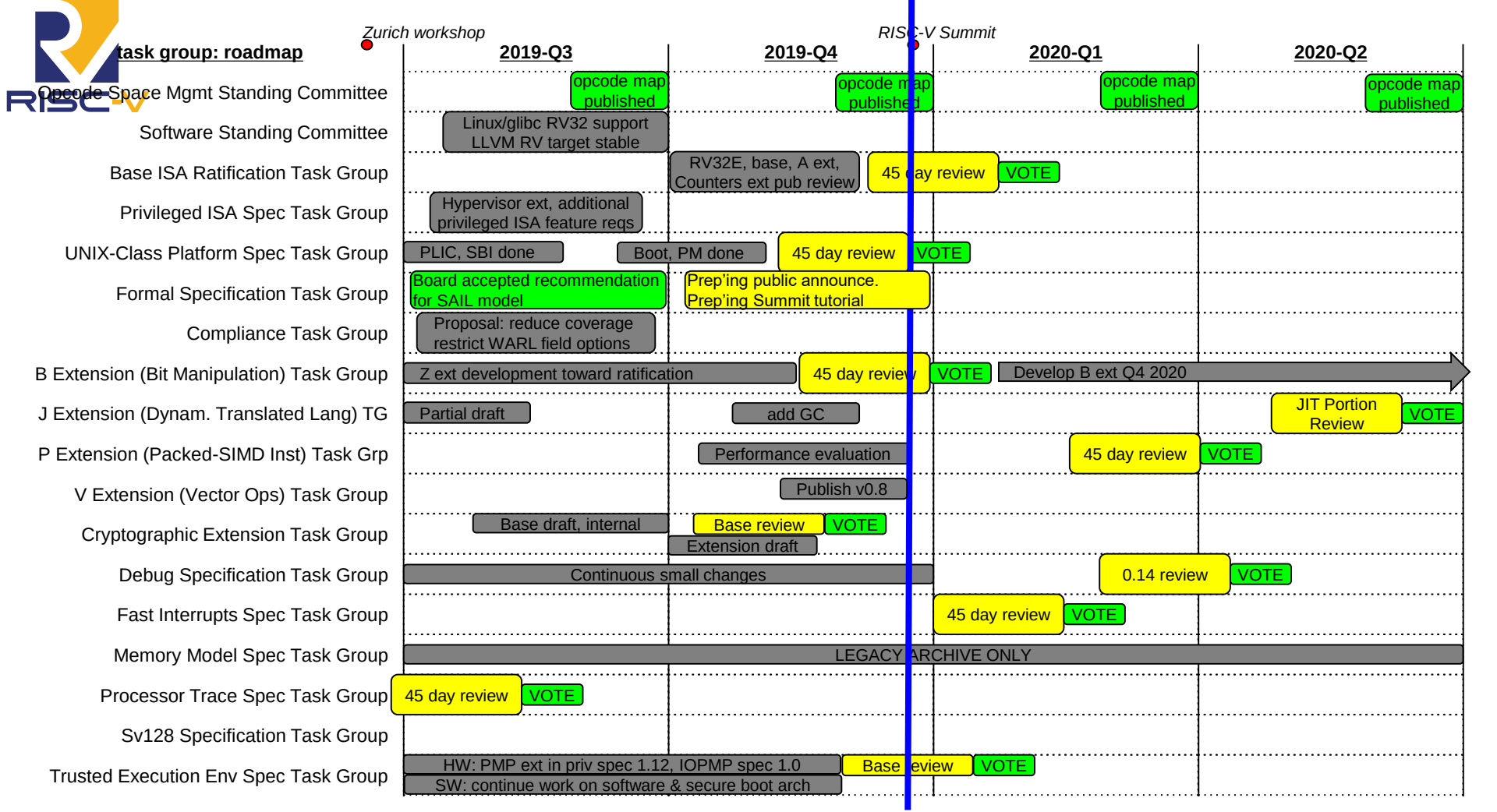
SIGs created for:

- FPGA soft cores
- Functional Safety
- HPC

- Provide a place to interact with broader community
- No specific deliverables, but can propose new initiatives within foundation

Standards Process

- Developing a written standards process to ensure all proposals take correct steps in formation/execution
- Desire to add lighter-weight standards path for simpler extensions
- All members interested in technical side of RISC-V should join general mailing list: **tech@lists.riscv.org**


