Oct 3, 2024 | RV Performance Event Sampling TG

Attendees: Snehasish Kumar tech.meetings@riscv.org Beeman Strong

Notes

- **Attendees**: Beeman, Ruinland, Chun, Snehasish, Dmitriy, Daniel, Atish
- **Slides/video** here
- Recap last week
    - Sample PC is useful with skidded implementations
        - Sample PC + CTR can help patch up call-stack, not sure if sample PC alone
            - Maybe sample PC is enough if only leaf node is different? Would require short skid. Google doesn't do this today.
            - Can patch with CTR freeze on trap in regular mode, or CTR freeze on overflow with RASEMU=1
    - Want an extension to indicate precise LCOFI
        - With some allowance for event (not cycle/inst) skid
    - Spec should indicate hierarchy of precision, and value of minimizing skid
        - Where did Intel announce improved skid? Beeman to check
- Proposed Ssplcofi (precise LCOFI) extension
    - Gives SW a way to indicate that LCOFI is precise
    - Will allow some event skid, where LCOFI can be taken after inst that causes overflow, or on some (shortly) later inst that still incurred the same event
        - Does not have cycle skid, can always associate epc with the event
    - Can do this because LCOFI is local. ARM had to use exceptions with SEBEC, so they could be synchronous.
        - Exceptions are before retire. SEBEC exception taken on overflow inst or next instruction? Beeman to check
    - Propose that Ssplcofi supports RET events only
        - SPEC and other events aren't guaranteed to trap following an inst that incurred the event, because the uarch doesn't track which insts get them
        - Reasonable to expect cycle skid on be reduced for SPEC events with this extension, since uarch supports trapping "on a dime". But no guarantees.
- Proposed Sspesa (precise event sample attribution) extension
    - As discussed last week, records only the precise sample PC
    - Is a step back from Ssplcofi in terms of precise, but cheaper to implement, and likely less impact to SW execution
    - Only records precise sample PC for a single counter's overflow, will indicate which in CounterID field
        - Multiple counter overflows for a single LCOFI seems like to rare of a corner case to bother to support (would need 2 CSRs per counter)
        - Sampling on multiple counters is believed to be quite rare. Even leader sampling only has one counter in sampling mode, others counting.

    - If an implementation has >32 counters (or if RV later standardizes more counters), would add another bit(s) to CounterID
    - Proposed that Sspesa and Ssplcofi are mutex. Is there value in using them together?
        - Because epc for Ssplcofi is always an inst that got the event, can't think of any reason to ever use Sspesa CSRs for sample PC
        - Worth more thought here
    - Even with Sspesa, no way to precisely attribute SPEC events?
        - No, because uarch didn't keep track of which inst got it
        - And sometimes there is no inst (e.g., a prefetch miss)
    - This could also help with profiling while interrupts are masked
        - Already have SBI's SSE for that, but if not using that for some reason then it's true, this will capture where the sample should have been despite interrupt skidding until interrupts are unmasked
- How can SW know the event skid with Ssplcofi?
    - Can check the counter value. If 0 then no event skid, >0 means some event skid. SW could then adjust reload value to keep sample rate consistent.
    - If counter is enabled for S-mode, this won't work bc counter will count events in LCOFI handler. Would need counter freeze. Later slide.
- Some confusion about whether Ssplcofi has skid or not, should improve terminology. Propose:
    - Event skid - LCOFI taken not after instruction that increments counter to 0, but after instruction that increments counter to some value >0. So sample inst (at epc) still got the event. This can simplify implementation, allowing HW to arm on overflow then trap on the next event occurrence.
    - Cycle skid - LCOFI traps some number of cycles after the counter overflows. No easy way to know which inst caused overflow, or got the event. Sspesa allows cycle skid, but records the sample PC.
- CounterID should be XLEN-relative rather than 63, or at bottom bits, to accommodate RV32
- **Out of time**, will continue next time

Action items

- [ ]