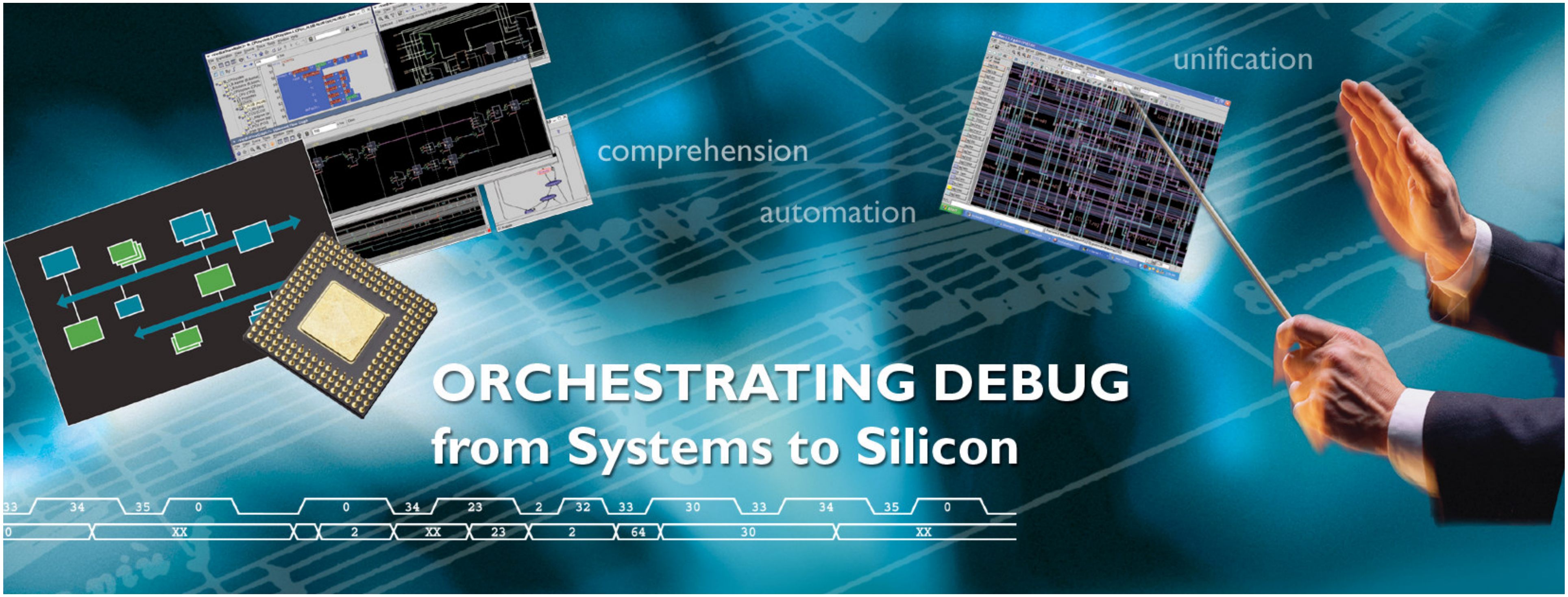


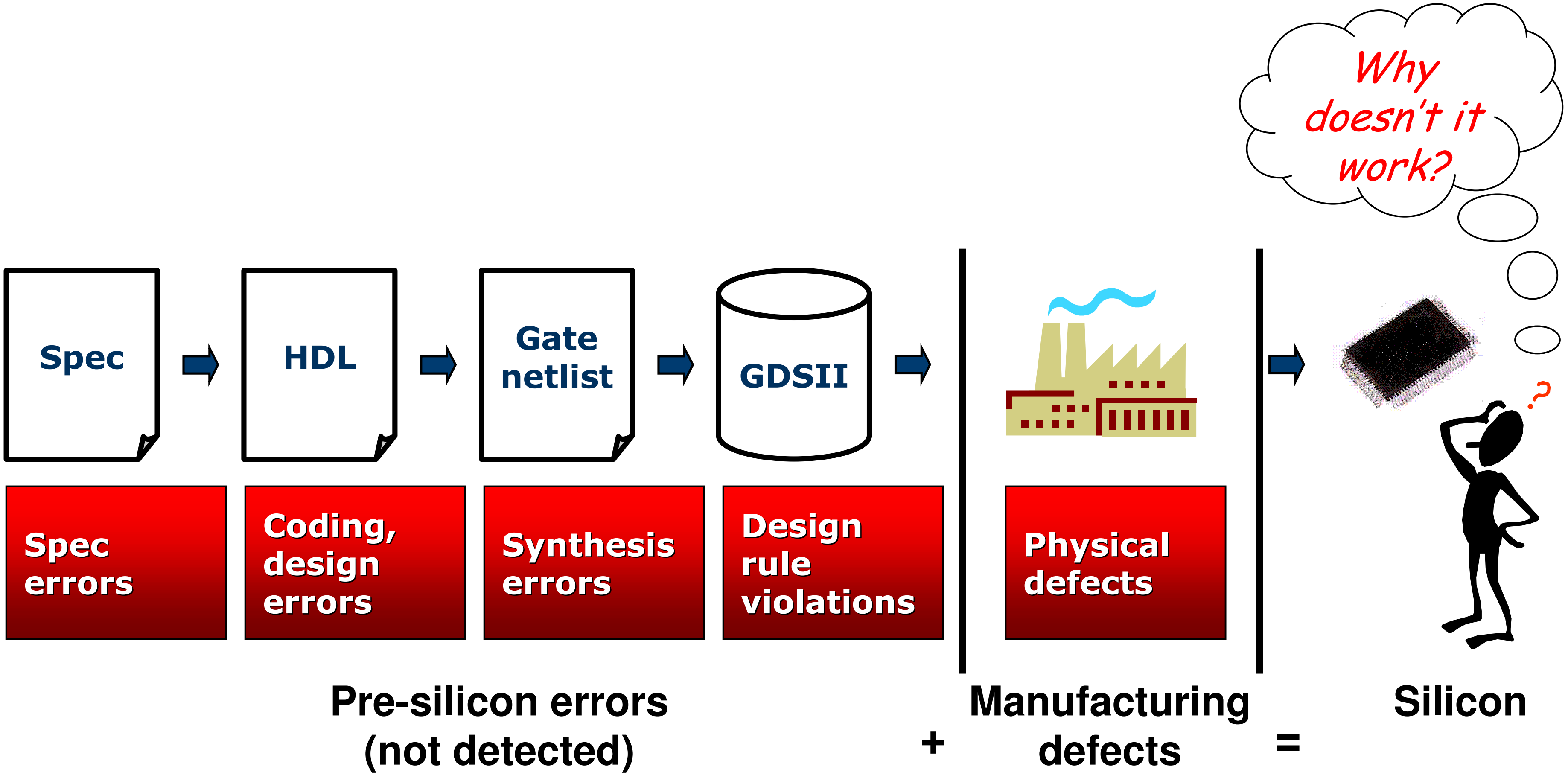
Efficient Debugging of Silicon Prototypes

