

COEN-4730 Computer Architecture
Lecture 13

Testing and Design for Testability (focus: Processors)

Cris Ababei
Dept. of Electrical and Computer Engineering
Marquette University

1

1

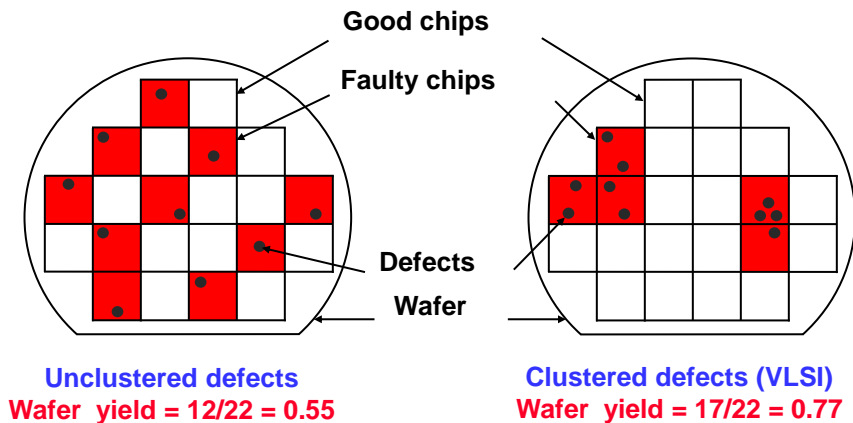
Outline

- **Testing**
- **Design for Testability (DFT)**
- **Microprocessors**

2

2

Quality of VLSI Circuits

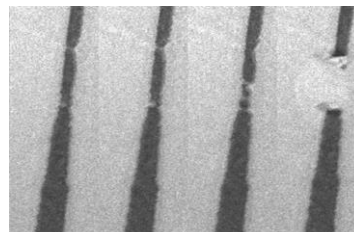
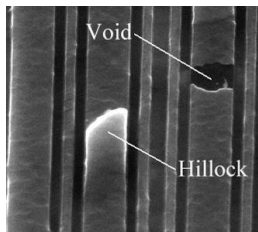


3

3

What is Testing?

- Testing
 - Experiment to detect if the operation of the fabricated physical circuit is affected by manufacturing defects
- Vs. Verification
 - Predictive analysis to ensure correctness of the synthesized circuit; when manufactured, the circuit will perform the given I/O function



4

4

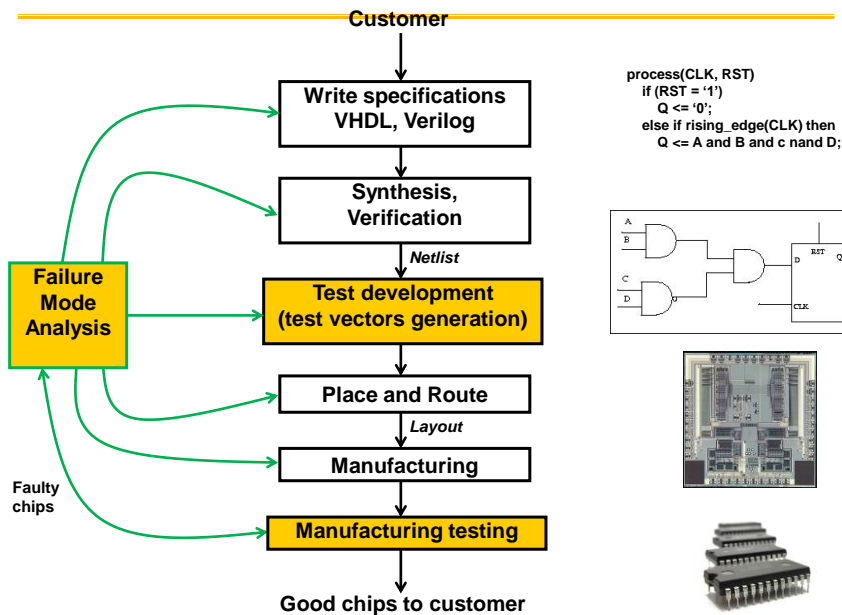
Why do we do Testing? – Roles of Testing

- **Detection:** Determination whether or not the **device under test (DUT)** has some fault.
- **Diagnosis:** Identification of a specific fault that is present on DUT.
- **Device characterization:** Determination and correction of errors in design and/or test procedure.
- **Failure Mode Analysis (FMA):** Determination of manufacturing process errors that may have caused defects on the DUT.

