Legal Disclaimer

The material contained herein is not a license, either expressly or impliedly, to any IPR owned or controlled by any of the authors or developers of this material or MIPI®. The material contained herein is provided on an "AS IS" basis and to the maximum extent permitted by applicable law, this material is provided AS IS AND WITH ALL FAULTS, and the authors and developers of this material and MIPI hereby disclaim all other warranties and conditions, either express, implied or statutory, including, but not limited to, any (if any) implied warranties, duties or conditions of merchantability, of fitness for a particular purpose, of accuracy or completeness of responses, of results, of workmanlike effort, of lack of viruses, and of lack of negligence. ALSO, THERE IS NO WARRANTY OR CONDITION OF TITLE, QUIET ENJOYMENT, QUIET POSSESSION, CORRESPONDENCE TO DESCRIPTION OR NON-INFRINGEMENT WITH REGARD TO THIS MATERIAL.

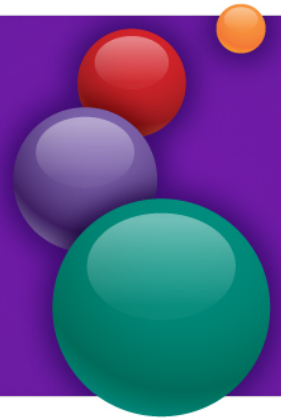
All materials contained herein are protected by copyright laws, and may not be reproduced, republished, distributed, transmitted, displayed, broadcast or otherwise exploited in any manner without the express prior written permission of MIPI Alliance. MIPI, MIPI Alliance and the dotted rainbow arch and all related trademarks, tradenames, and other intellectual property are the exclusive property of MIPI Alliance and cannot be used without its express prior written permission.

IN NO EVENT WILL ANY AUTHOR OR DEVELOPER OF THIS MATERIAL OR MIPI BE LIABLE TO ANY OTHER PARTY FOR THE COST OF PROCURING SUBSTITUTE GOODS OR SERVICES, LOST PROFITS, LOSS OF USE, LOSS OF DATA, OR ANY INCIDENTAL, CONSEQUENTIAL, DIRECT, INDIRECT, OR SPECIAL DAMAGES WHETHER UNDER CONTRACT, TORT, WARRANTY, OR OTHERWISE, ARISING IN ANY WAY OUT OF THIS OR ANY OTHER AGREEMENT RELATING TO THIS MATERIAL, WHETHER OR NOT SUCH PARTY HAD ADVANCE NOTICE OF THE POSSIBILITY OF SUCH DAMAGES.

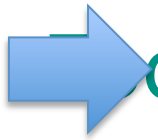
Webinar Topics

- **MIPI Alliance: A Brief Introduction**
  - Peter Lefkin, Managing Director, MIPI Alliance
- **MIPI Debug WG Webinar: Taking Debug Into the IoT**
  - Gary A. Cooper (Texas Instruments), MIPI Debug WG Chair

MIPI Alliance:
A Brief Introduction



Peter B. Lefkin
Managing Director

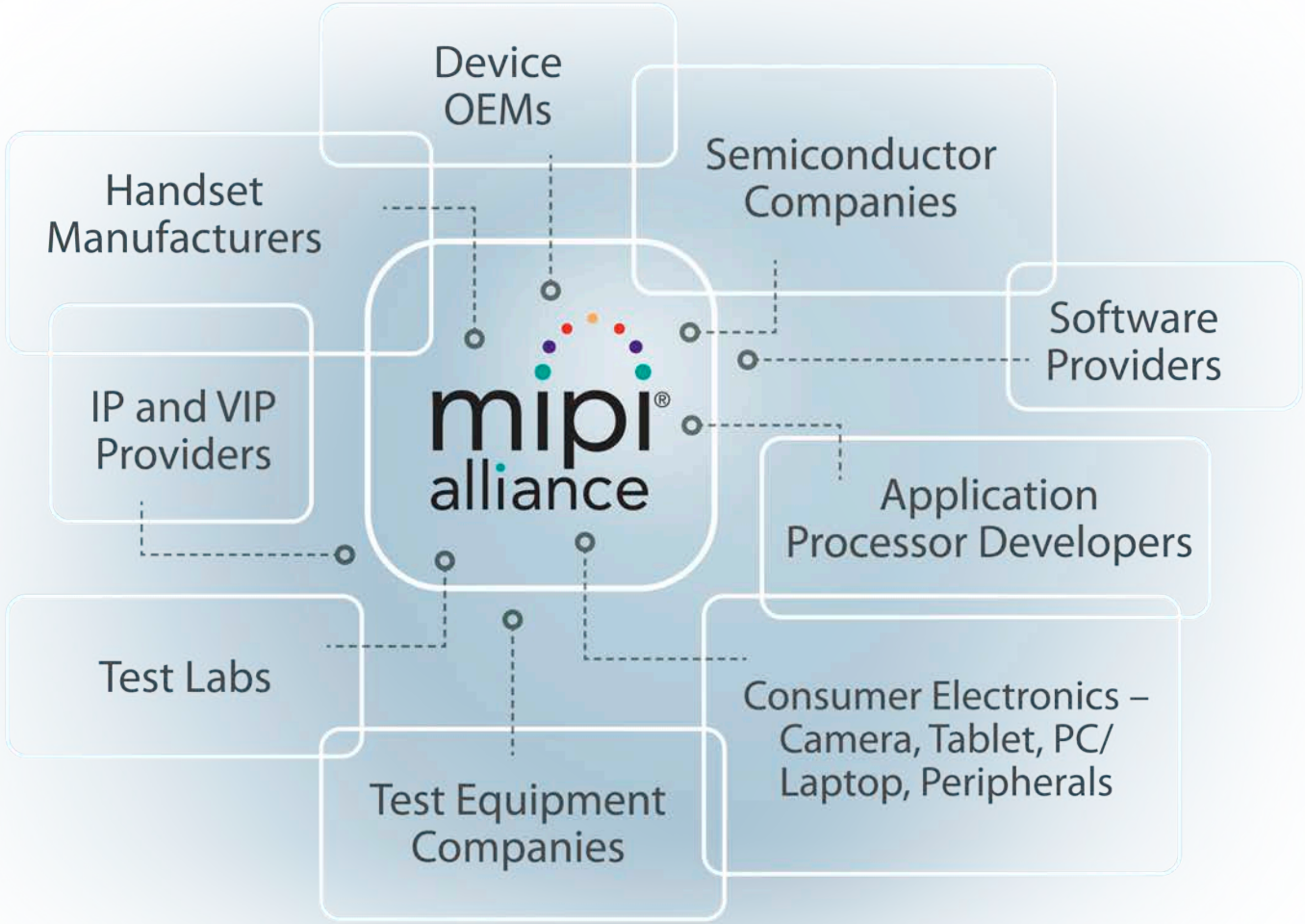
MIPI Alliance – Beginning  day

Since its inception MIPI Alliance has continued to remain focused and meet the needs of the mobile industry and beyond in the midst of significant change over the 10 year lifespan of the organization. It continues to do so with the development of new technologies and evolution of current specifications.

About MIPI Alliance

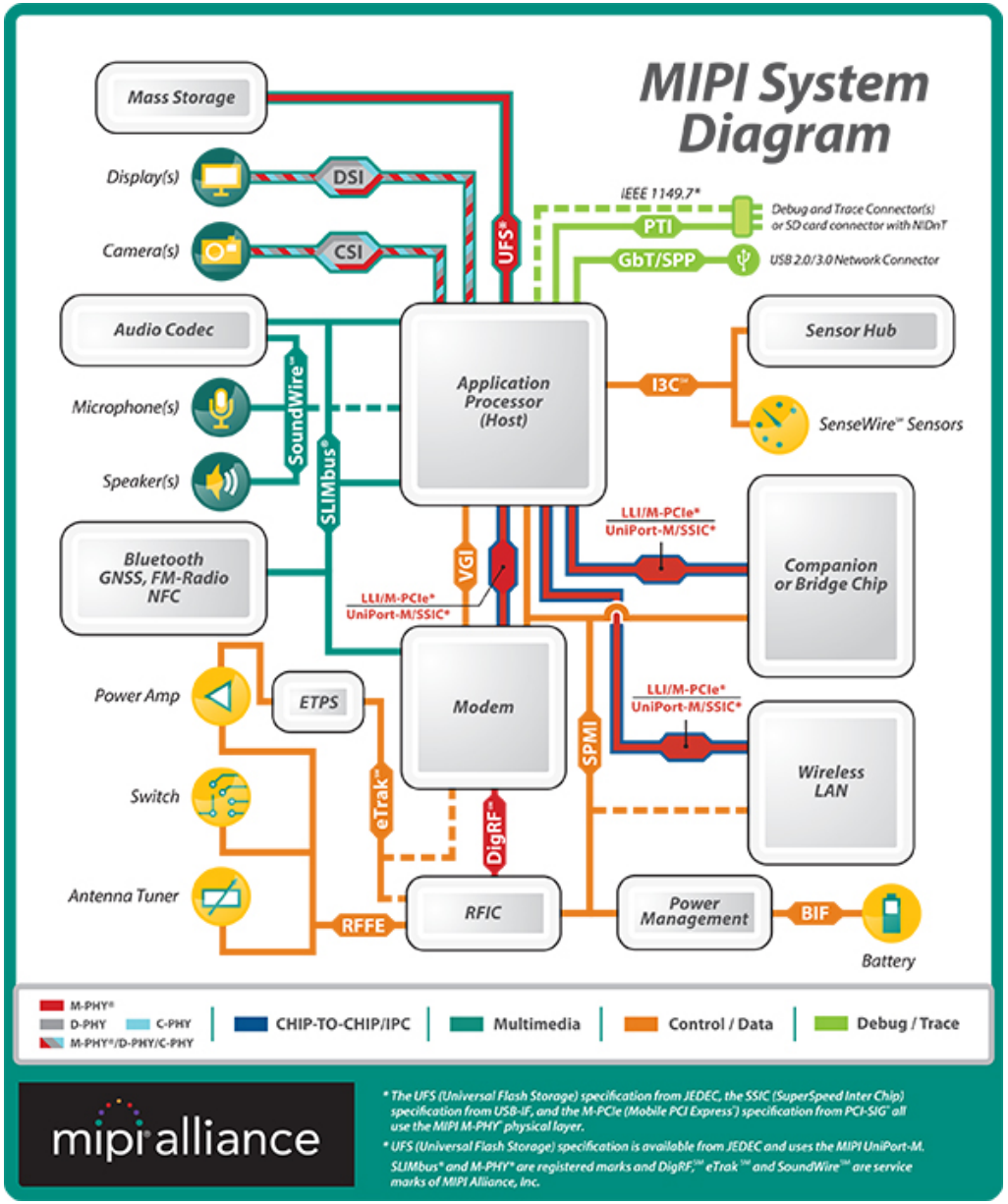
- 281 Members (as of 7 Nov 2014)
- 45+ specifications and supporting docs
- We drive **mobile** and **mobile-influenced interface** technology through the development of hardware and software **specifications**
- We work **globally** and **collaboratively** with other standards bodies to **benefit the mobile ecosystem**

