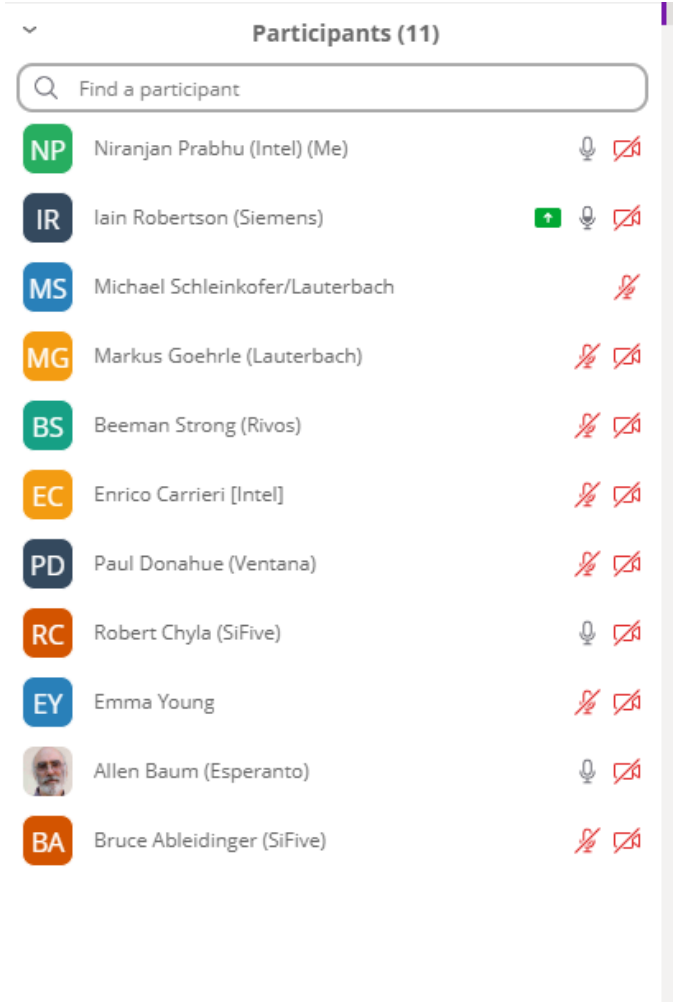

RISC-V®
DTPM SIG
01/11/2023

Agenda

- Disclosures
- Charter
- TG Status
  - Debug, Nexus
- Maintenance
- Proposed Focus Areas
- Gap Analysis Backlog
- Discussion
- Future meetings
- AOB



RISC-V®

Disclosures

Video link

Only RISC-V Members May Attend

- Non-members are asked to please leave except for Joint Working Groups (JWG).
- Members share IP protection by virtue of their common membership agreement. Non-members being present jeopardizes that protection. Joint working groups (JWG) agree that any IP discussed or worked on is fully open source and unencumbered as per the policy.
- It is easy to become a member. Check out riscv.org/membership
- If you need work done between non-members or or other orgs and RISC-V, please use a joint working group (JWG).
    - used to allow non-members in SIGs but the SIGs purpose has changed.
- Please put your name and company (in parens after your name) as your zoom name. If you are an individual member just use the word “individual” instead of company name.
- Non-member guests may present to the group but should only stay for the presentation. Guests should leave for any follow on discussions.

Antitrust Policy Notice

RISC-V International meetings involve participation by industry competitors, and it is the intention of RISC-V International to conduct all its activities in accordance with applicable antitrust and competition laws. It is therefore extremely important that attendees adhere to meeting agendas, and be aware of, and not participate in, any activities that are prohibited under applicable US state, federal or foreign antitrust and competition laws.

Examples of types of actions that are prohibited at RISC-V International meetings and in connection with RISC-V International activities are described in the RISC-V International Regulations Article 7 available here: https://riscv.org/regulations/

If you have questions about these matters, please contact your company counsel.

RISC-V®

Collaborative & Welcoming Community

RISC-V is an open ISA enabling a new era of processor innovation through open standard collaboration. Born in academia and research, RISC-V ISA delivers a new level of extensible software and hardware freedom on architecture, paving the way for the next 50 years of computing design and innovation.

