

ECE 269

VLSI System Testing

Krish Chakrabarty

Lecture 13:
Design for Testability (DFT): 1

Outline

- Motivation and Goals
- Controllability and Observability
- Ad Hoc DFT Methods
 - Control/Test Point Insertion
 - Circuit restructuring
 - Timing considerations

Why Design for Test?

- Test generation is complex (NP-complete), limiting exact application to circuits of moderate complexity
- Random logic circuits containing, say 10^6 or more gates, or 10^3 or more flip-flops may be too large for ATPG
- Heuristic methods generally used for testing complex circuits, e.g. microprocessors or high-density RAM chips.
 - The fault coverage of such methods is low and hard to determine
- To ensure high testability, **design for testability (DFT)** needed.
- Make circuits easy to test by design

Testability Definitions (Keiner 1980)

Testability

“A design characteristic which allows the status (operable, non-operable, or degraded) of a unit to be determined in a timely manner”

Design for Testability (DFT)

“A design process such that deliberate effort is expended to assure that a product may be thoroughly tested with minimum effort and cost, and that high confidence may be ascribed to the test results”

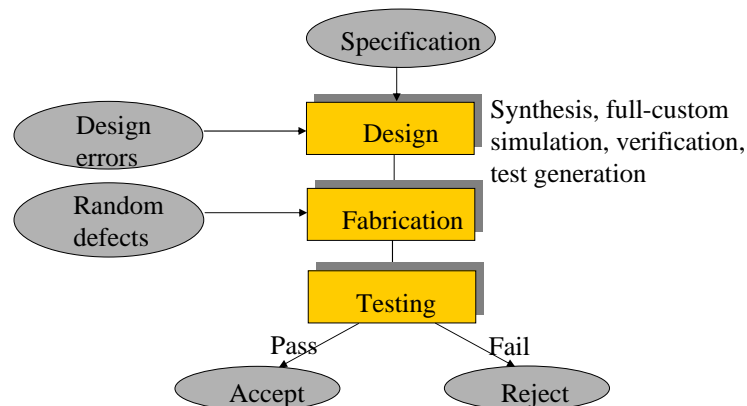
Testability Goals

- Maximize fault coverage
- Minimize test application time
- Minimize test data volume
- Minimize test generation effort
- Maximize fault isolation (isolating fault to smallest replaceable component)
- Minimize hardware/software overhead needed for testing
- Make the system as much self-testing as possible

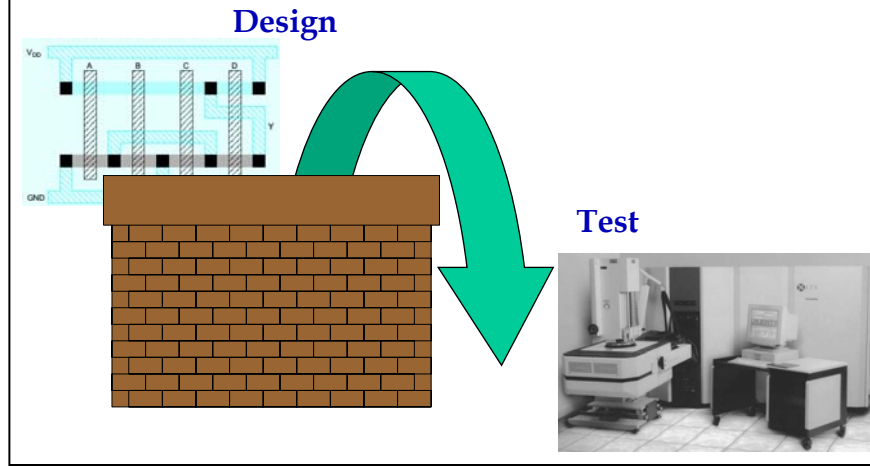
Trade-offs Needed!

