Built-In Self-Test for Regular Structure Embedded Cores in System-on-Chip

Srinivas Murthy Garimella
Master’s Thesis Defense

Thesis Advisor: Dr. Charles E. Stroud

Committee Members: Dr. Victor P. Nelson & Dr. Adit D. Singh

Department of ECE, Auburn University

The VLSI Design & Test Seminar Series March 23, 2005
Outline

- Motivation
- Background
- RAM BIST for Atmel SoCs
  - SoC Architecture
  - VHDL approach
  - AVR approach
- RAM BIST for Xilinx FPGAs
  - FPGA Architecture
  - VHDL approach
- Conclusions and Future Research
Motivation

- Testing of VLSI Chips
  - Up to 50-60% of manufacturing cost
- System-on-Chip (SoC)
  - $43.2 billion industry by 2009 (BCC)
- Embedded memory
  - Currently about 50% of SoC die area
  - Can grow up to 90% by 2010 (ITRS 2004)
Motivation (Contd..)

- Embedded memory cores
  - Increase density
  - Increase die size
  - Decrease yield

- Testing Embedded memory cores
  - Critical in SoCs
Background

- **System-on-Chip (SoC)**
  - Integration of various components (IP cores like processor, memory, FPGA) onto a single silicon chip
  - Elimination of interconnect effect on device performance
  - Realization due to advancement in semiconductor processing techniques
  - Communication products currently form its largest market segment
Configurable SoCs (CSoCs)

- SoCs with embedded Field Programmable Gate Array (FPGA)
  - Fixing design errors and reduce re-spin costs
  - Reusability for implementing different functions
  - Implementing DSP algorithms
  - Remote upgrades
Field Programmable Gate Arrays (FPGAs)

Programmable Elements

- Logic Blocks (PLBs)
- Interconnect network
- Input Output Buffers (IOBs)
- Embedded memory components
  - Coarse grained – dedicated memories
    - requires memory/logic partitioning during FPGA design
  - Fine grained – distributed memories
    - avoids poor memory utilization

Interconnect Network

PLB

IOB
RAM Functional Model

- RAM Types – SRAM, DRAM, Flash, etc.,
- Ports – Single-port, Dual-port, Multi-port
- Modes - Synchronous, Asynchronous
- Sizes – Different data widths and address widths
Memory Testing

- Functional testing
  - No knowledge of memory circuit implementation required
  - Fully functional test – $2^n$ complexity where $n = \text{number of cells in the array}$
  - Subset of likely to occur faults – Fault model
Fault Models

- Address decoder and Data line faults
- Refresh Logic and Sense Amplifier faults
- Cell-related faults
  - Stuck-At faults
  - Transition faults
  - Coupling faults (CFs)
    - Aggressor and Victim cells
    - Inter-word and Intra-word CFs
  - Neighborhood pattern sensitive faults (NPSFs) - base cell and neighboring cells
    - Static – base cell forced to a certain value
    - Passive – base cell changes
    - Active – base cell cannot change
- Multi-port memory faults
Memory Test Algorithms

- Traditional tests
  - Checkerboard, Walking 1/0, Butterfly
  - Order of complexity: $n^2, n \times \log_2 n$
  - Not sufficient for detecting all faults

- March Tests
  - Order of complexity: $n$
  - Different march tests for different fault models
  - Sequence of reads and writes
March Tests

- **Notation**
  - ↓: addressing downward
  - ↑: addressing upward
  - ‡: either way
  - w0: write 0
  - r1: read 1
  - Length = 16N, N = number of memory cells

