COEN 6521 VLSI Testing: Built-In Self Test

Zeljko Zilic

(In Ottawa)
zeljko@ece.mcgill.ca
Some slides based on the material provided by Bushnell and Agrawal
Overview

- Introduction to BIST
- Test Pattern Generators
- Response Compactors
Economics of BIST - Complexity

- Complexity
  - Cascaded devices
    - No simple way to obtain test vectors for the whole circuit even if tests for individual sub-blocks are given
      - Tests resulting in 100% fault coverage of individual elements do not guarantee 100% of the whole design: connections often untestable
    - BIST allows hierarchical decomposition of cascaded devices for test purposes
      - Example - Board with few chips: For the chip test system sends control signal to the board activating BIST on chip. Results are sent back to the system. In case of errors BIST hardware indicates which chip was defective. BIST tests all the embedded designs and interconnects, leaving only functional verification of the cascaded design to system-level tests.
Economic of BIST - Testing

- Test generation
  - Practically impossible to carry tests and responses involving hundreds of chip inputs through many layers of circuitry to chip-under-test. BIST localizes testing eliminating these problems

- Test application
  - BIST saves significantly test application time compared to external testers
  - BIST testing capabilities grow with the VLSI technology, whereas test capabilities always lag behind VLSI technology for external testing
  - Low development cost, as BIST is added to circuits automatically through CAD tools
Cost of BIST

- Chip area overhead for:
  - Test controller
  - Hardware pattern generator
  - Hardware response compacter
  - Testing of BIST hardware
- Pin overhead -- At least 1 pin needed to activate BIST operation
- Performance overhead – extra path delays due to BIST
- Yield loss – due to increased chip area or more chips in system because of BIST
- Reliability reduction – due to increased area
- Increased BIST hardware complexity – happens when BIST hardware is made testable
BIST Hierarchy
Random Logic BIST – Defs. 1

- **BILBO** – *Built-In Logic Block Observer*: A series of flip-flops with added testing hardware configuring flip-flops to work as: scan chains, *linear feedback shift register* (LFSR) pattern generator, LFSR-based response compactor, or, in the operation mode, as D flip-flops.

- **Concurrent Testing** – A testing process detecting faults during normal system operation.

- **Non-Concurrent Testing** – A testing process that requires suspension of normal system operation to test for faults.
Random Logic BIST – Defs. 2

- **Exhaustive Testing** – BIST approach in which all \(2^n\) possible patterns are applied to \(n\) circuit inputs.

- **Pseudo-Exhaustive Testing** – BIST approach in which a circuit having \(n\) primary inputs (PIs) is broken into smaller, overlapping blocks, each with < \(n\) inputs. Each of the smaller blocks is tested exhaustively.

- **Pseudo-Random Testing** – A BIST pattern generator that produces a subset of all possible tests having properties of randomly-generated tests.

- **CUT** – *Circuit-Under-Test*

- **TPG** – Hardware *Test-Pattern Generator*
Random Logic BIST – Defs. 3

- **LFSR** – *Linear Feedback Shift Register*.
  - Generates a pseudo-exhaustive random tests
  - Also used as test response compactor
- **Irreducible Polynomial** – A Boolean polynomial that cannot be factored
- **Primitive Polynomial** – A primitive Boolean polynomial $p(x)$ allows computing all possible non-zero polynomials of degree less than $p(x)$
BIST Process

1. **Hardware Pattern Generator**
   - Primary Inputs

2. **Input MUX**
   - Primary Outputs

3. **CUT**
   - Output Response Compactor
   - Signature

4. **Test Controller**

5. **ROM**
   - Reference Signature

6. **Comparator**
   - Good/Faulty
BIST Controllers

- **Test controller** – Hardware used to activate self-test simultaneously on all PCBs
- Each board controller activates parallel chip BIST Diagnosis effective only if very high fault coverage
Multi-Purpose BIST Implement.