MIPI Alliance Member Ecosystem



Active MIPI Alliance Working Groups

- Analog Control Interface
- Battery Interface
- Camera
- Debug
- Display High Speed Synchronous Interface
- Low Latency Interface
- Low Speed Multipoint Link (New - SoundWire℠)
- Marketing
- PHY (C / D / M)
- Reduced Input Output (RIO) (New)
- RF Front-End
- Sensor / I3C℠ (New)
- Software (New)
- Technical Steering Group
- Test
- UniPro℠



Recent Announcements

- 05 Nov 2014 -
  MIPI Alliance Introduces Sensor Interface Specification for Mobile, Mobile-Influenced and Embedded-Systems Applications

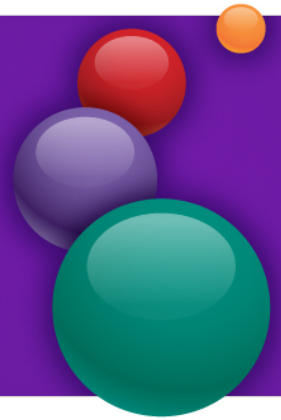
- 09 Oct 2014 -
  MIPI Alliance Introduces MIPI SoundWire℠, a Comprehensive Audio Interface for Mobile and Mobile-Influenced Devices

- 17 Sep 2014 -
  MIPI Alliance Introduces MIPI C-PHY™ Specification and Updates its D-PHY™ and M-PHY® Specifications

The Future of MIPI – Beyond Mobile

- Mobile influences **everything**
- Everything gets faster, smaller and lower power
  - MIPI will continue to evolve specs to take advantage of the evolution of technology in mobile devices

MIPI Debug WG Webinar
Taking Debug into the IoT



Gary A. Cooper
Texas Instruments Inc.
MIPI Debug WG Chair

19 November 2014

Agenda

- Why Debug?
- History of MIPI Debug
- Expanding the paradigm
- Gigabit Debug
- Future/Roadmap

Why Debug?

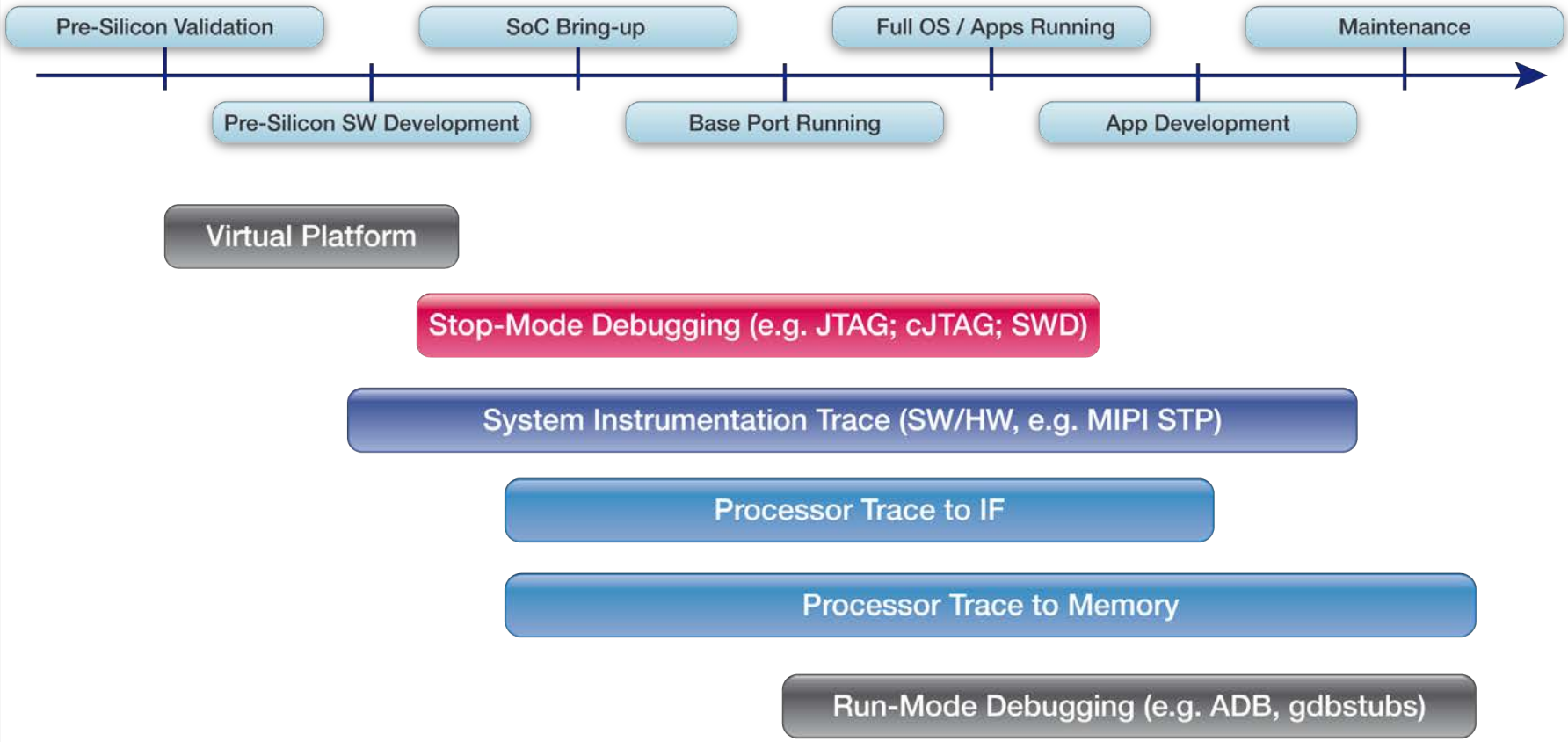
- Subtitle: An SoC Debug Architect's Lament

- Most attendees probably understand the issues debug faces:
  - We can be the heroes when we have the HW and SW in place to solve those nasty problems that block system deployment or impact stability
  - But most of the time we are considered an expensive (pins, area, power, resources) luxury
  - The constant struggle to address the rapidly evolving HW and system deployment challenges is fun, but sometimes frustrating

Requirements from Both Sides

- Silicon/System Vendors
  - Minimize gates: Reuse SoC infrastructure
  - Provide HW debug capabilities on customer platform for SoC triage
  - Provide full platform visibility (incl. SW, HW)
- OEMs
  - Visibility during the entire life-cycle
  - No additional cost:
    - No dedicated pins or connectors
    - Minimal footprint (ideally zero) for debug
  - Stable solutions
  - Easy to use
  - Ecosystem support when silicon is released
  - Standardized