- **Wider-Memories**
  - Background Data Sequences (BDS)
    - Detects intra-word CFs
    - Detects Bridging faults
    - Detects NPSFs
    - Number of BDS = \(\log_2(K)+1\), where \(K = \) data width
    - For e.g., BDS for 4-bit wide memory: 0000(1111), 0101(1010), 0011(1100)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Notation</th>
</tr>
</thead>
<tbody>
<tr>
<td>March LR w/o BDS</td>
<td>↓(w0); ↓(r0, w1);</td>
</tr>
<tr>
<td></td>
<td>↑(r1, w0, r0, r0, w1);</td>
</tr>
<tr>
<td></td>
<td>↑(r1, w0);</td>
</tr>
<tr>
<td></td>
<td>↑(r0, w1, r1, r1, w0); ↑(r0);</td>
</tr>
<tr>
<td>March LR with BDS</td>
<td>↓(w00); ↓(r00, w11);</td>
</tr>
<tr>
<td></td>
<td>↑(r11, w00, r00, r00, w11);</td>
</tr>
<tr>
<td></td>
<td>↑(r11, w00);</td>
</tr>
<tr>
<td></td>
<td>↑(r00, w11, r11, r11, w00);</td>
</tr>
<tr>
<td></td>
<td>↑(r00, w01, w10, r10);</td>
</tr>
<tr>
<td></td>
<td>↑(r10, w01, r01);</td>
</tr>
<tr>
<td></td>
<td>↑(r01);</td>
</tr>
</tbody>
</table>
AT94K SoC Architecture

- Three IP Cores
  - FPGA
    - Up to 48x48 array of PLBs
    - Embedded Free RAMs
  - SRAM
    - Dual-port Data SRAM
    - Single-port Program SRAM
  - 8-bit RISC Processor
    - Can write into FPGA configuration memory
      - Dynamic Partial Reconfiguration
FPGA Architecture

- Arranged in 4x4 array of PLBs
- PLB
  - Two 3-input LUTs
  - D Flip-Flop
  - Multiplexers
- Local routing to adjacent PLBs
  - 4 Directs (Y)
  - 4 Diagonals (X)
FPGA Architecture (cont…)

- **Global routing- 5 planes**
  - Two x8 lines per plane
  - Two x4 lines per plane
  - Repeaters provide buffering and connection

- **Free RAMs**
  - One 32x4 RAM for every 4x4 array of PLBs
  - Single-port or dual-port operation
  - Synchronous or Asynchronous operation
Free RAM Routing
Free RAM Testing

- Three modes of testing
  - Single-port synchronous
  - Single-port asynchronous
  - Dual-port synchronous
    - Asynchronous read
    - Not a true dual-port

- Test approach
  - Built-In Self-Test (BIST)
    - Advantages:
      - Minimal use of testers
      - At-speed testing
      - No X-states problem with memories
      - No fault-coverage problem as BIST relies on march tests for fault coverage
    - Disadvantages:
      - Area overhead (not for FPGAs)
      - Lower performance
BIST Approach

- Logic BIST approach
  - TPG
    - Single
    - Dual
  - ORA
    - Expected Data comparison
    - Adjacent RAM Comparison
- Number of test configurations
BIST Implementation (Dual-port testing)

- TPG – assuming logic and routing to be fault free
  - March DPR test algorithm (w/o BDS)
    - $\uparrow (w0:n); \uparrow (n:r0); \uparrow (w1:1;r1); \uparrow (w0:1;r0)$
  - VHDL Implementation – 66 PLBs
- ORA
  - Comparison with data from adjacent RAM
    - Lack of enough routing resources prevented implementation of expected data comparison
BIST Implementation (Single-port testing)

- **TPG**
  - March LR with BDS in synchronous mode
    - VHDL Implementation – 123 PLBs
  - March Y w/o BDS in asynchronous mode
    - \( \uparrow(w0) ; \uparrow(r0;w1;r1); \downarrow(r1;w0;r0); \uparrow(r0) \)
    - VHDL Implementation – 18 PLBs

- **ORA**
  - Comparison with expected data generated by TPG
Implementation Issues

- **VHDL approach**
  - No control over placement with Atmel’s tool – makes diagnosis difficult
  - Solutions:
    - Manual placement – tedious
    - Maintain mapping information – may change with synthesis
  - Irregular routing
    - Problems fitting BIST circuitry in smaller devices

- **Alternate approach**
  - Macro Generation language (MGL)
    - Similar to VHDL
    - Atmel proprietary HDL
    - Control over placement and routing
    - PLBs are primitive logic elements
Implementation Issues (cont...)