Electronics Design Process Symposium
2005

Agenda

- Overview of silicon prototype debug and diagnosis
- Economic issues of silicon debugging
- Silicon testing and debug issues
- Overview of current methods for silicon debugging
- New methods for silicon debug
- Experimental results
- Summary

What Do We Mean by Silicon Debug?



Spec → HDL → Gate netlist → GDSII →

Spec errors

Coding, design errors

Synthesis errors

Design rule violations

Physical defects

Pre-silicon errors (not detected) + Manufacturing defects = Silicon

Silicon Debug or Diagnosis?

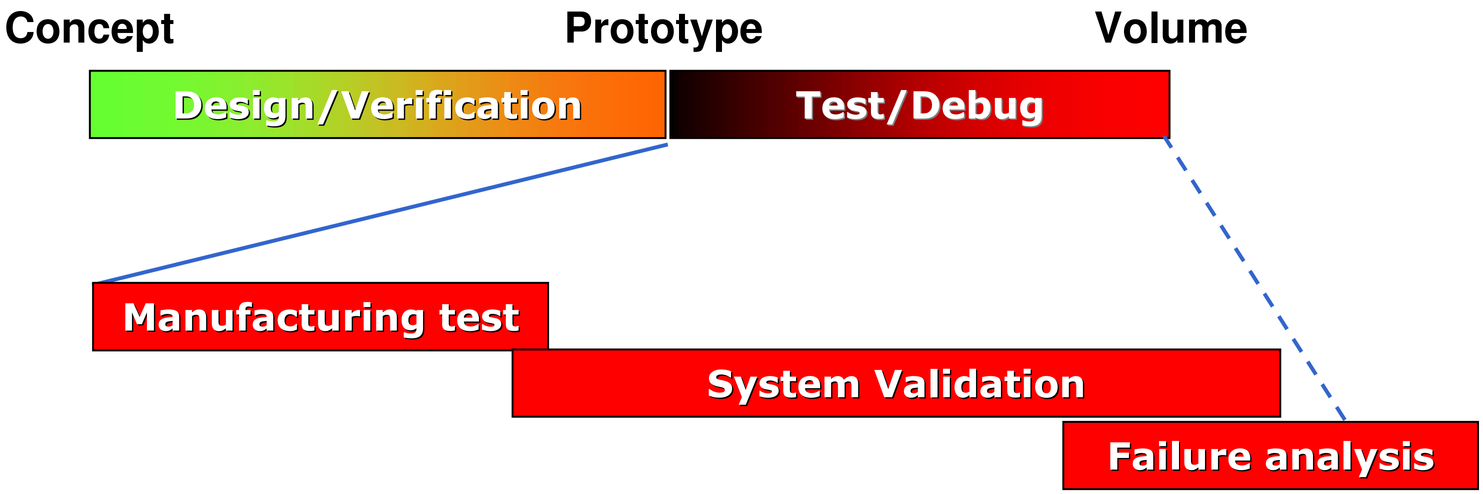
IEEE TTTC “Silicon Debug and Diagnosis” definitions:

Silicon debug = Finding/locating/identifying design errors on silicon

Silicon diagnosis = Finding/locating/identifying manufacturing defects or faults

Silicon debug and diagnosis performed when first silicon fails

Silicon Prototype Test and Debug Activity



Concept

Prototype

Volume

Design/Verification

Test/Debug

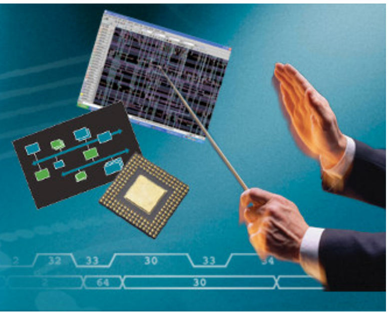
Manufacturing test

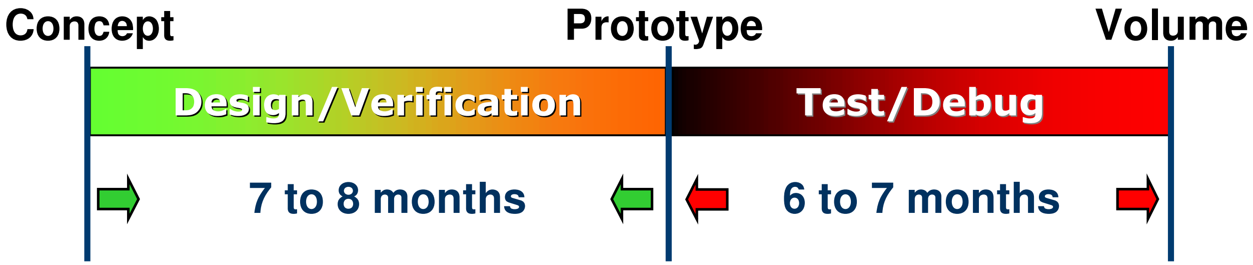
System Validation

Failure analysis

- **Manufacturing test** – *screen out physically defective parts*
- **System validation** - *verify silicon prototype works correctly in situ*
- **Failure analysis** – *find systemic defects to improve yield*

Economic Case for Quick Prototype-to-Volume

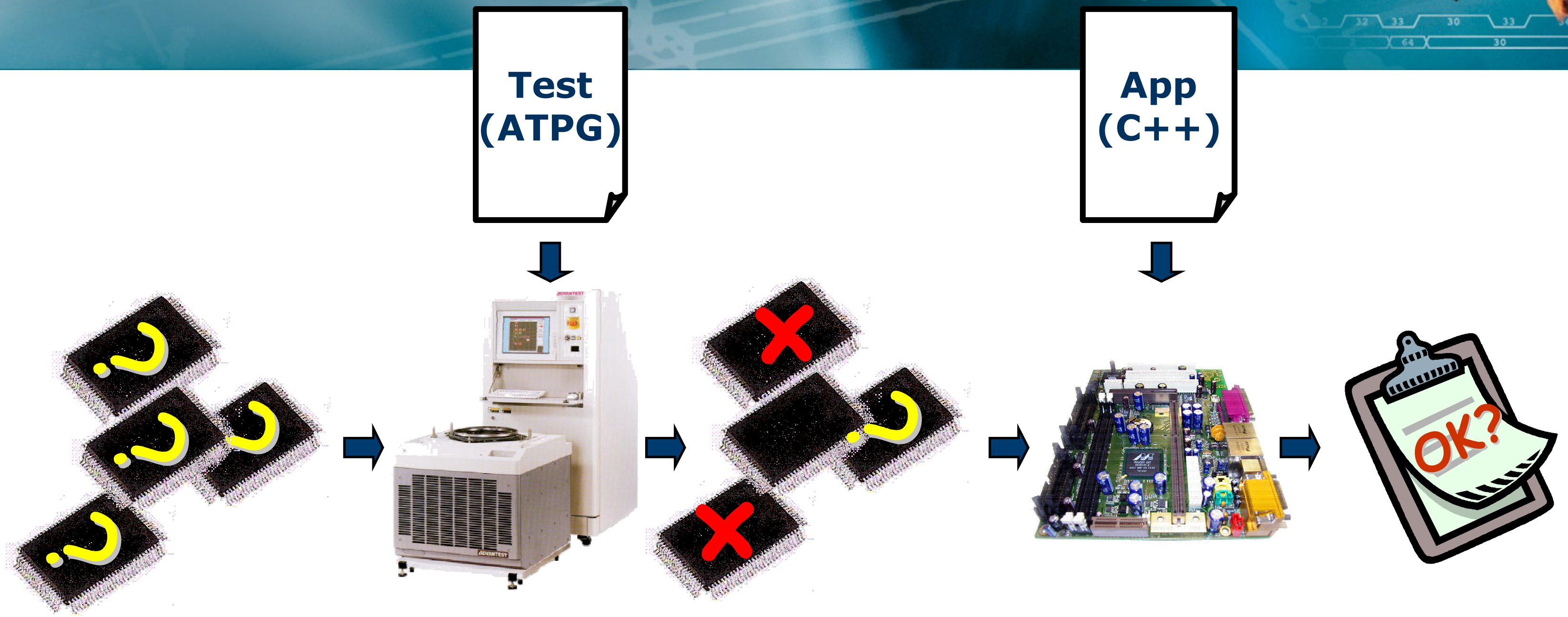




Prototype to volume is increasing!

