On Chip Debug - What Do We Have . . . What Do We Need

Dr. Neal Stollon
Neals@fs2.com

First Silicon Solutions Division
MIPS Technologies Inc.
www.fs2.com

Some quick definitions

Some Quick Definitions

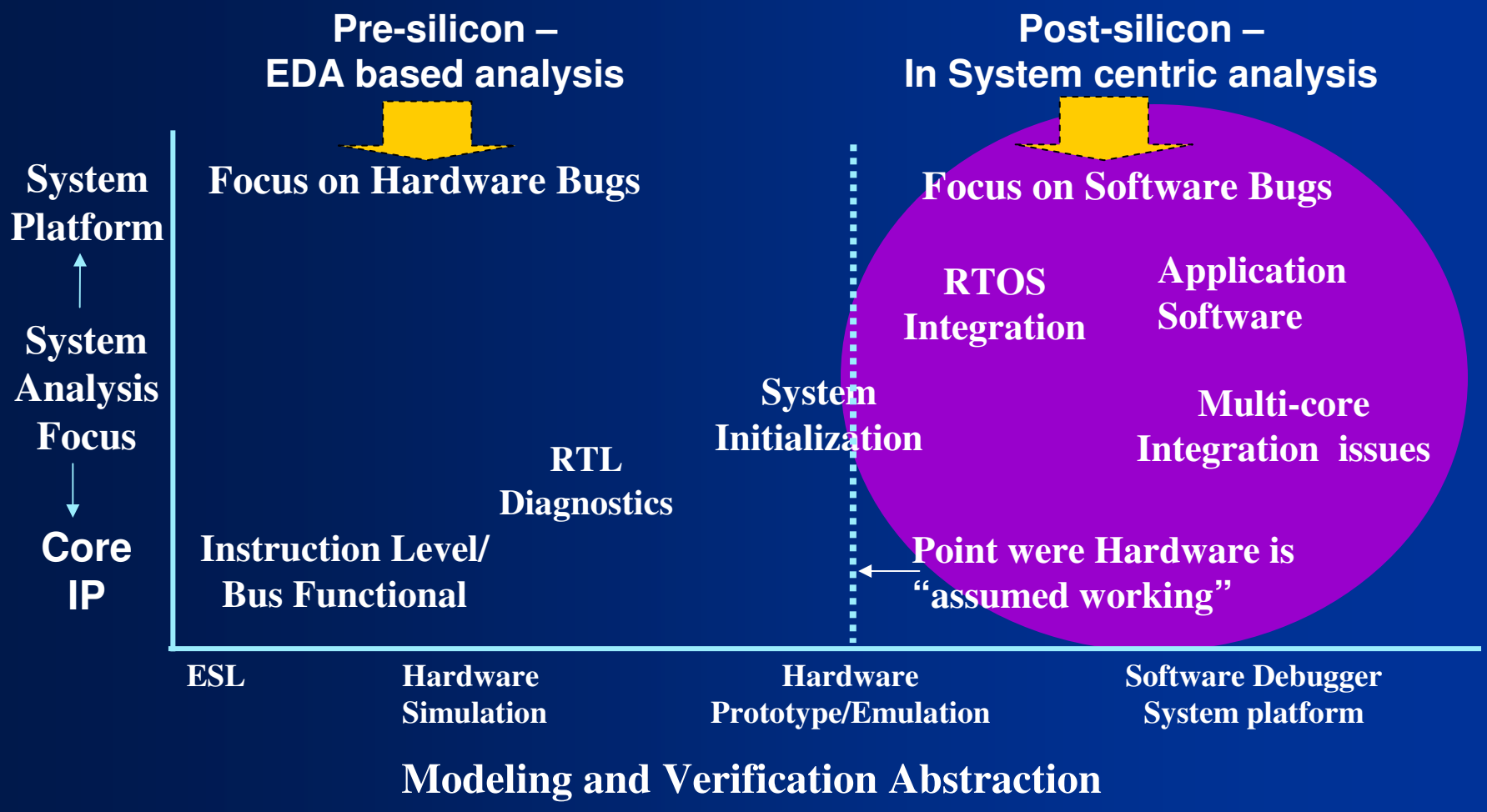
- Verification - process of analyzing, fixing errors, and optimizing a model or design
- Debug - process of analyzing, fixing errors, and optimizing a product (chip, board or box)
- EDA - Any tool used to manage the tedium and complexity of analysis and verification of a design.
- Instrumentation - any logic (on chip or off) used to manage the tedium and complexity of analysis and debug of a product

