The BIST History of FPGAs

Chuck Stroud
Electrical and Computer Engineering
Auburn University

Built-In Self-Test
Outline of Presentation

- Background
  - Overview of FPGAs & FPGA Testing
  - My Experience in FPGAs

- BIST Approaches for FPGAs
  - Logic BIST Approaches
    - CAD Tool Features vs. Testability
  - Routing BIST Approaches
  - Other Cores & Resources

- Embedded Processor-Based BIST

- Summary
FPGA Characteristics

- Configuration memory
  - 32K - 50M bits

- Array of Programmable Logic Blocks (PLBs)
  - 100 - 22,270 PLBs per FPGA
    - 1-8 4-input LUTs and 1-8 flip-flops per PLB

- Programmable interconnect network
  - Wire segments
    - 50 - 400 per PLB
  - Programmable switches
    - 80 - 2,400 per PLB

- Programmable I/O cells
  - Bi-direction buffer with flip-flops/latches
    - 50 - 750 per FPGA
Important Trends in FPGAs

- Dynamic partial reconfiguration
- Incorporating specialized cores
  - RAMs - single-port, dual-port, FIFO, ECC
    - 128 - 18K bits per RAM
    - 4 - 550 per FPGA
  - DSPs including multipliers, accumulators, etc.
    - 30 - 510 per FPGA
  - Embedded processor cores
    - Up to 2 hard cores per FPGA
    - Also support soft processor cores synthesized in FPGA
  - Internal access to configuration memory
    - Write and read access by embedded processor core
- FPGAs becoming more like SoCs
- ASICs & SoCs now incorporate FPGA cores
FPGA Testing Challenges

- Programmability
  - Must test all modes of operation
- Architectures designed for applications
  - Testing issues/problems left to product/test engineers
- CAD tools designed for high-level synthesis
  - Do not support control of proper test conditions
- Constantly growing sizes
  - Reconfiguration dominates test time
- Constantly changing architectures
  - Architectural features/limitations directly affect testability and test development
- Incorporation of many new/different cores
FPGA Testing

- Typically partitioned for logic and routing
  - But both resources needed to test each other
- External testing
  - Good for manufacture testing only
  - Tests applied via I/O pins
    - Package dependent and limited by I/O pins
  - Boundary Scan (only with INTEST)
    - Extremely long test time
- Internal Testing (BIST)
  - Good for manufacturing & system-level test
  - Good for embedded FPGA cores
FPGA Testing

- Application independent testing
  - Test all resources in FPGA
    - Good for manufacturing testing
  - Requires *many* test configurations
    - Long test time - downloads dominate test time
  - No area/performance penalty in system

- Application specific testing
  - Test only resources used by system function
  - Requires fewer configurations
    - But requires new tests for new applications
      - Good for system-level testing only
  - Area/performance penalty for test circuitry
System-Level FPGA Testing

- System-level test of FPGA-based designs
  - Diagnostic software for test in system mode
    - Many months of diagnostic code development
  - Good diagnostic resolution difficult to achieve
- DFT/BIST in FPGA (for system-level test)
  - Area penalty typically 10-30%
  - Performance penalty typically 2-3 gate delays
  - Less logic for system function
  - Longer design time
BIST for FPGAs

- **Basic idea:** reprogram FPGA to test itself
  - BIST logic disappears after test
  - No area overhead or performance penalties

- Applicable to all levels of testing
  - Application independent testing
  - A generic test for a generic component
  - Good diagnostic resolution
    - To faulty PLB or wire segment/switch within FPGA
    - No diagnostic code development or DFT design

- **Cost:**
  - Memory to store BIST configurations
    - **Goal:** minimize number of configurations
  - Download time to execute BIST configurations
    - **Goal:** minimize downloads
My BIST & FPGA Background

Bell Labs (1977-93)
- Telecommunications systems
- I designed
  - 21 production VLSI devices
  - Prototype boards for over half of these
  - 3 production printed circuit boards
- 1981 - began work on BIST
  - Most VLSI devices included BIST
- 1984 - began work on FPGAs
- 1985 - began work on synthesis tools
- 1987 - first mixed-signal BIST
Initial Work on FPGAs

- Bread board ASICs in design methodology
  - We used TTL and fused-based PALs in early 1980s
  - I designed RAM-based programmable device in 1984
    - Soft Programmable Logic Array Technology (SPLAT)
    - My original idea of BIST for FPGAs:
      - program for “easier” testing