5

5

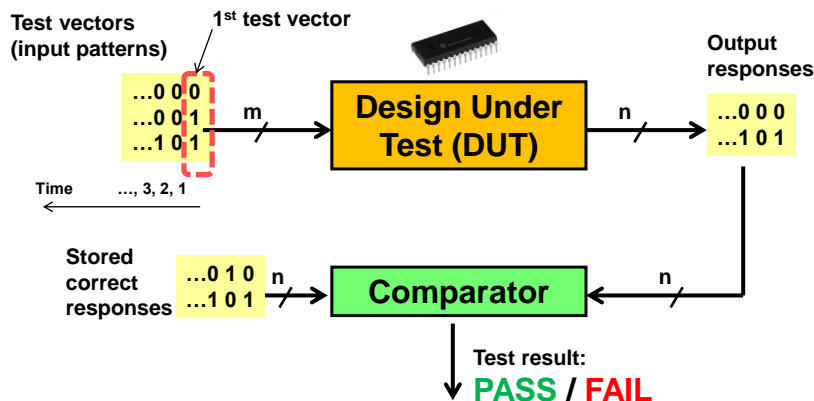
Typical Design Flow of VLSI Circuits



6

6

How do we do Testing? – Principle of Testing



7

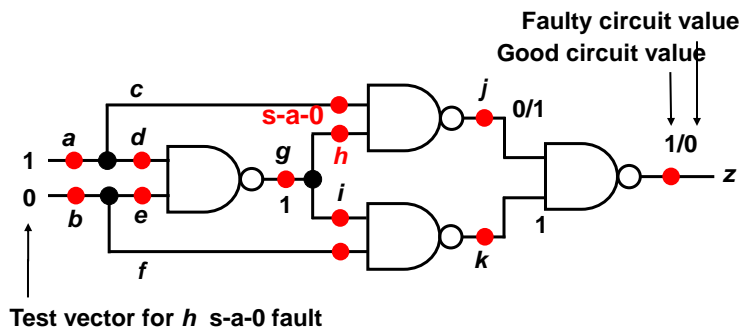
Defect, Fault, Error

- **Defect:** in an electronic system, is the unintended difference between the implemented hardware and its intended design
- **Fault:** a representation of a defect at the abstracted functional level
- **Error:** a wrong output signal produced by a defective system; is an effect whose cause is some defect

8

Single Stuck-at Fault Model

- Three properties define a single stuck-at fault
 - » Only one line is faulty
 - » The faulty line is permanently set to 0 or 1
 - » The fault can be at an input or output of a gate
- Example: XOR circuit has 12 fault sites (●) and 24 single stuck-at faults



9

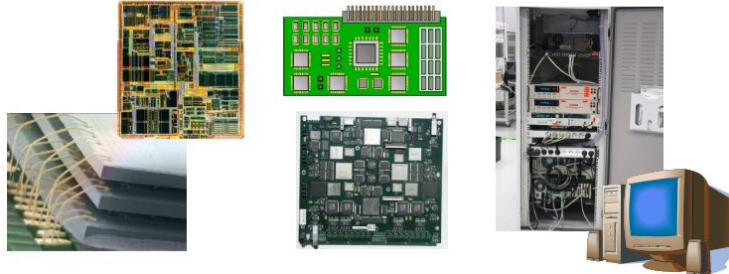
Single Stuck-at Fault Model

- How effective is this model?
 - Empirical evidence supports the use of this model (justified by the *frequent testing strategy*)
 - Has been found to be effective to detect other types of faults
 - Easy to use

10

Levels of Testing

Level		
Chip	Board	Rack
SoC, SiP, μ P, DSP, NoC, etc.	Motherboard, PCI card, etc.	PC, server, etc.