Verification and Debug are (should be) two sides of the same coin

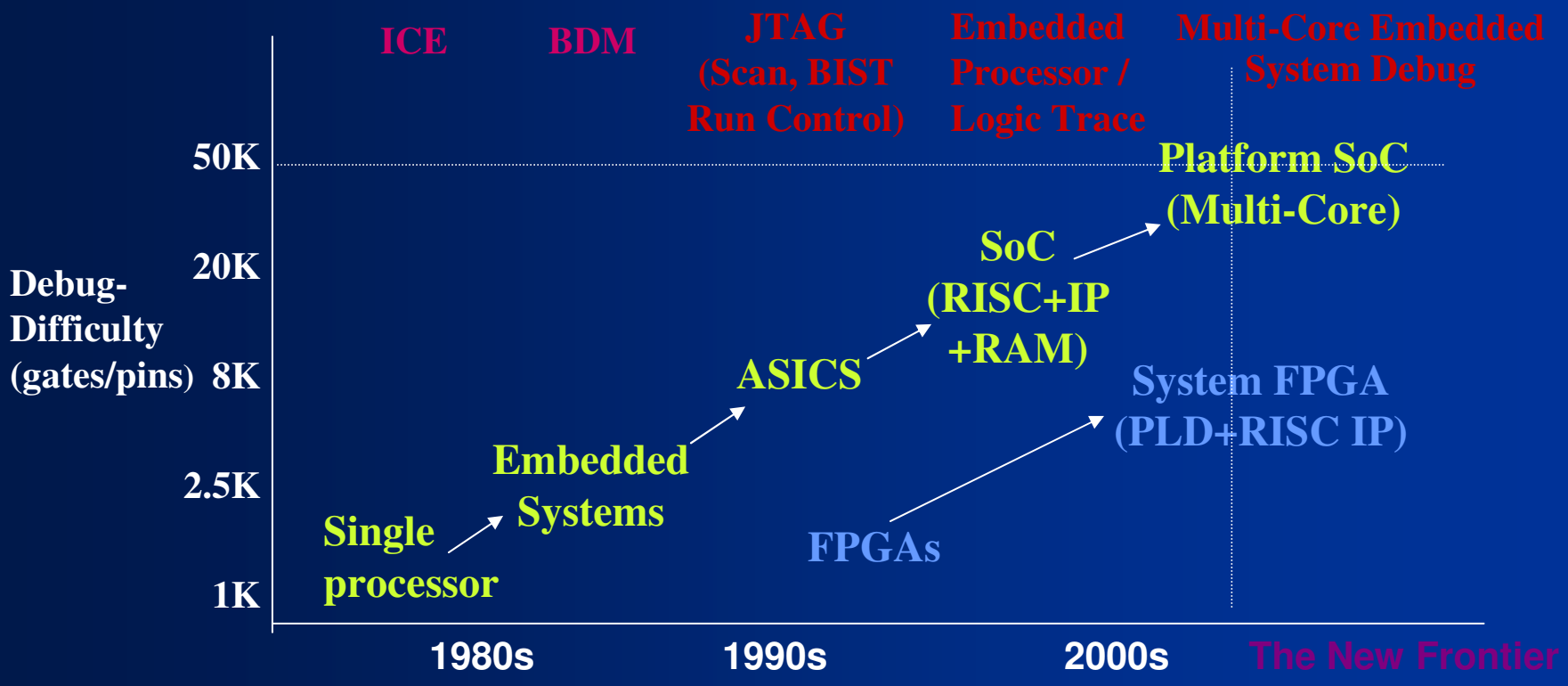
In ideal world, Verification and Debug should have seamless interface

- Reality is stitches are loose, broken, or missing
- So lets bridge the Verification / Debug gap

Verification vs. Debug Efforts



SoC Debug Evolution



Embedded Debug Complexity keeps increasing

- Gates increase geometrically - Pins increase linearly
- Significant debug difficulties for leading architectures
- More complex debug needs Instrumentation

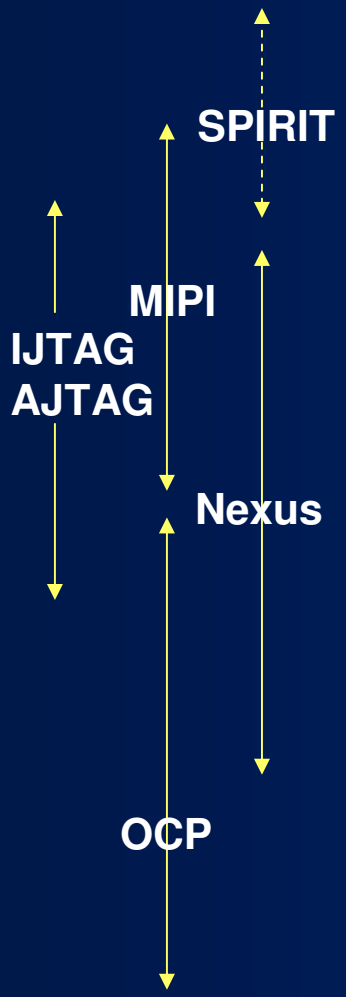
Key Emerging Debug Features

- Merging debug operations for several cores
  - Control/data transfer via a single JTAG compliant TAP
- Global control signals for multi-core cross triggering and synchronous actions (go, halt and breaks)
- Multi-core trace with timestamps
- Probe Hardware and tool API's to support multi-core trace
- Handle multiple instantiations of source level debuggers
- Customization to measure activity on buses, caches, execution cores, co-processors, interrupts, peripheral device events, . . .