- **On-time delivery is largest single factor for profitability** (Source: McKinsey)
- **Missed market opportunities worth millions $$**
- **Multiple spins are expensive: mask costs pushing $1 million**
- **Silicon test/debug utilizes personnel resources, which are better deployed on new projects**

Silicon Testing Issues

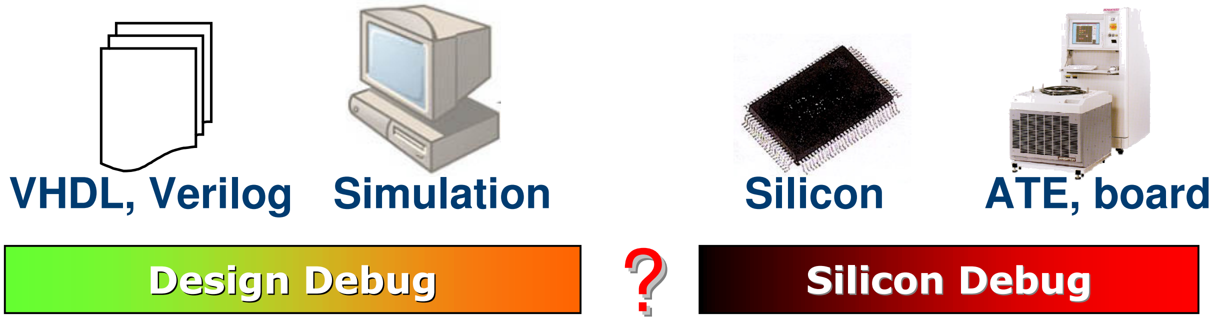


Test (ATPG)

App (C++)

- **Many defects are difficult to model and escape detection by testers**
- **Structured tests (ATPG, BIST, etc.) do not replicate real-word conditions, resulting in “over-testing” and “under-testing”, examples:**
  - Power consumption during for 65nm as high as 30x, creating data-corrupting voltage drops – change in V_{TH} (chip rejected)
  - Crosstalk effects are pattern sensitive and not targeted (chip passed)
- **More transistors increases time required to isolate causes of errors**

Silicon Debug – Data/Environmental Issues



VHDL, Verilog

Simulation

Silicon

ATE, board

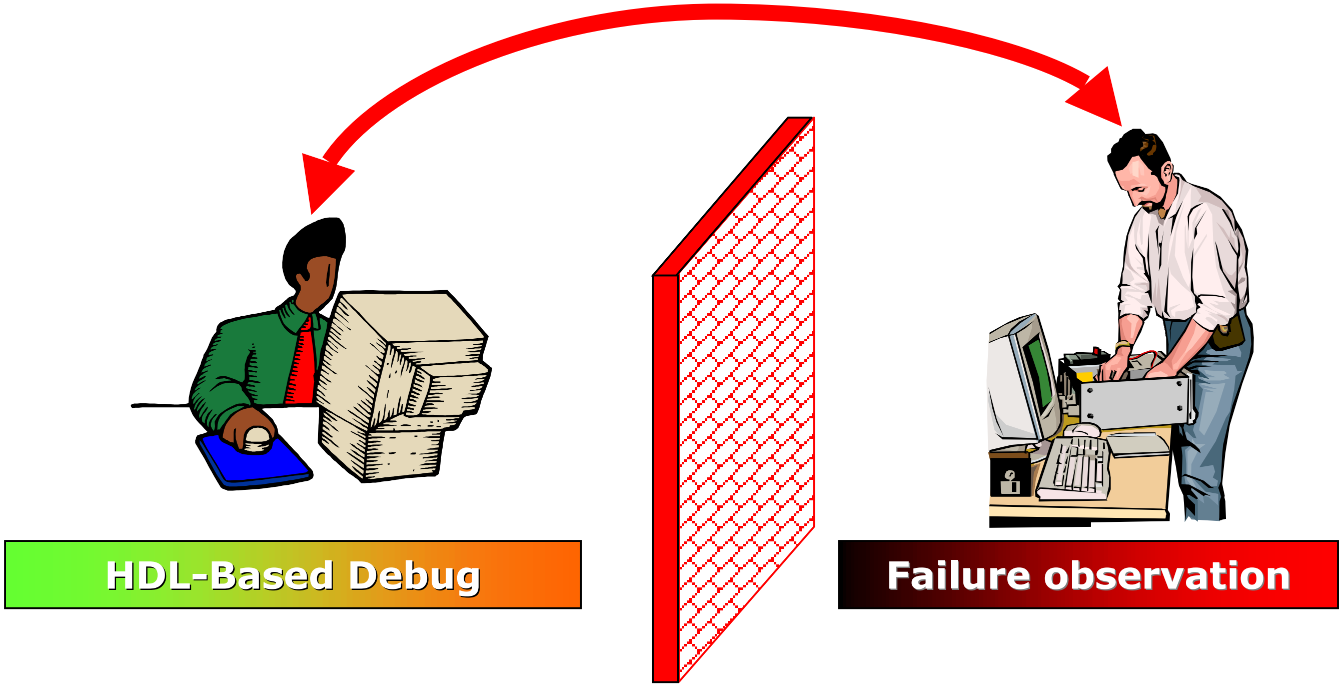
Design Debug

?