Design and Test Flow: Old View

- Test is merely an afterthought



“Brick Wall” Between Design and Test



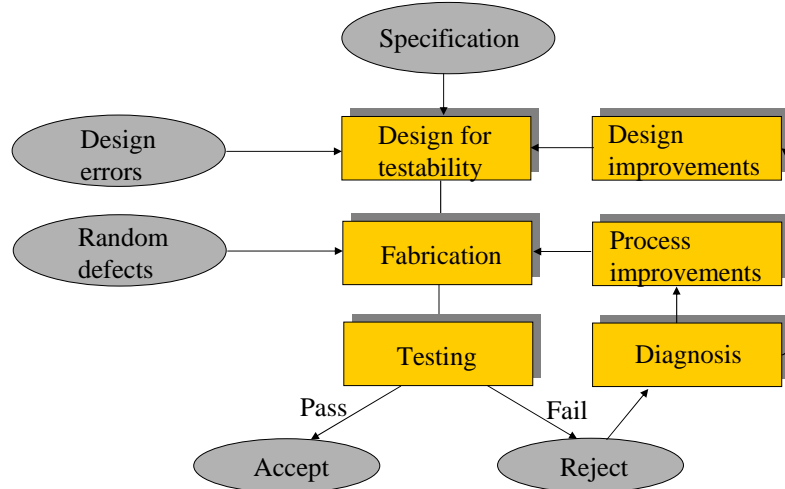
ECE 269

Krish Chakrabarty

7

Design and Test Flow: New View

- Design and test are tightly coupled



ECE 269

Krish Chakrabarty

8

Definition

- *Design for testability* (DFT) refers to those design techniques that make test generation and test application cost-effective.
- DFT methods for digital circuits:
 - Ad-hoc methods
 - Structured methods:
 - *Scan*
 - *Partial Scan*
 - *Built-in self-test* (BIST)
 - *Boundary scan*
- DFT method for mixed-signal circuits:
 - Analog test bus

Ad-Hoc DFT Methods

- Good design practices learnt through experience are used as guidelines:
 - Avoid asynchronous (unlocked) feedback.
 - Make flip-flops initializable.
 - Avoid redundant gates. Avoid large fanin gates.
 - Provide test control for difficult-to-control signals.
 - Avoid gated clocks.
 - Consider ATE requirements (tristates, etc.)
- Design reviews conducted by experts or design auditing tools.
- Disadvantages of ad-hoc DFT methods:
 - Experts and tools not always available.
 - Test generation is often manual with no guarantee of high fault coverage.
 - Design iterations may be necessary.

Impact of Testing Method

Consider testing a large combinational circuit. The precise impact of each testing method depends on the circuit size, circuit complexity, and practical design constraints

Testability parameter	<i>Testing method</i>		
	PODEM	Exhaustive	Random
Fault coverage	good	best	worst
Test application time	good	worst	fair
Test data length	best	worst	fair
Test generation effort	worst	best	good
Suitability for self-test	bad	fair	good

Testability Factors

Controllability and Observability

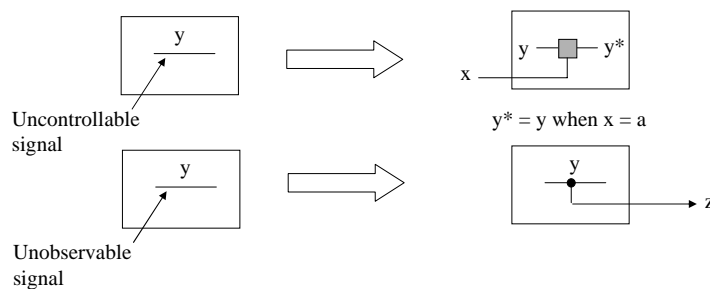
- Compute controllability and observability values for all signal nodes in the circuit
- “Bad” values can pinpoint hard-to-test areas of the circuit that need redesign
- Computation effort should be small, e.g., 10% of test generation effort. (Implies use of heuristics)

Controllability and Observability

- Several related schemes have been proposed, e.g.
 - TIMEAS (Stephenson and Grason, 1976)
 - SCOAP (Goldstein, 1979)
- Interpretation of controllability/observability values is difficult
- Often unclear how to modify circuit to improve specific controllability/observability values
- Systematic DFT can provide high levels *a priori*

