> ./rva23-in-the-linux-kernel

RVA23 Profile Support in the Linux Kernel

From Extension Definitions to Userspace Export

GX **Guodong Xu**
RISCstar
RISC-V International Advocate

CJ **Charlie Jenkins**
Rivos Inc. (Acquired by Meta)
Linux Kernel Developer

RISC-V Summit Europe 2026 · Bologna · Tue 9 Jun, 12:00 · plenary

> whoami

Guodong Xu

RISCstar · Linux Kernel Upstream

- ~20 yrs kernel & platform software (Arm + RISC-V)
- RISC-V mainline support · RVA23 enablement
- RISC-V International Advocate

I do RISC-V upstream in mainline Linux.

docularxu.github.io · GitHub: docularxu

Charlie Jenkins

Rivos Inc. (Acquired by Meta)

Linux Kernel Upstream · RVA23 profile support

ISA extension enablement (Zicboz, Zicbop, Svpbmt, CFI)

Linux Plumbers 2025 speaker (RVA23)

RISC-V and Linux enjoyer.

github.com/thecharlesjenkins

> the_hook

RVA23 — the mandatory baseline profile for application-class RISC-V

20 months

since ratification · Oct 2024 → today

Paper spec — or real?

two fronts: silicon / distros → today: the Linux kernel

SpacemiT K3 — the industry's first mass-produced RVA23 silicon

8x X100 (RVA23 core IP) + 8x A100 · ships Apr 2026

A Masterpiece Forged Over 3 Years and 6 Months of Intensive R&D

K3 Development Timeline

> June 22, 2022
> Wrote the first line of code for K3 (X100 Core)

SEPT 5, 2024	FEB 24, 2025	JUNE 18, 2025	SEPT 11, 2025	DEC 20, 2025	APRIL 2026
RVA23 Standard Freeze	X100 Code Freeze	A100 Code Freeze	K3 Successful First-Pass Tape-out	K3 Successful First-Pass Power-on	K3 Mass Production & Delivery

Source: SpacemiT

SpacemiT K3 — the industry's first mass-produced RVA23 silicon

8x X100 (RVA23 core IP) + 8x A100 · ships Apr 2026

K3 Development Timeline

SEPT 11, 2025

K3 First-Pass Tape-out

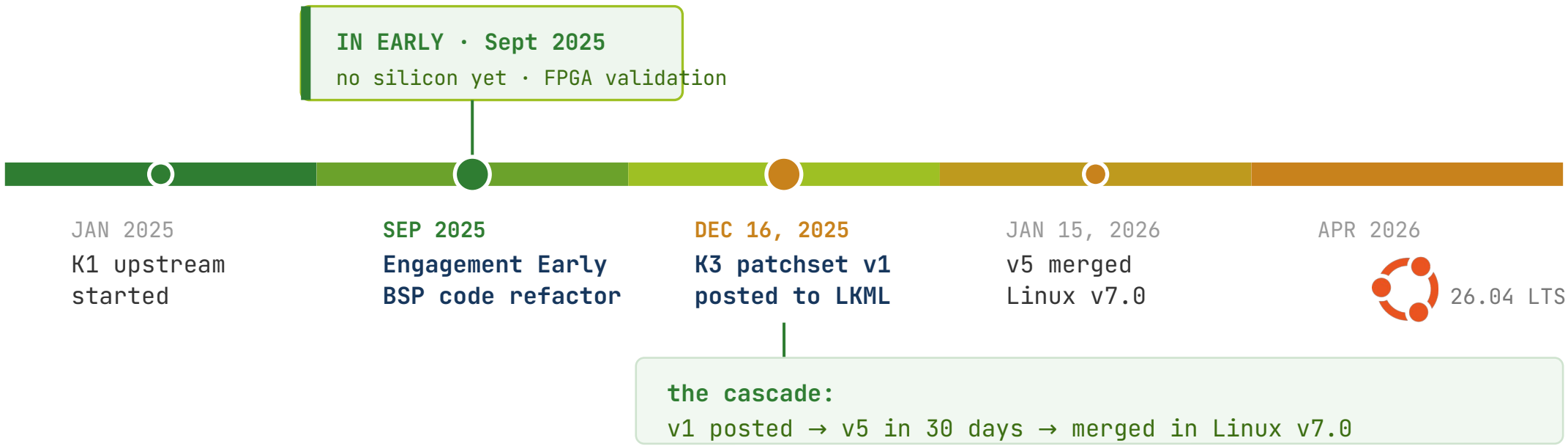
design sent to foundry

SEPT 5, 2024	FEB 24, 2025	JUNE 18, 2025	**SEPT 11, 2025**	DEC 20, 2025	APRIL 2026
RVA23 Standard Freeze	X100 Code Freeze	A100 Code Freeze	**K3 Successful First-Pass Tape-out**	K3 Successful First-Pass Power-on	K3 Mass Production & Delivery

