

RVA23 in practice: discovery, dispatch, and the hardware left behind

Guodong Xu, Qualcomm LPC 2026 RISC-V microconference

| [Linux v7.0-rc1: RVA23 mandatory extension coverage reaches 100%](#) (blog, 2026-03)

| [RVA23: From Ratification to Real-World Readiness](#) (blog on silicon and distros, 2026-08)

| [riscv: hwprobe: Expose RVA23U64 base behavior](#) (with Andrew Jones; v8 on linux-riscv, 2026-09)

2024-10

RVA23 ratified

2026

first RVA23 silicon

2026-07

glibc 2.44: RVV ifuncs

today

Ubuntu at RVA23
others at RV64GC

**The profile exists.
User space still cannot ask for it.**

Who knows the baseline?

	x86-64	arm64	RISC-V
named level	x86-64-v2 / v3 / v4	Armv8.x-a, Armv9.x-a	RVA20, RVA23
Debian, Ubuntu, Fedora, RHEL build for	v1; RHEL 10: v3	Armv8.0-A, all four	RV64GC Ubuntu 25.10+: RVA23
who computes it	glibc	nobody: apps see feature bits only	glibc, libgcc, each app: each on its own

RISC-V, two answers to "is this RVA23?", 39 mandatory extensions

GCC 16.1 libgcc: 24 checked

6 not

9 invisible

Canonical's checker (xypron/hwprobe): 40 checks, 3 hwprobe keys, needs kernel 7.3 for five of them

Zic64b: no bit

"Is this machine RVA23?" One bit from the kernel

```
riscv_hwprobe(KEY_BASE_BEHAVIOR) → value = IMA | RVA23U64 /* bitmask */
```

computed once at boot, from the full ISA bitmap

RFC v1 by Andrew Jones; carried on by Guodong Xu, now at v8

<https://lore.kernel.org/r/20260920-rva23u64-hwprobe-v2-v8-0-5f14bc5c23d3@oss.qualcomm.com>

```
IMA      = RISC_V_HWPBROBE_BASE_BEHAVIOR_IMA      _BITULL(0)    rv64ima, user ISA 2.2, priv 1.10
RVA23U64 = RISC_V_HWPBROBE_BASE_BEHAVIOR_RVA23U64  _BITULL(1)    all mandatory extensions of RVA23U64, profiles spec 0273f3c921b6
```

Ask 1 kernel maintainers: is this the right shape?

Dispatch, per function: arm64 vs the RISC-V glibc plan

arm64 today	RISC-V, Peter's plan
<pre>if (sve) { /* feature */ if (A64FX) /* CPU model */ return a64fx_version; return sve_version; } if (ORYON) /* CPU model */ return oryon_version; return generic;</pre>	<pre>if (RVA23) { /* profile */ if (TENSTORRENT) /* CPU model */ return tenstorrent_version; if (SIFIVE) /* CPU model */ return sifive_version; return zbb_version; /* design-time */ } return generic;</pre>
<pre>1st: feature bit 2nd: CPU model (MIDR)</pre>	<pre>1st: profile 2nd: CPU model (mvendorid, marchid) who answers "RVA23"? kernel or glibc: ?</pre>