Silicon Debug

- Limited silicon signal data prevents understanding of behavior
- Silicon errors may either be functional bugs or physical defects
- Silicon’s polygonal structure hinders functional understanding
- Silicon environment (file formats, tools, etc.) is different from design

Silicon Debug – Dependence on Designers

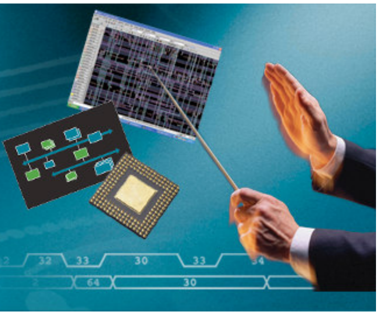


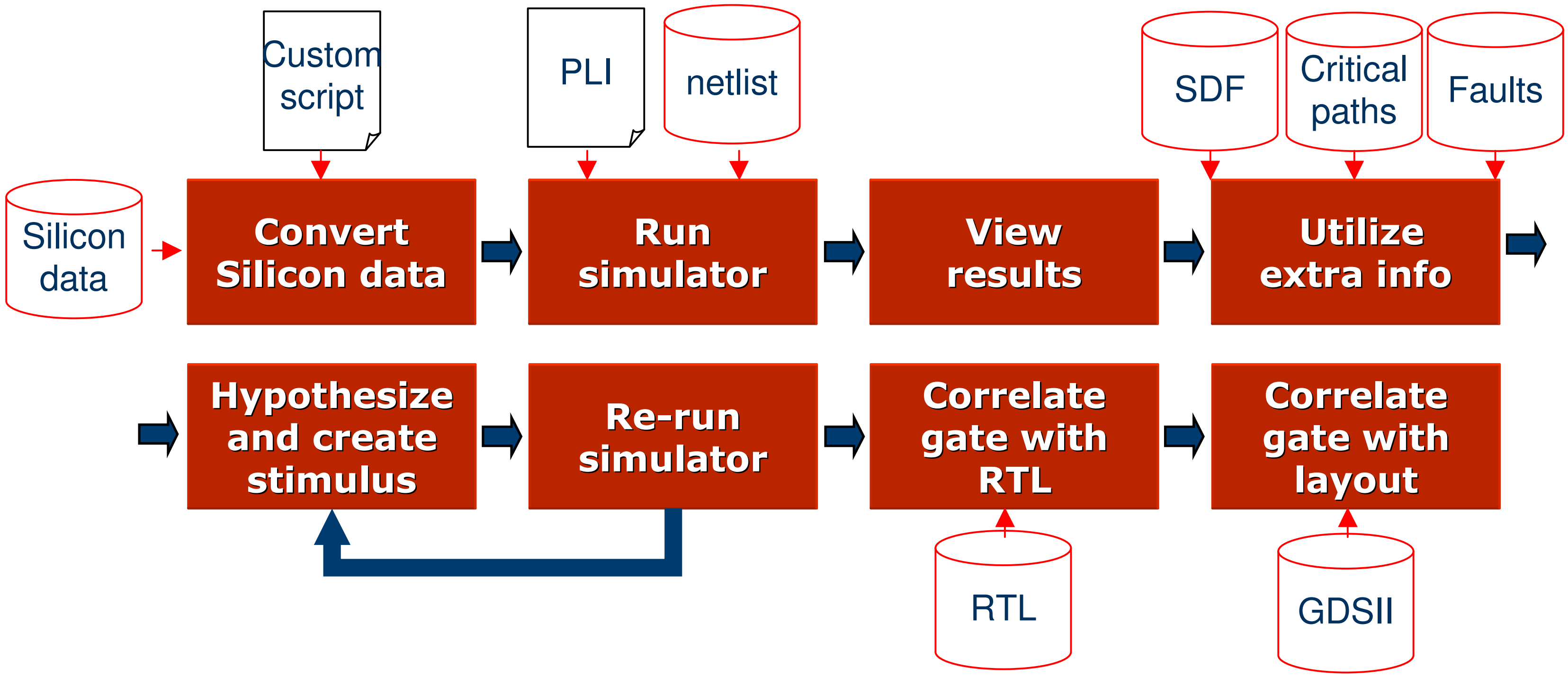
HDL-Based Debug

Failure observation

- Designers have function domain expertise
- Designers assist system validation engineers when functional error suspected and isolated to single silicon device
- Difficult to debug due to lack of post-silicon focused debug tools and methodology; designers rely on pre-silicon tools

