Western Digital®

SweRV Debug, Trace and On-chip Analytics for SOCs

Sesibhushana Bommana Rao

Mukesh Kumar Panda

June 12, 2019

SoC Design Challenges

- Time to Market
- Silicon & Design Complexity
  - Heterogeneous Block & Architecture
  - H/W & S/W Integration
  - Billion Transistor
  - Digital & Analog IP
- System Level Validation
- System Performance
- Functional Safety & Security

SoC Design Trend

- **33%** of ICs considered in 2012 were determined to be right-first-time
- **19%** of ICs required three or more spins before their products
- **49%** of silicon spins are still caused, by logic or functional flaws.
- As per Industry benchmarks post-silicon validation and debug processes combined can consume more than **35%** of the project design time or cost.
- SoCs produces 1Pbits **10^15 bits** of data per second.

 Reference taken from "Debugging Systems-on-Chip: Communication-centric and Abstraction-based Techniques" - 2014 Edition, Springer Publication

Problem Statement for DfDT

How Do We?

- ❑ select the internal signals that are to be observed
- ❑ observe the large number of internal signals with limited I/O pins
- ❑ reproduce faulty behavior of a silicon implementation
- ❑ make sure debug-functionality is not introducing new failures
- ❑ implement an efficient debug functionality
- ❑ make sure debug-functionality is not degrading performance
- ❑ implement a configurable & customizable debug-functionality

