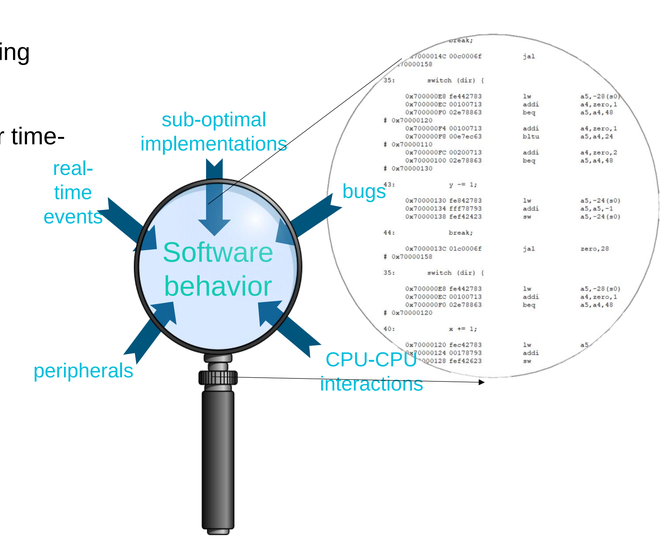
Unleashing the Power of RISC-V E-Trace with a Highly Efficient Software Decoder

Marcel Zak, Mat O'Donnell, & Vivek Chickermane

Tessent Embedded Analytics, Siemens EDA

What is Processor Trace?

- Debugging program misbehavior to isolate underlying Silicon problems is non-trivial
- Halting the core to debug is not always practical for time-critical applications
- Observation at full-speed is needed
  - Cycle accurate visibility of program execution
- *Processor Trace provides this visibility*
- Additional uses
  - Forensic debugging
  - Code profiling & coverage
  - Heisenbugs
  - Infrequent/Irregular bugs



Processor Branch Trace

- Start address is reported
- Only branches are reported
  - jump, call, return, interrupts, exceptions
- Sequential instructions - not reported
- Indirect jumps, interrupts and exceptions
  - Un-inferable program counter discontinuities
  - The destination address must be reported
  - Interrupts must also report PC at time of interrupt

```
 0:   00050793   mv   a5,a0      → start address
 4:   00100713   li   a4,1       ┐ not reported
 8:   00100513   li   a0,1       ┘
 c:   00f77e63   bgeu a4,a5,28   → branch not taken
10:   00100693   li   a3,1       ┐
14:   00078713   mv   a4,a5      │
18:   fff78793   addi a5,a5,-1   │ not reported
1c:   02e50533   mul  a0,a0,a4   │
20:   fed79ae3   bne  a5,a3,14   → branch taken
24:   00008067   ret             ┘
28:   00008067   ret
```

Branch Trace Log
1. start address
2. branch not taken
3. branch taken
4. branch not taken

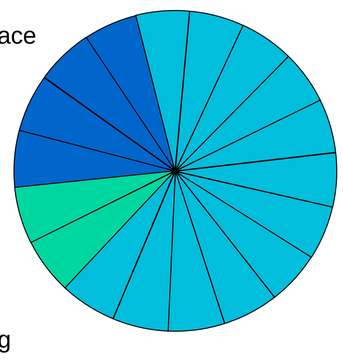
RISC-V E-Trace- Mandatory vs Optional

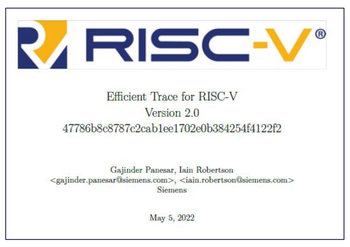
RISC-V Trace Mandatory Features

- Instruction trace
- Hart (CPU) to encoder interface
- 'Delta Address' trace mode
- Efficient packet format