- **Rule of 10x:** It costs 10 times more to test a device as we move to higher levels

11

11

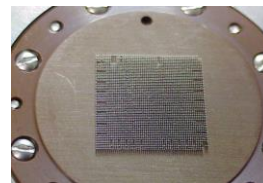
Automatic Test Equipment (ATE)



Production IC Probe floor



Cantilever Needle Card



Vertical card for area arrays
(3500 needles)

12

12

Memories

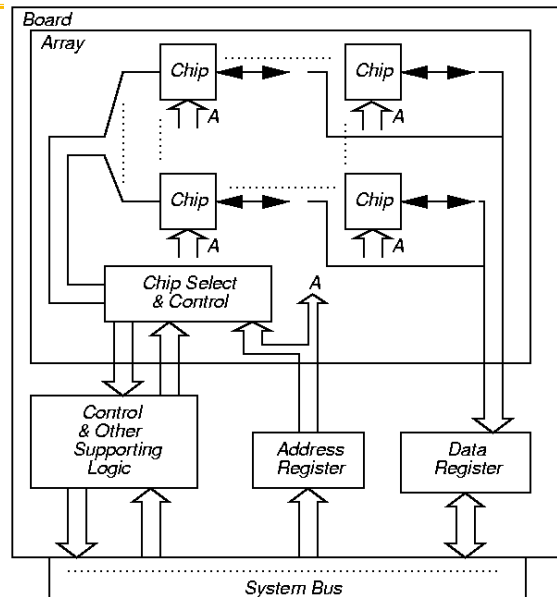
- Semiconductor memories are more than 35% of the entire semiconductor market
- **Memories are the most numerous IPs used in SoC designs**
- Number of bits per chip continues to increase exponentially and fault sensitivity increases; faults become more complex
- Less charge stored per memory cell, cells are smaller and closer → cell coupling faults
- Traditional tests require long test-times, which increases a lot with the increase of the memory size
- Test cost per memory chip must not increase significantly

13

13

Memory Testing

- Levels:
 - Chip, Array, Board
- **March tests:**
 - Family of tests called “marches”
- Neighborhood tests



14

14

Outline

- Testing
- **Design for Testability (DFT)**
- Microprocessors

15

15

Motivation for Design For Testability (DFT)

- Design techniques to add testability features to a design
 - Testing activities moved to on-chip and on-board!
- These added features make it easier to develop and apply manufacturing tests
- Goal: improve **controllability** and/or **observability** of internal nodes of a chip or PCB
- Benefits:
 - Reduced ATE cost due to self-test, decreased test times, inexpensive alternatives to burn-in test, reduced field repair cost, high quality of products delivered to customer
- Costs:
 - Reduced yield due to area overhead, increased power dissipation

16

16

Benefit/Cost of DFT

- If we consider life-cycle cost → DFT on chip lowers the costs at board and system levels!

Level	Design & manuf.			Field		
	Design and Test	Fabrication	Manuf. test	Diagnosis and repair	Maintenance test	Service interruption
Chip	+/-	+	-			
Board	+/-	+	-	-		
System (rack)	+/-	+	-	-	-	-

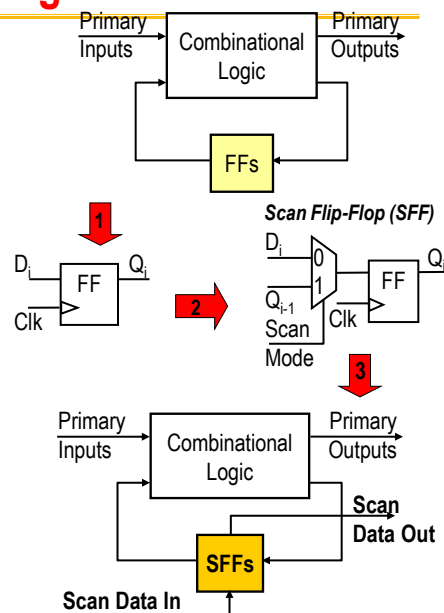
LEGEND: + Cost increase
 - Cost saving (i.e., reduction)
 +/- Cost increase may balance cost reduction