- BILBO1 configured as LFSR pattern generator to test CUT1. BILBO2 is a response compactor of CUT1. During this process CUT2 is ignored.
- Next, BILBO2 configured as LFSR pattern generator for CUT2, while BILBO1 works as compactor to compact test responses of CUT2. CUT1 is ignored.
Complex BIST Implementation

- **First Test Mode**
  - LFSR1 is used for simultaneous test pattern generation for CUT1 and CUT2
  - BILBO2 is configured as response compactor for CUT1, while LFSR3 works as response compactor for CUT2. CUT3 is held steady

- **Second Test Mode**
  - BILBO2 generates patterns for CUT3, while LFSR3 is test response compactor. CUT2 is held steady
Bus-Based BIST Architecture

- *Self-test control* broadcasting patterns to each CUT over bus – parallel pattern generation
- Awaits bus transactions showing CUT’s responses to the patterns: serialized compaction
BIST Pattern Generation

- **ROM**: ATPG test patterns stored in ROM on chip
  - Method expensive in chip area

- **LFSR**: Pseudo-random test patterns
  - Often more than 1 million tests needed to obtain a good fault coverage
  - Cheap in terms of chip area and dominant in BIST

- **Binary Counters**: Exhaustive tests
  - Very much time absorbing (64-bit counter, 100 MHz, requires 51,240,995.8 h to complete all $2^{64}$ patterns)
    - Solution: Partition the input space
  - More expensive in terms of area than LFSRs
BIST Pattern Generation cont.

- *LFSR and ROM*: LFSR used as the primary test mode. Process augmented with ATPG patterns stored in on chip ROM, which are missing from LFSR sequence.

- *Cellular Automata*: Each pattern generation cell has few gates, a flip-flop and is connected only to the neighboring gates. The cell is replicated to produce the cellular automata.
Exhaustive Pattern Generation

- Exhaustive tests are intended to show that:
  - Every intended circuit state exists
  - Every state transition works
- Example: $n$-input circuits, requiring all $2^n$ vectors
  - Impractical for $n > 20$
Pseudo-Exhaustive Patterns 1

- **Verification testing** (cone segmentation)
  - Large circuits partitioned into fanin cones by backtracking from each primary output (PO) through the circuit to the gate with influences the output
    - Fanin cones can often be tested in parallel – time reduction
    - Cones depend on subset of primary inputs (PIs) – reduction in pseudo-exhaustive patterns to test each cone

- Example: Two 5-input cones covering 8-input logic
  - Reduction in tests from $2^8 = 256$ to $2^5 \times 2 = 64$
  - Disadvantages: incomplete fault coverage
Pseudo-Exhaustive Patterns 2

Backtracking for Pseudo-exhaustive Testing
Pseudo-Exhaustive Patterns 3

- **Hardware partitioning**
  - Physical segmentation in which extra logic is added to the circuit in order to divide CUT into smaller subcircuits, each subcircuit directly controllable and observable
    - Each subcircuit tested exhaustively

- **Sensitized path segmentation**
  - Circuit is partitioned so that sensitizing paths are set up from PIs to the partition inputs, and then from the partition outputs to Pos
    - Each partition is tested individually
Random Pattern Testing

- Top curve - random pattern testable circuit
- Bottom curve - random pattern resistive circuit

(a) Top curve -- random pattern testing with acceptable fault coverage.
(b) Bottom curve -- unacceptable random pattern testing.
Pseudo-Random Pattern Generation

- **Standard Linear Feedback Shift Register (LFSR)**
  - Patterns generated algorithmically – repeatable
  - Most of desirable random # properties
    - Still patterns linearly dependant, CA sometimes better solution

- Need not cover all $2^n$ input combinations
- Long sequences needed for good fault coverage
Standard LFSR: Type 1

Type 1 LFSR: \( h_n \) are tap coefficients indicating presence or absence of feedback form DFF

- External XOR LFSR, as feedback network of XORs is feed externally from \( X_0, \ldots, X_{n-1} \)
LFSR Operations

- Operations of LFSR are in Galois Field GF(2) arithmetic
  - Multiplication by $x$ is equivalent to a right shift in LFSR by one bit
  - Addition is the XOR operation:
    - $0 - 0 = 0, \ 0 - 1 = 1, \ 1 - 0 = 1, \ 1 - 1 = 0$
LFSR Matrix Type 1

\[ X(t+1) = T_s X(t) \]
\[ X_0 = 1, \quad X_1 = x \cdot X_0, \quad X_2 = x \cdot X_1 = x_2, \ldots, \quad X_n = x \cdot X_{n-1} = x_n \]

\[ f(x) = |T_s - I X| = 1 + h_1 x + h_2 x^2 + \ldots + h_{n-1} x^{n-1} + x^n \]
Galois Field and FLSRs