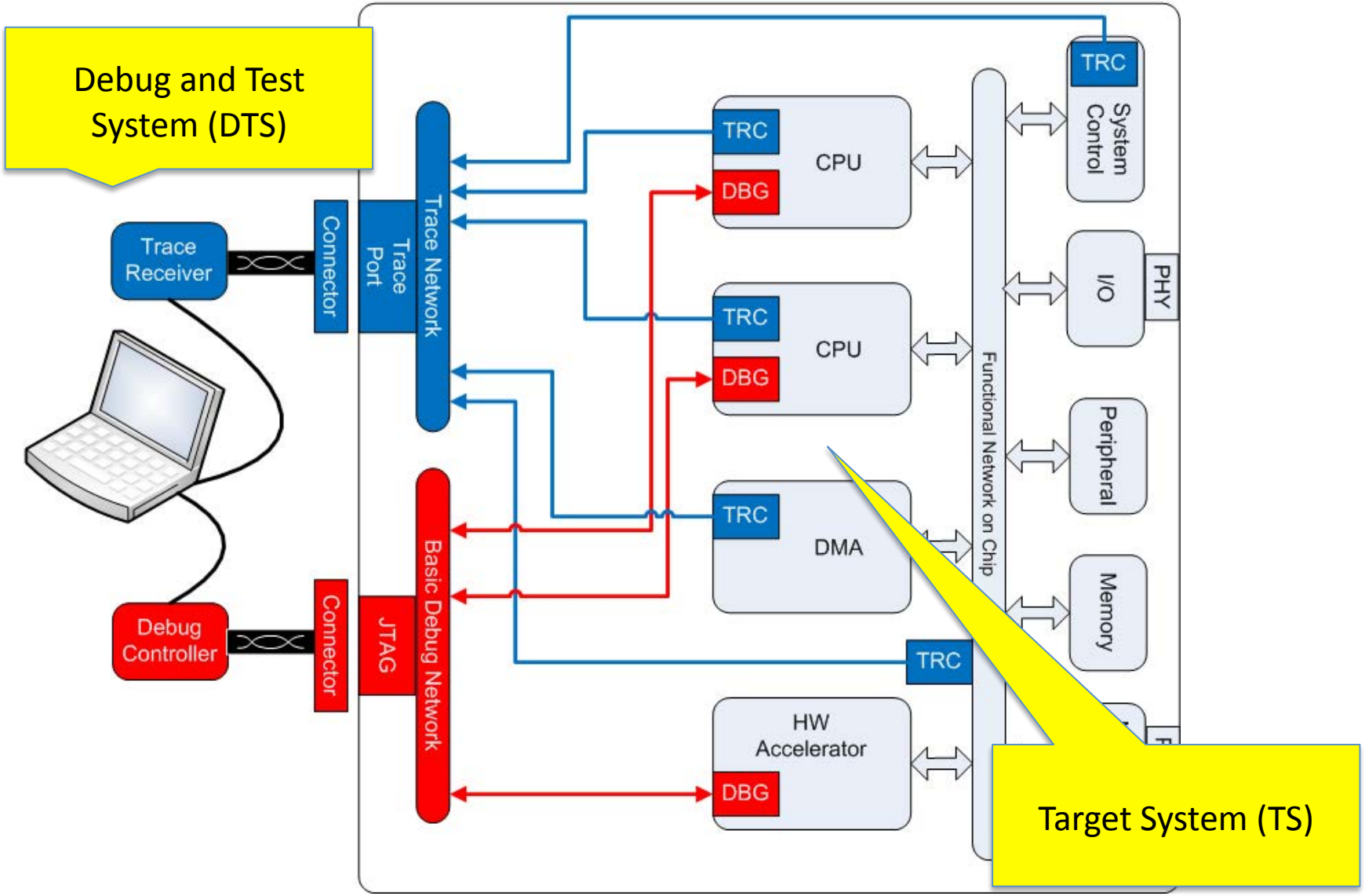
Standard System Development Timeline



MIPI Debug Terminology Cheat Sheet

- **Debug and Test System (DTS):** The combined HW and SW system that provides a system developer debug visibility and control when connected to a Target System.
- **Target System (TS):** The system being debugged. The TS may be a discrete device (a chip) or a collection of 1 to N discrete devices grouped on a board or collection of boards..

Debug in a Complex SoC



mipi alliance

Standard Debug in Action

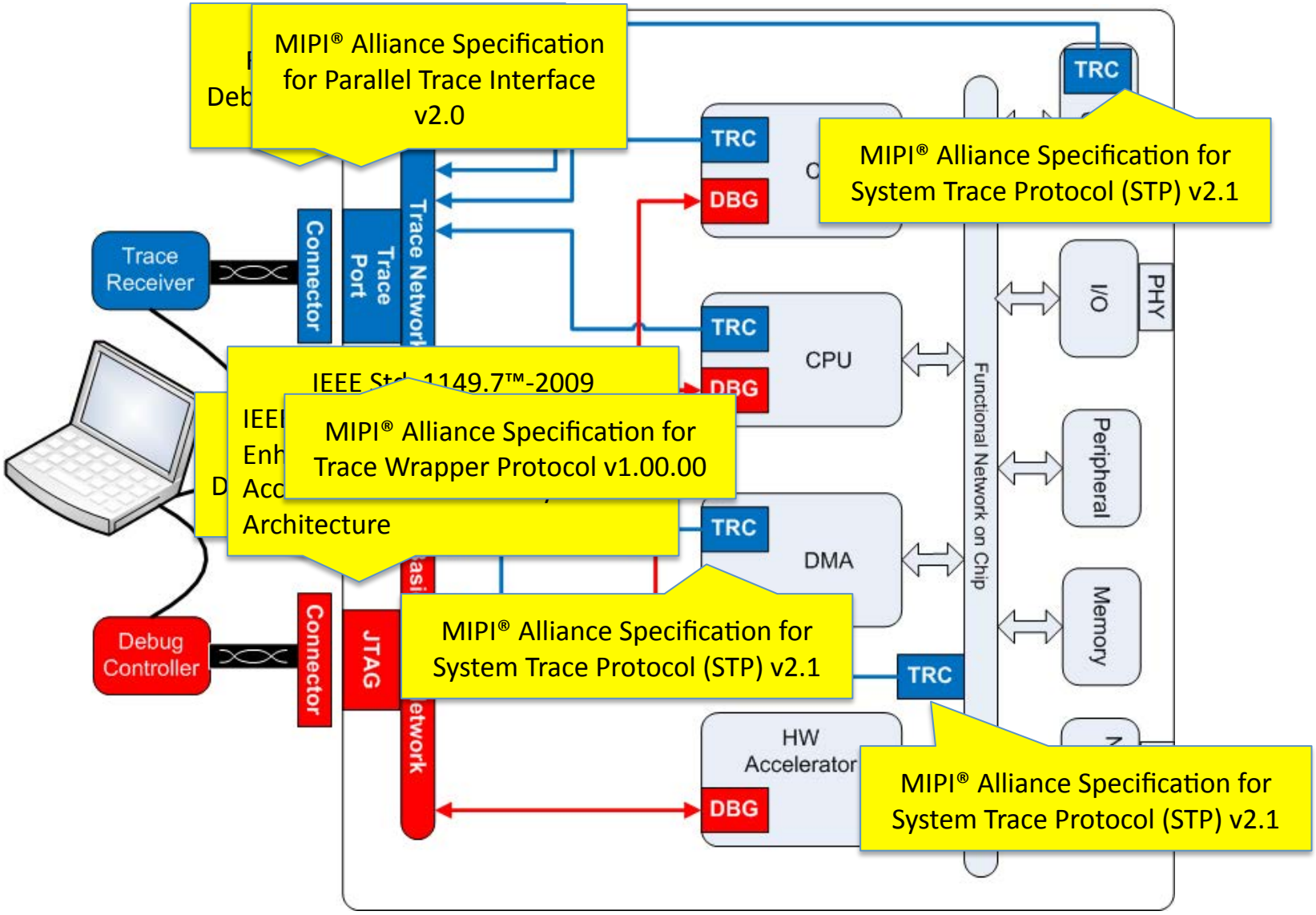
History of the MIPI Debug WG

- Formed as a MIPI IG and became a WG in 2004
- Always focused beyond mobile
  - Debug is a fairly universal problem space
- Initial focus on consolidating debug interfaces
  - Min Pin/1149.7
    - Reduced pin JTAG was spun out of MIPI to IEEE in 2006
  - Trace interface
  - Connectors
- Moved deeper into the platform (SoC and system HW)
  - Low level trace transport and merging protocols
- Current focus
  - Debug in form factor and remote debug
    - Debug reuse of functional interfaces
    - Debug over functional interfaces and networks

Overview of Current Documents

- MIPI® Alliance Recommendations for Debug and Trace Connectors Version 1.10.00
  - Basic Debug Connector (JTAG, cJTAG, SWD) connector
  - High-performance Trace Connector with Basic Debug
  - MIPI60 Trace Connector and MIPI Debug Connectors deployed in many EVMs and development systems. Supported by many debug tools vendors
- MIPI® Alliance Specification for Parallel Trace Interface v2.0
  - Interface timing specification
  - Training patterns
  - Supports multi-drop trace
  - A large number of SoC vendors and tools vendors support MIPI compatible PTIs
- MIPI® Alliance Specification for Trace Wrapper Protocol v1.00.00
  - Merging multiple byte-streams into a single trace stream
  - Padding and stream synchronization
  - Over 60 companies from across the industry have licensed MIPI-compatible IP from ARM. Multiple debug tools vendors support TWP decode.
- MIPI® Alliance Specification for System Trace Protocol (STP) v2.1
  - Supports HW and SW messaging in a simple nibble stream
  - Primitives for time-stamping, message framing, source identification, stream synchronization
  - Over 60 companies from across the industry have licensed MIPI-compatible IP from ARM. Multiple debug tools vendors support STP decode. Linaro effort underway to support STP compatible IP in Linux.
- MIPI® Alliance Specification for Narrow Interface for Debug and Test (NIDnT) Version 1.0
  - Maps Basic Debug and Trace signals to uSD connector
  - Interface switching protocol
  - Adopted by several SoC vendors and is supported by multiple tools vendors