- Needed synthesis tool to program SPLAT
  - Began development of CONES in 1985
  - Synthesized behavioral models written in C to
    - Standard cell based ASICs
    - PLD-based bread boards and PCBs
      - Used 22V10s and 16V8s and considered use of Xilinx LCA
      - Refined idea of BIST for FPGAs:
        - program for BIST and diagnosis – can route around faults
  - CONES became a design methodology
    - Widely used at Bell Labs until early 1990s (then VHDL)
BIST for FPGAs

- Started actual work on BIST for FPGAs in 1993
  - No one had done it (nothing in literature)
  - Good area for academic R&D starting at Univ. of Ky
- Off-line BIST and diagnosis of FPGAs
  - Funded by Bell Labs, NSF & UK CRMS (1993-1998)
  - ORCA 2C and 2CA series
    - Also some preliminary BIST work for ORCA 3C and 4 series
  - Miron Abramovici at Bell Labs joined me in 1994
    - Expertise in ATPG, DFT, and diagnosis of ASICs
- On-line BIST, diagnosis, and fault-tolerance
  - Funded by DARPA and Lucent→Agere (1997 - 2001)
  - Used ORCA 2C/2CA due to dynamic partial reconfiguration
    - Considered Xilinx 6800 series
  - John Emmert joined myself and Miron Abramovici in 1999
    - Expertise in fault-tolerant approaches in FPGAs
More BIST for FPGAs

- **Manufacturing test development and BIST**
  - Cypress Delta 39K series
    - Some initial work on 37K series CPLDs
  - Included (besides logic and routing)
    - BIST for embedded RAMs and FIFOs
    - Embedded logic analyzer
      - Similar to Xilinx ChipScope

- **Embedded processor-based BIST**
  - Atmel AT94K series FPSLIC
    - Also applicable to AT40K series FPGA
  - Included (besides logic and routing)
    - BIST for embedded RAMs and I/O buffers
    - Guard bands and fail-silent operation
BIST for Xilinx FPGAs

- **4000 & Spartan series**
  - 4000E, 4000XL/XLA, Spartan
  - BIST for logic and routing resources

- **Virtex-I, Virtex-II Pro, Spartan-II, Spartan-3**
  - Funded by NSA (2004-2005)
  - Preliminary work in
    - BIST and diagnosis
      - Logic, routing, I/O buffers, embedded RAMs, multipliers
      - Dynamic partial reconfiguration
      - Partial configuration memory read back
    - Guard bands and fail-silent operation
    - Embedded processor-based BIST for Virtex-II Pro

- **Virtex-4**
  - Funded by NSA (2005-2006)
  - BIST for logic, routing, I/O buffers, RAMs, DSPs
# Our FPGA BIST Configurations

<table>
<thead>
<tr>
<th>FPGA</th>
<th>Logic</th>
<th>Routing</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORCA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2C</td>
<td>9</td>
<td>27 (48)</td>
<td>66</td>
</tr>
<tr>
<td>2CA</td>
<td>14</td>
<td>48</td>
<td>76</td>
</tr>
<tr>
<td>Atmel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AT94K/40K</td>
<td>4</td>
<td>56</td>
<td>64</td>
</tr>
<tr>
<td>Cypress</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39K</td>
<td>20</td>
<td>419</td>
<td>459</td>
</tr>
<tr>
<td>Xilinx</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4000E/Spartan</td>
<td>12</td>
<td>128</td>
<td>152</td>
</tr>
<tr>
<td>4000XL/XLA</td>
<td>12</td>
<td>206</td>
<td>230</td>
</tr>
<tr>
<td>Virtex-I/Spartan-II</td>
<td>12</td>
<td>283</td>
<td>307</td>
</tr>
<tr>
<td>Virtex-4</td>
<td>15</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

Notes: Logic BIST configurations are applied 2 times
Configurations for embedded cores not included
Other Work in FPGA Testing

- Lombardi & Huang
  - External testing of logic and routing
    - Xilinx 4000 and Altera

- Renovell and Zorian (with others)
  - Test configurations for logic and routing
    - Xilinx 4000 and ORCA

- Harris and Tessier
  - BIST configurations for routing
    - Xilinx 4000 and Virtex

- Sun and Chan
  - Parity-based BIST for routing
    - Xilinx 4000

- Tahoori
  - Application dependent testing of logic and routing
    - Xilinx Virtex
First Logic BIST Approach

- Schematic entry difficult
  - Manual placement needed to test all PLBs
- Routing difficult with larger $N \times N$ arrays
  - Routing complexity = $O(N^2)$
  - Global routing resources heavily used