Zurich workshop
RISC-V Summit
task group: roadmap
2019-Q3
2019-Q4
2020-Q1
2020-Q2
Opcode Space Mgmt Standing Committee
opcode map published
opcode map published
opcode map published
opcode map published
Software Standing Committee
Linux/glibc RV32 support
LLVM RV target stable
Base ISA Ratification Task Group
RV32E, base, A ext, Counters ext pub review
45 day review
VOTE
Privileged ISA Spec Task Group
Hypervisor ext, additional privileged ISA feature reqs
UNIX-Class Platform Spec Task Group
PLIC, SBI done
Boot, PM done
45 day review
VOTE
Formal Specification Task Group
Board accepted recommendation for SAIL model
Prep'ing public announce.
Prep'ing Summit tutorial
Compliance Task Group
Proposal: reduce coverage
restrict WARL field options
B Extension (Bit Manipulation) Task Group
Z ext development toward ratification
45 day review
VOTE
Develop B ext Q4 2020
J Extension (Dynam. Translated Lang) TG
Partial draft
add GC
JIT Portion Review
VOTE
P Extension (Packed-SIMD Inst) Task Grp
Performance evaluation
45 day review
VOTE
V Extension (Vector Ops) Task Group
Publish v0.8
Cryptographic Extension Task Group
Base draft, internal
Base review
VOTE
Extension draft
Debug Specification Task Group
Continuous small changes
0.14 review
VOTE
Fast Interrupts Spec Task Group
45 day review
VOTE
Memory Model Spec Task Group
LEGACY ARCHIVE ONLY
Processor Trace Spec Task Group
45 day review
VOTE
Sv128 Specification Task Group
Trusted Execution Env Spec Task Group
HW: PMP ext in priv spec 1.12, IOPMP spec 1.0
Base review
VOTE
SW: continue work on software & secure boot arch

Formal Spec

- Foundation agreed to use SAIL formal spec as golden model for RISC-V
- Supports all features including memory model
  - except for floating-point (to be added)
- Will replace textual spec as the ultimate authority

- Tutorial on Formal Spec on Thursday 9am by Nikhil

Foundation Board of Directors' Award 2019

- **Rishiyur Nikhil**, Bluespec
- "For his leadership and technical contributions in the ISA Formal Specification and Compliance groups"

Compliance

- Significant progress on compliance framework and on actual tests
  - github:riscv/riscv-compliance
- Tests available for new B and V extensions
- Generating compliance tests is hard work, more help needed!

“B” Bit-Manipulation Extension

- Adds various bit-twiddling instructions to ISA
  - E.g., count leading zeros, popcount, byte reverse, bit fields
- Helps with software crypto algorithms,
  - E.g., rotate instructions, carryless multiply
- Also, help with address modes (shift+add)
- Reaching consensus on contents of B and other Zb* subsets

- Anticipate ratification first half 2020

“V” Vector Extension

- Provides scalable vector instructions
  - From 32x32b vector registers, to thousands bits per vector
  - From embedded fixed-point to HPC floating-point
- Release of version 0.8 of spec at 2nd Summit
- Binutils, spike, Imperas model and compliance tests available
- Multiple implementations in flight

- Anticipate freeze in mid-2020, ratify by end of 2020

”P” Packed-SIMD Extension

- For low-end cores, provides packed SIMD within x registers (e.g., 2*16b operations in 32b register)
- Some operations use paired even-odd registers for wide accumulators
- Version 0.9 release including toolchain
- Working on encoding and arranging instructions into instruction subsets

- Expect Ratification in Q3 2020

“J” Extension

- J TG focusing on support for dynamically translated languages
- Main topic is new instruction coherence scheme
  - Important for JIT compilers
  - FENCE.I was too heavyweight for some systems

Fast Interrupts ”CLIC” Task Group

- Provides low-latency, preemptive nested interrupts local to each core
- Most functionality defined in software, supports multiple ABIs
- Spec at version 0.9, refining details of what is supported in base profile

- To be ratified in early 2020

Privileged Architecture 1.12

- Priv 1.12 proposal to add set of features
  - Resumable NMI (current spec non-resumable only)
  - Counter overflow interrupts
  - Stimecmp as CSR
  - Redefine MPRV (passed)
  - Define Custom space for SYSTEM instructions
  - Mvtal atomicity
  - CEASE instruction for powerdown
  - Misaligned trap priority
  - RV32 satp change to accommodate more Modes
- Anticipate mid-2020 ratification

Hypervisor Update

- Version 0.5 released (in github:riscv/riscv-isa-manual)
- Supports Type 1 and Type 2 hypervisors
- Supported in QEMU
- KVM, Xen, seL4 ports underway
- Virtualized interrupt scheme to be defined, support for very large-scale systems

- Anticipate mid-2020 freeze, late 2020 ratification
  - Need hardware implementations and software ports

TEE and Crypto Task Groups

- TEE group proposal for additional PMP functionality to help build secure containers

- Crypto group adding instructions on top of vector unit using v registers, but also considering support within scalar x registers for smaller systems

Debug and Trace

- Debug group continuing enhancements towards 0.14 release, mid 2020
- Trace group defining new highly compressed trace format
- New effort to reuse portions of existing Nexus standard for RISC-V trace systems

Discovery Mechanism

- Many groups asking for some version of original “config string” to provide self-describing hardware
- Takes form of some data structure burned into ROM on machine
- Tim Newsome has been gathering inputs
- Need to form TG

Embedded ABI

- Original ABI developed for Unix platforms
- Embedded ABI goals
  - reduce interrupt latency by reducing argument registers, caller-saved registers
  - reduce code size by dropping 128b FP support
  - Provide compatible interface for RV32E and RV32I
- Proposal in https://github.com/riscv/riscv-eabi-spec
- Need help getting TG off ground

Zfinx “Float in X registers”

- Intended for embedded platforms
- Removes f registers from ISA
- Floating-point operations take values from x registers
- Supported for both E and I base ISAs

- Reduces system size, interrupt latency, simplifies ABI
- Frees up floating-point load/store opcode space in compressed extension
- Needs spec!

