# From Virtual Targets to USB: Upcoming SoC Debugging Approaches

**Michael Eick (Lauterbach GmbH)**
**Rolf Kühnis (Intel Finland Oy)**
2014/ 10 / 08

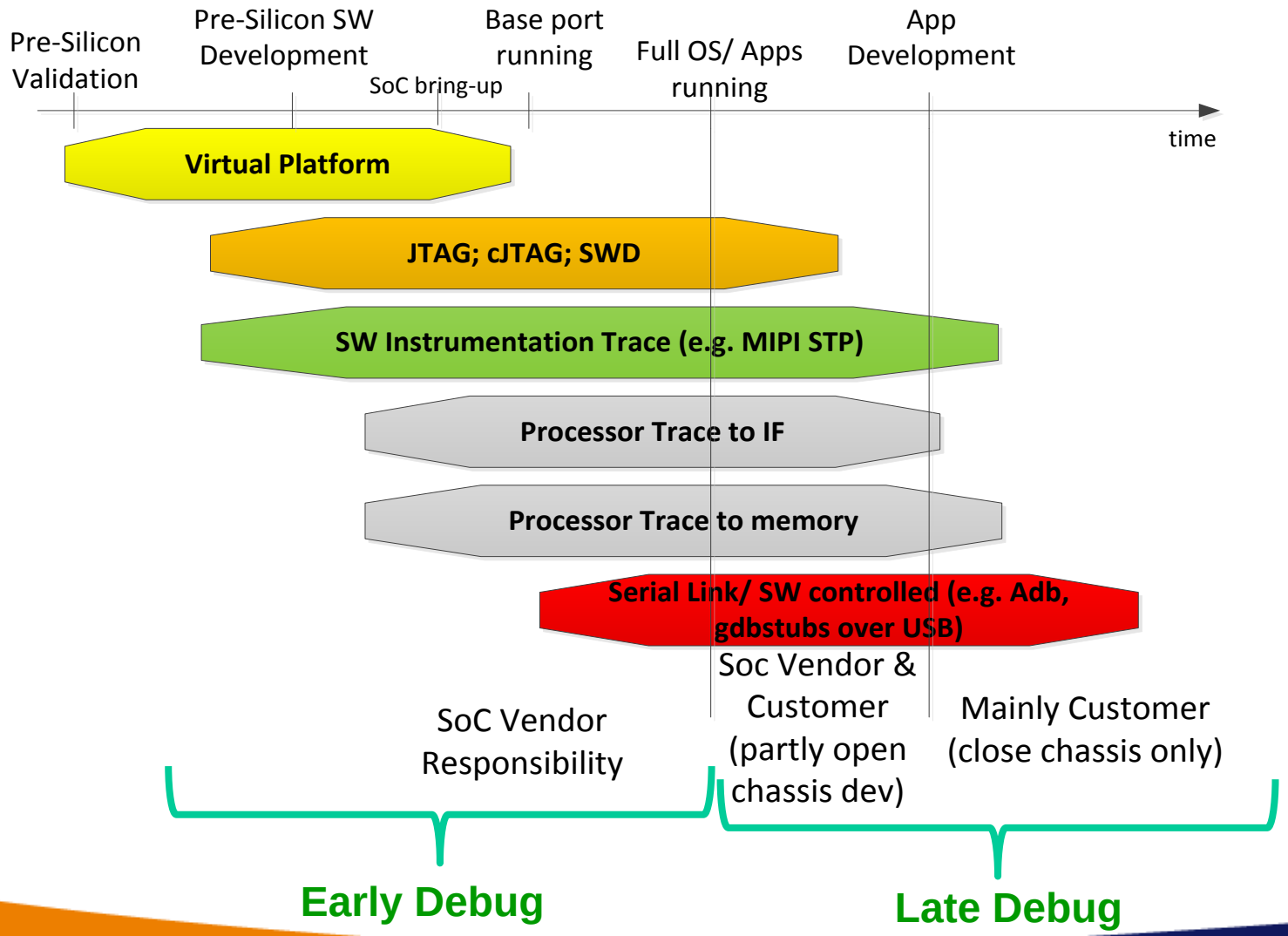


LAUTERBACH
DEVELOPMENT TOOLS

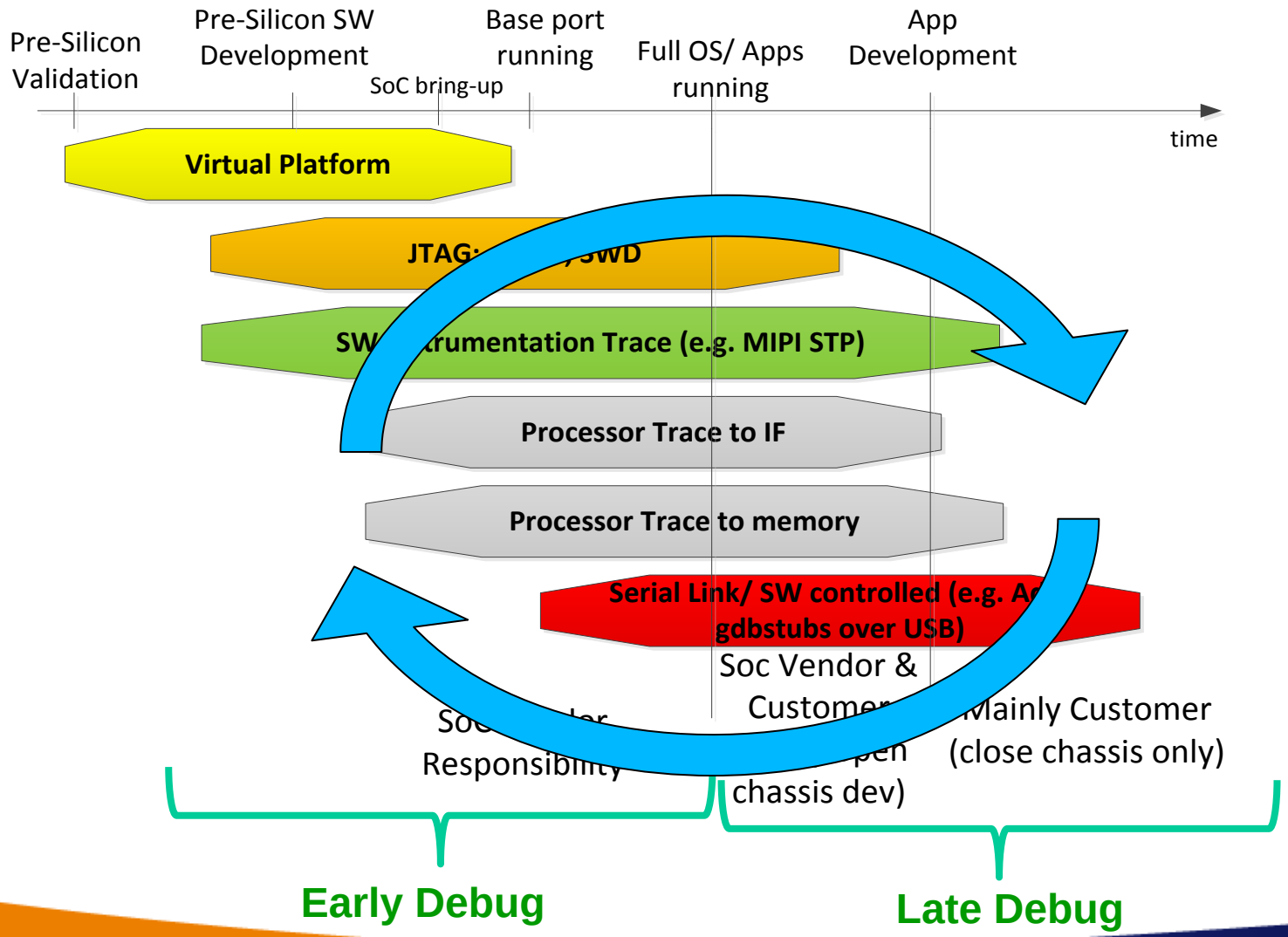
# Agenda

- Industry' desires
- Upcoming Approaches
  - Debugging
  - Trace
- Conclusions

# Development Cycle

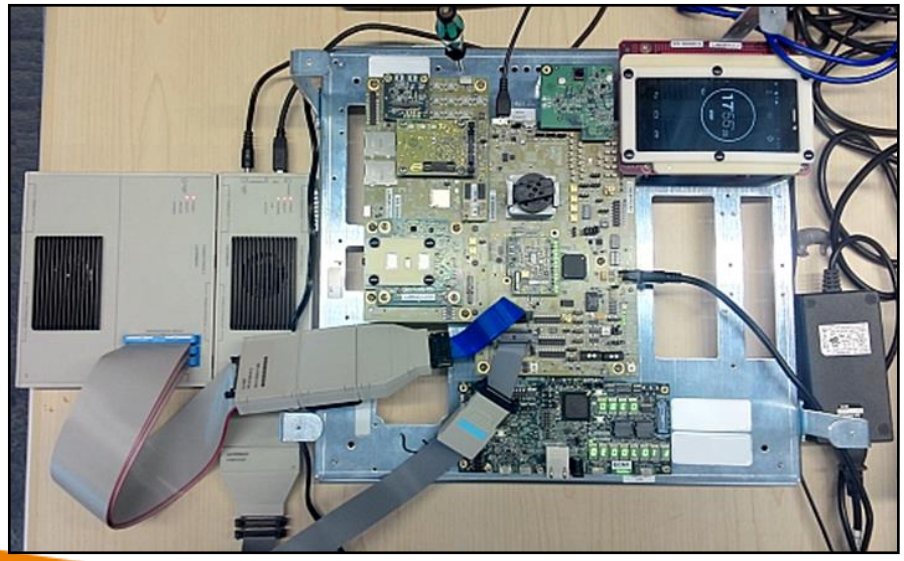


# Development Cycle



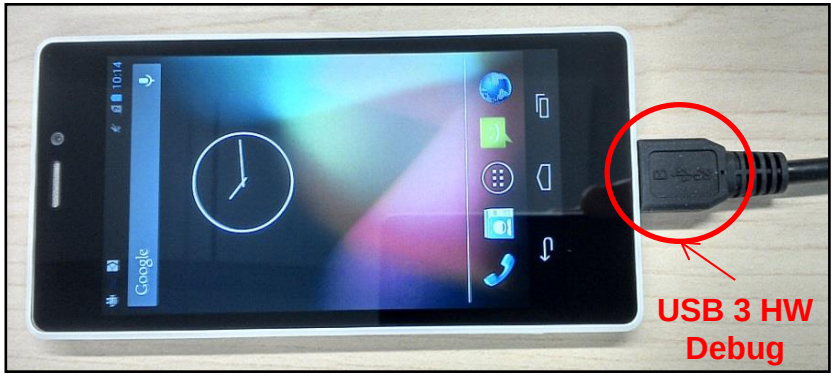