DfDT Techniques

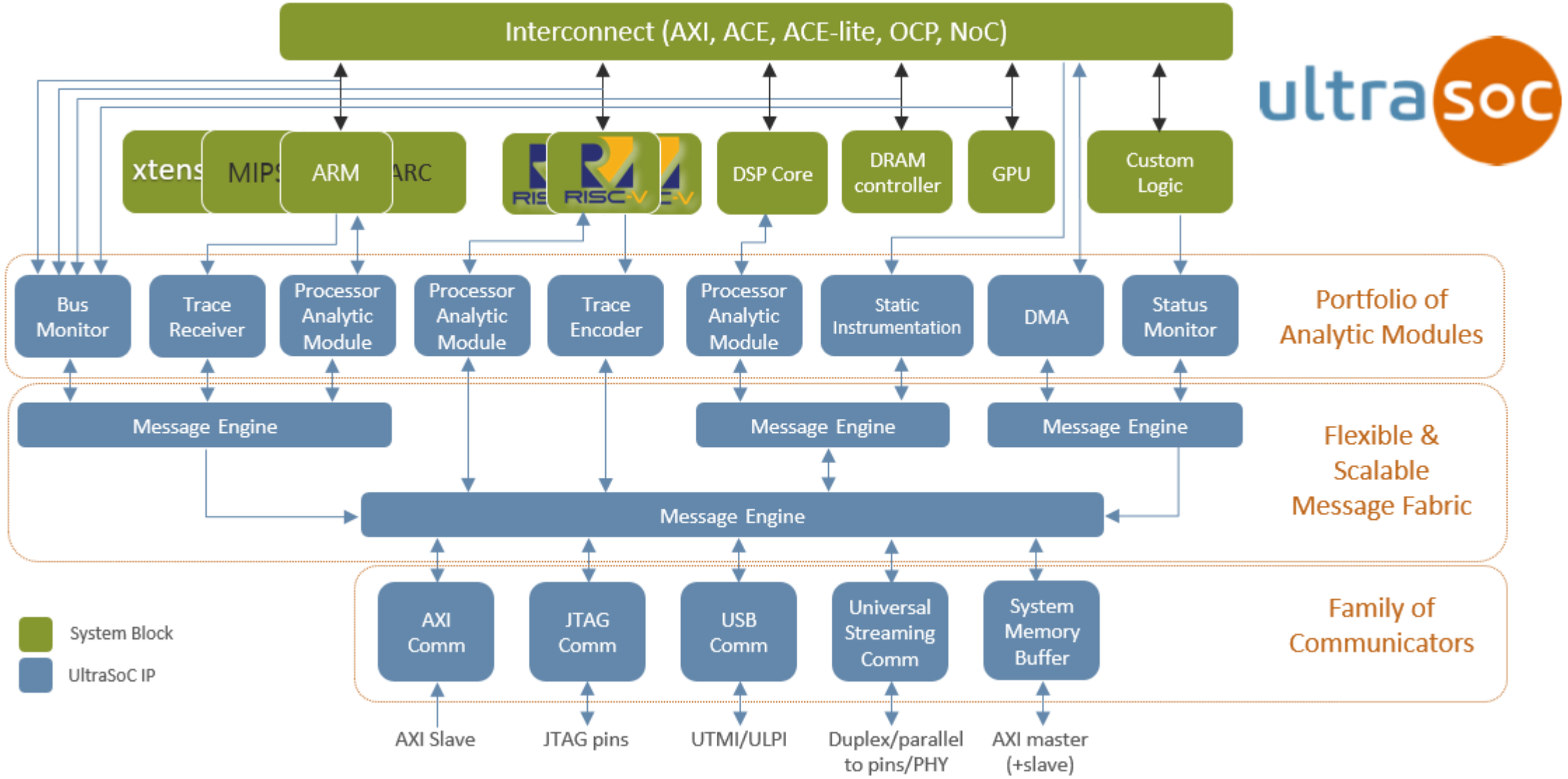
Trace Based Debug

- Record the set of internal signals into a memory & stream their values off-chip
- Implementation:
  - Configurable Sets of Internal Signals for trace
  - Add a trigger mechanism for when to sample & record
  - Adding an On-Chip Memory foe trace recording
  - Adding a mechanism to configure trace
- Advantage:
  - Ability to provide real time information
- Drawback:
  - Set of internal signals are limited to number of output ports
  - Total amount of trace information is less than information generated inside SoC
  - Silicon area associated with its on-chip H/W components