Today’s Ad-Hoc Silicon Debug Method





Custom script

PLI

netlist

SDF

Critical paths

Faults

Silicon data

Convert Silicon data

Run simulator

View results

Utilize extra info

Hypothesize and create stimulus

Re-run simulator

Correlate gate with RTL

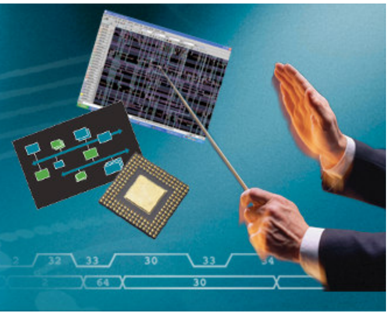
Correlate gate with layout

RTL

GDSII

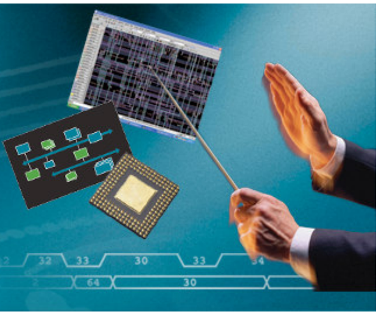
- Much effort to obtain little insight
- Many interfaces, tools, databases, and steps increases chance of error
- Little or no automation of these steps

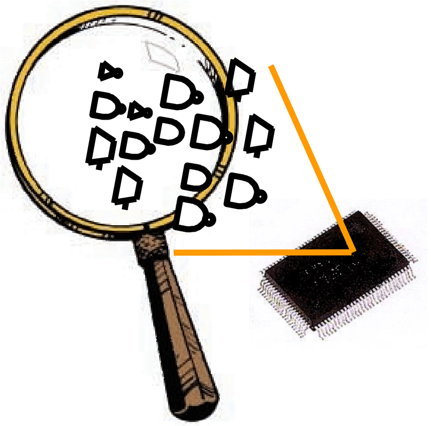
Proposal to Improve Silicon Debug for System Validation



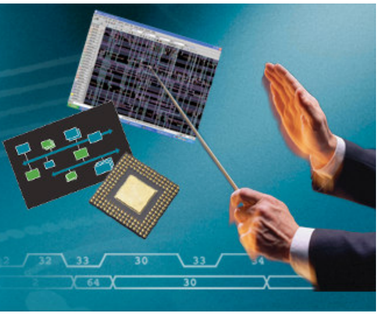
- Insert on-chip hardware to access “essential” signals
- Operate on-chip hardware orthogonal to system mode operation
- Process data onto HDL-oriented debug environment

On-Chip Hardware - Design-for-Debug



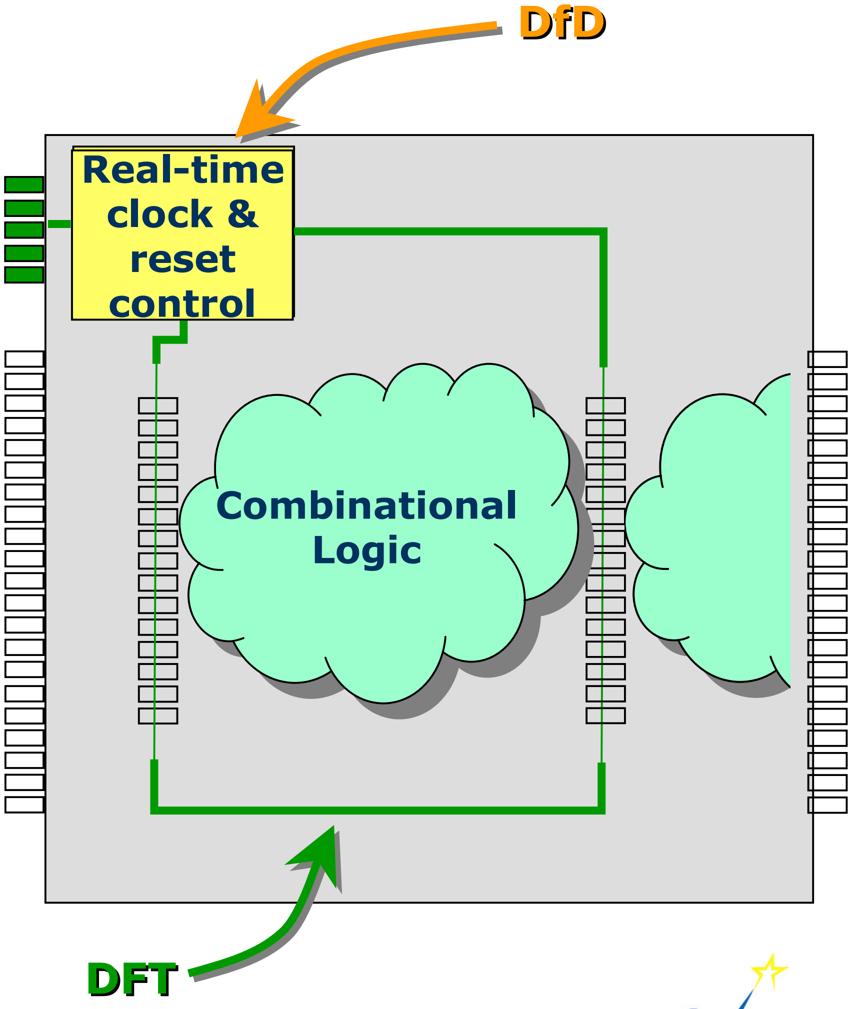


- Debugging silicon in situ easier when designed with “visibility” into internal nodes
- Logic which brings out data from the silicon while in the system is known as “design-for-debug” (DfD) logic
- Most DfD today is proprietary, although several vendors offer DfD IP
- DfD utilized when failures occur during system validation