# Requirements from Chip Manufacturers

- Minimize gates: Reuse SoC infrastructure
- Provide HW debug capabilities on customer platform for SoC triage
- Provide full platform visibility (incl. SW, HW)



# Requirements from OEMs

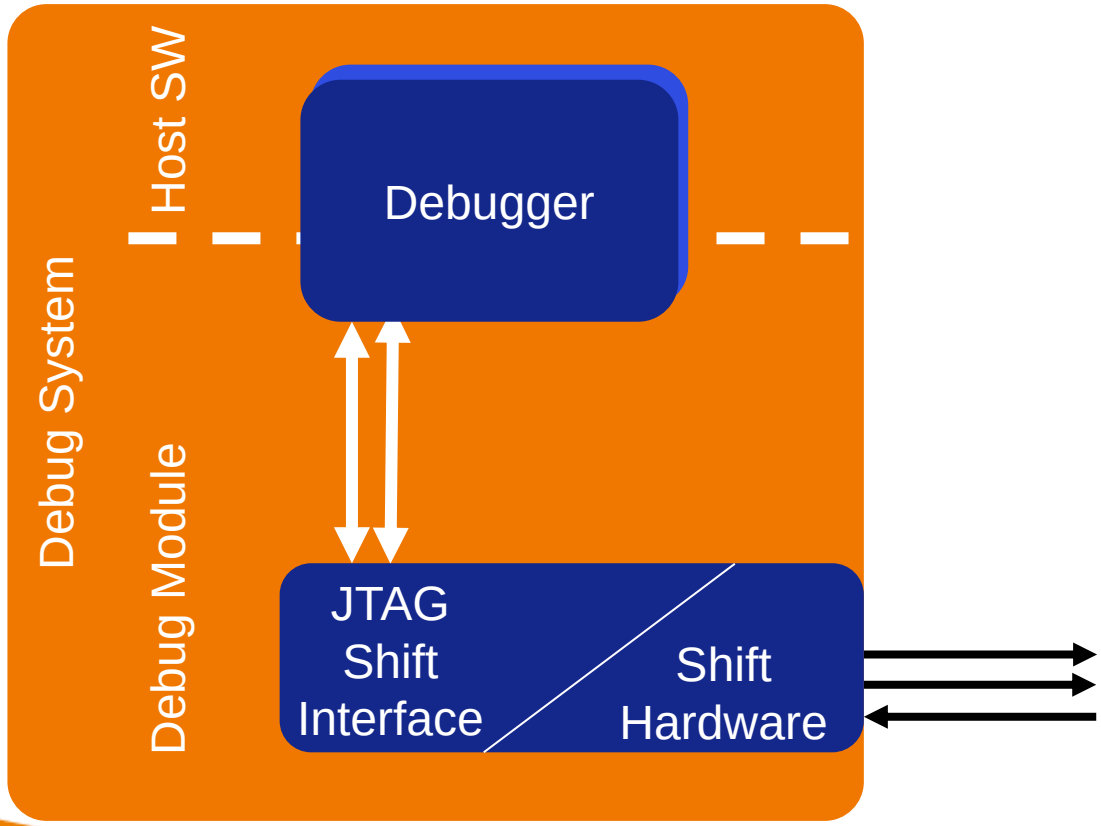
- No additional cost:
  - No dedicated connectors
  - Minimal footprint (ideally zero) for debug
- Stable solutions
- Easy to use
- Ecosystem support
- Standardized