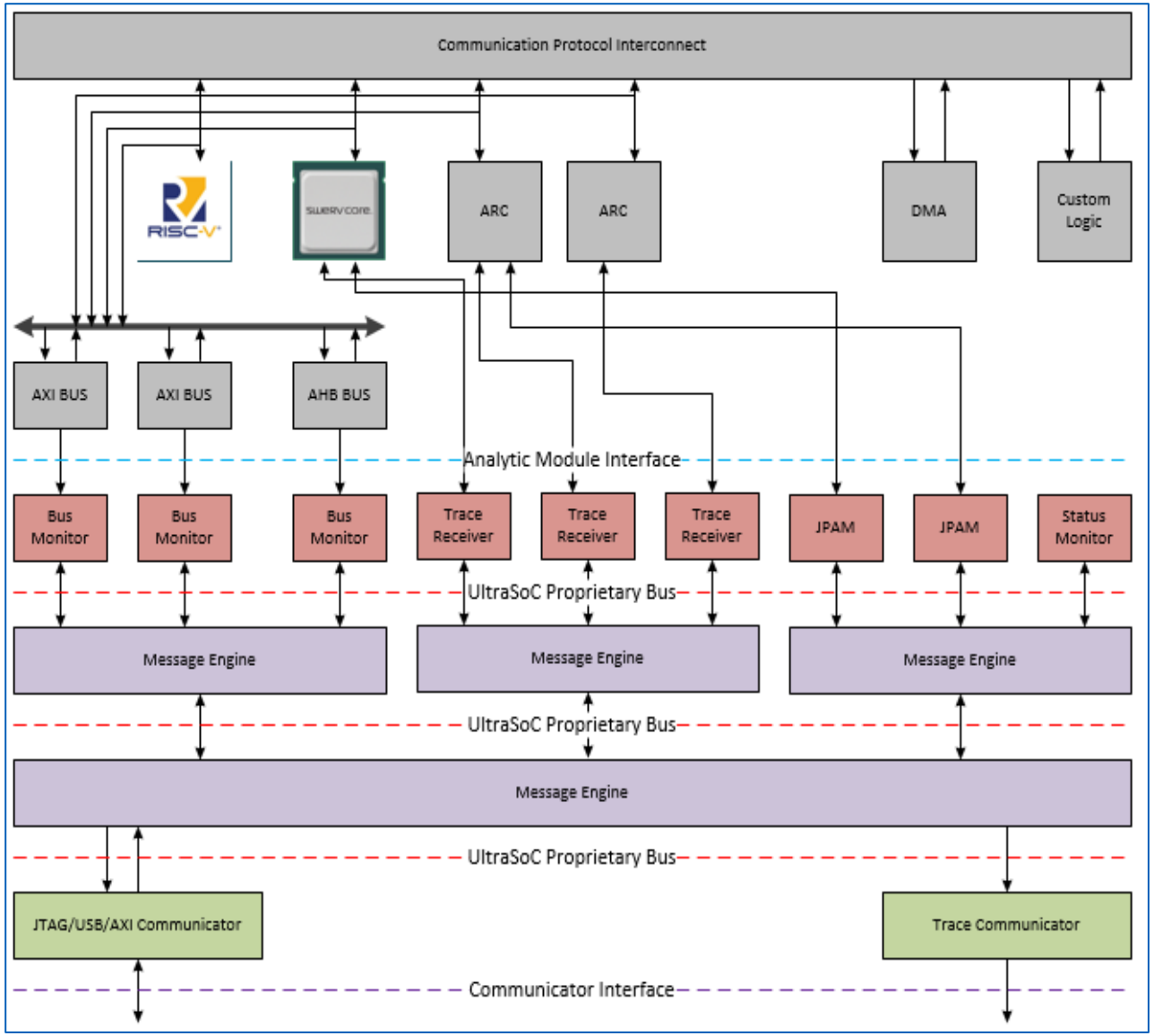
Run/Stop Based Debug

- Stop the execution of the SoC at an interested point of execution and extract the value of internal signals
- Implementation:
  - A trigger mechanism to determine when to execution stopped
  - Mechanism to stop the execution
  - Mechanism to access the state of the SoC
  - Mechanism to configure debug functionality
- Advantage:
  - Ability to provide observability and control over all internal signals
- Drawback:
  - Full control & observability is available only after execution is stopped
  - Silicon area cost associated with its on-chip hardware components

