This document is capture of live chat session. "RC: … answer here…" answers were added after.

NOTE: I removed personal information from this file.

From … to Everyone: 02:10 PM

How much is the IAR adapter?
Can IAR do reverse execution?

RC: We provide IAR adapter to our individual RISC-V customers. I will ask internally about possible selling it individually.

RC: In general nobody can fully "reverse execution" as certain actions (like sending character by UART). However debugger allows navigation over recorded trace back and forth. This tutorial was not focused on 'how to use trace', but rather 'how to implement trace'.

From … to Everyone: 02:24 PM

What is uninferable-jump?
unconditional jump?

RC: Uninferable jump is a jump, where destination address cannot be inferred from binary code. So address must be sent 'somehow' in trace packet – easiest way would be to send full-address, but it is usually sent somehow 'compressed'.

From … to Everyone: 02:25 PM

jump to register -- requires the
new address to be transferred

RC: That's correct. This is an example of uninferable jump.

From … to Everyone: 02:26 PM

uninferable jump is one in which the destination cannot be determined from the opcode

From … to Everyone: 02:27 PM

The destination of an inferable jump can be determined simply from the opcode

RC: Right.

From … to Everyone: 02:27 PM

Thanks

From … to Everyone: 02:44 PM

Return stack, recent address cache, branch prediction, all exist in many microarchitecture. Does the encoder replicate this hardware, or does any implementation actually use this hardware?

RC: The idea is to have trace encoder as much 'separated' from core as possible (via ingress port 'wires') – however nothing prevents to have specific encoder implementation to have 'private-links' to get/share more status from/with core.

From … to Everyone: 02:46 PM

The encoder replicates it.

RC: I would say 'almost always', but it would be practically hard.

From ... to Everyone: 02:49 PM

So is E-trace RISC-V specific? (Nexus is architecture independent).

RC: Nexus is 'architecture independent' indeed, but certain 'details' (like exception codes, etc.) may be. Nexus Trace TG is trying to define these 'details'.

From … to Everyone: 02:50 PM

E-Trace is RISC-V specific.
It was optimized for things that are particular to RISC-V.

RC: Technically speaking E-Trace protocol is not 'RISC-V only', but it was designed having RISC-V architecture in mind. So Paul's answer is correct.

From ... to Everyone: 02:52 PM

Does RISC-V have a concept similar to the CoreSight event triggering mechanism (I forget the exact terminology)?

From … to Everyone: 02:53 PM

What type of event? Something related to hart execution or from some source outside the hart?

From ... to Everyone: 02:55 PM

Coresight doesn't assign semantics, but a typical use case would be a hart hitting a breakpoint could cause other harts to stop at the same time. (This is done with the Cross-Triggering Matrix).

From … to Everyone: 02:57 PM

That's part of the debug spec, not trace. Yes, that type of thing is supported. I think it's not quite as full-featured as CTM but it handles the use-cases that I've used CTM for.

RC: I had a 'bonus-slide' about triggering and I wanted to show it in case I will have time. I will post these bonus-slides somewhere. In general core watchpoint logic (used for debugging …) reserve 3 'action-types' for trace - it is very simple although very powerful. It is up to trace encoder to define what to do with these events (original 'debug spec' defined these 3 as start/stop/event, but it was made more generic).

RC: RISC-V debug architecture does not define anything similar to CTM, but at certain point of time it most likely will. CTM events are used to cooperate with trace.

RC: Encoder may also implement other triggers (for example start trace on TRGIN pin on trace connector).

From … to Everyone: 03:02 PM

And it may require hooking things up in a particular way. Like if a trigger (breakpoint) hooks to input of the HPM (performance counter) then upon counter overflow that can put info into the trace stream (if HPM is connected to the encoder) and/or halt this hart (if HPM output hooks to the trigger module) and potentially other harts (if the debug module is hooked up in the right way).

RC: Exactly – trace is accepting different triggers and it is up to Trace Control layer to define what to do with them. Usual events (like core start/stop) are reported by ingress port and will cause trace to stop/start. Possibilities are endless. However in certain situations you collect entire trace and analyze it later.