MIPI Debug Specifications and the SoC

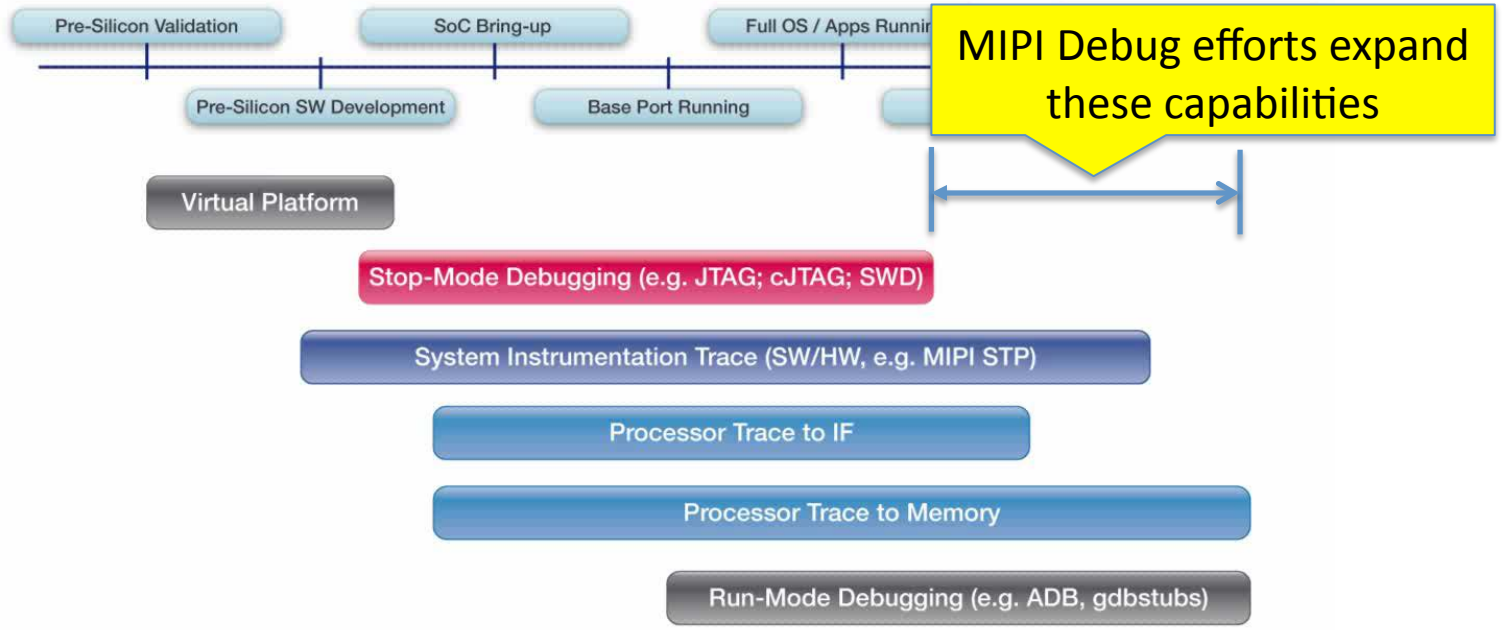


mipi alliance

Expanding the Debug Paradigm

- Problem statement
  - Systems are physically small but internally more complex
  - Short Time-to-Market
    - Development windows decreases
    - Debug in form factor device must have access to all debug features in the SoC
    - Remote debug via the network (IoT) must have access to all debug features in the SoC
- MIPI Debug Solutions
  - Temporarily repurpose functional interfaces when debug is required
  - Leverage functional interfaces (like networks) for interacting with debug while coexisting with functional traffic

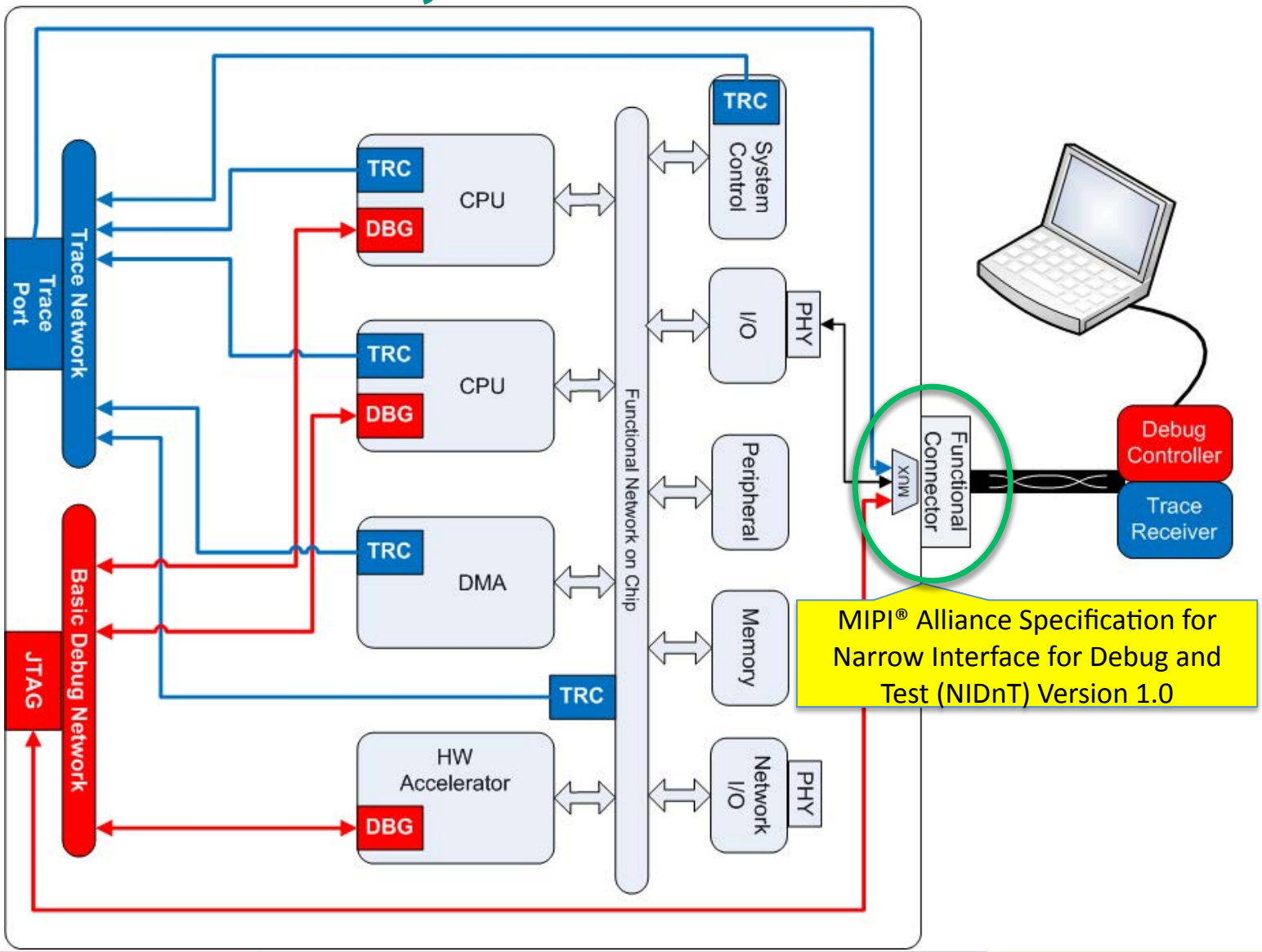
MIPI Debug Expands the Development Timeline



Expanding the Paradigm - NIDnT

- Narrow Interface for Debug and Trace
- More debug capabilities enabled on a form factor device
- Pin/interface reuse
  - We are stealing the pins. Normal functionality is disabled
  - Pins use existing debug protocols and tools
- Initial focus on uSD interface
  - Multiple pin maps to support different debug scenarios
    - Debug and Trace possible
  - Graceful interface switching methodology
- More interfaces to come in the future