DfDT Blocks for Whole SoC



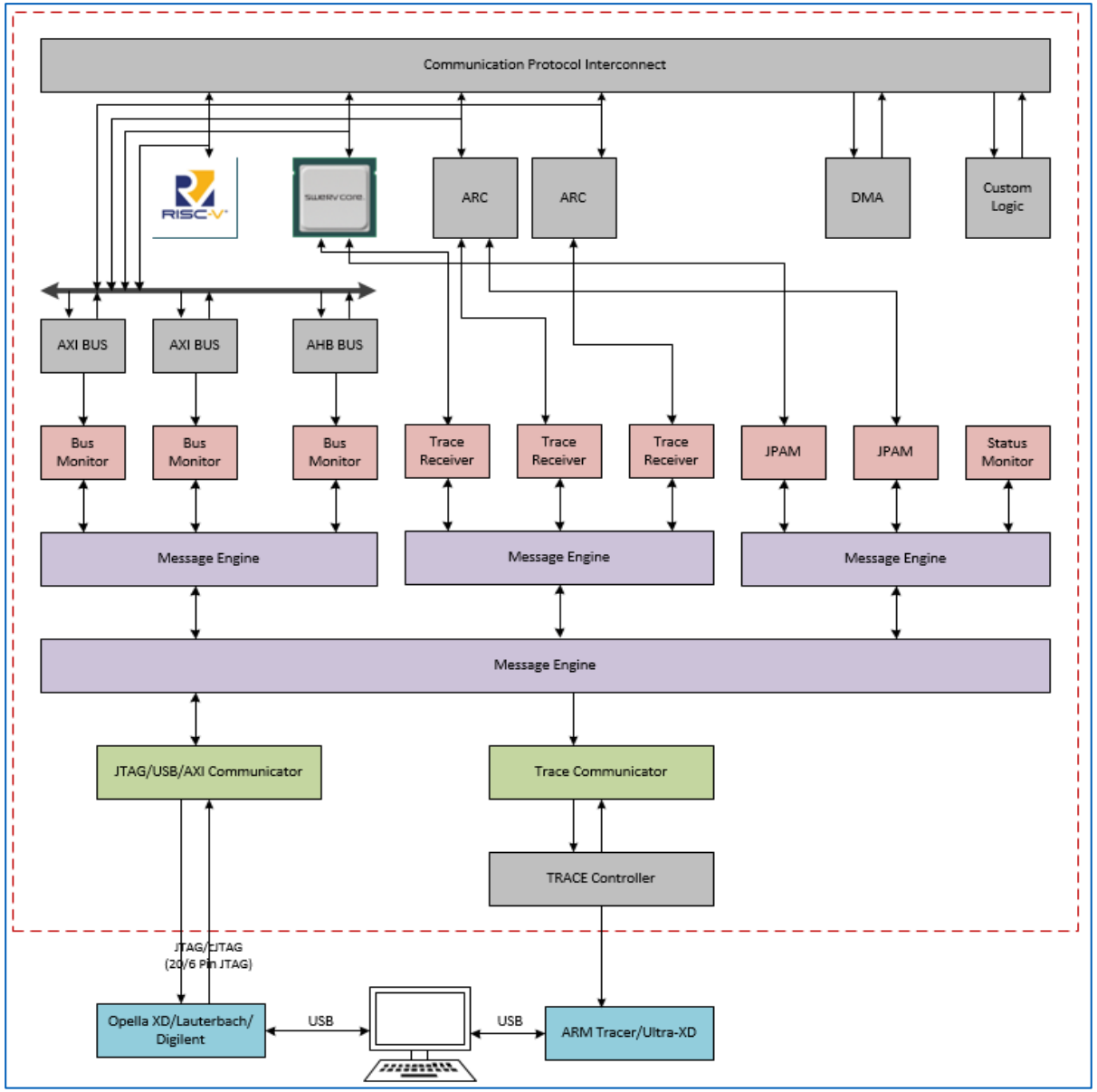
Communication Through Different DfDT Blocks



- Analytic Modules:
  - Bus Monitor
  - Status Monitor
  - Processor Analytic Module (PAM)
  - Processor Trace Receiver
- Message Infrastructure (Interconnect)
- Communicator:
  - JTAG Communicator
  - AXI Communicator
  - USB Communicator

- Plug & Play Message Infrastructure:
  - Two Asymmetric Channel UpStream(Configuration) and DownStream(Observability)
  - Tree Hierarchy: Message Engine allow multi level of hierarchy
  - System-Discovery identifies the complete debug & trace system

Multi Core based SoC Design for Debug & Trace (DfDT)