BIST start

TPG

BUT

LUT

FF

ORA

pass/fail

Routing complexity = $O(N^2)$

Global routing resources heavily used
Second Logic BIST (Iterative Logic Array)

Advantages:
- Linear routing complexity
- Easily scaleable
- Algorithmic PLB placement & routing with NCL

Disadvantages:
- 3 test sessions
- Difficult to propagate test patterns through BUTs
  - Particularly for sequential logic functions
Third Logic BIST (Hybrid)

- Two test sessions
  - Row or column orientation
- Good balance of global & local routing
- Algorithmic placement & routing with NCL
  - Easily scalable
  - Good for dynamic partial reconfiguration
BISTory of the ORA (BISTora)

- ORAs impact BIST architecture
  - Both logic and routing BIST
  - ORA design has had big impact on history of BIST for FPGAs
    ✓ Greater than TPGs

- A history of ORAs in BIST for FPGAs could take a most of a seminar by itself
  - But I don’t want to BORA you!!!
Latest Logic BIST (Circular)

- Circular comparison of BUTs
  - Better diagnostic resolution
  - Possibly better fault detection

- Need TPGs
  - Embedded processor
  - Other cores
    - DSP
    - Embedded RAM
      - DSP counter reads
      - RAM (ROM) with test patterns

- Need sufficient routing resources
  - Available in many newer FPGAs
Logic BIST for Large FPGAs

- Need to manage loading on TPGs
- Signals degrade completely after 200 PIPs
- Quad BIST structures in large arrays
- Small number of rows with BIST structure across all columns

Repeat to fill array

BUT

ORA

BUT
Other Logic BIST Approaches

- Ping-Pong (BIST)
  - One test session by combining TPGs & BUTs
    - FSM approach with current state as test patterns
  - No ORAs
    - Config memory readback every BIST clock cycle
  - Only tests LUTs and Flip-flops
    - More reconfigurations in one test session

- BORA-BORA (Tahitian BIST)
  - One test session by combining BUTs & ORAs
  - No TPGs
    - Test patterns from embedded processor
  - Only tests LUTs and FFs
CAD Tool Features vs. Testability

- Controlling test conditions with CAD tools
  - Oriented for design
  - Oriented for synthesis
  - **For testing we need to:**
    - Control unselected inputs to logic multiplexers
      - Test for stuck-at faults
    - Control opposite logic values on at least one unselected input for MUX PIPs
      - Test for PIP stuck-on faults
      - # test configurations = # MUX inputs
  - DRC complaints about antennas & stubs
    - Delete signals for test conditions
Logic BIST Place & Route

- **Problem:** PAR re-maps BUT configurations
  - Supposedly improves routability
  - Destroys test conditions for fault detection

- **Solution:** Develop our own routing heuristics in C program for logic BIST
  - Maintains test conditions
  - Faster clock frequencies
  - Longer BIST development time
NCL-XDL Example for ORA

inst "ora_1_1" "CLB", placed R1C1 CLB_R1C1,
cfg "F::#LUT:F=(F1@F2)+(F3@F4) F4MUX::F4I
  G::#LUT:G=(G1@G2)+(G3@G4) G3MUX::G3I G2MUX::G2I
  H::#LUT:H=F+G+H1 H0::G H1::C4 H2::F
CLKY::CLK DY::H YQMUX::QY SRY::RESET FFY::#FF
SRX::RESET FFX::#FF ";

net "ora_1_1_yq",
  outpin "ora_1_1" YQ,
  inpin "ora_1_1" C4,
;
ip R1C1 CENTER_GYQ -> CENTER_GYQ_VERT,
ip R1C1 CENTER_GYQ_VERT -> CENTER_H2R,
ip R1C1 CENTER_H2R -> CENTER_C4,
;
BLO1
  BRO1
  BLO2
  BRO2
LUT G

BLO3
  BRO3
  BLO4
  BRO4
LUT F

LUT H
YQ

unrouted
net in XDL

routed
net in XDL
Automated BIST Configurations

- C program generates .XDL file
- .XDL converted to .NCD
  - xld –xdl2ncd bist.ncd
- FPGA Editor
  - Design Rule Check
  - Route
    - “no pin swap” option
- .NCD converted to .BIT file to download into FPGA for BIST
Programmable Routing Network

- Wire segments of varying length
  - \(xN = N\) PLBs in length
    - \(N = 1, 2, 4, 6\) are common
  - \(xH = \) half the array in length
  - \(xL = \) length of full array