17

17

DFT #1 - Scan Path Design

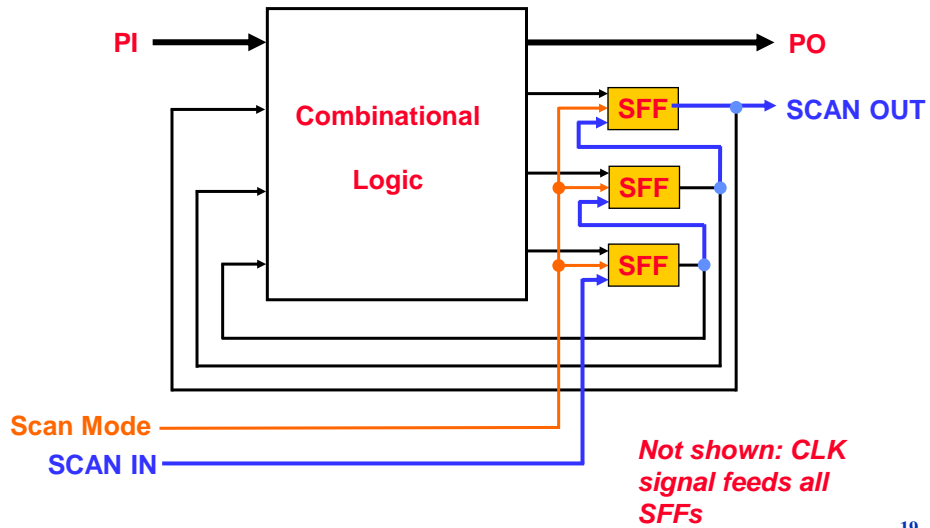
- Scan design
 - Replace all selected storage elements with scan cells
 - Connect scan cells into scan chains (shift register)
 - Scan mode facilitates
 - Shifting in test vectors
 - Shifting out responses
- Good CAD tool support
 - Transforming flip-flops to shift register
 - ATPG



18

18

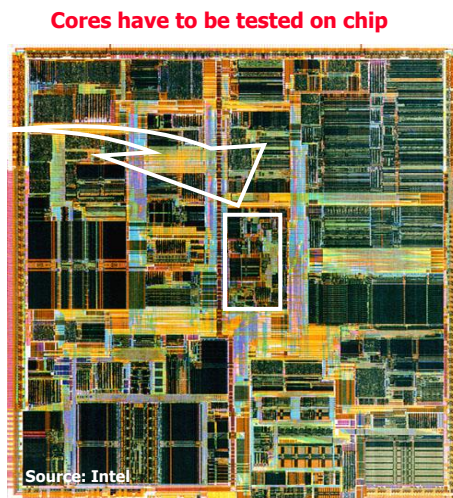
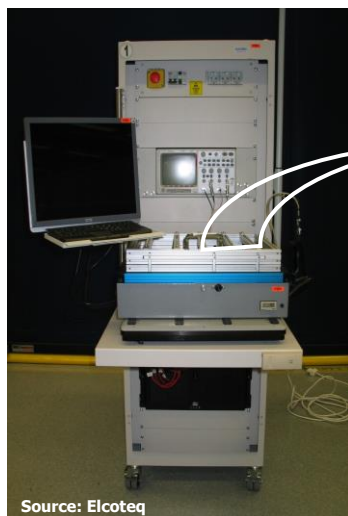
Scan Path Design



19

19

DFT #2 - Built-In Self-Test (BIST)



20

20

Built-In Self-Test (BIST)

- Motivation for BIST:

- Need for a cost-efficient testing (general motivation)
- Increasing difficulties with TPG (Test Pattern Generation)
- Growing volume of test pattern data
- Cost of ATE (Automatic Test Equipment)
- Test application time
- Gap between tester and DUT (Design Under Test) speeds

- Drawbacks of BIST:

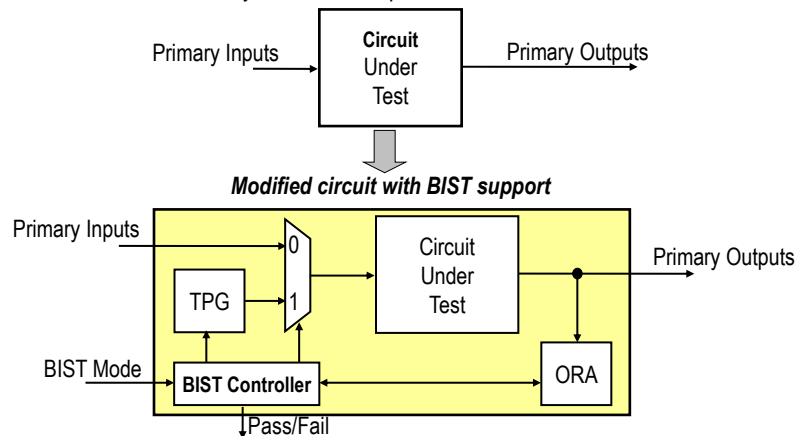
- Additional pins and silicon area needed
- Decreased reliability due to increased silicon area
- Performance impact due to additional circuitry
- Additional design time and cost

21

21

Built-In Self-Test (BIST)

- Incorporates test pattern generator (TPG) and output response analyzer (ORA) internal to design
 - Chip can test itself!
- Can be used at all levels of testing
 - Device → PCB → system → field operation

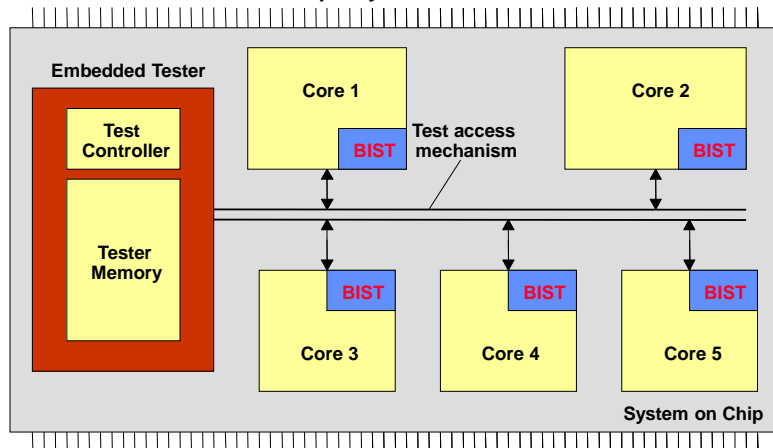


22

22

System-on-Chip (SoC) BIST

- testing time ↓
- memory cost ↓
- power consumption ↓
- hardware cost ↓
- test quality ↑



23

23

Outline

- Testing
- Design for Testability (DFT)
- **Microprocessors**

24

24

Challenges in Microprocessor Testing

- Today's microprocessors consist of billions of transistors operating at extraordinarily high speeds
- Large number of registers
- Large number of small buffers or queues
- Different sizes of memories
- Complex random logic (control path & datapath)
- Board level testing
- Test integration & scheduling
- CAD tool support

25

25

Part1: Systems with Microprocessors

- A system is an organization of components (hardware/software parts and subsystems) with capability to perform useful functions
- Systems with a microprocessor can use it to implement testing strategies for the whole system. The uP can self-test itself too

26

26

Part2: Microprocessors Testing

- **Structural testing**

- Faults defined in conjunction with a structural model: **structural fault models**. Main types of structural faults are shorts and opens and they are mapped into **stuck-at** and **bridging** faults
- Test generation methods are based on the structural model of a system under test: produce tests for structural faults. Examples: PODEM, CONTEST, etc.
- Test generation difficulty increases with the increase of processor complexity. Addressed partially by DFT techniques.

- **Functional testing**

- **Functional fault model** at Register Transfer Level (RTL): represent the effect of physical faults on the operation of a functionally modeled system. Example: addressing fault affecting register-decoding
- Difficult to automate

27

27

Functional Testing

- Functional testing reduces the complexity of the test generation problem by approaching it at higher levels of abstraction → **higher efficiency in test time**
- The process of test generation is **difficult to automate**. It is often a manual process – time consuming, prone to errors
- The applicability of a functional testing method is limited to systems described via a particular modeling technique
- **1. Testing without a fault model**
 - Quality of the functional tests is unknown
 - Typically does not check that unintended operations do not occur (e.g., In addition to a correct transfer of data into register R1, the presence of a fault may cause the same data to be written into register R2)
- **2. Using specific fault models**
 - We do not know the comprehensiveness of a functional fault model → functional fault coverage is not meaningful