NIDnT in the System



mipi alliance

NIDnT in Action

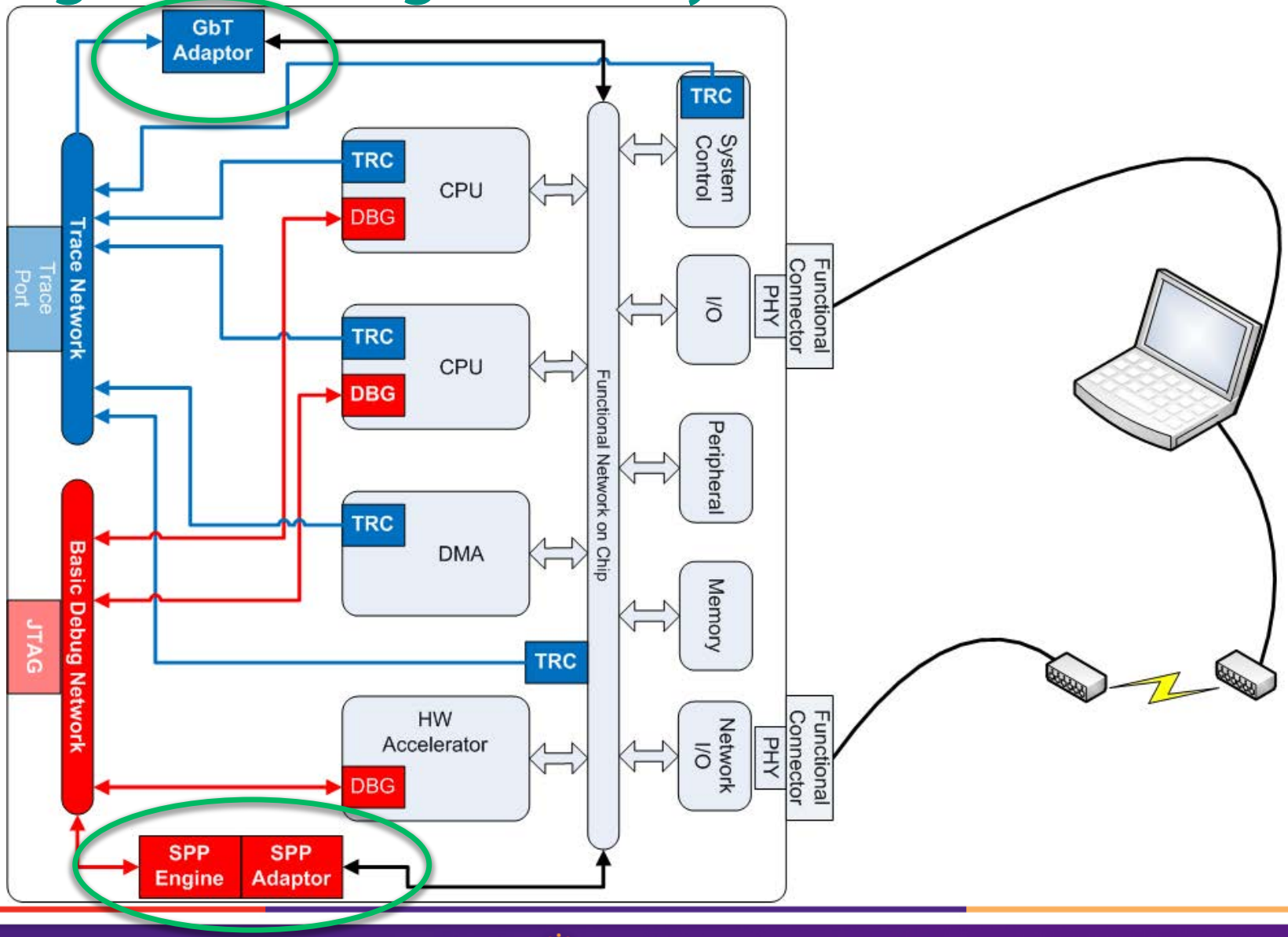
Expanding the Paradigm - Gigabit Debug

- Complex SoC debug often requires significant bandwidth for debug functions like trace
- High-performance (Gigabit) interfaces now available on many systems
- Many/most of these interfaces support transport layers that enable use by multiple clients
- *Why not leverage these powerful data paths for debug use when required?*
- The goal for GbD is to expand the availability of deeply embedded debug resources so developers and system tuners can always interact with them with the same level of visibility as the lab
- Expose these features with minimal software/system intrusion
- Debug and functional traffic ***share*** the interface/link so full system debug is possible
- In many cases, we can eliminate the need for expensive debug tools hardware (e.g. dedicated Trace Receivers)

MIPI Gigabit Debug Framework

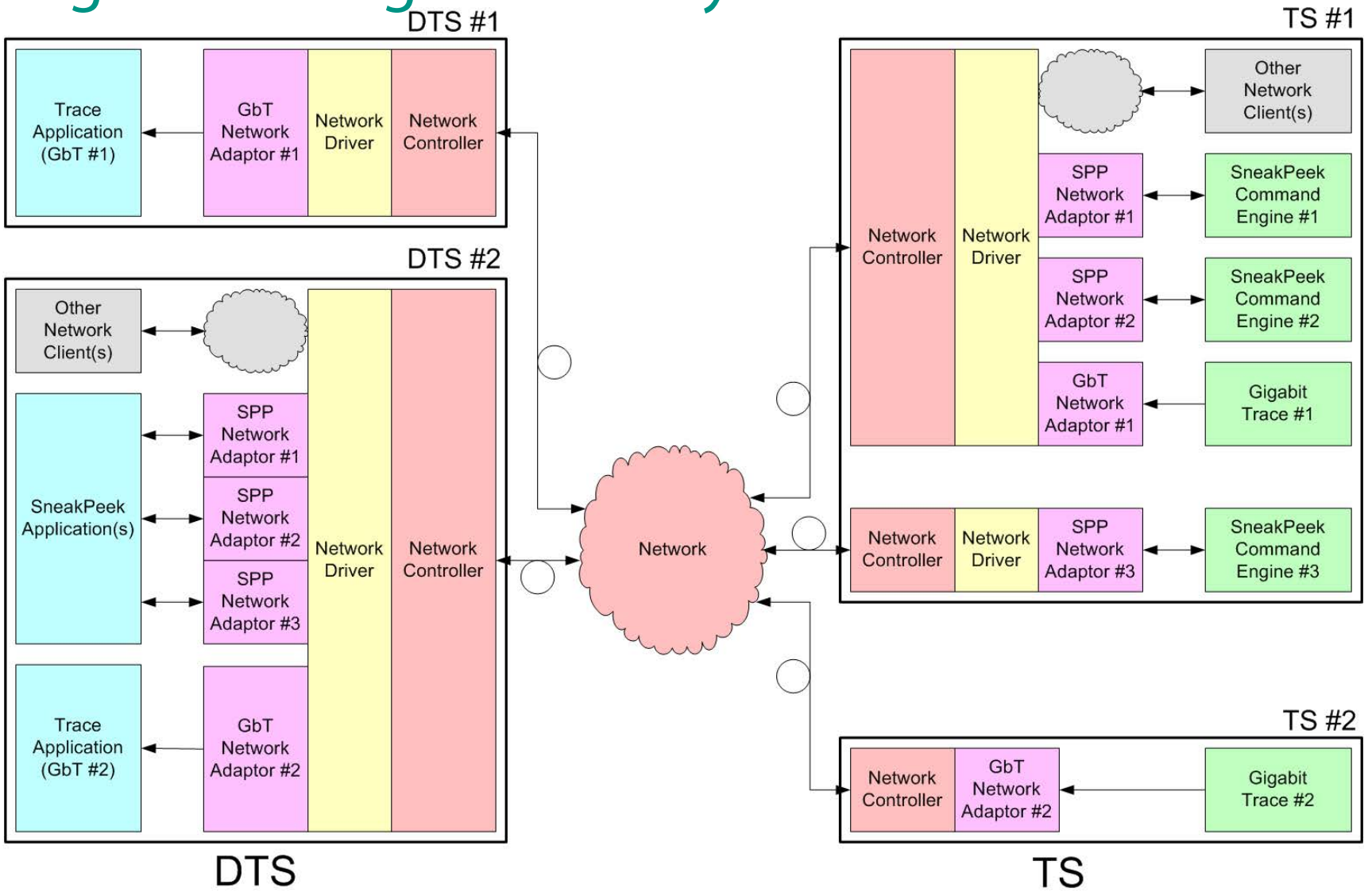
- Develop interface independent protocols and frameworks to address the new debug paradigms
  - SneakPeek Protocol (SPP) – Provide debug visibility and system control (JTAG replacement)
  - Gigabit Trace (GbT) – Streaming non-intrusive debug data to the host
- Map these protocols to specific functional interfaces/ networks
  - Current Efforts
    - Gigabit Debug for USB
    - Gigabit Debug for IP Sockets (UDP/TCP)
  - More to come

Gigabit Debug in the System



mipi alliance

Gigabit Debug – Macro System View



GbD in Action

USB

GbD in Action in the IoT

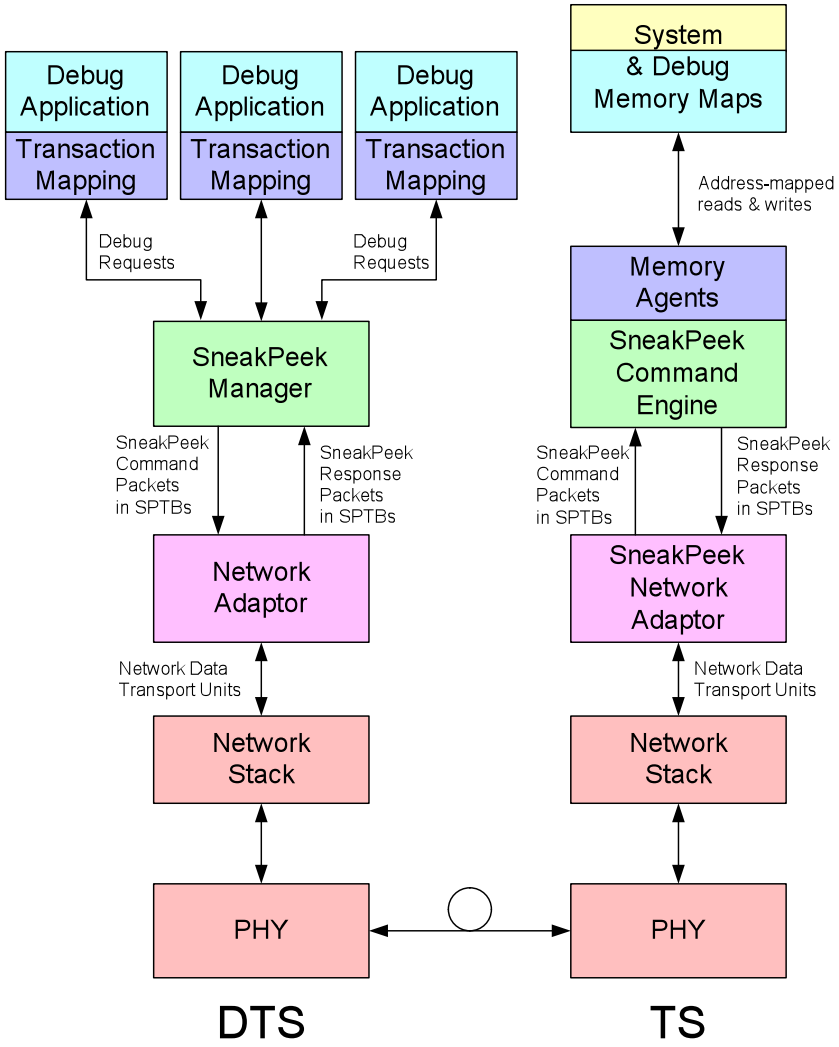
But people always debug using USB???