Embedded Analytics Additional Functionality

- Cycle accurate trace
- Custom Instructions decoding

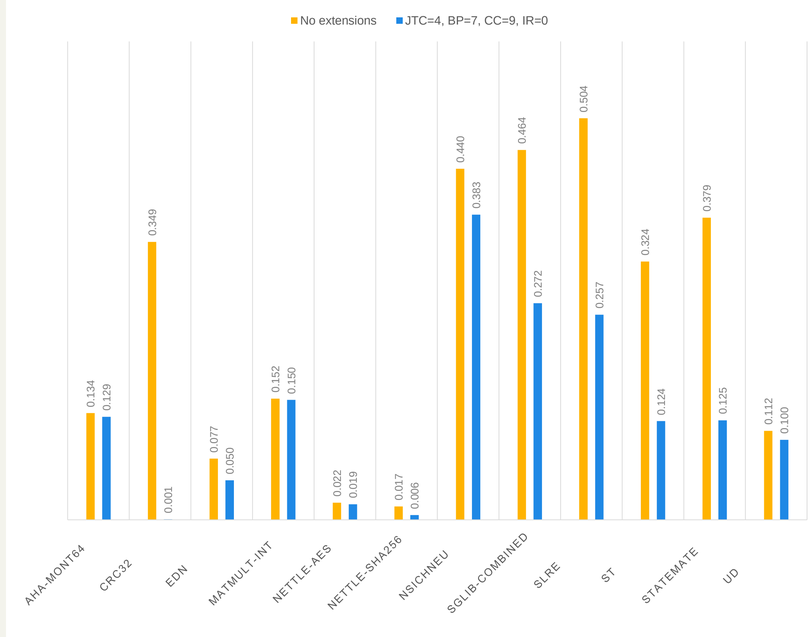


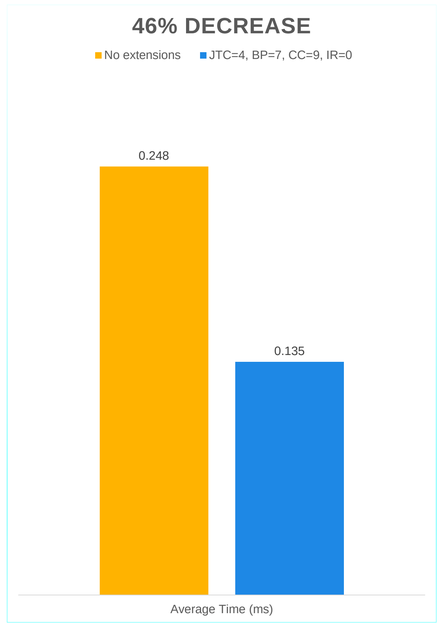


RISC-V Trace Optional Features

- Multiple instruction retirement
- Implicit exception mode
- Sequentially inferable jump mode
- Implicit return mode
- Branch Prediction mode
- Jump Target Cache mode
- Full Address mode
- Sign-based compression
- XOR data trace compression
- Filtering
- Timestamps

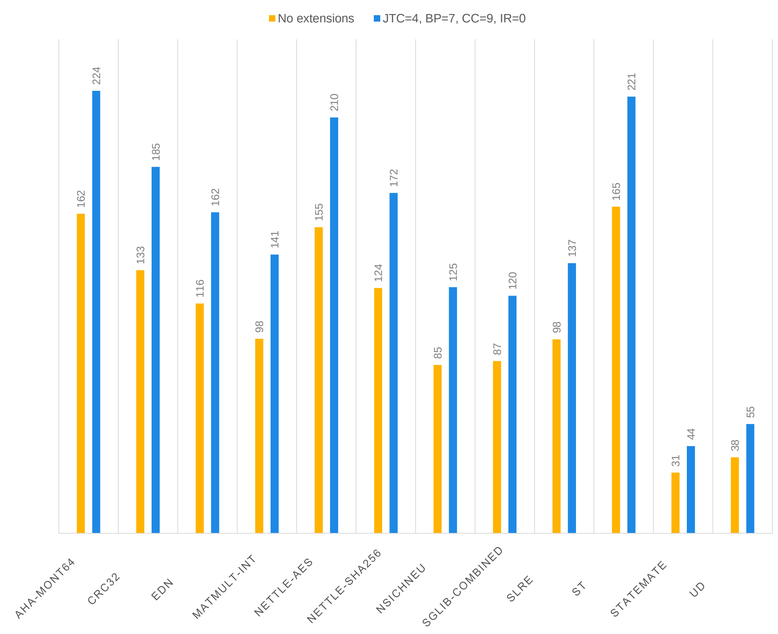
Embench™ : HW Trace Encoder – bits per instruction

