



DTPM SIG

07-Jan-2025

Meeting minutes are in the speaker notes for the relevant slide



Disclosures

tech.riscv.org



Agenda

- Disclosures
- Status
 - HTI fast track status
 - Awaiting ARC approval
 - E-Trace maintenance
 - Most PRs approved. A couple of new ones still to address.
- Instrumentation Trace
 - Charter discussion
- AOB

PD Paul Donahue (Ventana) (Me)

IR Iain Robertson (Siemens)

JG Jay Gamoneda (NXP)

BS Beeman Strong (Rivos)

BG Bo Gan (Individual)

BA Bruce Ableidinger (MIPS)

 Carsten Gosvig (SiFive)

MS Michael Schleinkofer (Lauterbach)

 Rafael Sene (RISC-V)

RC Robert Chyla (MIPS)

Instrumentation Trace – Draft Charter

As RISC-V cores continue to find homes in embedded, RTOS and many core systems, users will need the ability to accurately monitor device activities. In large SoC's full program trace of each core may not be practical or precise enough. This can be achieved by directly instrumenting execution code some type of *printf-like* event. Competing ISAs already offer similar *printf-like* trace support. The ARM-M family of cores contain the Instrumentation Trace Macrocell Unit which generates trace packets with direct writes to "ITM stimulus registers". Motorola/Freescale PowerPC cores provide *printf-like* capability using the Nexus standard, generating trace message when an specific ISA register was written. The goal of the Instrumentation Trace TG will be to provide the RISC-V community a standard guideline for *printf()* trace support.

Instrumentation trace gives the user a *printf()* style debugging tool which can be used in embedded cores, Real Time Operating Systems, or applications which are to be monitored. Instrumentation trace allows users the ability to timestamp specific events in code which might be missed with standard program trace messages. When not enabled, the code to generate the instrumentation trace message will run as a NOP. The trace messages can be stored locally in a RAM to be processed later, or egress off chip via the PIB module. The TG will propose a standard way for a core to produce the *printf()* event/message with either a memory mapped region, local registers, or possibly an instruction.

The proposed RISC-V Instrumentation Trace TG will collaborate to produce:

1. A trace agnostic definition of instrumentation trace
2. Extension to Trace Control Interface to support instrumentation trace
3. Extension to N-trace specification to include new packet message
4. Work with E-trace representatives to ensure compatibility/support

The TG will work together with the Debug, Trace, and Performance Monitoring SIG to complete the instrumentation trace details.



Instrumentation trace charter discussion:

printf is one use case but is too prominent in the draft charter. Also, mentioning other ISAs may be inappropriate.

We may not be able to make this a fast-track if the scope is not narrow. Ultimately, it's not up to us. Let's figure out what we want to do and then we'll let the appropriate people decide whether this is eligible for fast-track.

If we do the MMIO store then it's not really a NOP if it's disabled. It still does the store but it's a NOP

with respect to trace. Advantages of MMIO is that there's no change to the hart at all. This capability can be added to existing harts. Having an instruction would allow it to be non-intrusive and pass data to the trace encoder without going through the memory subsystem. Other than PTWRITE, the industry seems to favor MMIO.

Consensus that we should define both ways.

If we're extending N-Trace then we also need to extend E-Trace as more than an afterthought. We're talking about adding a new packet definition for E-Trace encapsulation, not E-Trace itself.

Instrumentation trace:

Power Architecture Data Acquisition: two SPRs are defined. One has an 8b tag and the other initiates the message (using the tag from the first register). For RISC-V, we had talked about MMIO and special instructions but this is a third way: CSRs. The CSRs could be implemented in the encoder or in the hart (which then sends info to the encoder as necessary).

Siemens: Static Instrumentation via a set of MMIO

mailboxes. There are groups of mailboxes where high address bits define which mailbox and low bits enable different features within that mailbox. Standalone block, architecture agnostic, can be shared between multiple harts.

Rafael: There's a new process for creating a TG. It may take up to 5 months because it requires approval all the way up to the RVI board of directors. We should focus on technical content and not worry about bureaucracy.

Future Meetings / AOB

- Next meeting is 4-Feb at 10am Pacific (18:00 UTC)
- AOB



Thank You