- Yes, but not in the way you can with GbD
- Gigabit Debug **focuses on leveraging the same on-chip debug resources used in early-phase development**
  - Fine grained visibility (into HW and SW) with low intrusion
  - Tools reuse throughout the development timeline
  - Can get you closer to the "bare-metal" debug experience
    - Stop-mode debug possible
    - High-bandwidth, non-intrusive trace from HW sources
- Run mode debug tools like GDB, ADB, system logs etc. are software/OS centric
  - Rely on large parts of the system being functional
  - Relatively high level of intrusiveness
  - Not closely coupled to on-chip debug hardware
- Not everyone needs GbD, but the power is there when you do

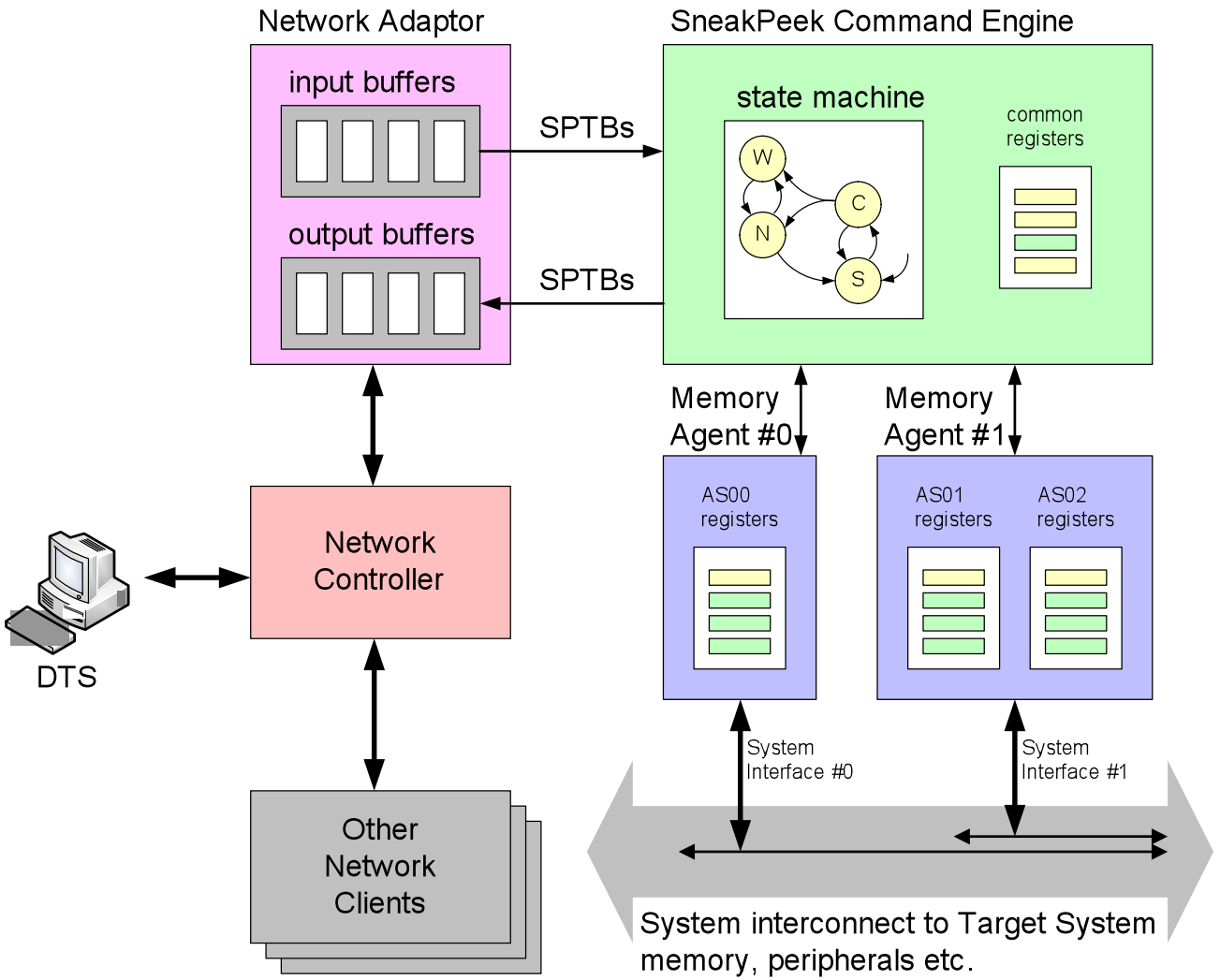
SPP Overview

- Command/Response protocol driven by the tooling
  - Command and Response Packets collected in SneakPeek Transfer Blocks (SPTBs)
- Focused on system and debug peripheral memory access
  - Use of internal scan for debug is waning
  - Layered approach abstracts the SPP protocol from any specific debug functionality
    - We are not developing a new debug command/control protocol
    - Most existing protocols can sit on top of SPP
- Packet based protocol
  - **Simple enough to be implemented in dedicated HW modules**
  - Easily adapted to various transport/network layers
- Network latency tolerant for higher throughput while maintaining error feedback through command/response

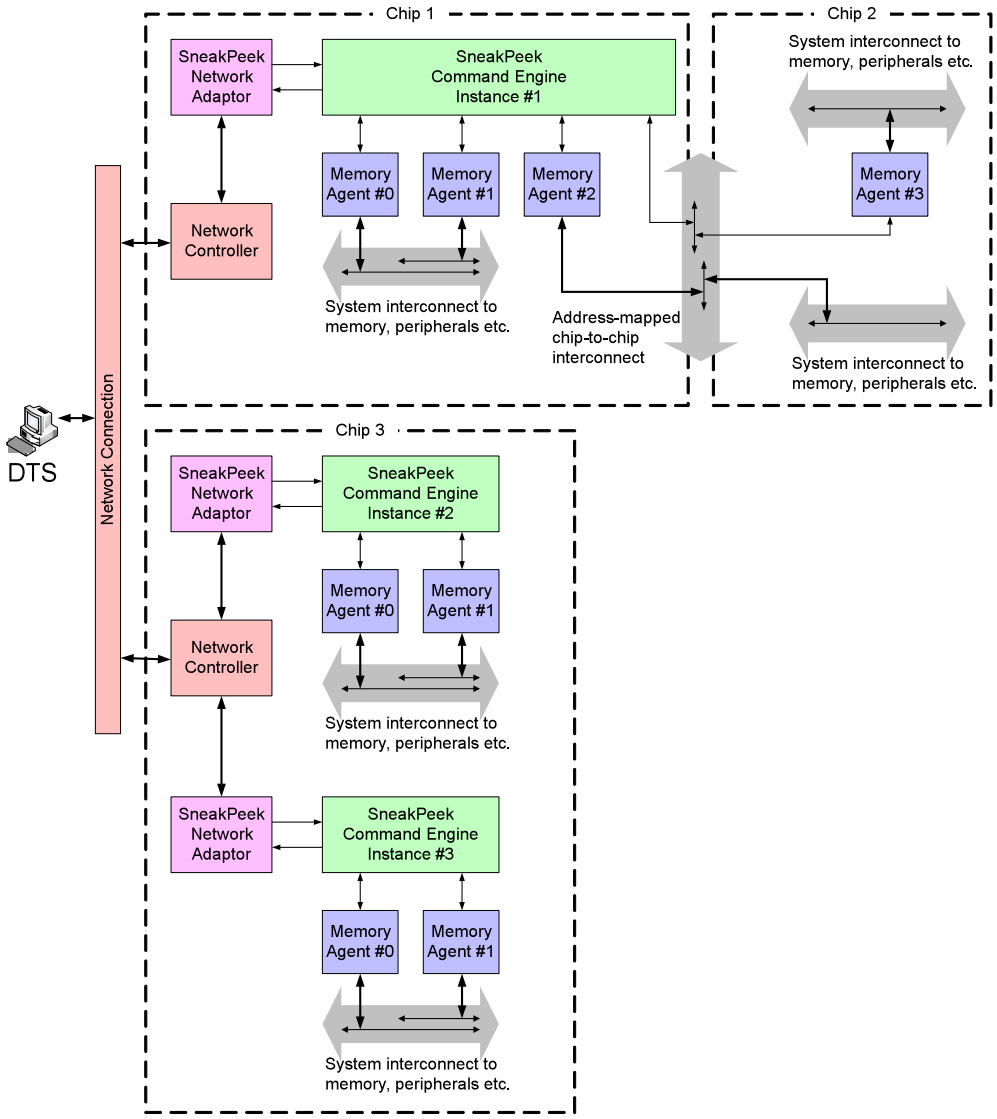
SPP Stack View



SPP Implementation Overview (Conceptual)