Control/Test Point Insertion

- Make hard-to-control internal signals controllable via extra primary inputs and logic (control points)
- Make hard-to-observe internal signals observable via extra primary outputs and logic (test points)



Control/Test Points

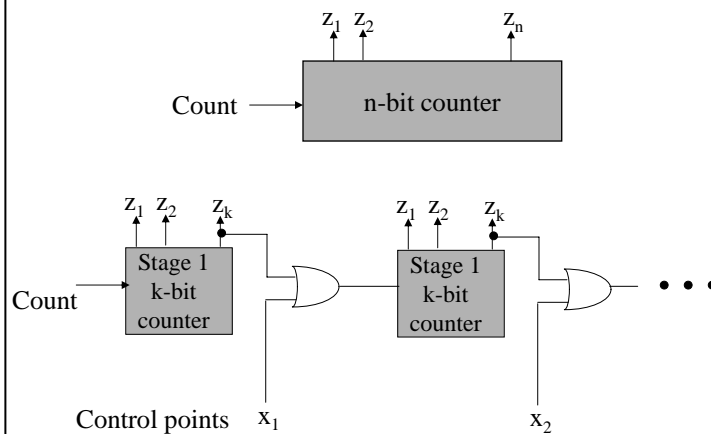
Site Selection

- Memory elements determining a system's control states
- Long narrow subcircuits with buried components
- Short wide subcircuits with high fan-in or fan-out
- Internal system buses
- Redundant circuits containing undetectable faults
- Circuit bottlenecks determined by programs such as SCOAP

Major Limitations

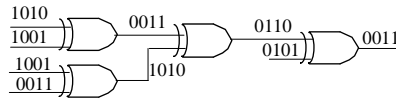
- Availability of spare I/O pins
- Cost of added test/control circuits

Circuit Restructuring



XOR-Based Designs

- Circuits composed exclusively of XOR (and XNOR) are easy to test
 - Easy to observe
 - Easy to control
- Every all-XOR/XNOR circuit composed of 2-input XOR gates can be tested for all SSL faults with 3 test patterns (Homework problem!)
- Every all-XOR/XNOR circuit composed of 2-input XOR gates can be tested for all cell faults with 4 test patterns

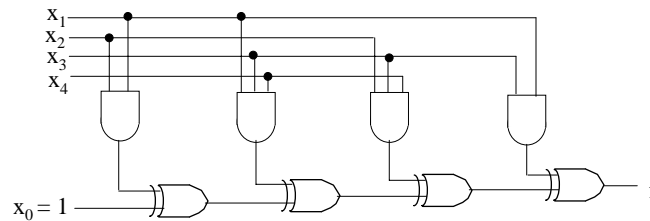


Reed-Muller Circuits

- Every Boolean function can be expressed in the “Reed-Muller” form consisting of an XOR sum of product terms with no complementation

Example:

$$f(x_1, x_2, x_3, x_4) = 1 \oplus x_1x_2 \oplus x_1x_3 \oplus x_1x_2x_3 \oplus x_2x_3x_4$$



Reed-Muller Circuits

- An n -input Reed-Muller circuit can be tested for all SSL faults using $n+4$ tests patterns
- It requires an extra AND gate, a control input and a test point
- The number of gates and the circuit delay grows exponentially with n .

$$\begin{aligned}f &= a + b \\ &= ab + \bar{a}b + a\bar{b} \\ &= ab \oplus \bar{a}b \oplus a\bar{b} \text{ (sum of minterms)} \\ &= ab \oplus (1 \oplus a)b \oplus a(1 \oplus b) \\ &= ab \oplus b \oplus ab \oplus a \oplus ab = a \oplus b \oplus ab\end{aligned}$$

Design Rules Summary

- Design controllable (e.g. initializable) and controllable circuits with careful selection of control/test points
- Partition large hard-to-test circuits into small testable components
- Allow feedback paths to be opened and closed
- Avoid redundancy, or allow it to be overridden during testing
- Avoid asynchronous circuits and provide access to clock signals
- *Often ad hoc design modification is too late to significantly improve a circuit's testability*