Upstream Early, Land Fast

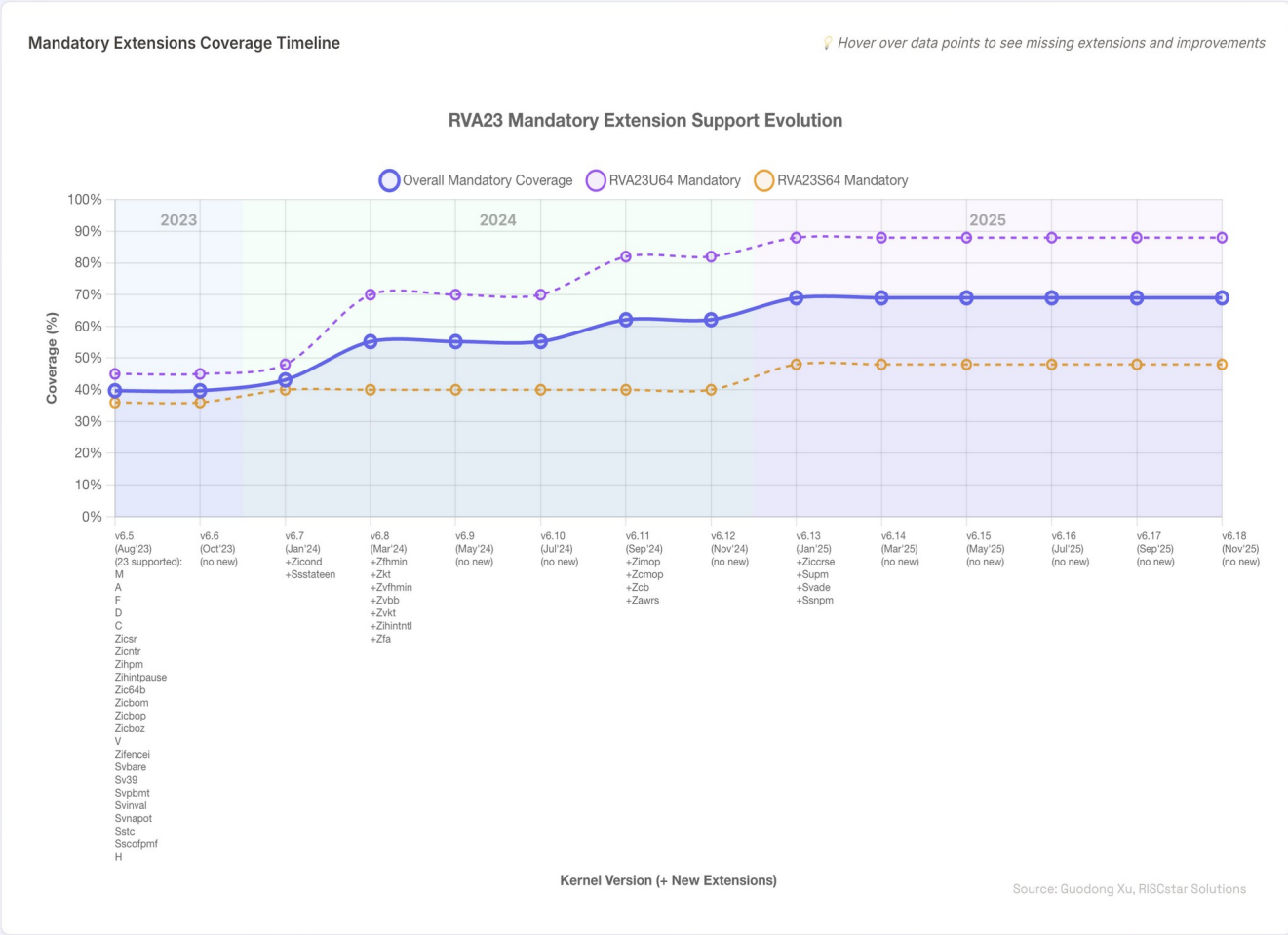
Engaged early — analyzing K3 code pre-silicon, then v1 → merge