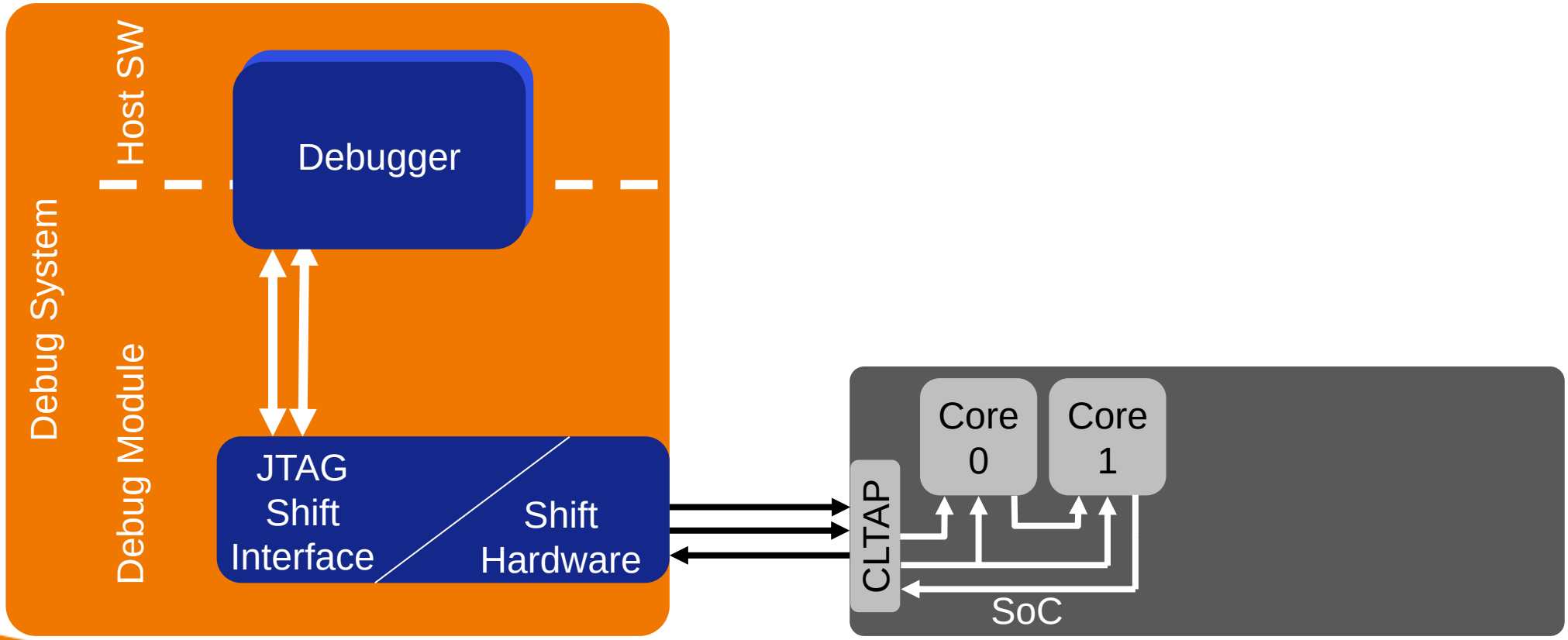
# Agenda

- Industry' desires
- Upcoming Approaches
  - Debugging
  - Trace
- Conclusions

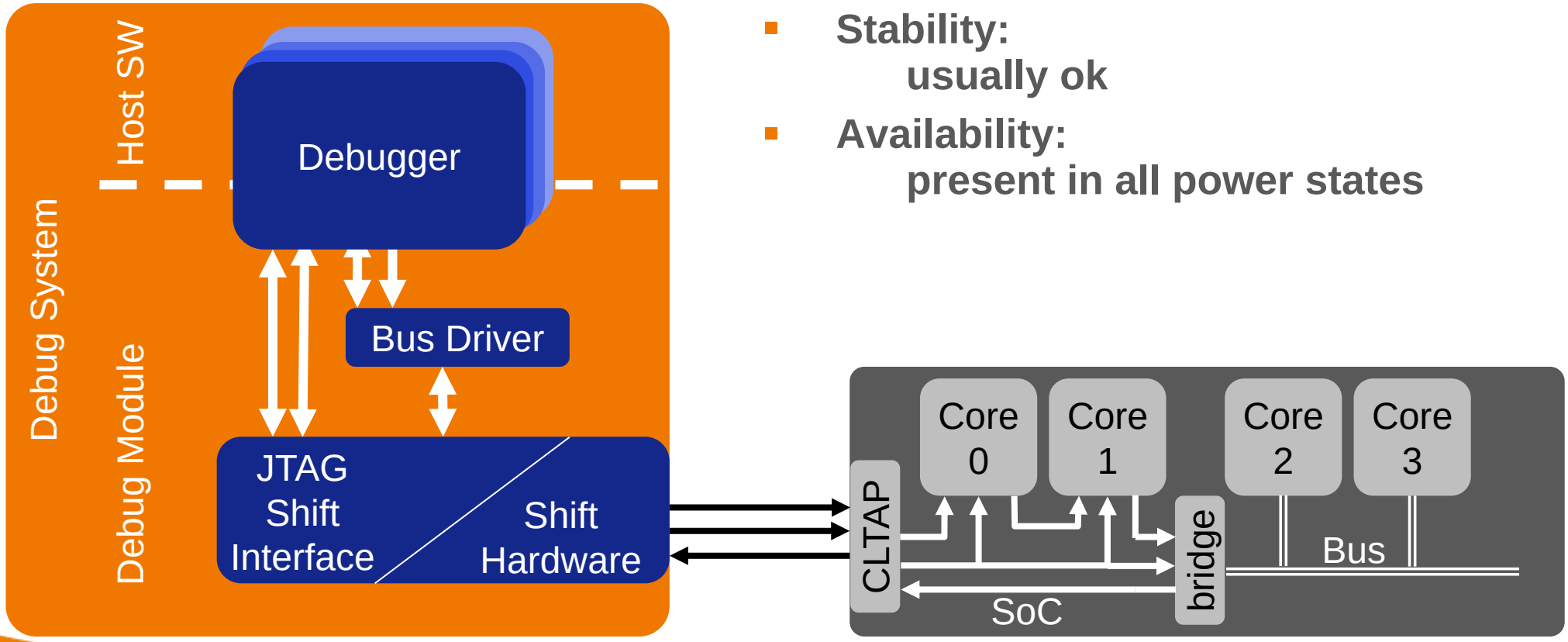
# State of the Art: JTAG



# State of the Art: JTAG