We are a transparent, collaborative community where all are welcomed, and all members are encouraged to participate. We are a continuous improvement organization. If you see something that can be improved, please tell us. help@riscv.org

We as members, contributors, and leaders pledge to make participation in our community a harassment-free experience for everyone.

https://riscv.org/community/community-code-of-conduct/

Conventions

- **For one hour meetings, please start at 5 after the start time** in order to allow people going to other meetings have time for a short break between meetings. 30 minute meetings start on time.
- Unless it is a scheduled agenda topic, we don’t solve problems or detailed topics in most meetings unless specified in the agenda because we don’t often have enough time to do so and it is more efficient to do so offline and/or in email. We identify items and send folks off to do the work and come back with solutions or proposals.
- If some policy, org, extension, etc. can be doing things in a better way, help us make it better. Do not change or not abide by the item unilaterally. Instead let's work together to make it better.
- Please conduct meetings that accommodates the virtual and broad geographical nature of our teams. This includes meeting times, repeating questions before you answer, at appropriate times polling attendees, guide people to interact in a way that has attendees taking turns speaking, …
- Where appropriate and possible, meeting minutes will be added as speaker notes within the slides for the Agenda

Charter

Full verbiage is here. Key points:

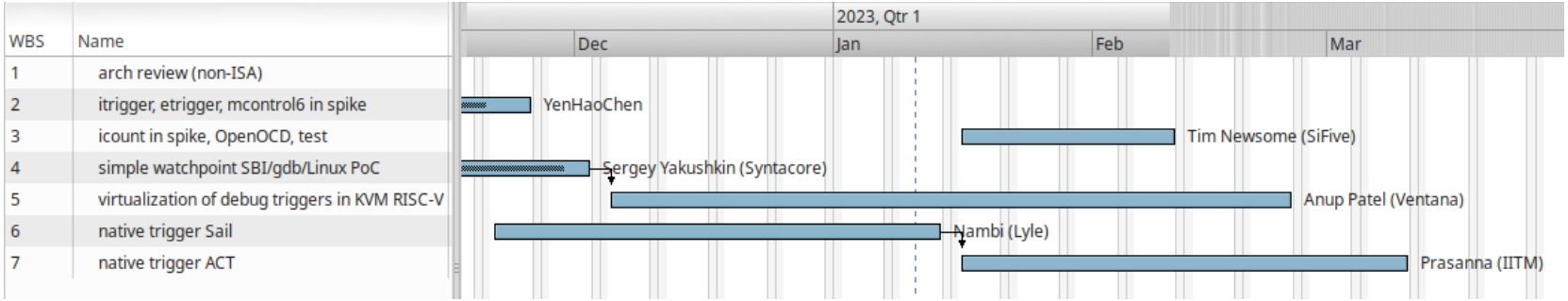
The goal for the DTPM SIG shall be to define a strategy to establish specifications and guidelines for the interfaces, transports, and other pieces of the SoC infrastructure

The DTPM SIG shall focus on the following areas:

- Standard programming interface to aid interoperability
- Protocols and formats for debug/trace/performance-monitoring encapsulation
- Unified security architecture for debug/trace
- Cables and connectors
- Debug, Trace, and SoC performance monitoring features

The DTPM SIG shall work on a gap analysis by starting with a lay of the land review on RVI debug, trace, SoC performance-monitoring capabilities

TG Status - Debug



- Still shooting for Q2 for ratification

- Several minor follow-on items deferred to v1.1:
  - #562, let the debugger flush caches when there is no program buffer
  - #390, optionally don't reset debug CSRs during reset

N-Trace Status (part 1)

- Nexus TG was following full, ambitious and wide agenda:

  The following parts of Nexus specification will be addressed:
  - Nexus compatible trace encoding
  - Trace control
  - Trace configuration
  - On-chip and off-chip trace routing
  - Physical trace connector options

  This group will not address the debug part of the Nexus standard.

- Nexus TG Group clearly wants to ratify all ‘components’ together
  - It was key ambition to provide full end-to-end-trace, consistent and integrated solution
  - During development of this standard, I’ve seen several implementations based on preliminary material.