Dispatch, whole library: x86-64 vs RISC-V with glibc-hwcaps

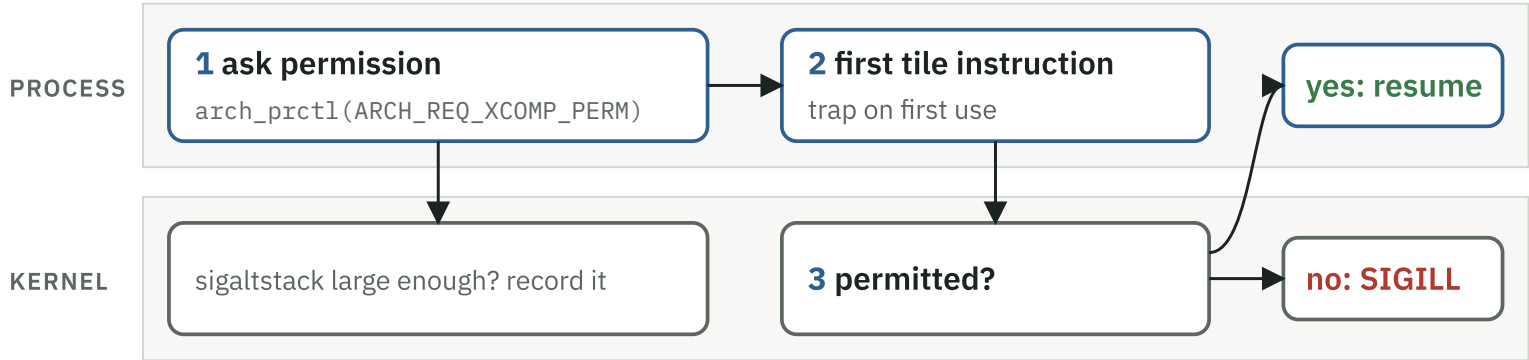
x86-64 today	RISC-V, with glibc-hwcaps
-----	-----
ld.so, at start-up:	ld.so, at start-up:
<pre>level = from CPUID /* glibc */ if (level >= v3) /* psABI */ open glibc-hwcaps/x86-64-v3/libc.so.6 else open libc.so.6</pre>	<pre>level = from kernel RVA23U64 bit /* hwprobe */ if (RVA23) /* profile */ open glibc-hwcaps/rva23u64/libc.so.6 else open libc.so.6</pre>
inside x86-64-v3/libc.so.6, memcpy resolver:	inside rva23u64/libc.so.6, memcpy resolver:
<pre>if (AVX512_prefer) /* CPU pref */ return avx512_version; return avx2_version; /* v3 guarantees */</pre>	<pre>if (TENSTORRENT) /* CPU model */ return tenstorrent_version; return rvv_version; /* design-time */</pre>
-----	-----
1st: psABI level, whole library 2nd: CPU preference, per function	1st: profile, whole library 2nd: CPU model, per function
level computed by glibc from CPUID	level computed by the kernel

Ask 2 distros: ship `rva23u64` builds of selected libraries?

Three enable models. One exception.

	always-on	dynamically enabled, per process x86 term	opt-in
x86-64	AVX-512 state buffer from task creation, no trap	AMX	CET
arm64	SVE, SME first-use trap allocates the state		BTI, GCS, MTE
RISC-V	V, revocable first-use trap allocates the state		Zicfilp, Zicfiss

AMX on x86



RISC-V V, revoked by prctl: AT_HWCAP sees it, hwprobe does not. SIGILL before main.

From the 09-16 WG call. What is open.

DISCUSSED, RISE PLATFORM WG CALL 09-16

No opt-out for future extensions.

Signal frame: walk the chain, fail closed. Samuel Holland

OPEN, FOR THIS ROOM

Deprecate the V prctl: discussed on the call, not proposed on linux-riscv. Peter Bergner, Andy Chiu, Samuel Holland

Matrix: always-on, or dynamically enabled per process, as x86 chose for AMX?

Who writes the contract into Documentation?

Ask 3 kernel maintainers: deprecate the V prctl?

Ask 4 one enable model for future extensions: always-on, or dynamically enabled per process?

Next steps, as I see them

item	status	closest to it
hwprobe RVA23U64 bit	v8, in review	RISC-V maintainers; Guodong Xu answers
deprecate the V prctl	to propose	kernel: Andy Chiu, Samuel Holland
signal-frame contract, Documentation	to write	kernel: Andy Chiu, Samuel Holland, or me
glibc, profile-based rework	in progress	glibc: Peter Bergner
libgcc, read the profile bit	to discuss	GCC: Jeff Law
rva23u64 library builds via glibc-hwcaps	pending kernel merge of the RVA23U64 bit	distros; Debian, and all

- Ask 1 kernel maintainers: is the RVA23U64 bit the right shape?
- Ask 2 distros: ship `rva23u64` builds of selected libraries?
- Ask 3 kernel maintainers: deprecate the V prctl?
- Ask 4 future extensions: always-on, or dynamically enabled per process?