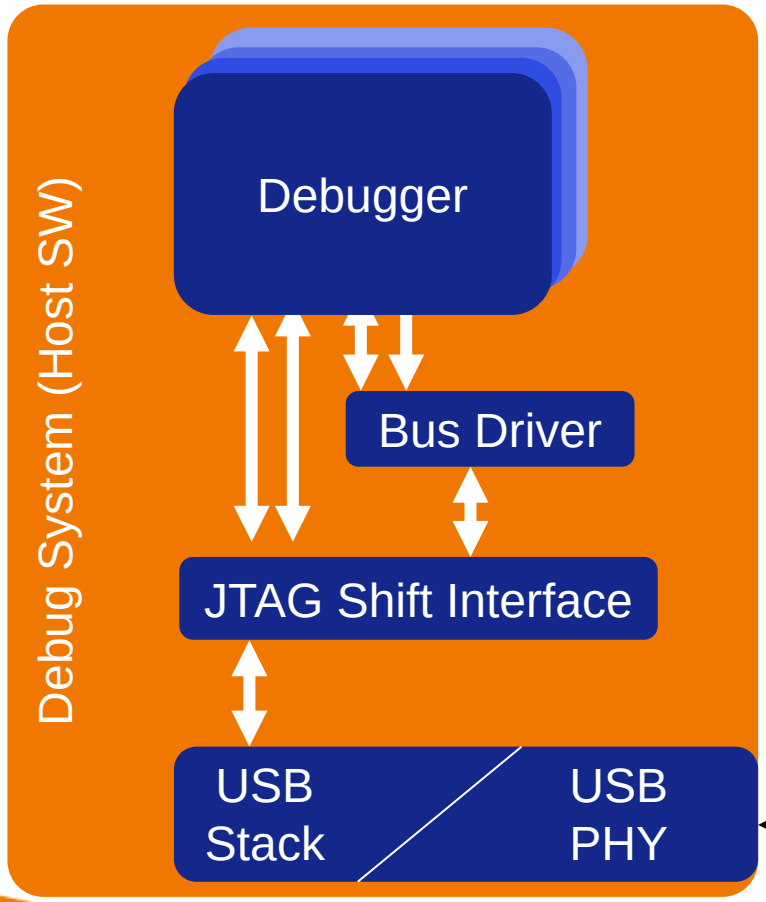


# State of the Art: JTAG

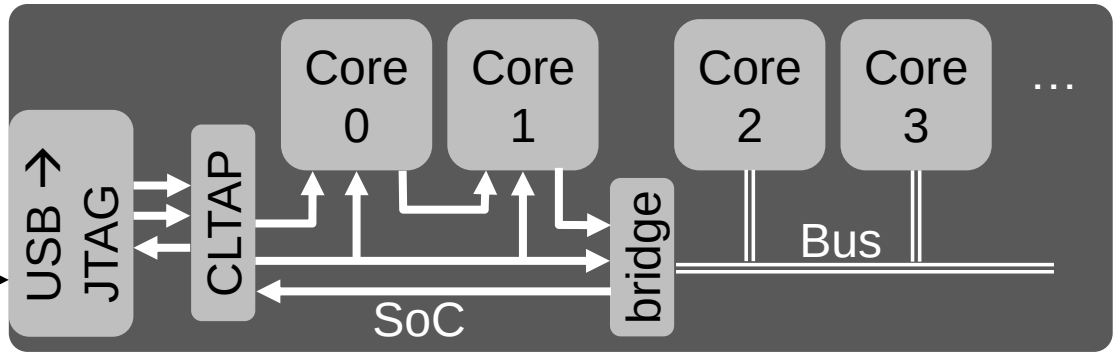



- **Dedicated port:**
  **yes**
- **Stability:**
  **usually ok**
- **Availability:**
  **present in all power states**