- Trace Connectors (‘**practically frozen**’)
  - Based on updated (by Nexus TG request!) MIPI standard (+ small Arm-compatible extension).
  - Fully compatible with RISC-V Debug Connectors (pure extensions).
  - Unchanged for long time (some formatting and unification changes still possible).
  - Lauterbach (key player in high-end trace) contributed significantly. Myself as well (when I was with IAR there).
  - Several trace probe vendors OK-ed the spec already

N-Trace Status (part 2)

The following parts of Nexus specification will be addressed:

- Nexus compatible trace encoding
- Trace control
- Trace configuration
- On-chip and off-chip trace routing
- Physical trace connector options

This group will not address the debug part of the Nexus standard.

- Official TG agenda (as a reminder …) in the upper right corner …
- Control, Configuration, On-chip, off-chip trace routing (‘**practically frozen**’)
  - It was the biggest ‘headache’ from many reasons:
    - Desire to make it 100% shared with E-Trace
    - E-Trace in meantime went through data trace ratification (I on purpose slow-down on N-Trace MESSAGES to NOT create ‘mixed feeling’ in community).
    - Agreed (but a bit late …) discussion/decision to spit into separated IP blocks (for easier SoC integration).
    - A lot of ADOC formatting, housekeeping and learning curve (diagrams, tables, etc.).
    - Original, donated specification was NOT designed as guide for implementation, so a lot of clarifications were necessary.
  - No significant changes/notes for some time
    - Markus from Lauterbach from tools perspective and Iain from E-Trace angle were contributing and validating a lot (big thanks ☺!)
    - Some new active members (from IP side) as well (more clarifications).
  - This is a standard for ‘years to come’, so should NOT be rushed.
- Messages (‘**in working state**’)
  - It was pushed as ‘last-item’ as it is easiest and least ‘externally sensitive’ and ‘easy to agree’ part of specification.
    - Messages (and fields) are ‘taken’ from Nexus standard ‘as-is’.
  - Reference encode and decoder as well as preliminary ADOC with messages were NOT changed for long time.
    - Reference code included ‘machine-readable’ pseudo-formal spec of RISC-V messages (fully Nexus compliant!)
  - Now ADOC is being ‘converted’ into RISC-V Style PDFs
  - Some clarifications are still needed, but basically this is fully ‘defined’ by Nexus, reference encoder/decoder/dumper and ADOC as well.
  - Nexus is flexible and extensible format, but with ‘default profile’, it provides better than ‘good enough’ compression.

N-Trace Status (part 3)

The following parts of Nexus specification will be addressed:

- Nexus compatible trace encoding
- Trace control
- Trace configuration
- On-chip and off-chip trace routing
- Physical trace connector options

This group will not address the debug part of the Nexus standard.

- Official TG agenda (as a reminder …) in the upper right corner …
- Proof of Concept and Reference implementation[s]
  - Original version of trace control (so called ‘version 0’ has several implementations in silicon – it is NOT only SiFive).
    - Current version can be considered as ‘extension’ of original donated by SiFive
  - Reference encode/decoder provided long time ago (when I was with IAR).
    - That reference was used by others to do own N-Trace implementations.
  - Recently slightly refreshed to run test used by E-Trace during ratification.
    - **TODO:** Need to find a way to not duplicate ELF files from E-Trace (update script …).
- Amount of work left and formal steps
  - I estimate 3 to 4 weeks of work (TG is meeting weekly) to have ‘message PDF’ completed.
    - Other PDFs are ‘**practically frozen**’, so we can focus. Vice chair (Jay from NXP is an expert on Nexus messages).
    - I created infrastructure to generate PDFs easily.
    - With so good compressions, we will NOT went into area of optimizing encoding too much. It will provide some gain, but it is NOT needed.
    - We can always create another revision (only addressing more optimal encoding)
  - I am preparing formal ‘Ratification Plan’ (only recently I have to admit – but planning principles were also evolving …)
    - Preparing a plan sooner would NOT make much difference as key components were ‘floating’ – now there is no ‘unknowns’ ahead.
    - I am personally far from pushing for ‘impossible deadline’, but I am rather seeking completeness, uniformity and excellence.
    - Again – this is a spec for years to come!
  - I want to seek formal ratification (**of entire package**) in Q1 2023
    - This is non-ISA specification, so formal steps are easier and shorter!
  - IMO there is no need for any waivers (**reference implementations both in SW and in IP are already available**)

Trace Compression Comparison (recent)

- **These are Preliminary!**
- I included row/column for easier reference.
- **‘embench-’** tests are real-life programs, but average is nearly the same
- Average is about 0.36 bits/instr, (both E-Trace and N-Trace). It means that 1GHz core with 1 IPC, will produce 360Mbps of trace (45MB/s).
- This bandwidth is EXCELLENT!
  - Mictor connector is capable of **12.6Gbps** bandwidth!
  - MIPI20 connector has **1.6Gbps (4 cores!)**
- **IMPORTANT:** Both N-Trace and E-Trace on ‘default’ settings and both will do better using return stack compression.
- Some unusual **Bits/inst** values should be looked at and understood.
- I want to trace full Linux boot – this will be ultimate ‘real-life’ program!
- Compression is APP and compiler-dependent ☹ (lower IPC, the better!).