28

28

1. Without fault models

- Develop test programs that can be executed on the processor
- Method:
 - Test each instruction
 - Test each subunit such as ALU
 - Test buses, register file and decoders
 - Test sequencing of instructions
- **Key idea:** Start small – test components and instructions that are easy to test and then use the tested parts to test other parts

29

29

2. Using specific fault models

- Graph model for microprocessors: based on architecture and instruction set
- Fault classes:
 - Addressing faults affecting the register-decoding function, instruction-decoding, and instruction sequencing
 - Faults in the data-storage, data-transfer, data-manipulation functions
- Fault model development
 - Determine which instructions are “easy” to execute – such as uses fewest resources, fewest cycles – easy to control and observe
 - Use such instructions to read and write register file to test register file and address decoding logic
 - Test buses by moving different types of data on buses
 - Test ALU by executing ALU related instructions such as ADD, SUB, ...
 - Buses: stuck-at and bridging faults
 - ALU, register file: stuck-at
 - Instruction decoder:
 - » No instruction is executed
 - » Different instruction is executed
 - » An additional instruction is also executed
- Algorithm development
 - Develop simple sub-programs for each sub-unit testing
 - Put them together

30

30

“Snapshot” of Selected Microprocessor Testing and DFT Research Papers

- **Intel**

- D.M. Wu et al., An Optimized DFT and Test Pattern Generation Strategy for an Intel High Performance Microprocessor, ITC, 2004.

- **AMD**

- A. Sehgal et al., Test cost reduction for the AMD Athlon processor using test partitioning, ITC, 2007.

- **SUN Microsystems**

- R. Molyneaux et al., Design for Testability Features of the SUN Microsystems Niagara2 CMP-CMT SPARC Chip, ITC, 2007.

- **IBM**

- R. Franch et al., On-chip Timing Uncertainty Measurements on IBM Microprocessors, ITC, 2007.

31

31

1. Intel

- Intel high performance 3GHz uProcessor, multiple clock domains, multi-cycle paths, domino logic
- Concerns: silicon area, leakage power, scan performance impact
- DFT uses a Hierarchical Scan Architecture (“divide and conquer” strategy)
 - Design partitioned into clusters (e.g., floating point execution cluster)
 - A cluster contains more units
 - Each cluster has one cluster test controller (CTC) and at least one unit test controller (UTC)
- Each CTC has 36 scan chains that allow testing of partitions formed by selected clusters, units or combinations

32

32

Intel

- Scan chains not in partition under test can be bypassed
- ATPG patterns are generated using the scan-based ATPG tools
- Skip scan methodologies
 - Skip scan technique or Data Path Interleaved Scan (DI-Scan)
 - Follow a set of DI-Scan rules: DI-Scan used only in datapath pipelines, control logic is full scan, etc.
- Cache/memory testing
 - Programmable built-in self-testing (PBIST)
 - Access to all portions of PBIST is available through the JTAG TAP controller
 - Direct access testing (DAT): 100 times faster production test
 - Programmable weak-write test mode (PWWTM): to detect stability types of defects in memory cells

33

33

Intel

- Integrated test controller (ITC) includes the TAP logic
 - Complies with JTAG (IEEE 1149.1)
 - Provides access to testability and debug features:
 - » Micro-breakpoints
 - » Control register bus access
 - » Scan, Scanout, Signature mode
 - » Thermal sensor control
 - » Fuse programming, DAT mode
 - » Boundary scan register
- Full chip ATPG methodology with a very low scan overhead
- On-chip weighted random patterns BIST structure including a test compression structure

34

34

2. AMD

- 33-element partitioning → 80% reduction in test time compared to a flat model
- Advantages of modular test:
 - Reduced ATPG run-time
 - Greater test reuse
 - Simplified verification and scan chain failure debug
 - Reduced test time
- Note that this is [similar to the Intel approach](#) as divide-and-conquer