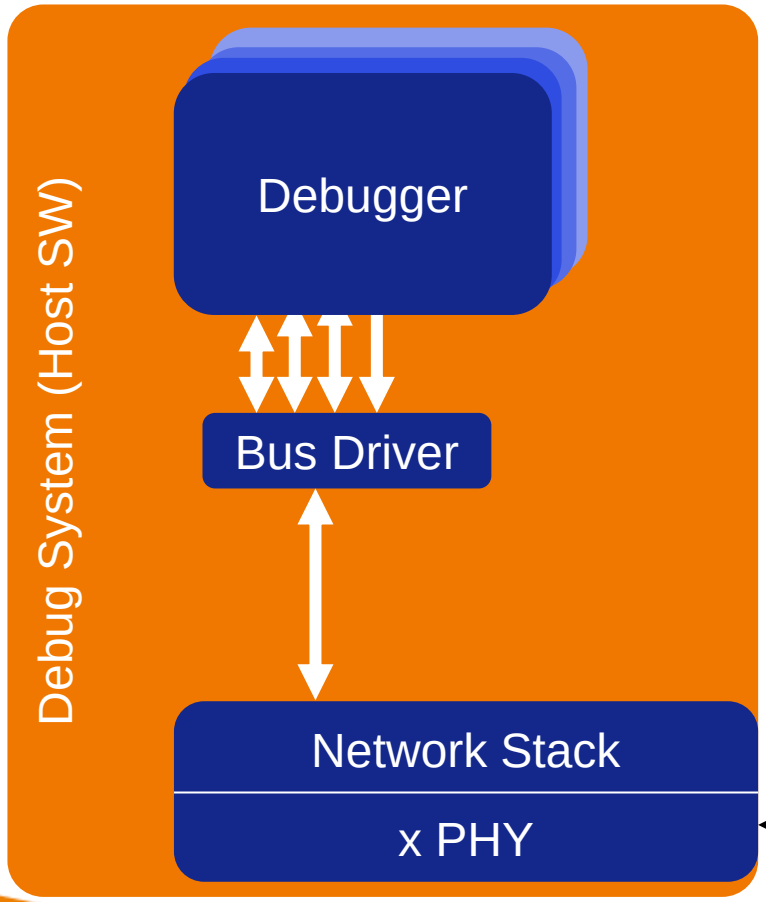
# JTAG over USB



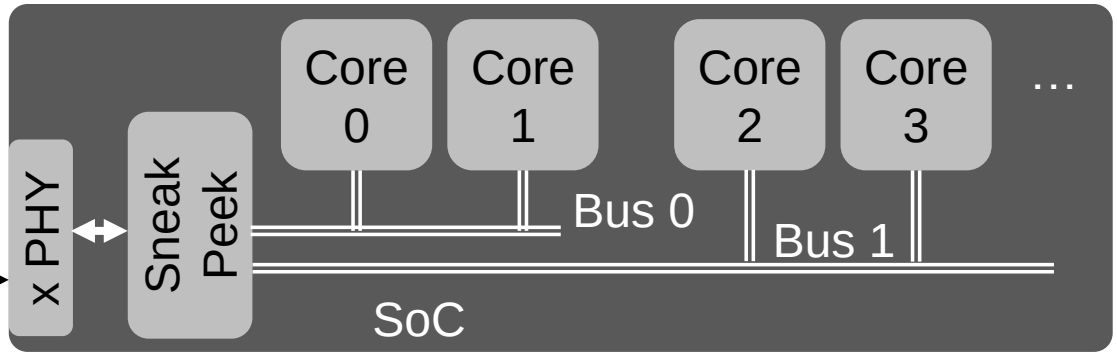

- **Dedicated port:**
  **no**
- **Stability:**
  **depends on USB port**
- **Availability:**
  **might be off in some power states**



# MIPI: SneakPeek$^{SM}$



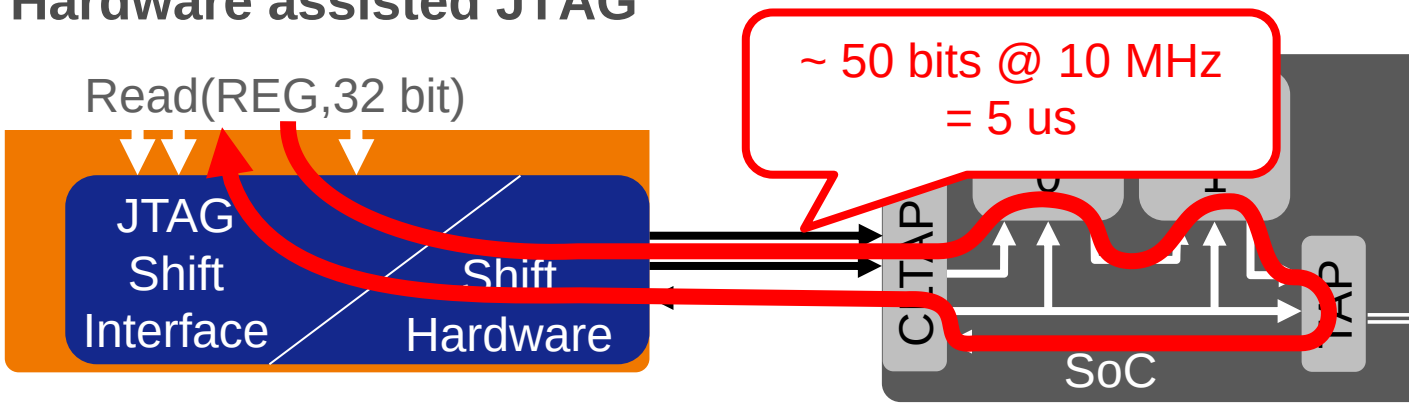

- **Dedicated port:**
  **no**
- **Stability:**
  **depends on “PHY”**
- **Availability:**
  **might be off in some power states**




[MIPI Architecture Overview for Debug; www.mipi.org]