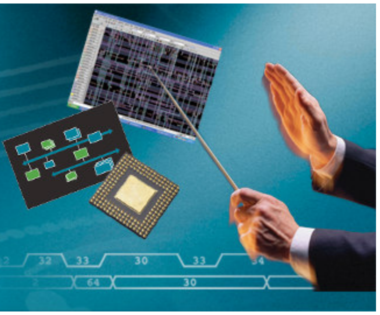


Design-for-Debug (DfD) on the Chip

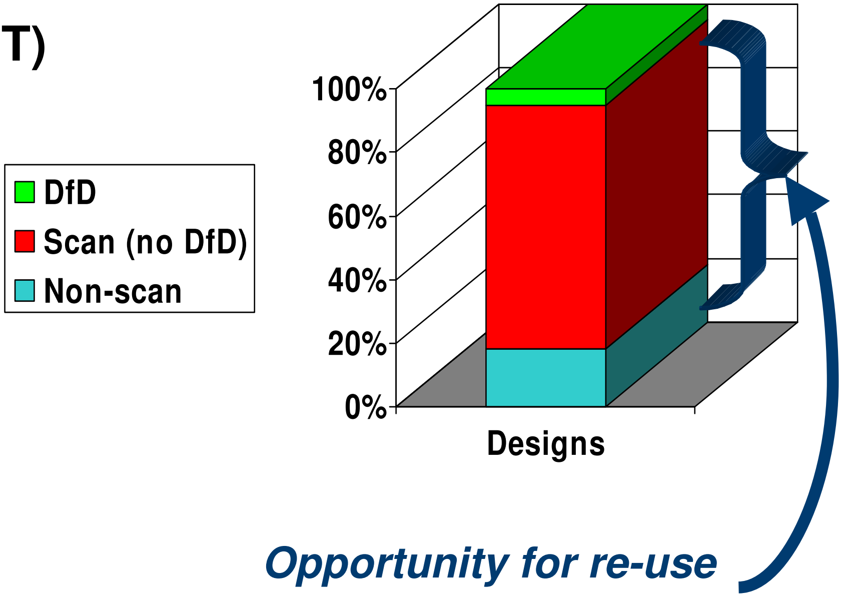
- **Most implementations leverage design-for-test (DFT):**
  - IEEE 1149.1 controller
  - Internal scan chains
- **DfD sophistication varies**
  - Simple – in situ shift out of scan data
  - Complex – breakpoints, assertions, debug busses
- **DfD must be planned and implemented as part of standard design flow prior to tape-out**



Opportunity for DfD

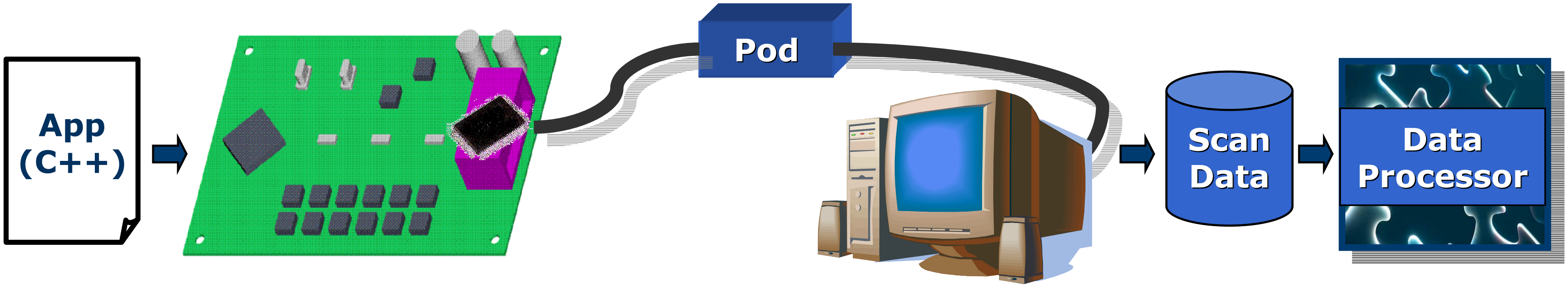


- >80% of designs contain scan chains (DFT)
- Only 5% of designs have design-for-debug (DfD) now
- DfD builds on DFT and adds as little as 0.3% area for 8 million gate design



