Apr 4, 2024 | RISC-V Perf Analysis SIG Meeting

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

Notes

- **Attendees**: Beeman, MattT, Snehasish, BruceA, BoG, ChunL, Dmitriy, DanielB
- **Slides/video** here
- Opens/updates
  - Smcdeleg/Ssccfg ratified
- Quick review of sampling extension plans
- Picked up with options for storing sample data discussed last time
  - CSRs
  - Store to memory
  - Store to HW buffer, SW flushes to memory
- Complexities with store to memory:
  - Handling page faults
    - Not due to the current instruction
    - Need to resume uarch flush after faultk
  - Multiple addressing modes for various usages (virtual, phy, guest phys)
  - Interrupt on (almost) full
  - Wrap vs stop on full
- Page fault could undermine perf benefit of memory approaches
- Another option: push data to N-trace, SiFive likely to do this
  - Agreed that output to trace is valuable, but don't want this to depend on implementation of trace. So want to have base mechanism, but a further extension could enable pushing it to trace instead.
- Why do memory for this but not for CTR?
  - Agree. CTR has more data.
- Could pin the memory buffer to avoid page faults
  - Or use phys addressing
- ARM AMU has very low overhead, just free-running counters
  - Useful for DVFS
  - Could we do something as low-overhead as AMU, for usages like fleet-monitoring where PMU might be always-on?
- Google typically samples for 10-30s, with sample rate chosen to keep overhead <5%
  - Different use case from typically perf troubleshooting
  - Call-stack is most important thing, so hard to avoid interrupts
- Go with CSRs. We could add an output to memory option in the future, if we have a way for HW to collect call stack, and hence sampling without interrupts could be the common case
  - Options for call-stack that we've discussed:
    - CTR RASEMU mode - tools today use analogous LBR call-stack mode only as complementary with SW call-stack, since prone to corruption

        - CFI shadow-stack - new, would need broader adoption (despite perf overhead). Would still require a memcpy, can HW really do this faster than SW?
        - Copy SW stack - ARM considered this, not sure it was ever ratified/implemented. Again, this is just a memcpy.
- Context switch
    - If timer interrupt is taken between save of sample data and subsequent interrupt
    - This window should be very small, since LCOFI should be precise with instr sampling. Essentially requires a higher-priority interrupt on precisely the same instr.
    - SSE would reduce problem window, since only higher-priority M-mode interrupts could intervene
    - IBS also stores sample data in CSRs, check what perf does for it
        - PEBS/SPE are different, since they store to memory. That's an advantage to that approach, SW doesn't need to save/restore (assuming the implementation guarantees that those stores always complete before any interrupt can be taken)
    - Simplest approach would be to drop, especially if this is really rare
    - Could be like CTR, only save/restore when in use. Just adds a check, compiler-optimized.
- **Concluded the strategy discussion around sampling**. Next step will be to form a TG to work out the details. Let Beeman know if you are interested in a chair role.

Action items

- [ ] Beeman Strong - Jul 28, 2022 - Reach out about proprietary performance analysis tools
- [ ] Beeman Strong - Jul 28, 2022 - Reach out to VMware about PMU enabling