K3 Kernel Upstream Timeline



> challenge_1

The kernel didn't yet have RVA23's mandatory extensions



68%

mandatory coverage — stuck ~1 year
(v6.13 → v6.18, when I picked up K3)

Heinrich Schuchardt (Canonical) · 2025-12-16

"This is not the description of an RVA23S64 cpu. It is not even RVA23U64, e.g. supm is missing. Is the description incomplete or is the CPU not compliant?"

my K3 v1 cover letter · 2025-12-16

"...only declares the isa-extensions currently supported by the kernel bindings."

> challenge_1 · the lesson

The reflex was wrong: don't strip — add upstream

the reflex · "platform problem"

binding doesn't know these yet →
omit them from the K3 dtsi

strip

```
riscv,isa-extensions =
  "i","m","a","c","v","h", ...
  "b","sha","ziccif","za64rs",
  "sstvecd","ssu64xl", ...
  (18 extensions dropped)
```

✗ K3 looks less capable
than it really is

→

the fix · platforms are changeable

not in the kernel yet? submit them.
then the dtsi can describe the real CPU

submit

```
riscv,isa-extensions =
  "i","m","a","c","v","h", ...
  "b","sha","ziccif","za64rs",
  "sstvecd","ssu64xl", ...
# +18 extensions, now in binding
```

✓ 18 extensions added to
extensions.yaml upstream

> *"...platforms are amenable to change and that one does not have to live with second-rate support."*
>
> — Jonathan Corbet, "The platform problem", LWN.net, 2011

> challenge_1_cont

The deepest nail — then coverage hits 100%

Supm: a profile abstraction, not hardware

(the real hardware underneath is Smnpm / Ssnpm)

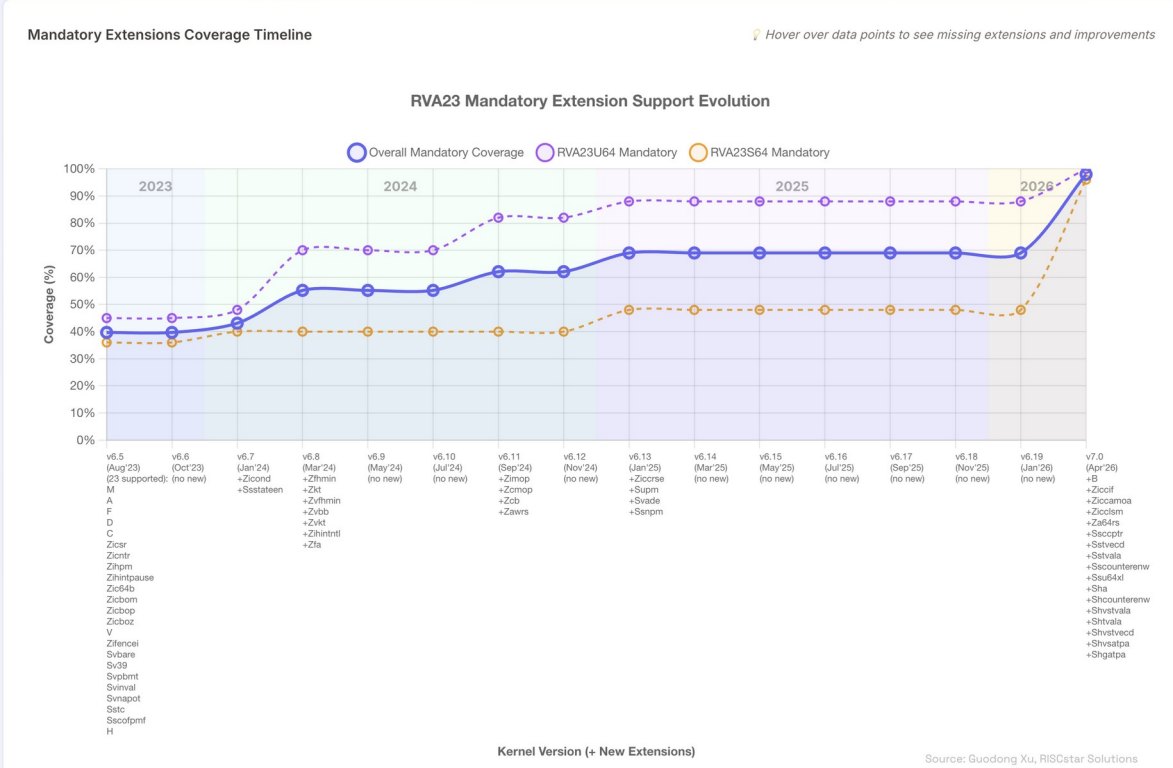
Conor Dooley (riscv DT maintainer) · 2025-12-22