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1		N-Trace (default=HTM + RepeatedBranch)						N/E Ratio			E-Trace (without return address stack)							
2	Name	Bytes	MsgCnt	Bytes/msg	InstrCnt	Bits/inst		With Hrd1	With Hrd2		InstrCnt	PktCnt	Payload	Bits/inst	With Hdr1	Bits/hdr1	With Hdr2	Bits/hdr2
3	br_j_asm	Timer interrupts									10027	802	1636	1.305	2438	1.945	3240	2.585
4	coremark	1497855	224644	6.67	33399227	0.359		93%	110%		33399232	256974	1140853	0.273	1397827	0.335	1654801	0.396
5	dhrystone	44157	8041	5.49	215010	1.643		81%	102%		215015	8960	26919	1.002	35879	1.335	44839	1.668
6	embench-aha-mont64	108591	15698	6.92	4541661	0.191		96%	113%		4541666	17662	86957	0.153	104619	0.184	122281	0.215
7	embench-crc32	3290	694	4.74	4028857	0.007		13995%	19584%		4028862	197016	296355	0.588	493371	0.980	690387	1.371
8	embench-cubic	366621	51664	7.10	7724337	0.380		86%	102%		7724342	57853	257932	0.267	315785	0.327	373638	0.387
9	embench-edn	80683	11504	7.01	3493777	0.185		66%	82%		3493782	12943	40254	0.092	53197	0.122	66140	0.151
10	embench-huffbench	116641	16675	6.99	2461117	0.379		93%	109%		2461122	18714	89937	0.292	108651	0.353	127365	0.414
11	embench-matmult-int	92476	13214	7.00	2891806	0.256		82%	98%		2891811	14837	61450	0.170	76287	0.211	91124	0.252
12	embench-minver	260604	37878	6.88	2620515	0.796		83%	99%		2620520	42615	173737	0.530	216352	0.660	258967	0.791
13	embench-nbody	354340	50579	7.01	6394542	0.443		88%	104%		6394547	56604	255846	0.320	312450	0.391	369054	0.462
14	embench-nettle-aes	17481	2466	7.09	4523969	0.031		95%	111%		4523974	2765	13954	0.025	16719	0.030	19484	0.034
15	embench-nettle-sha256	17661	2866	6.16	3874834	0.036		76%	95%		3874839	3226	10093	0.021	13319	0.027	16545	0.034
16	embench-nsichneu	173916	24848	7.00	2241141	0.621		97%	113%		2241146	27899	141492	0.505	169391	0.605	197290	0.704
17	embench-picojpeg	109997	15432	7.13	4012848	0.219		69%	85%		4012853	17255	58662	0.117	75917	0.151	93172	0.186
18	embench-qrduino	126037	18037	6.99	3426824	0.294		92%	108%		3426829	20218	95171	0.222	115389	0.269	135607	0.317
19	embench-sglib-combined	200093	27606	7.25	2269619	0.705		93%	110%		2269624	34910	150738	0.531	185648	0.654	220558	0.777
20	embench-slre	269441	39637	6.80	2622477	0.822		84%	100%		2622482	44636	180799	0.552	225435	0.688	270071	0.824
21	embench-st	302136	42576	7.10	4412657	0.548		85%	102%		4412662	49692	208311	0.378	258003	0.468	307695	0.558
22	embench-statemate	82615	11805	7.00	1038135	0.637		84%	100%		1038140	13282	55820	0.430	69102	0.533	82384	0.635
23	embench-ud	28190	4453	6.33	1277146	0.177		94%	112%		1277151	5012	21499	0.135	26511	0.166	31523	0.197
24	embench-wikisort	244839	40063	6.11	2346529	0.835		82%	100%		2346534	44813	156347	0.533	201160	0.686	245973	0.839
25	hello_world	PK loader	For later?								358426	14343	30901	0.690	45244	1.010	59587	1.330
26	median	1565	247	6.34	15010	0.834		83%	101%		15015	277	1030	0.549	1307	0.696	1584	0.844
27	mm	5359	809	6.62	297033	0.144		67%	84%		297038	909	2691	0.072	3600	0.097	4509	0.121
28	mt-matmul	2892	463	6.25	41449	0.558		68%	86%		41454	522	1452	0.280	1974	0.381	2496	0.482
29	mt-vvadd	6295	1000	6.29	61067	0.825		66%	84%		61072	1126	3049	0.399	4175	0.547	5301	0.694
30	multiply	4601	768	5.99	55011	0.669		68%	87%		55016	862	2267	0.330	3129	0.455	3991	0.580
31	new_hw	PK loader	For later?								356199	14285	30828	0.692	45113	1.013	59398	1.334
32	pmp	Trap handlers									1110463	11047	54900	0.396	65947	0.475	76994	0.555
33	qsort	14494	2097	6.91	235010	0.493		86%	102%		235015	2351	10039	0.342	12390	0.422	14741	0.502
34	rsort	4910	729	6.74	375011	0.105		63%	79%		375016	812	2268	0.048	3080	0.066	3892	0.083
35	spmv	1896	296	6.41	70010	0.217		94%	111%		70015	331	1451	0.166	1782	0.204	2113	0.241
36	towers	1811	354	5.12	15011	0.965		77%	99%		15016	396	1007	0.536	1403	0.747	1799	0.958
37	vvadd	880	148	5.95	10011	0.703		70%	89%		10016	164	455	0.363	619	0.494	783	0.625
38	xrle	3389	485	6.99	164959	0.164		74%	90%		164959	546	1964	0.095	2510	0.122	3056	0.148
39	**Embench sum & average**	2955652	427695		66202791	0.357		103%	126%		66202886		2355354	0.285	3037306	0.367	3719258	0.449
40	**All sum & average**	4545756	667776		101156610	0.360		101%	122%		102991880		3669064	0.285	4665723	0.362	5662382	0.440
41																		
42	**TODO (in N-Trace refcode):**		<- To be fixed and hand-checked							**Notes:**	N-Trace is self-synchronizing (start/stop message can be detected). E-Trace needs syncs.							
43		PK loader	<- Test is using PK loader (ELF is incomplete)								E-Trace requires packet size (1 byte at least as defined by the spec). This is 'With Hdr1'.							
44											ATB (Arm Trace Bus) will add 1 byte extra for every source switch event. This is 'With Hdr2'.							
45											ATB (Arm Trace Bus) will add 1/16=6.25% ID-handling fixed overhead (even for single hart!)							