# Does it behave like JTAG?

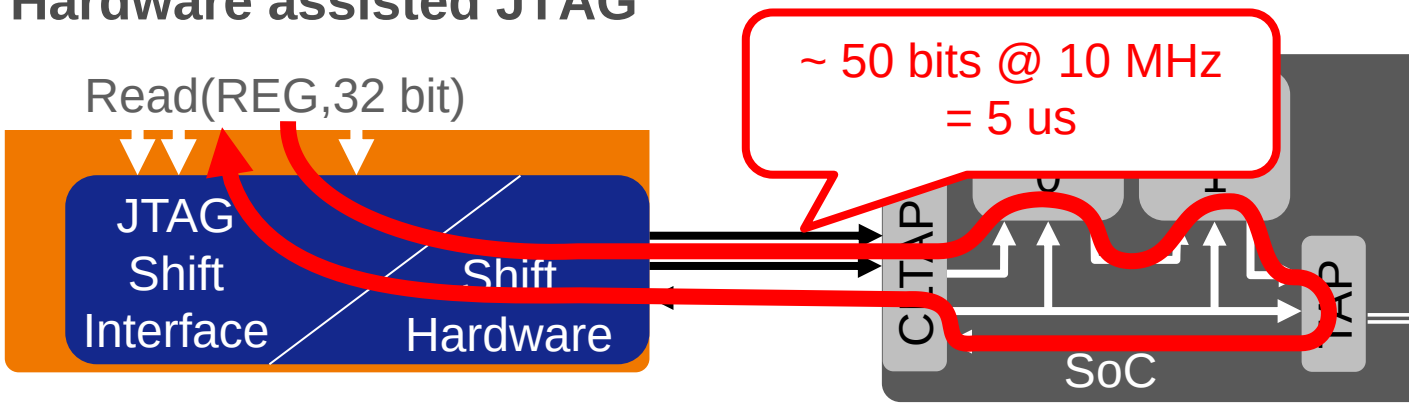
- **Hardware assisted JTAG**




**Total time**
**~ 5us**

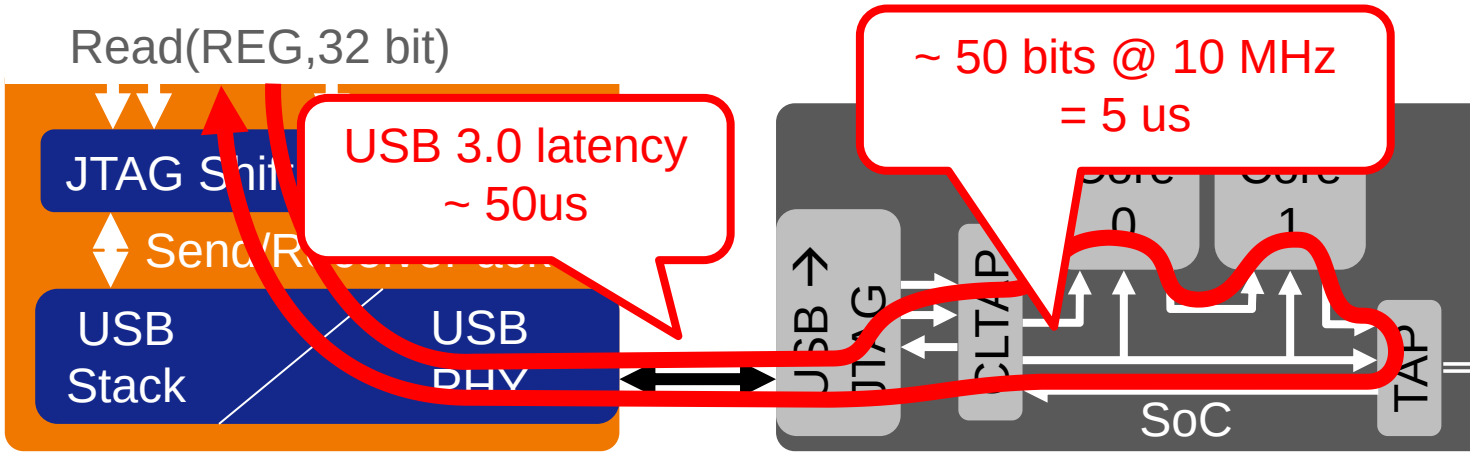
# Does it behave like JTAG?

- **Hardware assisted JTAG**




**Total time**
**~ 5us**

- **JTAG over USB / SneakPeek / …**

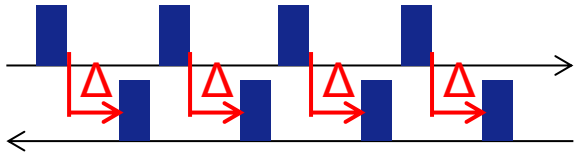



**Total time**
**~ 55us**

# Performance Considerations

**Example: Block read**

**Read single 32-bit words**

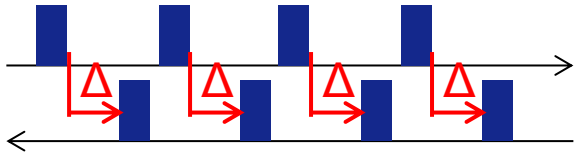


- Data rate strongly depends on latency → USB slower!

# Performance Considerations

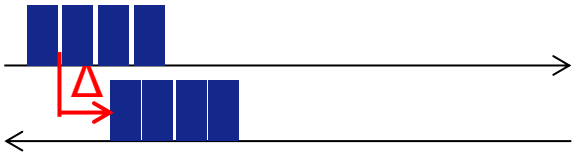
## Example: Block read

### Read single 32-bit words

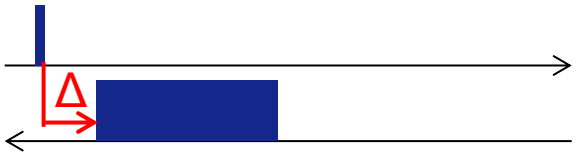


- Data rate strongly depends on latency → USB slower!

### Read single 32-bit words (pipelined)



### Dedicated block access

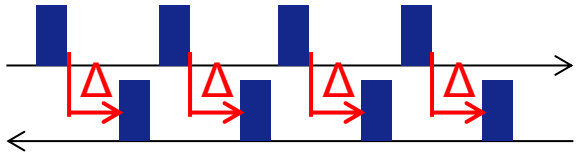


- Latency has less influence
- Availability depends on
  - Debug protocol
  - Hardware capabilities

# Performance Considerations

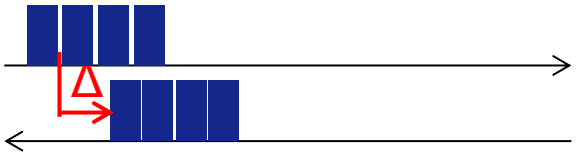
**Example: Block read**

**Read single 32-bit words**

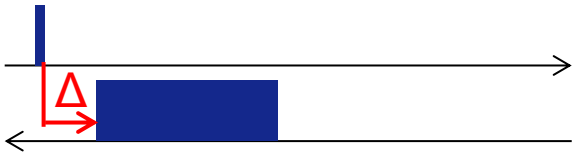


- Data rate strongly depends on latency → USB slower!