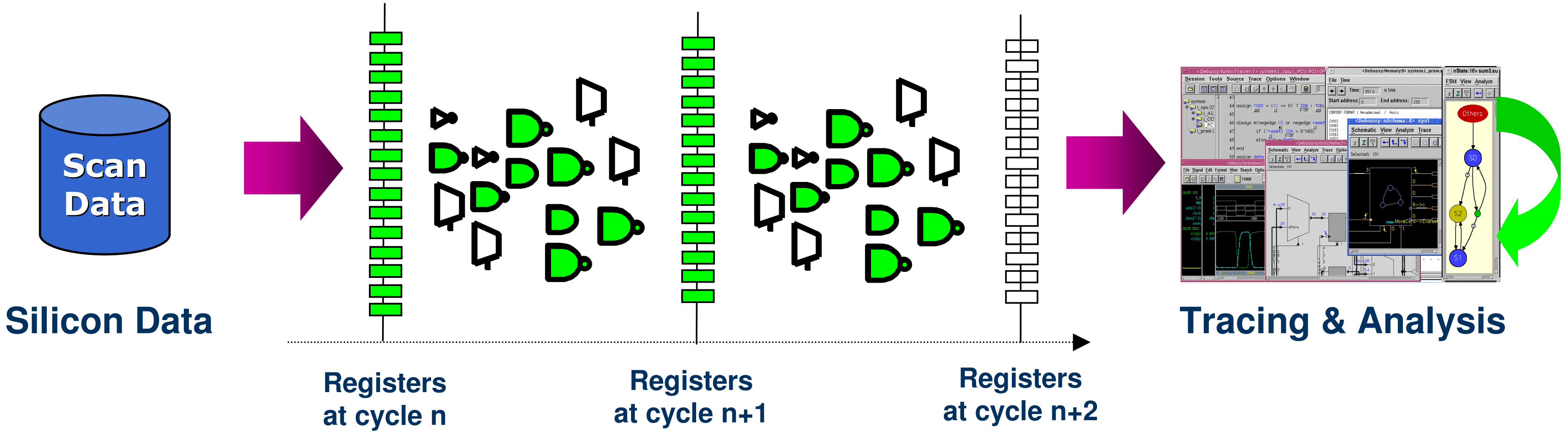
Opportunity for re-use DFT for DfD

System Validation, DfD, and Real-Time Access



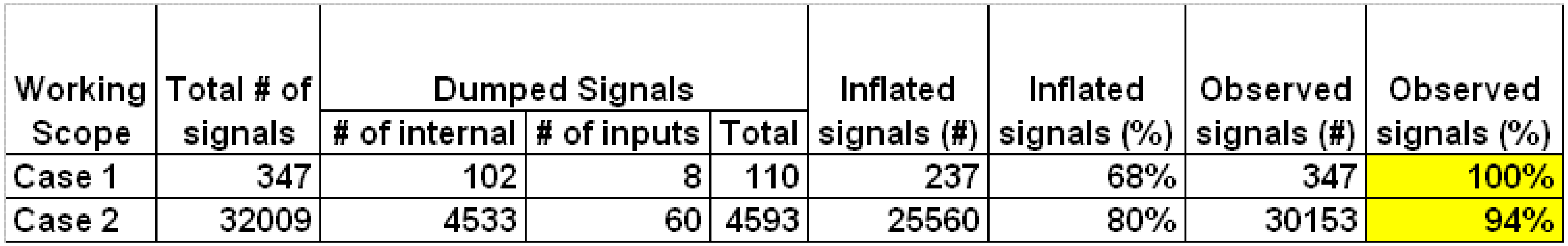
- On-chip DfD typically hooks up to a pod and is activated by a separate software-based control program
- The pod is a device that makes the electrical characteristics between the DfD port and the PC port compatible
- The DfD control program accesses data such as the internal registers
- This data must then be processed for debug tools

Data Inflation



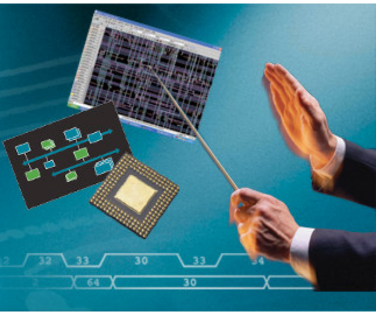
- **Data inflation engine makes limited data more useful**
  - Convert limited scan register data to namespace, time
  - Inflate (compute) unobserved nodes
  - Elegantly deal with missing state data
- **View and trace activity using advanced HDL debugging techniques**
- **HDL-oriented debug of silicon quickly leverages designers' expertise**

Experimental Results



Working Scope	Total # of signals	Dumped Signals			Inflated signals (#)	Inflated signals (%)	Observed signals (#)	Observed signals (%)
		# of internal	# of inputs	Total				
Case 1	347	102	8	110	237	68%	347	100%
Case 2	32009	4533	60	4593	25560	80%	30153	94%

- Two cases of actual silicon signal dumps
- By dumping 14% of signals, 94% of device signals became visible (case #2)
- DfD significantly increases observability
- Increased observability has huge potential to decrease silicon debug time



Summary of Silicon Prototype Debugging

- Silicon debug/diagnosis is a bottleneck to time-to-volume
- Existing silicon debug flows are ad-hoc, inefficient
- Silicon debug flows must better connect the system validation engineers and designers
- Emerging DfD techniques combined with new technologies will enable more efficient debug of silicon prototypes