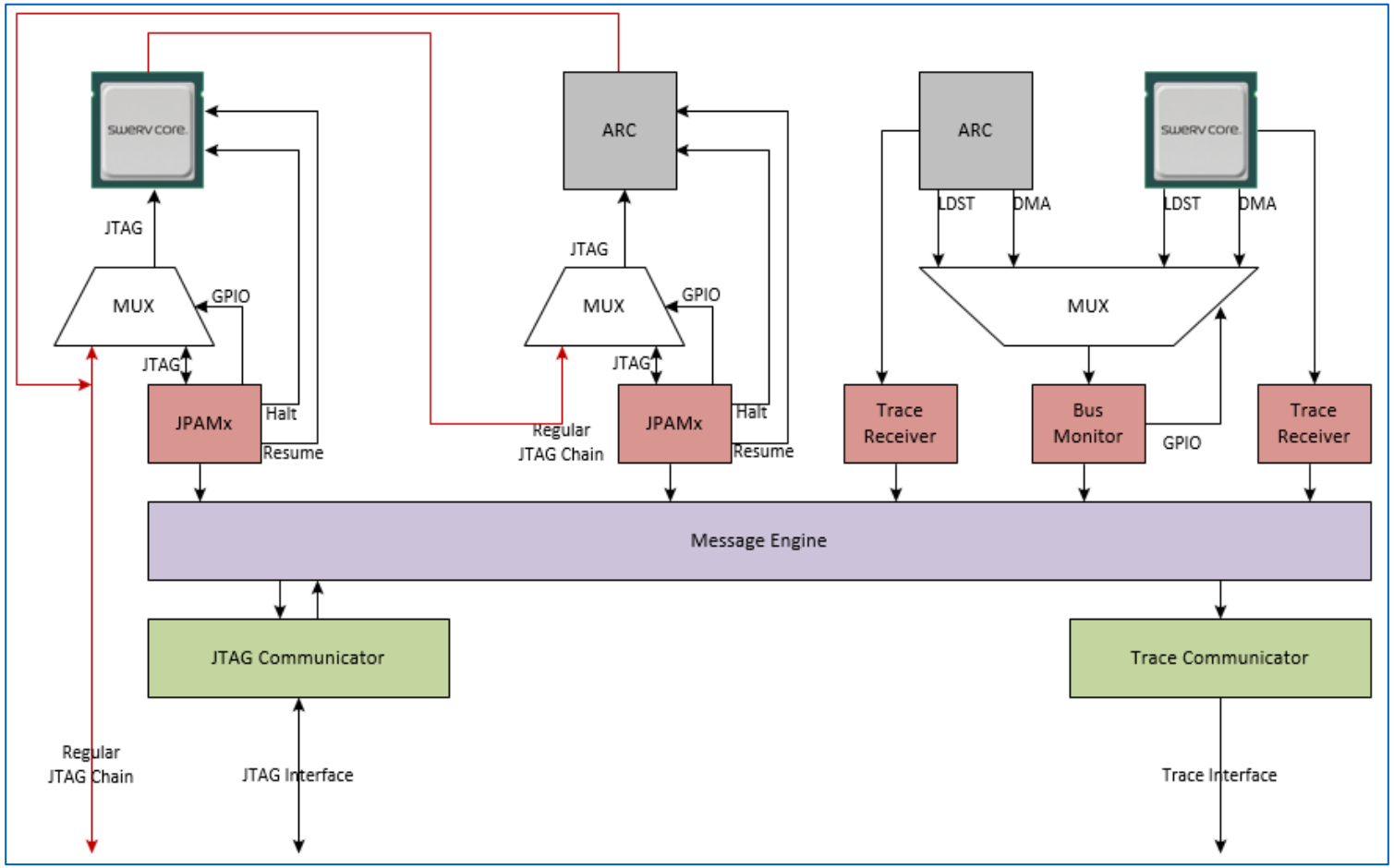
Debug & Trace Challenges:

- ❑ How quick can we halt a processor
- ❑ How to stop multiple core at same real time
- ❑ How to access debug-registers of multiple core parallel
- ❑ How to guarantee safety & security

Debug & Trace Analytics Modules:

- ❑ Processor Trace Encoder (TE)
- ❑ Processor Analytics Module (PAM)
- ❑ Bus Monitor
- ❑ Status Monitor
- ❑ Message Engine (Interconnect)
- ❑ JTAG Communicator
- ❑ Trace Communicator