- *Galois field* (GF2) (mathematical system):
  - Multiplication by $x$ equivalent to right shift of LFSR
  - Addition operator - XORing of two or more values

- $T_s$ companion matrix:
  - 1st column 0, except $n$th element which is always 1 ($X_0$ always feeds $X_{n-1}$)
  - Rest of row $n$ – feedback coefficients $h_i$
  - Rest is identity matrix $I$ – means a right shift

- Near-exhaustive (maximal length) LFSR
  - Cycles through $2^n - 1$ states (excluding all-0)
  - 1 pattern of $n$ 1’s, one of $n-1$ consecutive 0’s
LFSR Theory

- All-zero LFSR DFFs initialization forbidden
  - No other than all-zero test pattern resulting
- If $X$ is initial state, states to follow: $T_s X$, $T_s^2 X$, $T_s^3 X$, ...
- **Matrix period:**
  Smallest $k$ such that $T_s^k = I$
  - $k$ LFSR cycle length
- Described by characteristic polynomial:
  \[
  f(x) = |T_s - I X| = 1 + h_1 x + h_2 x^2 + \ldots + h_{n-1} x^{n-1} + x^n
  \]
Maximum-lengths Sequences

- There is one pattern of $n$ consecutive ones and one pattern of $n-1$ consecutive zeros
- There is an autocorrelation property
  - Any two sequences, the original and the circularly shifted sequence (in the same LFSR) will be identical on $2^{n-1}$ positions
Example: Standard LFSR

Characteristic polynomial $f(x)$ of external-XOR LFSR is read from right to left. Since the rightmost Dff is always tapped, this polynomial has a $1(x^0)$. Middle Dff is also tapped, so $x$ term is present ($h_1 = 1$). There is no $x^2$ term, as Dff3 is not tapped. There is always $xn$ term in characteristic polynomial for standard $n$-bit LFSR, so $x^3$ is included. Overall, $f(x) = 1 + x + x^3$. 

| X0 | 1 0 0 1 0 1 1 1 0 |
| X1 | 0 0 1 0 1 1 1 0 0 |
| X2 | 0 1 0 1 1 1 0 0 1 |

$X0, X1, X2_{initial} = \text{"001"}$
Type 2 LFSR: $h_n$ are tap coefficients indicating presence or absence of feedback from DFF

- Internal XOR LFSR – faster than external XOR LFSR
LFSR Matrix: Type 2

\[
\begin{bmatrix}
X_0(t+1) \\
X_1(t+1) \\
X_2(t+1) \\
\vdots \\
X_{n-2}(t+1) \\
X_{n-1}(t+1)
\end{bmatrix}
= 
\begin{bmatrix}
0 & 0 & 0 & \ldots & 0 & 0 & 1 \\
1 & 0 & 0 & \ldots & 0 & 0 & h_1 \\
0 & 1 & 0 & \ldots & 0 & 0 & h_2 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
0 & 0 & 0 & \ldots & 1 & 0 & h_{n-2} \\
0 & 0 & 0 & \ldots & 0 & 1 & h_{n-1}
\end{bmatrix}
\begin{bmatrix}
X_0(t) \\
X_1(t) \\
X_2(t) \\
\vdots \\
X_{n-2}(t) \\
X_{n-1}(t)
\end{bmatrix}
\]

\[X(t+1) = T_M X(t)\]
\[X_0 = 1, X_1 = x X_0, X_2 = x X_1 = x^2, \ldots, X_n = x X_{n-1} = x^n\]

\[f(x) = \left| T_M - I X \right| = 1 + h_1 x + h_2 x^2 + \ldots + h_{n-1} x^{n-1} + x^n\]
Example: Modular LFSR

\[
\begin{bmatrix}
X_0(t+1) \\
X_1(t+1) \\
X_2(t+1) \\
X_3(t+1) \\
X_4(t+1) \\
X_5(t+1) \\
X_6(t+1) \\
X_7(t+1)
\end{bmatrix} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 
\end{bmatrix} \begin{bmatrix}
X_0(t) \\
X_1(t) \\
X_2(t) \\
X_3(t) \\
X_4(t) \\
X_5(t) \\
X_6(t) \\
X_7(t)
\end{bmatrix} = f(x) = 1 + x^2 + x^7 + x^8
\]
Primitive Polynomials