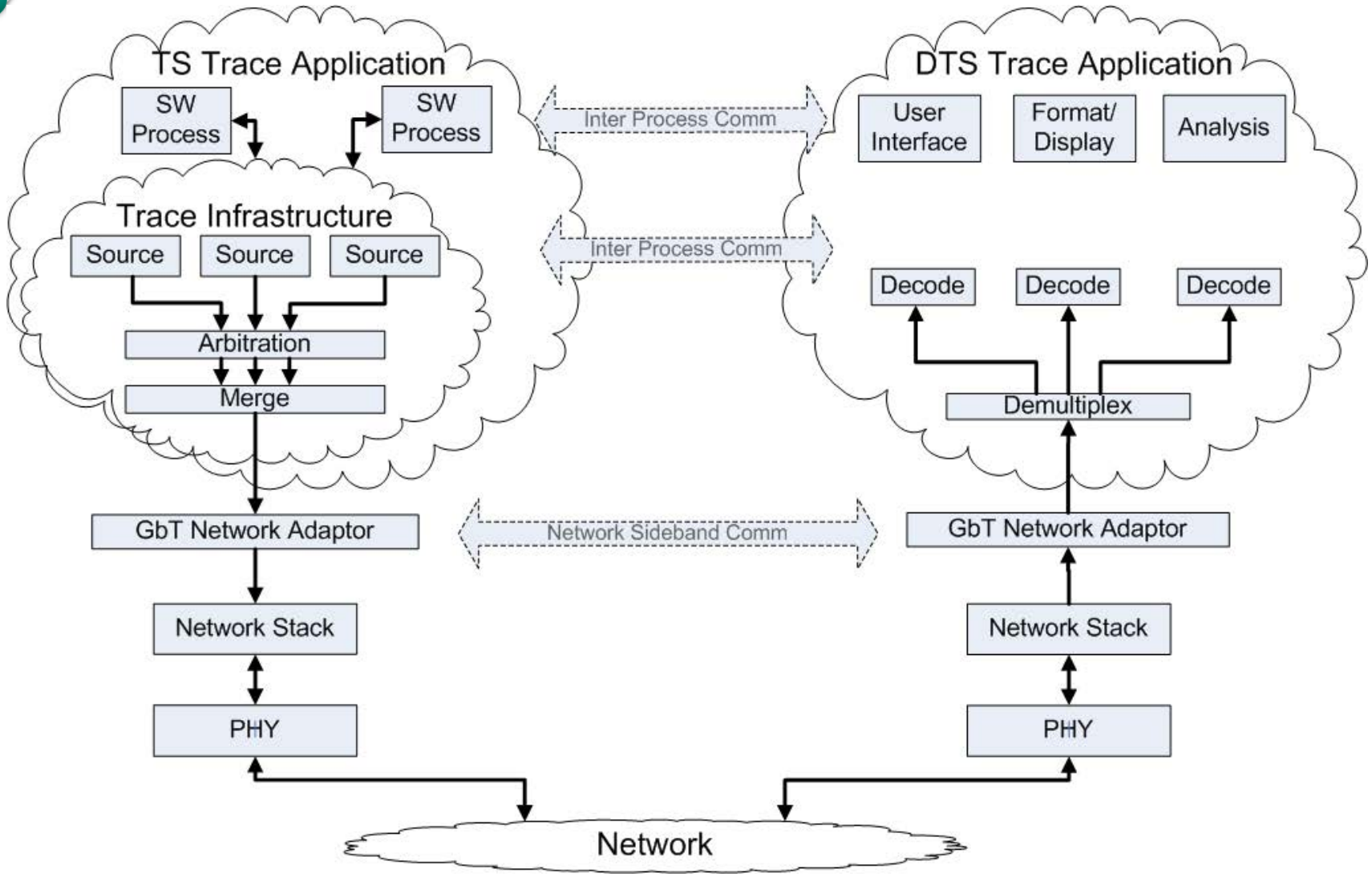
SPP in a Complex System Architecture



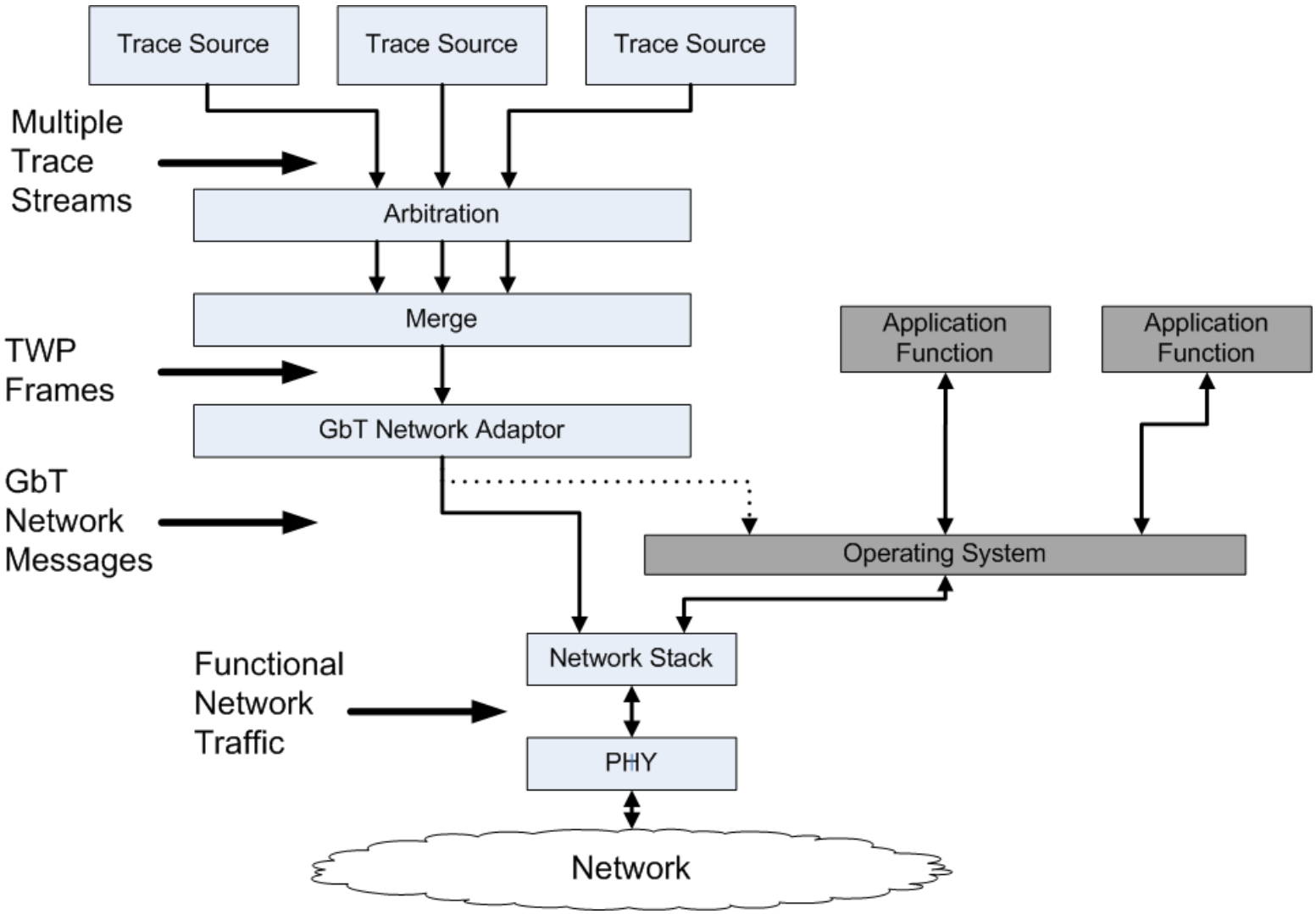
GbT Overview

- Streaming protocol support spontaneous data from the TS
- Leverages the existing primitive structures defined in the Trace Wrapper Protocol specification
  - Network messages consist of the standard TWP Data Frames, Syncs and Padding
  - HW and tooling support already exists
- Packet based protocol
  - **Simple enough to be implemented in dedicated HW modules**
  - Easily adapted to various transport/network layers
- Comprehends the need for trace specific functionality like flushing the trace infrastructure