Multi Core based SoC Design for Debug & Trace (DfDT) cont…



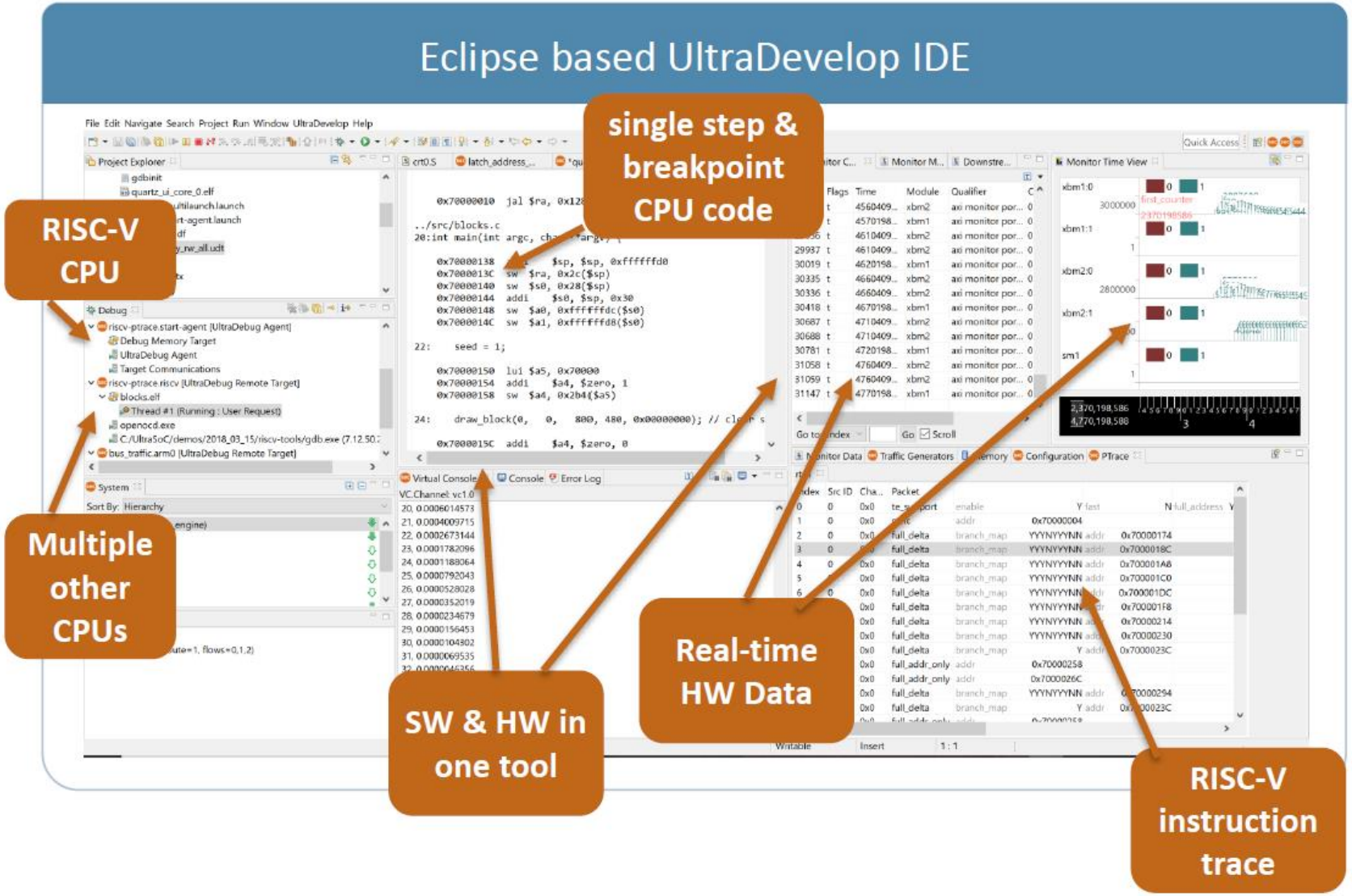
Operating Mode Support:

- ❑ Normal Mode Support
- ❑ Debug Mode through Core IDE
- ❑ Debug Only Mode Support using UltraSoC IPs
- ❑ Debug & Trace Support

Feature:

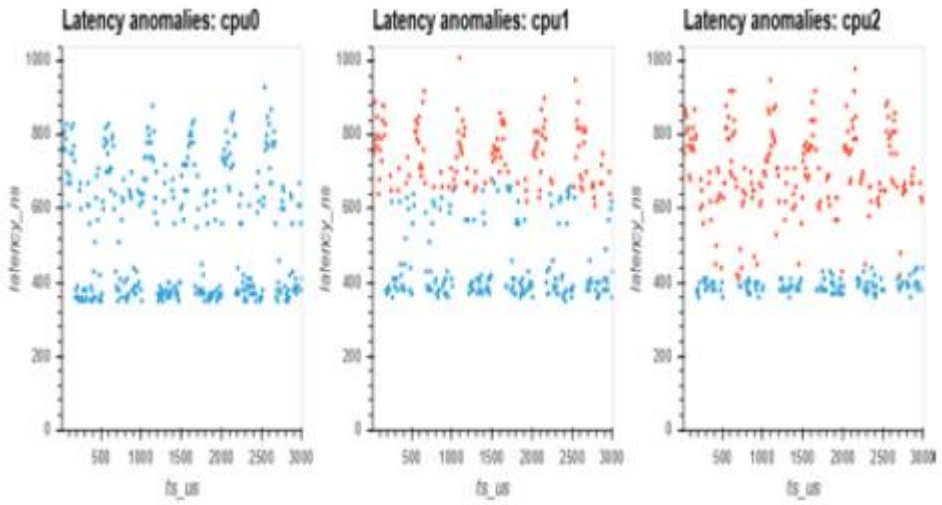
- ❑ Parallel Access of Core Debug Interface
- ❑ Simultaneous Access of Multi Core Halt & Resume Interface
- ❑ Simultaneous access to both debug and trace interface
- ❑ Stop the execution of Multi Core for a safety and security alert