"I'm not yet convinced that this makes sense as this describes how the consumer of the devicetree behaves not the hardware."

...and a swarm of dependency-check questions

- B = Zba + Zbb + Zbs — alias, bidirectional check
- Sha implies H + 6 others — v2 strict, v3 relaxed
- Ziccamoa / Za64rs → new checks on a / zaamo

K3 series: v1 → v5 in 30 days; Supm split out



100% RVA23 mandatory coverage at v7.0-rc1 (22 Feb 2026)

Supporting an extension: how much work depends on stateful vs stateless

> kernel enablement — the full path

1 · add DT binding
extensions.yaml recognizes the hardware

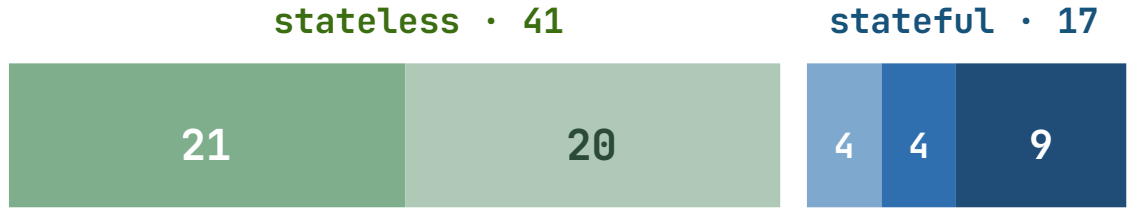
2 · cpufeature.c
kernel-space detection (ISA bitmap)

3 · hwprobe export
surface to userspace (/proc/cpuinfo + hwprobe)

4 · instructions / logic
save/restore state, signal frame, ABI...

how many steps depends on the class →

> RVA23 mandatory · 58 extensions, by arch-state



- insn-only 21 · e.g. Zicond
- property 20 · e.g. Sv39 (no insns, no state)
- fp 4 · e.g. Zfa (mstatus.FS)
- vector 4 · e.g. V (mstatus.VS)
- csr-only 9 · e.g. Supm

41 of 58 are stateless — only discoverable via hwprobe, no OS work.

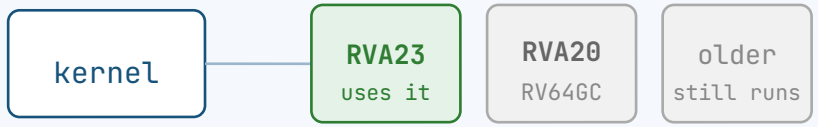
my classification · docularxu.github.io/rva23/rva23-extension-m html

Two halves of RVA23 in the kernel — complementary, both needed

binding defined

Detect at runtime

one binary · adapts at boot



uses RVA23 where present · /proc/cpuinfo + hwprobe

Andrew Jones · Guodong Xu

portability over specialization

Compile with RVA23

one kernel · tuned for RVA23

runs only where RVA23 is present · -march CFLAGS

Charlie Jenkins

performance over portability

Detect at runtime — a collision that became a handoff

Andrew Jones (Qualcomm) → my PATCH 1 · 2026-02-08

"Hi Guodong, As part of the 'hwprobe: Introduce rva23u64 base behavior' RFC I posted a similar patch where I also added B to hwcap.

Can you take a look at that?"