- Highly desirable that LFSR generate all possible $2^n-1$ patterns (implement *primitive polynomials*)
- Requirements for primitive polynomials:
  - Polynomial must be monic, i.e., coefficient of the highest-order $x$ term of characteristic polynomial must be 1
    - Modular LFSRs: All Dffs must right shift through XORs from X0 to Xn-1, which must then feed back directly to X0
    - Standard LFSRs: All Dffs must right shift from Xn-1 to X0, which must then feed back into Xn-1 through XORing feedback network
- Characteristic polynomial must divide the polynomial $1+x^k$ for $k = 2^n - 1$, but not for any smaller $k
Test Response Compaction

- Severe amounts of data generated in CUT response to LFSR patterns – example:
  - 5 million random patterns
  - CUT - 200 outputs
  - Test results: 5 million x 200 = 1 billion bits response
- Uneconomical to store and check all of these responses on chip
- Responses need to be compacted
BIST Response Compaction

- **Definitions**
  - Compaction: reducing #bits of response
  - Compression: reversible reduction of #bits
  - Signature: describes good circuit response
  - Aliasing: bad circ. signature = good circ. sign.

- Response compaction: parity, ones count, transition count, syndrome testing
Some Compaction Definitions

- **Signature analysis** – Compaction of good machine response into good machine signature
  - Actual signature generated during testing, and compared with good machine signature

- **Aliasing** – Information loss during compaction causing signatures of good and some bad machines to match

- **Compaction** – Drastic reduction # bits in original circuit response – lose information

- **Compression** – Reduction of # bits in original circuit response – no information loss – fully invertible (can get back original response)
Transition Count

- Transition Count Response Compaction – Count # transitions from 0 to 1 and 1 to 0 as a signature

(a) Logic simulation of good machine and fault a stuck-at-1.

(b) Transition counts of good and failing machines.
LFSR Response Compaction

- LFSRs implement Cyclic Redundancy Check (CRC) codes
  - Division by polynomial – remainder is a signature
  - After testing – comparison of signature in LFSR to known good machine signature
  - Critical: Must compute good machine signature
Modular LFSR Response Compactor

- LFSR seed value is "00000"

Characteristic Polynomial $x^5 + x^3 + x + 1$
Modular LFSR Compactors - a Good Choice?

- Problem with ordinary LFSR response compacter:
  - Too much hardware if one of these put on each primary output (PO)
- Solution: MISR – compacts all outputs into one LFSR
  - Works because LFSR is linear – obeys superposition principle
  - Superimpose all responses in one LFSR – final remainder is XOR sum of remainders of polynomial divisions of each PO by the characteristic polynomial
Multiple Input Signature Register (MISR)

- Superimpose linearly multiple outputs to a single LFSR

- Example: Outputs \( d_i \) superimposed
Aliasing

- Aliasing – when bad machine signature equals good machine signature
- MISR - more aliasing than LFSR on single PO
  - Error in CUT output $d_j$ at $t_i$, followed by error in output $d_{j+h}$ at $t_{i+h}$, eliminates any signature error if no feedback tap in MISR between bits $Q_j$ and $Q_{j+h}$. 
Theorem 15.1: Assuming that each circuit PO $d_{ij}$ has probability $p$ of being in error, and that all outputs $d_{ij}$ are independent, in a $k$-bit MISR, $P_{al} = 1/(2^k)$, regardless of initial condition of MISR

- Not exactly true – true in practice
Aliasing Theorems, cont.

- **Theorem 15.2**: Assuming that each PO $d_{ij}$ has probability $p_j$ of being in error, where the $p_j$ probabilities are independent, and that all outputs $d_{ij}$ are independent, in a $k$-bit MISR, $P_{al} = 1/(2^k)$, regardless of the initial condition.
Built-in Logic Block Observer

- BILBO – combines:
  - DFF
  - test pattern generation
  - response compactor
  - scan chain function
Other BIST Techniques

- Test Point Insertion
- Memory BIST
- Complex Systems – Networking etc.
- Delay Fault BIST
- Mixed-signal BIST