Maintenance

- eTrace:
  - Clarifications on push/pop support for data trace agreed with requestor (Seagate)
    - Examples added to section 4.3 and pull request made
    - No functional change
    - Not merged to main yet. Wait for…
  - Need to remove register descriptions and cross-reference common-control when it is frozen
  - Corner-case bug in reference encoder model found, fixed, regressed and merged

Proposed Next Focus Areas

- Trace
  - eTrace packet encapsulation for transport
    - I propose we form a task group to define this
  - eTrace vector extension support
    - I believe nothing is needed for instruction trace
    - Data trace may require additional packet formats – needs further discussion
- Competitive analysis of debug/trace features from different architectures
  - Any volunteers?
- Debug Authorisation framework
  - Some discussions underway…
    - Various debug levels End-User, Expert or OEM, etc.
    - How do we authorize different levels?
    - Root of trust being established to debug and identify if debug is enabled on a platform
    - What can be part of debug vs what cannot be part of debug?
    - Virtualization for trace to memory

Gap AnalysisBacklog

- Trace
  - cycle accurate
  - trace to memory
- Performance counting
  - Collaborate with Perf Analysis SIG as required
- Remote Telemetry
- Self hosted debug
- Debug spec, post ratification:
  - Vector extension impact
  - Couple of minor issues deferred (see slide 9)
- Alternatives to GDB with multi-core support

Future Meetings / AOB

- Monthy from now on – 2nd Wednesday of each month
- AOB

RISC-V®


Thank You
RISC-V®