Andrew's friendly heads-up triggered my in-depth review; we now collaborate over the air :)

K3 (SpacemiT X100)

```
~ # cat /proc/cpuinfo
processor   : 0
hart        : 0
isa bases   : rv64ima rva23u64
mmu         : sv39
uarch       : spacemit,x100
```

isa bases : rv64ima

rva23u64 = the v2 deliverable, on real silicon

In open source, we're all working toward the same goal — upstream shares the work, not divides it.

rva23s64 · rva22 · rva20 ... export coming next — a follow-up patchset, once v2 settles upstream

> over_to_charlie

Guodong Xu

→

Over to Charlie Jenkins

Compile with RVA23 — the compile-time half

Compile with RVA23 — Charlie's series, and the reviewer's gate

CJ

Charlie Jenkins

[PATCH RFC 00/10] riscv: Add support for rva23

framework → CFLAGS

CONFIG_RVA23

optimize riscv_has_extension_*()

```
CONFIG_RISCV_ISA_RVA23=y
CFLAGS += -march=..._zba_zbb
# gcc now emits Zba/Zbb in plain C
```

based on Linux 6.18 · LKML 2025-12-10 · Plumbers 2025

goal: a kernel specialized for RVA23 (perf)

PW

Paul Walmsley (riscv maintainer)

review · 2026-01-14

IS THIS A WIN?

→ SHOW HARDWARE NUMBERS

framing correction:

"...not 'RVA23 support' patches ... these just enable building an RVA23-specific kernel."

the gate: 2024 precursor had only Spike / QEMU numbers

The answer needs hardware numbers — on real RVA23 silicon → (Page 10)

Answering Paul: real RVA23 silicon (SpacemiT K3 / X100)

no real silicon, no answer — the hardware is what unlocked the reviewer's gate

SPACEMIT 进迭时空

first buyable RVA23 silicon

2024 — Spike/QEMU only

Jan 2026 — Paul: bring hardware numbers

Apr 2026 · K3 ships globally — the enabler

May 2026 — K3 + K1 measured →

Spike's 4.9% (sim) → measured on silicon:

up to -42% per-primitive · -2.0% whole-kernel instructions

+ reproduced cross-uarch on K1 X60 (non-RVA23).

Three Hardware Setups

3 / 3 ✓

✓ **RVA23-compiled kernel**
RVA23=y · -march=..._zba_zbb
× RVA23 hw — K3 X100

✓ **vanilla mainline kernel**
stock build · no Zba/Zbb
× RVA23 hw — K3 (controlled)

✓ **RVA23-compiled vs vanilla**
reproduced cross-uarch
× non-RVA23 hw — K1 X60

ANSWERING PAUL · THE NUMBERS (backup)

