Feb 23, 2023 | RISC-V Perf Analysis SIG Meeting

Attendees: Beeman Strong tech.meetings@riscv.org Marc Casas

Notes

- **Attendees**: Beeman, JohnS, BryanV, BruceA, Ved, Robert, Atish
    - Intro: Bryan Veal, part of Intel Foundry Services, team doing performance analysis
- **Slides** here (no video, apologies)
- Quickly viewed proposed standard events from Oct '22, but will leave further refinement to a future TG
- Event discovery
    - All architectures have adopted JSON event file format, used by Linux kernel (and other tools) to provide event name, desc, encoding, etc
    - Proposed that RISC-V standardize event names and descriptions, and lean on event file to provide encoding/config info
        - Maintains flexibility for implementations to choose best encodings for them
    - Ved: JSON should indicate if the event is precise, or if it counts across harts. Could be used to gate exposure of the event to a VM.
    - Robert: we should look at unified discovery
        - But not for events
    - Robert: events for SiFive are from a bigger file with other stuff
- Required events
    - Could include required events as part of an extension, or could have a PMU version (like ARM) that dictates the required events
    - Ved: or could let profiles dictate which events are required
        - Consistent with including events as part of extensions
        - Robert: must have all or none of an extension
        - Ved: or profile could specify required events on a per-event basis. Doesn't have to be a bucket of events as part of an extension.
            - Standard event list doesn't specify what is required
        - Robert: profiles can't specify which optional parts of profiles
            - Ved: RVA profile does so
        - Then profiles can deprecate an event
        - Agree with this approach. TG will develop a list of standardized events, then profiles can determine which (if any) are required
- Counter/event asymmetries
    - Occur when not all events can be counted by all counters. ARM doesn't allow them, Intel does (and indicates them in the event file)
    - Bruce: SiFive uncore blocks have own counters and events
        - Uncore PMUs require own driver for now, not part of this discussion. Should look at standardizing in the future.

        - Robert: fixed counters are essentially asymmetric counters. Should have flexibility to support all implementations, which may have asymmetries. Not that hard.
            - Ved: agree, allow for most efficient implementations
            - Atish: asked question to mailing list. PeterZ (maintainer) says it's a big pain to manage these constraints. They tend to grow, x86 keeps adding more. Prefer ARM approach of all-to-all.
            - But if building an asymmetric counter there must be a good reason
                - avoids hierarchy of muxes
            - Want to allow for custom extensions, like a fixed-counter, that really requires asymmetry
                - Don't want a dedicated driver to use it
            - Ved: could define allowed asymmetries. E.g., allow which events are supported per counter, but not whether priv mode filtering is supported, or width, per counter.
            - Intel didn't define counter masking from the beginning, had to do too much based on CPUID family/model/stepping before
            - Atish: AMD has weird cases of overlapping constraints, we should get more info on that
- Bruce: distinguish between edge (or occurrences) events vs cycle events
- Metrics
    - Formulas of events, now published by vendors
    - Should develop standard metrics, use them to drive need for standard events
- **Group agrees that a Performance Events TG is warranted, to define standard events and metrics**

Action items

- [ ] Atish Kumar Patra - Aug 25, 2022 - check on how to read multiple counters in perf when taking an interrupt on one
- [ ] Beeman Strong - Jul 28, 2022 - Reach out about proprietary performance analysis tools
- [ ] Beeman Strong - Jul 28, 2022 - Reach out to VMware about PMU enabling
- [ ] Beeman Strong - Jul 28, 2022 - Talk to security HC about counter delegation