- Programmable Interconnect Points (PIPs)
  - Also known as Configurable Interconnect Points (CIPs)
  - Transmission gate connects to 2 wire segments
  - Controlled by configuration memory bit
    - \(0 = \) wires disconnected
    - \(1 = \) wires connected
PIPs

- **Break-point PIP**
  - Connect or isolate 2 wire segments

- **Cross-point PIP**
  - 2 nets straight through
  - 1 net turns corner and/or fans out

- **Compound cross-point PIP**
  - Collection of 6 break-point PIPs
    - Can route to two isolated signal nets

- **Multiplexer PIP**
  - Directional and buffered
  - Select 1-of-$N$ inputs for output
    - Non-decoded MUX PIP – 1 config bit per input
    - Decoded MUX PIP – $N$ config bits select from $2^N$ inputs
  - Major routing resource in new FPGAs
Routing BIST

- Program PLBs as TPGs and ORAs
  - Like in logic BIST
- Program groups of wires under test
  - Wire segments
  - Programmable Interconnect Points
- Tests partitioned for local and global routing resources
  - Must route through PLBs for local routing
- Fault models
  - Bridging faults and opens in wire segments
  - Line stuck-at faults
    - Shorts to Vdd and Vss
  - PIPs stuck-on and stuck-off
- Test conditions
  - Opposite logic values on wires/PIPs
  - Monitor both logic values
First Routing BIST Approach

- Originally thought logic BIST would test routing resources
  - Not true (only 55% in ORCA)
- Comparison-based
  - ORAs compare two groups of WUTs
    - Similar to logic BIST
- Tried to test as much routing as possible at one time
  - Poor diagnostic resolution
  - Difficult to develop configurations
Second Routing BIST

- Developed during on-line BIST project
  - Testing restricted to routing resources for 2 rows or columns of PLBs
  - Small Self-Test AReas (STARs)
  - Comparison-based BIST
- Applied to off-line BIST
  - Fill FPGA with STARs
  - Tests run concurrently
  - Diagnostic resolution to STAR
- Easier BIST development
  - But more BIST configurations
    ✓ 27 vs. 48 for ORCA 2C
Other Routing BIST Approaches

- **Parity-based (Sun and Chan)**
  - Xilinx 4000
  - Parity bit routed over fault-free resources
    - What is fault-free until you’ve tested it?

- **Harris and Tessier**
  - Used our comparison-based approach
    - Pointed out 2-testing requirement

- **Renovell and Zorian**
  - Minimum test configurations for switch boxes

- **Modified parity-based approach**
Latest Routing BIST

- Comparison-based BIST
  - No good for small PLBs and difficult to route

- Modified parity-based approach
  - $N$-bit up-counter with even parity, and
  - $N$-bit down-counter with odd parity
    - Gives opposite logic values for
      - Stuck-on PIPs & bridging faults
  - Parity used as test pattern
    - $N+1$ wires under test
  - Good for small PLBs

- Make STARs as small as possible

![Diagram of test pattern generator (TPG)](image)
### Our FPGA BIST Configurations

<table>
<thead>
<tr>
<th>FPGA</th>
<th>Logic</th>
<th>Routing</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORCA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2C</td>
<td>9</td>
<td>27 (48)</td>
<td>66</td>
</tr>
<tr>
<td>2CA</td>
<td>14</td>
<td>48</td>
<td>76</td>
</tr>
<tr>
<td>Atmel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AT94K/40K</td>
<td>4</td>
<td>56</td>
<td>64</td>
</tr>
<tr>
<td>Cypress</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39K</td>
<td>20</td>
<td>419</td>
<td>459</td>
</tr>
<tr>
<td>4000E/Spartan</td>
<td>12</td>
<td>128</td>
<td>152</td>
</tr>
<tr>
<td>4000XL/XLA</td>
<td>12</td>
<td>206</td>
<td>230</td>
</tr>
<tr>
<td>Virtex-I/Spartan-II</td>
<td>12</td>
<td>283</td>
<td>307</td>
</tr>
<tr>
<td>Virtex-4</td>
<td>15</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

Notes: Logic BIST configurations are applied 2 times
Configurations for embedded cores not included
Comparing FPGAs

- **Routing BIST**
  - Routing resources per PLB
    - 4000XL/XLA has 25% more than ORCA 2C/2CA
    - ORCA 2C/2CA has 48% more than 4000E/Spartan
  - Routing BIST configurations
    - 206 for 4000XL/XLA
    - 48 for ORCA 2C/2CA
    - 128 for 4000E/Spartan