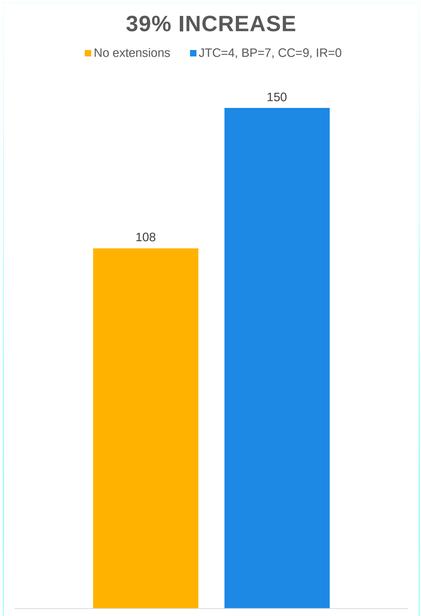




CPU: Intel® Xeon® W-2225 @ 4.10GHz; RAM: 2x8GB (HMA81GR7CJR8N-XN) @ 2934 MT/s; OS: Rocky Linux 8.10 (Green Obsidian)

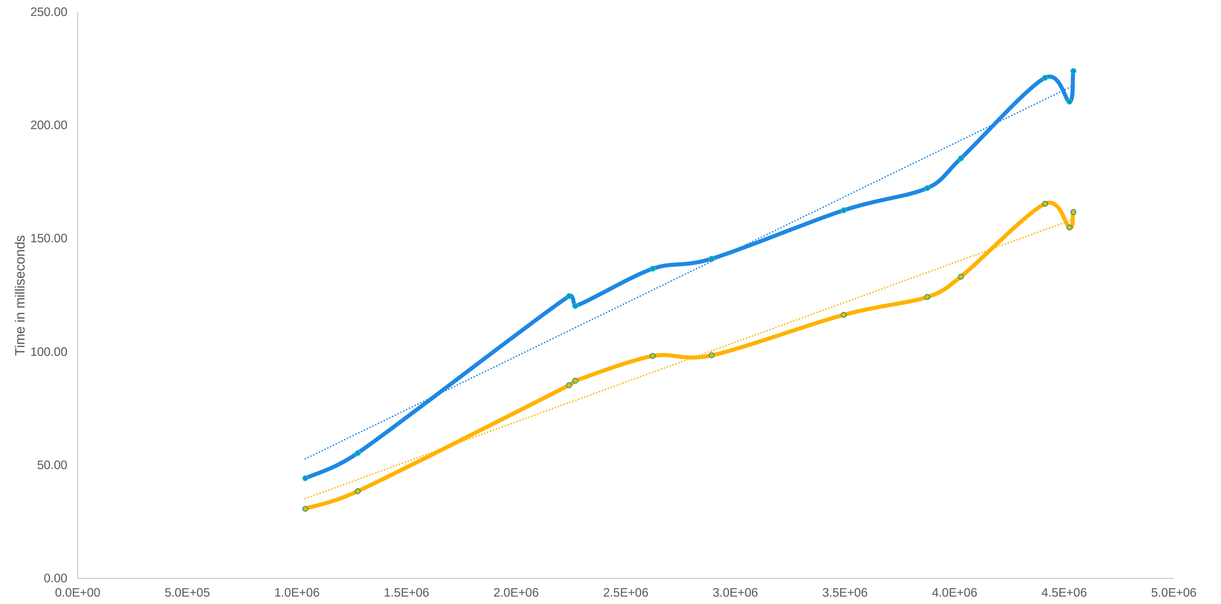
Embench™ : SW Trace Decoding Speed





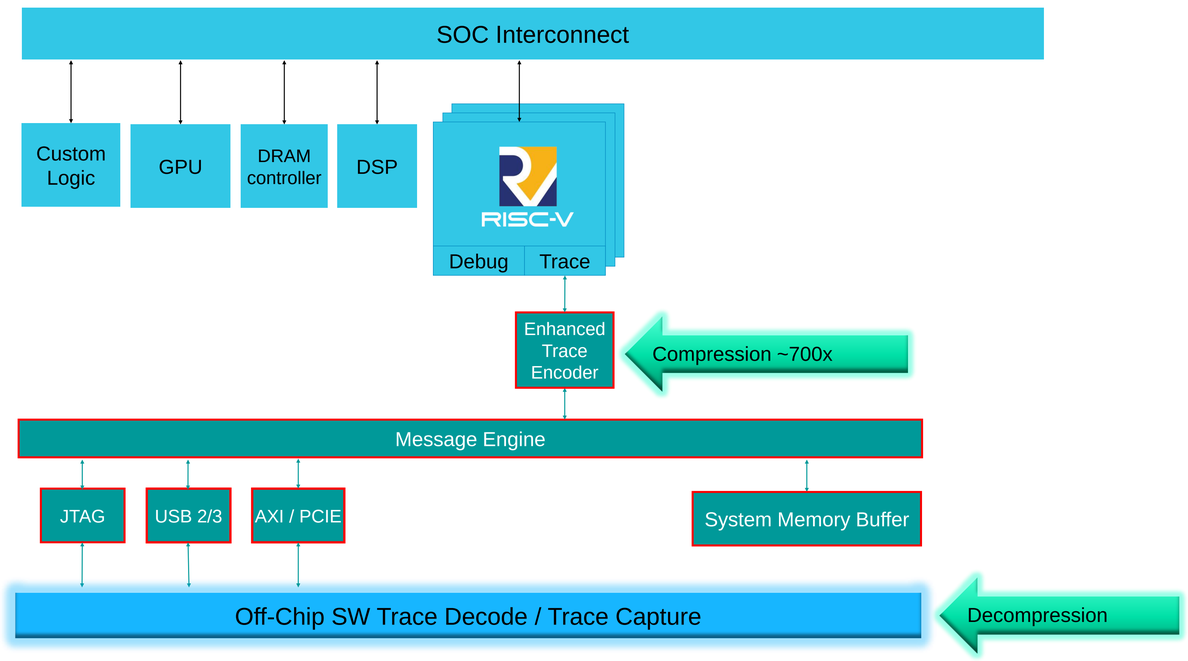
CPU: Intel® Xeon® W-2225 @ 4.10GHz; RAM: 2x8GB (HMA81GR7CJR8N-XN) @ 2934 MT/s; OS: Rocky Linux 8.10 (Green Obsidian)

Embench™ – SW Trace Decoding Complexity - O(n)

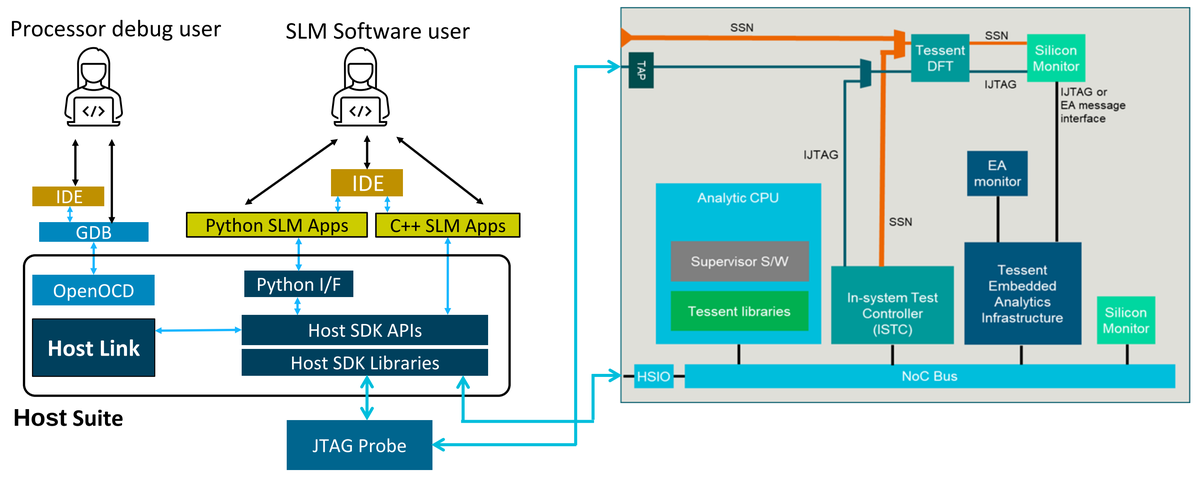


CPU: Intel® Xeon® W-2225 @ 4.10GHz; RAM: 2x8GB (HMA81GR7CJR8N-XN) @ 2934 MT/s; OS: Rocky Linux 8.10 (Green Obsidian)

E-Trace for SoC Debug and Continuous Monitoring



Silicon Lifecycle Management: Software Stack



- SoC access via JTAG probe or high-speed I/O: PCIe, USB, Aurora SerDes etc.
- Concurrent data acquisition from Trace Monitor and a multitude of embedded instruments
- SLM User can develop Python or C++ SW stack using the EA Host Suite
- Integration with IDEs such as VSCode, interactive Python sessions, and Jupyter Notebooks
- Access to cloud-based DB and AI/ML Analytics