Aug 15, 2024 | RV Performance Events TG

Attendees: Beeman Strong Dmitriy Ryabtsev tech.meetings@riscv.org

Notes

- **Attendees**: Dmitriy, Beeman, Snehasish, Bruce
- **Slides/video** here
- Breakdown of TOPDOWN.BAD_SPEC
    - Add CONTROL_FLOW and MEM_ORDERING sub-events
    - Any other sources would be represented by OTHER metric, should be very small %
    - Intel has L2 events sum to L1, but L3/L4 are often estimate metrics and do not
        - E.g., CORE_BOUND and MEM_BOUND sum up to BACKEND_BOUND
- Breakdown of MEMORY_BOUND
    - Add ADDR and DATA sub-events
        - ADDR sub-events: AGEN, L1TLB, L2TLB, …, MMU
        - DATA sub-events: L1, L2, L3, …
    - AGEN probably won't get slots, but covers all address
    - Haven't added TLB levels yet
        - Last level could be PGWALK, TLBMISS
            - TLBMISS sounds like L1TLB, so prefer PGWALK
            - Or just LxMISS?
        - How are these counted?
            - If blocking op doesn't have a VA, count towards AGEN
            - If blocking op has a VA, count just like L1/L2/etc cache events (formulas based on L1TLB_MISS, L2TLB_MISS, etc)
    - How many levels of cache/TLB to define? How will tools know which ones apply?
        - Should define enough for any foreseeable implementation (implementations can always add custom additions if needed)
        - Tools will know which are supported using the json file
    - How to deal with different memory end-points?
        - Cache misses aren't always to DRAM, may hit HBM, device, CXL, …
        - For implementation that support those, may be nice to know which end points are causing bottlenecks
            - But such events are expensive to implement. Without a simple hierarchy, can't use formulas of X_MISS events, like what's proposed for L1_BOUND, L2_BOUND, etc.
            - Instead, have to count cycles/slots for LLC miss, and wait for data to return with the data source. Then add the accumulated count to the proper event.
            - Some implementations may opt to support that, but many will not. Leave it up to them

        - Easiest solution is to simply use L3_MISS to catch everything beyond the caches
            - What about implementations that have a different number of caches? Better to use LLC_MISS?
            - LLC is confusing since the level varies
            - Already need Lx_MISS events for Lx_BOUND formulas, so just use whichever is the last one as the catch-all for external access
    - L1_BOUND/L2_BOUND implementation
        - L1_BOUND like just MEMORY_BOUND.DATA - L1.MISS, L2_BOUND = MEMORY_BOUND.DATA - L2.MISS, …
    - Fix L2_BOUND formula typo
    - Should rename DRAM_BOUND to something more generic
        - EXT_BOUND? EXTERNAL_MEMORY?
- Where does serialization fit in to TMA?
    - FENCE.I will naturally cause FE_BOUND, since it blocks fetch while waiting for stores
    - FENCE/FENCE.TSO arguably should be MEM_BOUND, since waiting on mem ops to complete
    - CSR write serialization could be CORE_BOUND
    - Intel lumps all serialization into CORE_BOUND
        - Not sure they have anything like FENCE.I?
        - May be due to original MEM_BOUND implementation being formulas based on cache misses, etc. So everything else just fell in CORE_BOUND
    - Want to support both simple and complex implementations, but don't want to require something unnatural
        - E.g., making FENCE.I be CORE_BOUND would be unnatural because it would cause the FE to not deliver uops, which is fundamentally FE_BOUND
    - **AI** Beeman (et al) to check with implementers on what might work best here
- Exception events
    - Not sure how useful for performance, other than maybe page fault and misaligned
    - Switch "load page fault" to "data page fault", since even post-retire stores typically do translation before retire
    - Have all exceptions event?
        - Yes, could have EXCEPTIONS.ALL
    - This could complicate virtualization use
        - A VM can count most exceptions, but shouldn't see virt-inst exceptions or guest page faults (second-level), otherwise virtualization hole
        - Want to avoid standardizing events that might require a hypervisor to trap on event selector writes to filter the value
        - **Out of time**, finish this discussion on mailing list

Action items

- [ ] Beeman Strong - Aug 15, 2024 - check on implementation of TMA events for serialization
- [x] ~~rdb197@gmail.com - Aug 8, 2024 - Add MEM_BOUND breakdown to PR for review~~
- [ ] rdb197@gmail.com - Aug 8, 2024 - Include Bfloat16
- [ ] rdb197@gmail.com - Aug 8, 2024 - Check on single/double vs 32/64 terminology for FP events
- [x] ~~rdb197@gmail.com - Aug 8, 2024 - Check on including metrics in metric formulas~~
    - Answer: nested metrics are allowed
- [ ] rdb197@gmail.com - Aug 8, 2024 - Consider adding events for VSETVL, div/sqrt, etc
- [ ] Beeman Strong - May 23, 2024 - check on idea to count remote HITMs locally