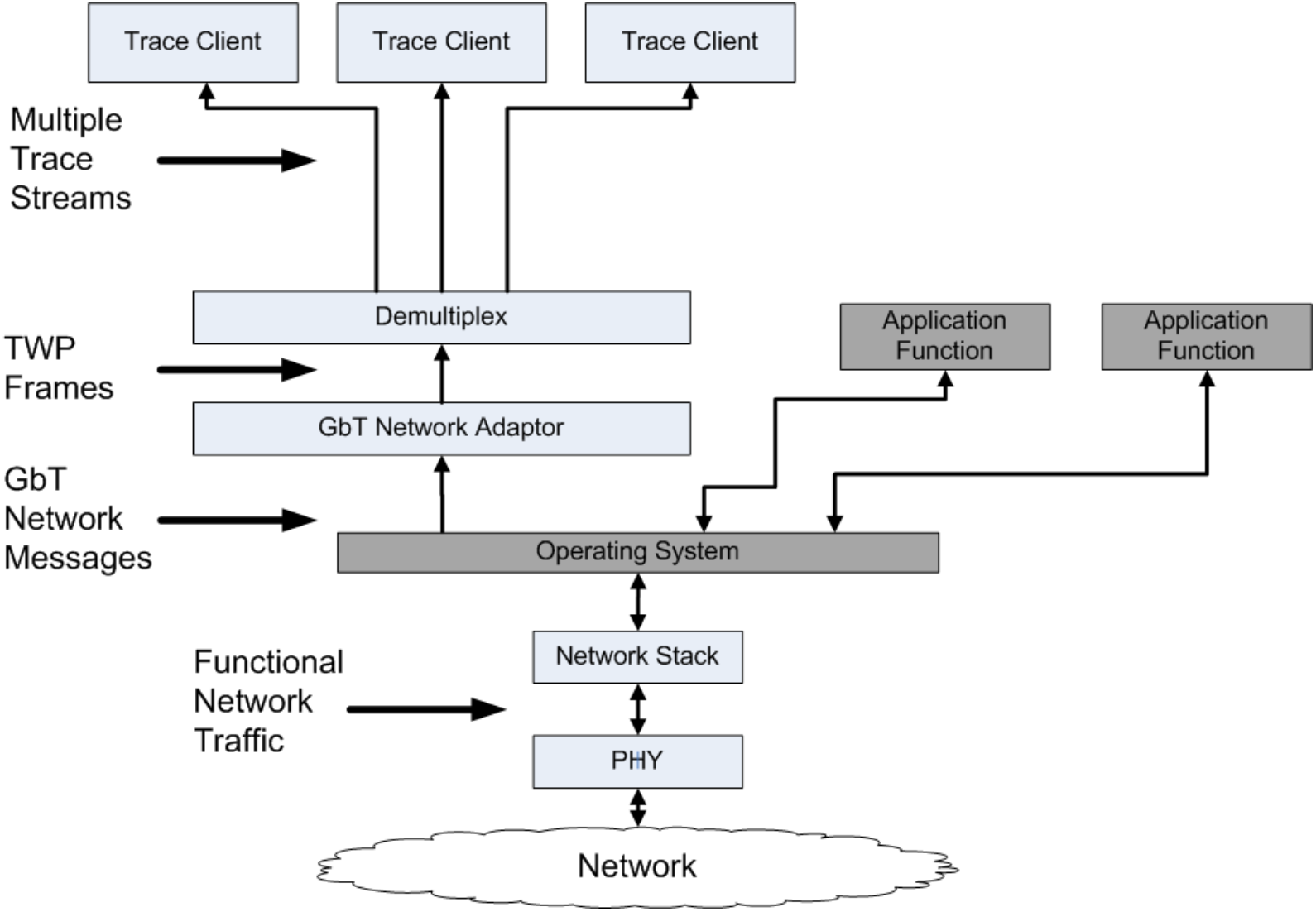
Conceptual View of a GbT System



GbT Data Flow in the TS



GbT Data Flow in the DTS



Of Course Everything Isn't Perfect…

- Expanding the paradigm introduces numerous challenges
  - Security
    - Debug is always considered a weak point and now we are making it more visible
    - The MIPI approach is to push the security requirements into the transport/network and/or application layers
  - Bandwidth
    - Mainly a GbT issue over routable networks
    - Local network bandwidth will support most trace scenarios, but the usability will decrease as distance impacts throughput
    - GbT will still be very useful for many debug and performance profiling scenarios where non-intrusive visibility is important and bandwidth requirements are not intensive
  - Power
    - Debug may require changes to power management to keep interfaces and infrastructure active when normal functionality would allow them to power down
  - Complexity and Intrusiveness
    - Some network transports (like TCP) generally require a significant amount of OS/SW in the stack
    - GbD for IP Sockets is developing a "lossless UDP" layer that can be implemented in HW

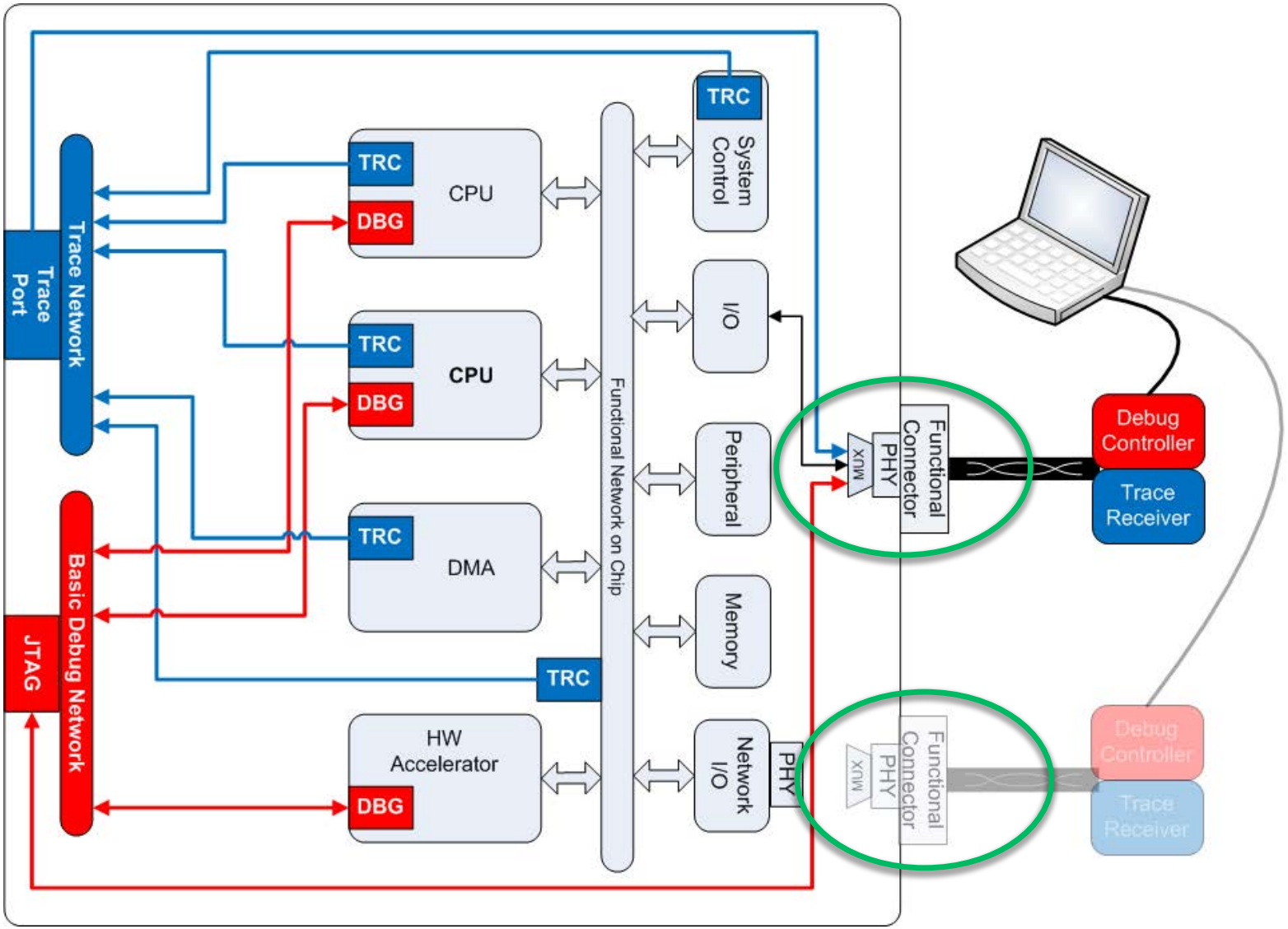
Gigabit Debug Status

- Gigabit Trace is defined in an update to the Trace Wrapper Protocol Specification that is currently in approval pipeline for MIPI Specifications
- The Sneak Peak Protocol Specification is in final editorial cleanup and will enter the review and approval pipeline soon
- Gigabit Debug for USB is in final technical edits and will be released into the approval pipeline very early in 2015
- Gigabit Debug for IP Sockets should be released into the approval pipeline in the first half of 2015

Future MIPI Debug Efforts

- New NIDnT Interfaces
  - Moving from pin reuse to PHY reuse
  - Leverages the performance of these interfaces for increased trace bandwidth
- GbD over new interfaces and networks

NIDnT with PHY Reuse



mipi alliance

Conclusion/Summary

- More information:
  - MIPI Debug WG Public Page
  - MIPI Architecture Overview for Debug
  - MIPI Debug on Wikipedia
- Join the MIPI Debug WG and save the world
  - Currently around 10-12 companies actively participating
    - SoC, IP, Tools vendors
  - Requires MIPI Contributor membership
  - Telcos every other Tuesday at 1600 or 1700 UTC
    - Summer/daylight time switch
    - 90-120 minutes depending on the agenda
  - Three F2F sessions with all MIPI WGs
    - Generally meet for 4 days

Q and A

MIPI Debug WG members at the Shanghai F2F in October 2014