35

35

AMD

- Partitioning in three major steps:
 - Disposition of partition boundaries – done by surrounding each test module with a core test wrapper
 - Connect partition module to the test resources – known as providing a test access mechanism (TAM). Use 40 scan chains
 - Test boundaries are selected considering:
 - » Maximizing test coverage
 - » Minimizing pattern count
 - » Using the shortest possible scan chains
 - » Minimizing routing overhead
 - » Re-using existing scan registers at partition boundaries
 - » Allowing parallel module testing if desired

36

36

AMD Athlon Chip

- Design partitioned into 10 top-level modules and 33 second-level modules
- Test time is reduced with 38% compared to the non-modular approach. Attributed to reduction in the scan chain lengths in each module compared to the length of 4000 in the flat case
- Number of flops increased with 5% (due to wrapper cells)
- The cumulative pattern count of the modules 370% higher

37

37

3. SUN

- Niagara2 SPARC:
 - 8 processor cores, 1.4GHz, 4MB on-chip L2 cache, 65nm technology
 - 8 clock domains, mixture of full custom, semi-custom, and ASIC design styles, 300 SRAMs
- Level sensitive scan architecture
- Every SRAM tested with at-speed MBIST
- Scan chains
 - More than 1 million flops are organized as 32 scan chains
 - 84 JTAG scan chain configurations, 2 manufacturing scan chains
 - 35 MBIST chains for rapid programming of MBIST configuration registers

38

38

SUN

- Stuck-at test coverage: 98.5%
- Transition test coverage: 82%
- Path delay testing
 - 15,000 paths in each core tested at-speed
- SRAM access
 - Each SRAM is equipped with scanable input flops
 - Micro Test: process used to access the SRAM during debug from the JTAG port
- Memory BIST: at-speed testing
 - 80 MBIST engines
 - March C- forms the basis of the test algorithm
 - Read after write worst case (RAWWC) test
- FIFO memories (200Kbits): equipped with custom clock MUX
- CAM, Double-pumped memories (network interface unit): March tests
- Direct Memory Observe test: combination of MBIST with direct pin access to facilitate embedded SRAM bitmapping
- Support for JTAG 1149.1 boundary scan testing

39

39

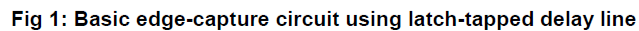
4. IBM

- Timing uncertainty comprised of:
 - PLL jitter
 - Clock distribution skew
 - Across chip variations
 - Power supply noise
- On-chip measurement macro called SKITTER (skew + jitter): measures timing uncertainty from all combined sources; 5-8ps resolution
- Very sensitive monitor of power supply noise, as dominant factor of timing uncertainty

40

40

- SKITTER used in IBM microprocessors: PPC970MP, XBOX360, CELL broadband engine, POWER6



41

- Changing location of the edges in the Skitter sampling latches is a good indicator of the variations in chip timing
- The Skitter itself can be self-monitoring and trigger a readout if an edge is detected in a bin where it is not expected
- Measurements can be converted from bin counts into picoseconds
- The shift in an edge bin can be converted into mV of VDD noise
- Duty cycle measurements useful
- Multiple Skitters on chip: for each core, at different locations on chip, in the front side bus, etc.

42

Conferences

- For the latest research on Testing, not only of processors, check out the latest papers published in the following conferences and journals:
- IEEE International Test Conference (ITC)
 - <https://easychair.org/smart-program/ITC2024/index.html>
- IEEE International Conference on Design & Test of Integrated Micro & Nano-Systems (DTS)
 - <https://ieeexplore.ieee.org/xpl/conhome/1832104/all-proceedings>
- IEEE International Automatic Testing Conference, AUTOTESTCON
 - <https://ieeexplore.ieee.org/xpl/conhome/1000070/all-proceedings>
- IEEE Design & Test Magazine
 - <https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6221038>

43

43

Conclusions

- Design hierarchy & circuit partitioning - **Divide-and-conquer** seems to be a successful testing strategy
- **(M)BIST for large memories/arrays**
- **Special BIST for small buffers**
- **Scan for random logic**
- Full chip testing employs multiple scan chains, MBIST, boundary scan, transition and path delay tests
- Other design for testability and debug/diagnosis: Skitter, software-based defect detection and diagnosis
- Fault tolerance techniques: self repairing microprocessor arrays

44

44