**Read single 32-bit words (pipelined)**



**Dedicated block access**

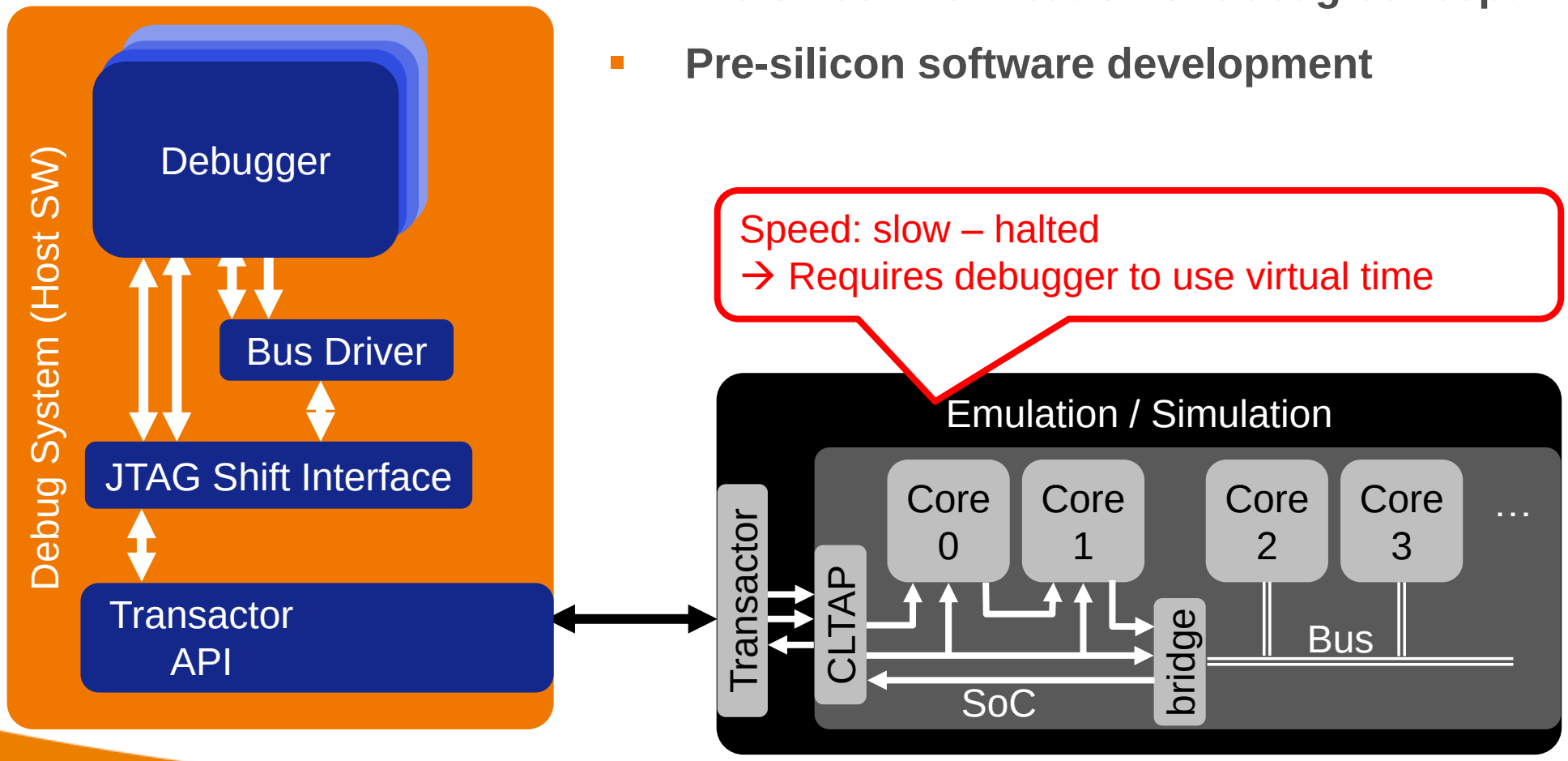


- Latency has less influence
- Availability depends on
  - Debug protocol
  - Hardware capabilities

optimal solution depends on debug link, memory, …
high software complexity → high implementation effort
→ high verification effort

# Debugging using Virtual Prototypes

- Pre-silicon verification of debug concept
- Pre-silicon software development

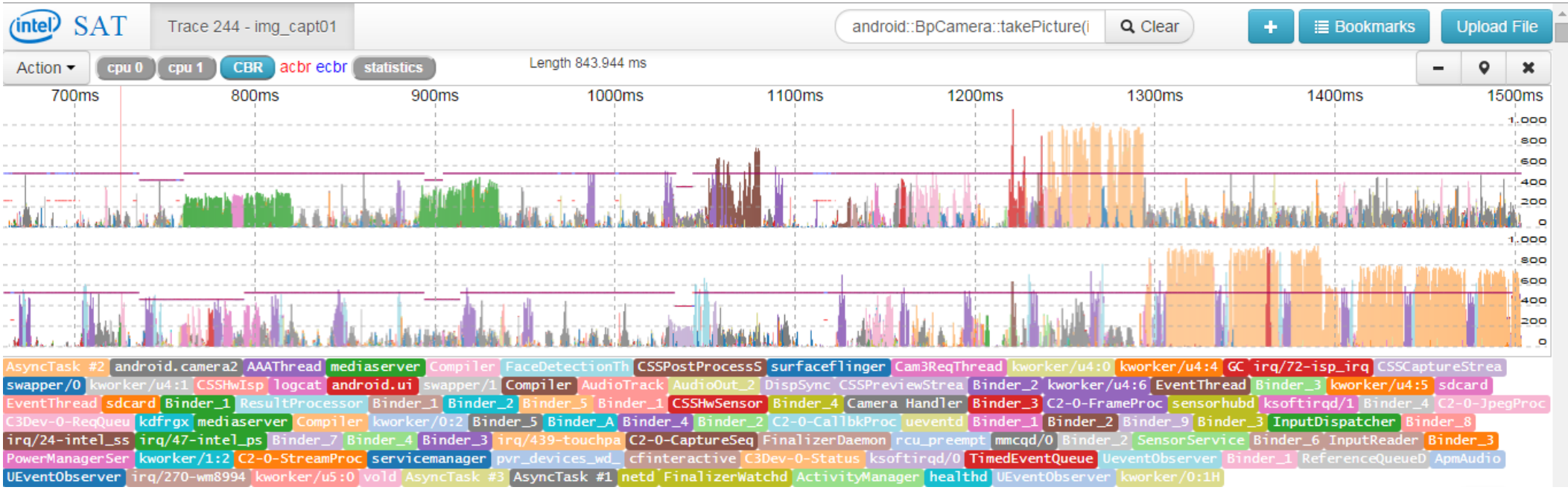


# Agenda

- Industry' desires
- Upcoming Approaches
  - Debugging
  - Trace
- Conclusions

# Trace

- Trace is one of the most valuable debug features
- Used for SW and HW investigations
- MIPI STP/ PTI widely used