Improving Embedded Code Size

- Better documentation/ white papers on existing techniques for RISC-V
- Improved library code for code size reduction, including new embedded ABI
- Make compilers aware of compressed instructions
- Further compiler optimizations
- Repurpose FP load/store encodings for Zfinx
- Other new instructions
- Needs work and TG chairs

Floating-Point Updates, New TG

- Requests for several features:
  - Support for half-precision (scalar and vector)
  - Support for alternate formats (bfloat16, Posits)
  - RV32D/RV64Q f-to-x register moves
  - Support for new IEEE-754/2019 FP additions
- Need interested volunteers to work on spec

New Priv-Arch TG Proposals

- Support for larger pages (e.g., 64KiB) and other VM page-table enhancements
- Dynamic physical memory attributes (PMAs)
- Cache-maintenance operations
  - Software coherence, power-down, security, performance
- IOMMU/IOPMP

- Proposals being prepared for TC

Call for Standards Participation

Design is easy & fun, standards are hard work

- Debating all the fine technical details
- Separating facts from opinions
- Evaluating hardware and software implications
- Understanding commercial realities
- Backwards compatibility
- Pruning unnecessary options (saying “No!”)
- Reaching consensus

- ***We need more skilled participation!***

RISC-V Predictions for 2020

- In 2020, see significant investment in high-end RISC-V implementations (server, mobile, auto)
- We're laying the ISA and software foundations to enable these
  - Hypervisor
  - Vectors
  - Security
  - Power-management
  - Server platform profile
  - Functional Safety

Summary

- Core standards ratified
- Widespread acceptance and adoption
- Community growing and growing
- Software ecosystem filling out rapidly
- Standards efforts filling in gaps
- Lots of work ahead…

RISC-V
Technical Committee

RISC-V Software

RISC-V Software Progress

RISC-V Technical Committee

- **An ISA is as successful as its software/ecosystem**

- **Major strides in RISC-V software ecosystem over past 4 years**
  - See Randy Allen's "RISC-V Software State of the Union" talk
  - For full details: https://riscv.org/software-status/

- **RISC-V software ecosystem highlights:**
  - **Toolchains**: Upstream GCC and clang/LLVM
  - **Libc:** Upstream glibc, newlib, musl, ...
  - Upstream **Linux kernel**
  - **Linux distributions** (not yet upstream): Debian, Fedora, OpenSUSE, Yocto, …
  - **Boot/Firmware:** Upstream u-Boot, Grub, coreboot, OpenSBI, EDK2 out for review
  - **Simulation/Emulation:** Upstream QEMU, Renode, Spike, …
  - **Other OSes:** Upstream FreeBSD, Zephyr, FreeRTOS, seL4, RTEMS, Tock, **...**

Software Status Page

RISC-V Technical Committee

RISC-V®

ABOUT | MEMBERSHIP | SPECS & SUPPORT | CORES & TOOLS | NEWS | EVENTS

Software Status

/ Software Status

RISC-V SPECIFICATIONS

- Unprivileged Specification
- Privileged ISA Specification
- Debug Specification

RISC-V SOFTWARE

- Software Status

RISC-V CORES

- RISC-V Cores

RISC-V EDUCATION

- RISC-V Educational Materials
- RISC-V Books
- RISC-V Academic Papers

This content is reproduced for convenience from the GitHub Wiki page on this topic.

The original page is available at:

https://github.com/riscv/riscv-software-list

RISC-V Software Ecosystem Overview

This document captures the status of the RISC-V Software Ecosystem. Please add to the list and fix inaccuracies.

- Simulators
- Object toolchain
- Debugging
- C compilers and libraries
- Boot loaders and monitors
- OS and OS kernels
- Compilers and runtimes for other languages
- IDEs
- Security

Accelerating the software ecosystem

RISC-V Technical Committee

- We have an aggressive roadmap for 2020
- Current RISC-V software/ecosystem gaps:
  - RV32 support for Linux
  - Upstream EDK2
  - Upstream Linux distribution support
  - Rust Linux/RV64 support
  - Support for new RISC-V extensions (Hypervisors, BitManip, Vector)
  - Toolchain code size optimizations
  - Javascript JITs
  - Browsers
  - OpenJDK
  - Low cost Linux based development boards

We need your help!

RISC-V Technical Committee

- RISC-V BoD wants to invest more in software enablement as well as prevent potential fragmentation in the community

- We will reach out to you shortly
  - What do you need from the RISC-V software ecosystem?
  - What RISC-V software are you working on internally?
  - Are there ways where you can contribute to the RISC-V ecosystem?

- Together, we can help build out the missing pieces in the software ecosystem and make RISC-V the default ISA for computing.