- **Number and size of multiplexer PIPs**
  - N=5 for ORCA 2C multiplexer PIPs
  - N=35 for 4000XL/XLA multiplexer PIPs
  - **Bad News:** more & larger MUX PIPs in new FPGAs
    - Even more routing BIST configurations
Comparing Routing Architectures

- PLB input/output access to busses
  - More difficulty routing to/from wires = more configs

- Shared vs. dedicated busses to each PLB
  - Routing conflicts from TPGs to ORAs = more configs

Xilinx

FGC1 FGC2 FGC3 FGC4 X/XQ Y/YQ

Atmel

ORCA

F1-4 G1-4 C1-4 O1-4

repeaters

long lines

by-1 lines

by-4 lines
Current R&D

- Use embedded processor core to
  - Reconfigure FPGA for BIST
  - Execute BIST and retrieve BIST results
  - Perform diagnostic procedure
  - Perform fault injection emulation
    - Methodical verification of BIST configurations

- Processor must access configuration memory
  - Configuration memory read very helpful

- Implemented in Atmel AT94K
  - 8-bit AVR microcontroller
  - Configuration memory write access only

- Currently implementing in Virtex-4 for NSA
  - PowerPC (hard core) & MicroBlaze (soft core)
  - Configuration memory write and read access
Logic BIST Architecture - Virtex II Pro

- 4 Test Sessions
  - Right Half – East & West
  - Left Half – East & West

- Circular comparison for better diagnosis
  - TPG incorporated in the processor half of FPGA
  - Replace TPG column with ORAs
  - Extra routing needed for BUT-to-ORA connections at edge
# Test Time Results for AT94K

<table>
<thead>
<tr>
<th>Resource</th>
<th>Function</th>
<th>Download</th>
<th>Processor</th>
<th>Speed-up</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic BIST</td>
<td>Download</td>
<td>7.680 sec</td>
<td>0.101 sec</td>
<td>76.0</td>
</tr>
<tr>
<td></td>
<td>Execution</td>
<td>0.016 sec</td>
<td>0.085 sec</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td>Total time</td>
<td>7.696 sec</td>
<td>0.186 sec</td>
<td>41.4</td>
</tr>
<tr>
<td>Routing BIST</td>
<td>Download</td>
<td>20.064 sec</td>
<td>0.110 sec</td>
<td>182.4</td>
</tr>
<tr>
<td></td>
<td>Execution</td>
<td>0.026 sec</td>
<td>0.343 sec</td>
<td>0.075</td>
</tr>
<tr>
<td></td>
<td>Total time</td>
<td>20.090 sec</td>
<td>0.453 sec</td>
<td>44.3</td>
</tr>
<tr>
<td><strong>Total Test Time</strong></td>
<td><strong>27.786 sec</strong></td>
<td><strong>0.639 sec</strong></td>
<td></td>
<td><strong>43.5</strong></td>
</tr>
</tbody>
</table>

- 60 BIST configurations
- Maximum download clock frequency = 1MHz
- Maximum processor clock frequency = 25 MHz
Memory Reduction Results

<table>
<thead>
<tr>
<th>Resource Tested</th>
<th>Download</th>
<th>Processor</th>
<th>Memory Reduction Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average</td>
<td>File Size</td>
<td># Files</td>
</tr>
<tr>
<td>Logic</td>
<td>60 Kbyte</td>
<td>12 Kbyte</td>
<td>16</td>
</tr>
<tr>
<td>Routing</td>
<td>57 Kbyte</td>
<td>14 Kbyte</td>
<td>44</td>
</tr>
<tr>
<td>Combined</td>
<td>58 Kbyte</td>
<td>22 Kbyte</td>
<td>60</td>
</tr>
</tbody>
</table>

- Download approach needs 3.5 Mbyte for storage
  - Requires external control for BIST and diagnosis
- Processor-generated BIST program
  - Includes all control and diagnostic routines
  - Will fit in program memory of embedded processor
  - Or can be easily downloaded when needed
Summary

- Growing use of FPGAs in systems and SoCs
- FPGA testing is necessary but difficult due to
  - Programmability
  - Complex interconnect network
    - Logic BIST is simple compared to routing BIST
  - Constantly growing size and changing architectures
  - Incorporation of new and different cores
- Test development is time consuming
- New FPGA capabilities help BIST
  - Dynamic partial reconfiguration and readback
  - Configuration by processor cores