From … to Everyone: 03:03 PM

May I request if we can have the presentation, we are totally new hearing about IAR. I plan to evaluate the tool.

RC: Presentation will be provided for everyone (either on RISC-V or IAR pages).

From ... to Everyone: 03:06 PM

Can SW output log e.g. ASCII messages through the trace infrastructure as well?

RC: Yes. This is NOT yet 'elaborated', but for example SiFive implemented it. I was showing this during short demo.

From … to Everyone: 03:07 PM

What day was the Nexus Trace talk?

RC: This is live – video and presentation will be provided later (hopefully soon).

From … to Everyone: 03:13 PM

E-Trace is currently working on data trace. I haven't followed that closely but it may (or may not) enable ASCII messages by storing ASCII characters to some special address being traced. The pmp "benchmark" is not a benchmark. It's a directed test that tests whether memory accesses properly trap based on how PMP (physical memory protection) is setup. So there are lots of mode changes and traps. It's not going to compress well and it's not representative of any real program.

RC: That 'ASCII messages' may be implemented as 'selective-data-trace' (write to specific address/address range is 'intercepted' and trace packet is created). But it may also be on dedicated interface, where core has dedicated 'peripheral', which provides ASCII bytes to encoder via 'private' side-channel as Ingress Port does not define it.

RC: Current 'E-Trace' data trace work is focusing on full-blown data trace (which is very different 'animal' than intercepting and sending some ASCII bytes once in a while …).

From … to Everyone: 03:13 PM

Is there anyone who knows if any open source trace encoder in HW description language is available now?

RC: Very good and important question. I may only say 'I hope for it'. IAR provide C-model (link in presentation) for Nexus encoder. I wrote core of that encoding algorithm in such a way that it may be re-written in Verilog/VHDL easily.

From … to Everyone: 03:13 PM

The trace compression ratio is not just about the programs, but also the choices made by the compile in code generation.

RC: Certainly! Code produced by different compilers and/or different optimization options may trace very differently. That's why publishing trace benchmark without saying what was exact compiler and compiler options may be very misleading. May idea for providing NexRv tool (link to github in presentation) was to see what will be compression ratio for your ELF file (which is your application).

From ... to Everyone: 03:15 PM

Nexus has the Data Acquisition Message (DQM) which supports tracing data values. The SiFive trace supports Instrumented Trace using this message.

RC: Right – I was showing this during live demo portion. E-Trace does not define that functionality. This was really something what made SWO trace for Cortex devices very popular!

From ... to Everyone: 03:19 PM

Are the "Aurora" high-speed links still of interest for trace?

RC: “Aurora” is part of original Nexus specification. Nexus Trace TG is NOT discussion that option, but it should be followed. This is just very fast way of sending data out of chip. But basic question is what to do with that amount of data? Nexus Trace TG does not want to step into that territory. It took many years after Arm defined HST (High Speed Trace).

From … to Everyone: 03:25 PM

How do people switch off trace securely (i.e., it's useful for debugging, but it's a security hole)?

RC: Security is not actively discussed, but certainly cannot be ‘neglected’. As I said verbally either core may ‘turn-off’ ingress port or trace encoder may ‘filter-out’ secret code. One may imagine, you need to do some ‘cryptographic’ authentication before you will be allowed to turn on trace. Trace without ELF program is not very useful, but certainly some ‘secrets’ can be revealed by looking at it.

From … to Everyone: 03:28 PM

Looking forward to downloading the slides

RC: If you are reading this, then you probable did it already ☺.

From … to Everyone: 03:35 PM

good presentation and lot of support links than you

From … to Everyone: 03:36 PM

if anybody need to join and contribute to Trace

From … to Everyone: 03:37 PM

Qns:what is the technical background needed to participate with trace like a VLSI graduate?

RC: Anyone may participate (and contribute!) to any RISC-V groups.

From … to Everyone: 03:49 PM

Thank you very much.

From … to Everyone: 03:50 PM

thanks for the elaboration and Q &A

From … to Everyone: 03:50 PM

Thank you for joining us today. Please return to the agenda to find your next session.

RC: Thank anyone for attending and providing live-answers via chat. I hope these Q&A answers (post-factum …) are useful.