Lessons Learned since 2002

- JTAG remains key –JTAG as the catchall debug interface is still strongly held and pervasive opinion
- Keep stuff small – major sensitivity to debug IP gate count for by many designers
- Cross Triggering is key – Important for debug of multiple cores
- Customers like standards…sometimes … still lots of proprietary buses, IP debug approaches, etc.
- Custom(er) is king – SoC architectures vary, therefore debug configurations and solutions will need to as well
- Work with your Eco-system - Debug does not stand-alone – close ties to EDA flow, IP, test, SW are needed …
- It is all about the tools – *Interfaces / SW tools are significant and ongoing effort to use new debug features*

The Debug Landscape

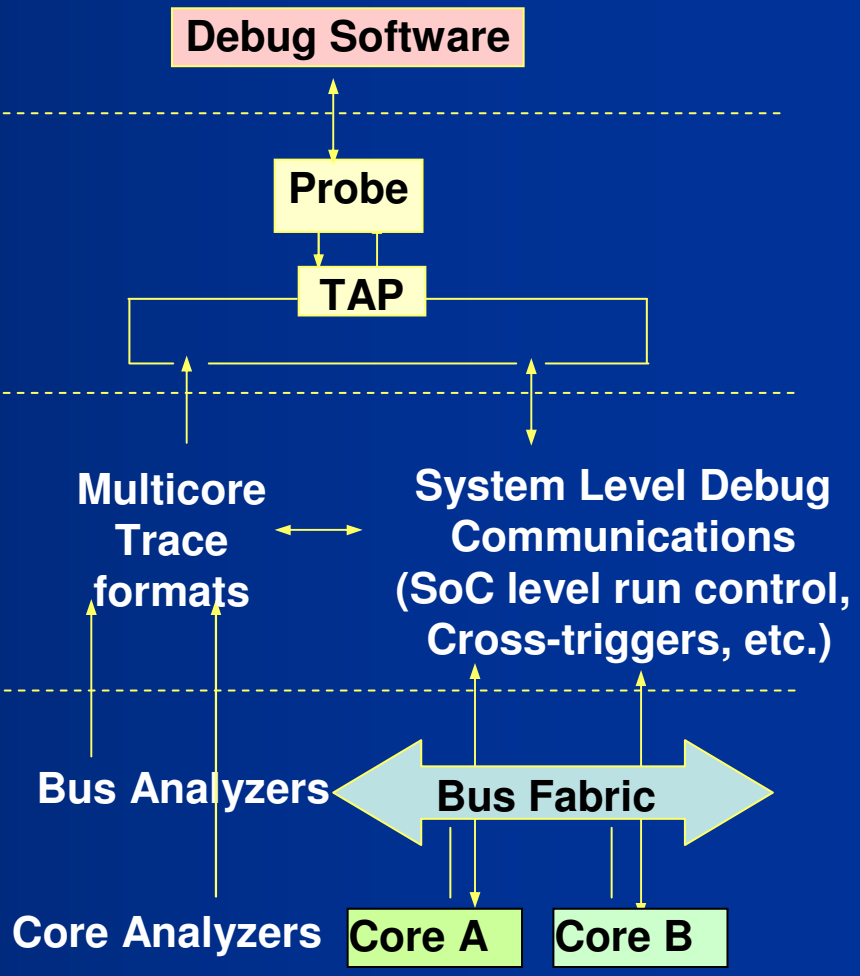


Tool Level of Integration
- SW/EDA Tool level APIs
- HW tools level APIs

IO Level of Integration
- JTAG – different flavors
- Nexus 5001
- MIPI NIDnT

Protocol Level of Integration
- Nexus 5001
- Multicore IP & probe APIs
- Other Proprietary

Instrument Level
Many vender/IP solutions
- MIPS EJTAG
- ARM ETM
- Bus Analyzer



What are the risks

- Lots of good essential work being done
  - Number of WGs has basically doubled every 2 years
  - 2 in 2002, 4 in 2004, 8 in 2006
- Lot of overlap efforts in focusing on advanced features
  - Next generation JTAG
  - SW interface and API development
  - High performance trace formats
  - Debug description approaches
- Contributions of more interested parties is needed
  - SoC Designers - Make sure solutions meet your needs
- We need to avoid standards turf wars on the horizon
  - Been there, done that, hopefully we are smarter

Summing it up

- Multi-core SoC requires new next wave in debug
  - Architectures, interfaces, tools
  - ASIC/EDA/systems players start to recognize the need and opportunities in Debug
- IC debug is still running on 15 year old JTAG concepts
  - Concurrent debug of multi-core SoC needs new concepts
  - Embedded instrumentation is one next logical step
- Debug fills a hole in SoC design/verification flow
  - Not being addressed by ASIC, IP, or EDA community
  - Not just debug - multi-core optimization, performance analysis, etc.

Investing gates to get more design visibility, better debug
= proven faster time to market