Software tools for data analytics from UltraSoC

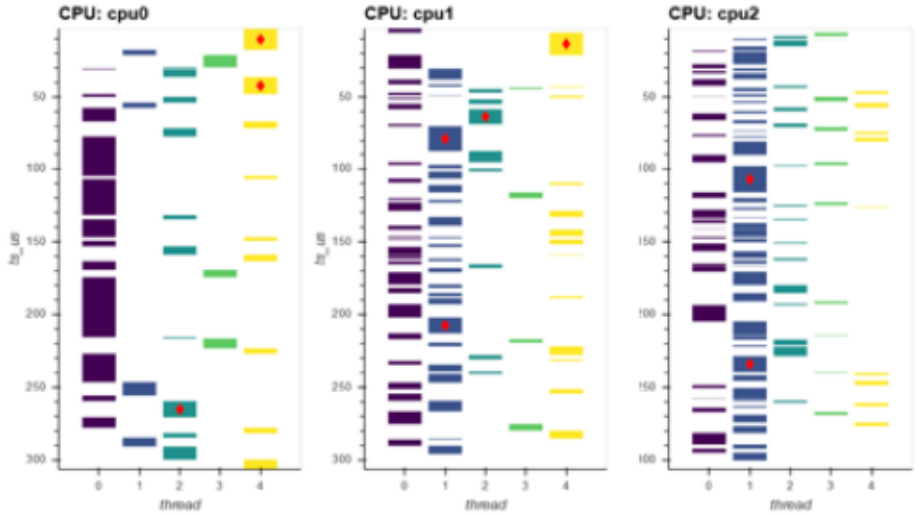


Real Time Monitoring and Analytics

Security: H/W Based Attack Detect

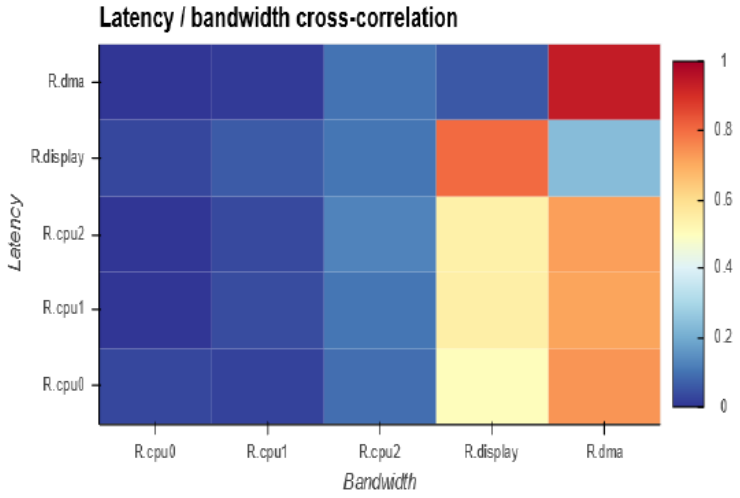


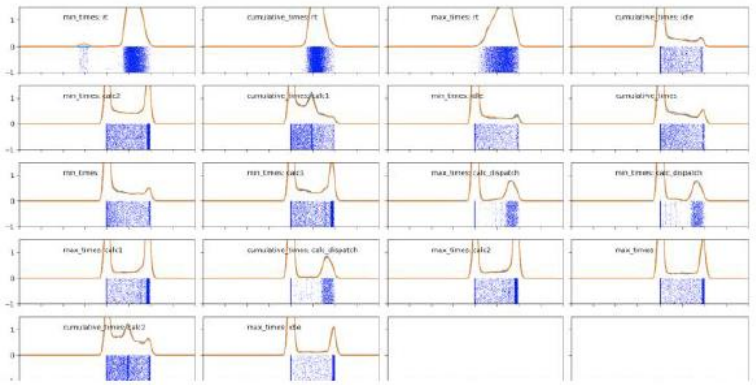
Performance Monitor: Run Time

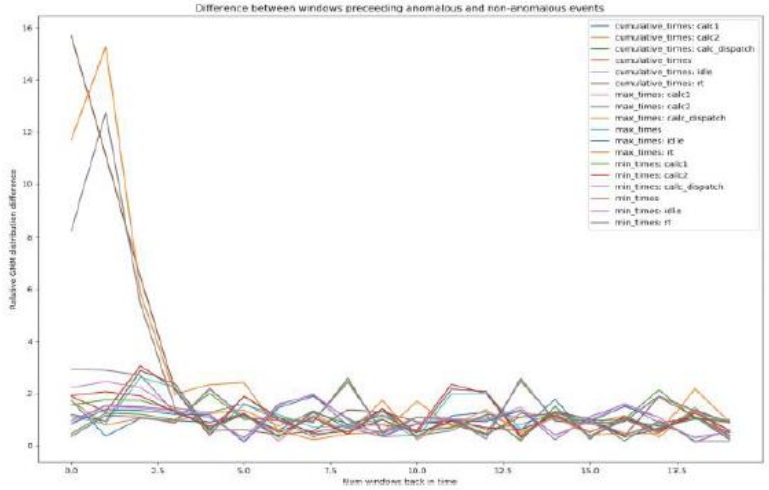


- Hardware: Fast, react at Hardware timescale, invisible to software
- Visibility: Analyze software and system everywhere in SoC, see any problem
- Non-Intrusive: No performance impact or warning due to DfDT

Localization of Root Cause using Data Science







- ❑ Anomaly Detection
- ❑ Root Cause Detection
- ❑ Big Data is Powerful but UltraSoC delivers rich data

Root Cause Localization:

- ❑ Control over SoC operating mode, clock and reset
- ❑ Communication Monitoring
- ❑ Communication Control Support
- ❑ Fast and Scalable Event Distribution
- ❑ Debug monitor has to be accessible from off-Chip debugger S/W

Summary

- ❑ On Chip Analytics tool for viewing performance metrics inside the SOC
- ❑ Real time and off-chip post processing support.
- ❑ Examine performance data in a graph, histogram, report or waveform.
- ❑ It can also check for the Bus Protocol compliance of any interface connected.

Combined Debug and System monitoring in the SOC:

- ❑ Provide depth insight of SoC to the system designers
- ❑ Analyze the localize the performance bottlenecks of the target
- ❑ Bus transactions are the most interesting to observe through on-chip trackers
- ❑ By combining the Debug and the Performance Monitors will enable the user to get a command over the debugging vs the longer debugging cycles associated with corner cases.
- ❑ For Bigger Chip all analytic modules can be included keeping the area in mind
- ❑ For Smaller Chip only required analytic module can be used along with a JTAG Communicator, this will reduce the area cost.

Western Digital®

THANK YOU