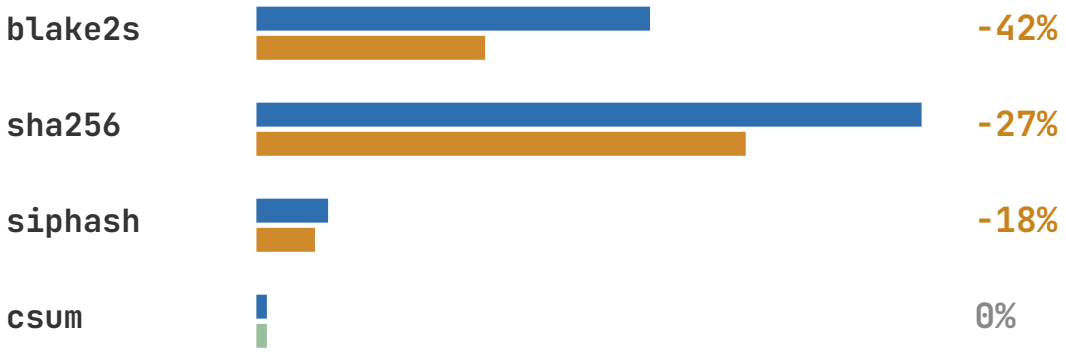
Compile-time Zba/Zbb on K3 X100

✓ cross-uarch: K1 X60 reproduces dd -0.44% / -3.11%

A = RVA23-compiled
Zba/Zbb in CFLAGS → compiler emits them in plain kernel C

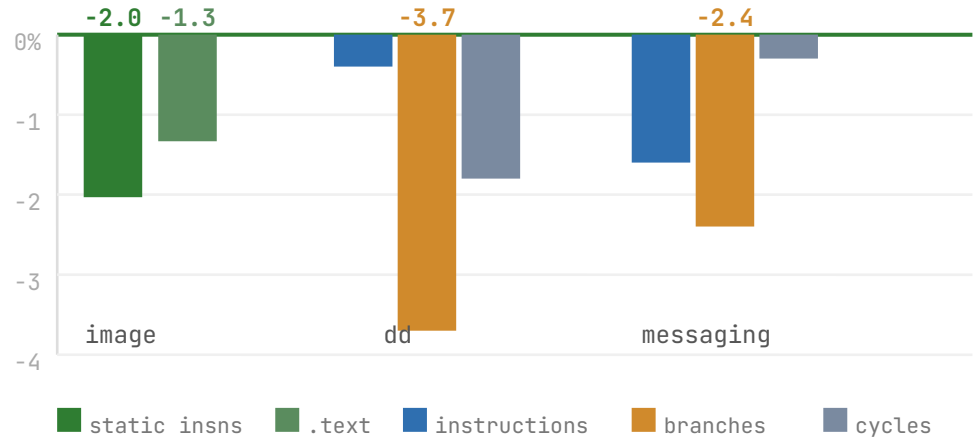
B = vanilla baseline (control)
same .config — only difference: Zba/Zbb NOT in CFLAGS

FASTER · ns to hash 4 KiB · shorter = faster



csum already has a runtime Zbb path → 0%. Wins live in pure-C primit

SMALLER · % reduction (A vs B) · down = less



WHY IT SHRINKS · static codegen, whole vmlinux (A vs B)

+68,054
compiler Zba/Zbb ops in kernel C

1 Zbb op = ~ 2.3 base ops → fewer, denser

blake2s_compress_generic (insns)

-32% (matches the -42% dynamic)

same math on X60: dd -0.44% · siphash -37.8% (narrower pipe → bigger win)

Distros: one moved to RVA23 — three are waiting for the hardware

Debian · Fedora · Red Hat — still here

RV64GC (RVA20) — legacy baseline

Ubuntu 26.04 LTS arrived

RVA23 — new baseline

Debian

RV64GC · Trixie added riscv64 wait until RVA23 hw is mainstream (build-machine migration risk) · Forky ~2027

Aurelien Jarno, via Yu Bo (ISCAS)

Fedora

RV64GC · RISC-V SIG explicitly paused the RVA23 transition

Feb 2026 SIG · koji stays rv64gc

Red Hat (RHEL 10)

RV64GC · developer preview

RHEL 11 ~2028

Ubuntu

RVA23 · first LTS to mandate RVA23; full archive rebuild

"it just works." — Heinrich (Canonical)

More shipping RVA23 silicon (like K3) is what moves the waiters → adopters.

Thank you

> with thanks to the community

AJ **Andrew Jones**
Qualcomm · runtime

CD **Conor Dooley**
Microchip · riscv DT

CJ **Charlie Jenkins**
Rivos→Meta · kernel dev

HS **Heinrich Schuchardt**
Canonical · Ubuntu

PW **Paul Walmsley**
riscv maintainer

SH **Samuel Holland**
SiFive · Supm review

Guodong Xu
RISCstar · RISC-V International Advocate · docularxu.github.io

BACKUP · CROSS-UARCH DETAIL

K1 X60 (non-RVA23) reproduces the K3 win

K1 = SpacemiT BananaPi-F3 · X60 · non-RVA23 (RVA22-class, has Zba/Zbb) · A1 = RVA23-compiled (generic+Zba/Zbb), G = vanilla baseline · 2026-05-31

CROSS-UARCH · dd (A1 vs G), retired

metric	K3 X100	K1 X60
instructions	-0.41%	**-0.44%**
branches	-3.71%	**-3.11%**
cycles	—	-2.30%

→ matches to 3 sig figs (different core, same win)

K1 per-primitive (A1 vs G), ns/op

blake2s	**-39.3%**	(K3 -41.1%)
sha256	**-29.6%**	(K3 -26.6%)
siphash	**-37.8%**	(K3 -17.7%)
csum	**0%**	(runtime alt)

messaging: instructions -2.61% · branches -3.40%

Narrower X60 pipeline → some primitives gain MORE: siphash -37.8% vs K3 -17.7%.

zbb helps entry-level (non-RVA23) cores at least as much, not less.

✓ CHECKSUM_KUNIT pass:5 fail:0 — all six boots (K3 A,B,G,A1 + K1 G,A1)

caveats: only G & A1 boot K1 (A/B's RVA23=y traps on K1's missing exts) · K1 GNU dd vs K3 busybox dd · K1 maxcpus=4 vs K3 8 → messaging absolute counts differ · clocks/IPC differ (K1 ~1.59 GHz, K3 ~2.18 GHz)