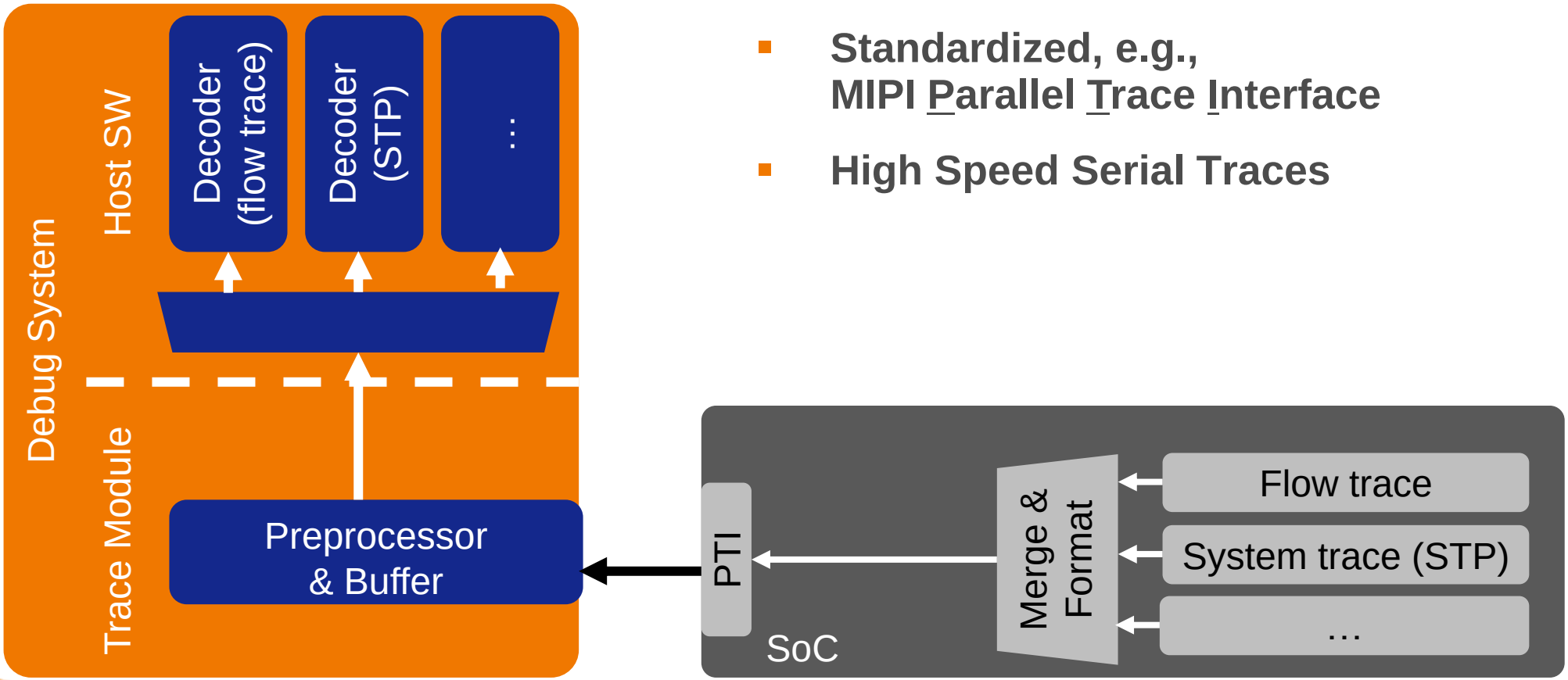





LAUTERBACH DEVELOPMENT TOOLS

# State of the Art: Dedicated Trace Ports

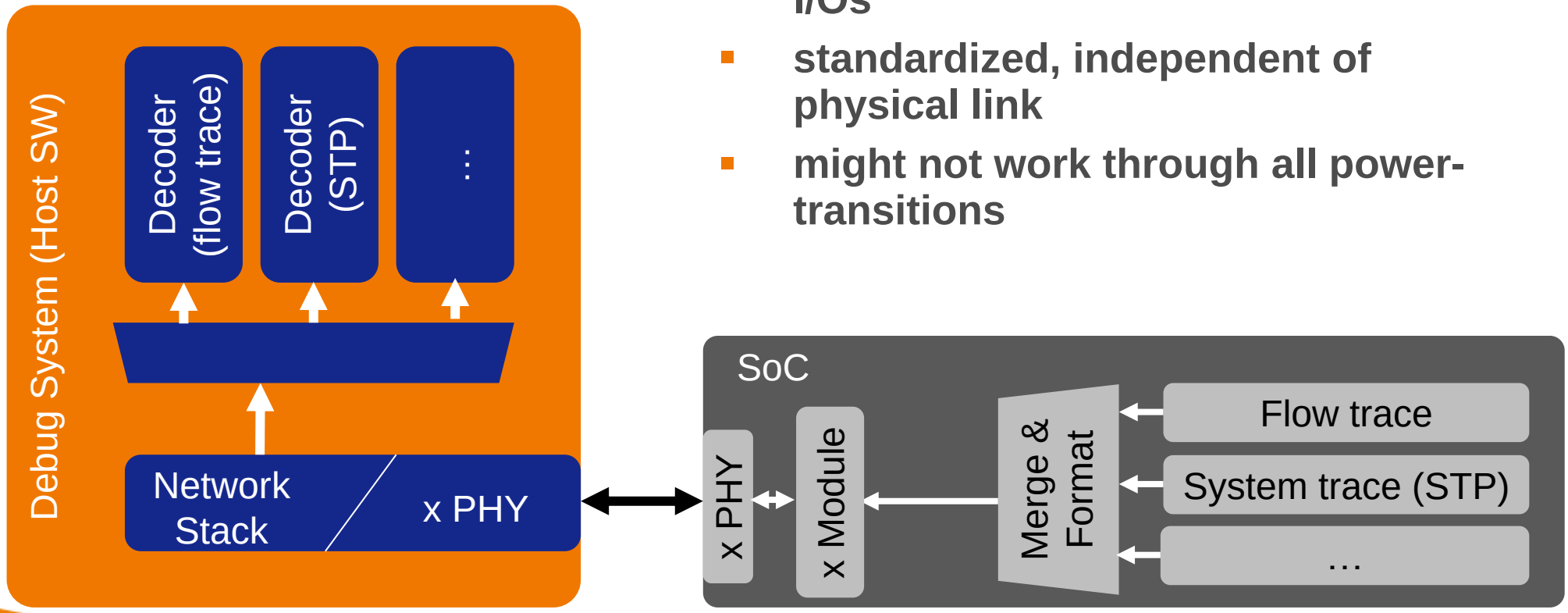
- **Real-time streaming**
- **Standardized, e.g., MIPI Parallel Trace Interface**
- **High Speed Serial Traces**




[MIPI Architecture Overview for Debug; www.mipi.org]

# MIPI Gigabit Trace

- no dedicated trace port
- can profit from high speed standard I/Os
- standardized, independent of physical link
- might not work through all power-transitions




[MIPI Architecture Overview for Debug; www.mipi.org]

# Conclusions

- Debugging of Virtual Prototypes allows pre-silicon
  - Verification of debug concepts
  - Software development
- Industry demands use of functional interfaces
  - Cost reduction
  - Full platform visibility in late development stages
  - Homogenous tool-chain
- Challenges
  - Very different properties of link types
  - Combination with many different core architectures
  - Stability and availability of link

▪ **Michael Eick (Lauterbach GmbH)** ▪
michael.eick@lauterbach.com
▪ **Rolf Kühnis (Intel Finland Oy )** ▪
rolf.kuhnis@intel.com

# Questions?



LAUTERBACH
DEVELOPMENT TOOLS