- Define relative placement of RAMs and ORAs
- Define interconnection between RAMs and ORAs
- TPG implementation in MGL
  not simpler
- VHDL-MGL mixed approach
  - Best solution
  - Implement TPG in VHDL
  - Implement rest of the BIST circuitry in MGL
  - Helps in implementing processor BIST
Mixed-approach Results

- #Collapsed faults = 1870
- Undetected faults = 6
- Fault Coverage = 99.81%

- Three BIST configurations developed
  - Tested free RAMs in AT94K10 (24x24 PLBs) and AT94K40 (48x48 PLBs) devices with fault injection
  - Requires three separate downloads
    - BIST-time = 300 us  Download-time = 1500 ms (for AT94K40)
  - BIST results read into PC and diagnosis performed inside PC
    - Locates faulty RAM and also faulty bit in the RAM
Free RAM BIST from embedded processor core

- Decrease total test time
  - Decrease number of downloads
  - Combine three configurations into one
  - Take advantage of dynamic partial reconfiguration from AVR
  - Move TPG into AVR
  - ORAs inside the FPGA core
  - Control the BIST from AVR
    - Start BIST
    - Retrieve BIST results at the end of BIST
    - Diagnose the BIST results

- Processor assumed to be fault-free

Processor assumed to be fault-free
Processor BIST approach

- One single download
  - Download Dual-port BIST configuration
  - Reconfigure ORAs and RAMs to test remaining configurations
  - Controlled with instructions from a higher controlling device (PC)
  - Performs diagnostics if instructed
Processor BIST Implementation

- Registered TPG signals
  - Since AVR Data bus is 8-bit wide
  - FPGAWE and FPGARE are clocks
  - IOSEL lines functions as clock enables for registers and for BIST clock
  - ORA results read from AVR Data bus

- One configuration developed
  - Downloaded into AT94K10 and AT94K10 devices and tested with fault injection
  - On-chip diagnostics implemented
    - MULTICELLO for dual-port
    - Simpler diagnostics for single-port

- Test-time = 8.8ms Download time=600ms
  - BIST-time increases
    - Parallel execution in FPGA replaced with sequential program execution in AVR

- Total test-time improved by a factor of 2.5
- Memory required for storing BIST configurations is reduced by a factor of 2.5
Processor BIST with no download

- Regular BIST structure inside the FPGA
  - TPG registers contribute to irregularity in both routing and logic
  - ORAs and RAMs are very regular
  - BIST-time = 20m Download-time = 200m
  - Total-test time improved by a factor of 7.5
  - Memory required for storing BIST configurations is reduced by a factor of 9
  - Increased BIST time at the cost of decreased download time
  - This approach helps in combining logic BIST and routing BIST with RAM BIST
### Diagnosis and BIST summary

<table>
<thead>
<tr>
<th>Function</th>
<th>Execution Cycles</th>
<th>Program Memory (bytes)</th>
<th>Data Memory (bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIST</td>
<td>398,100</td>
<td>1,860</td>
<td>72</td>
</tr>
<tr>
<td>Diagnostics</td>
<td>110,000</td>
<td>1,330</td>
<td>132</td>
</tr>
<tr>
<td>Total</td>
<td>508,100</td>
<td>3,190</td>
<td>204</td>
</tr>
</tbody>
</table>

- Diagnosis procedures are implemented in C
  - Can be used with BIST for other resources like logic and IOBs
  - Cannot be applied for routing
Embedded SRAM Memory

- 36KB in size
  - Shared by both AVR and FPGA
  - 20 KB of dedicated program memory
  - Cannot be tested from AVR
  - 4KB dedicated data memory
  - Remaining 12KB can be configured to be data or program memory
  - 96 bytes of dedicated FPGA memory
  - Testable portion from AVR and/or FPGA is 14KB + 96 bytes
Data SRAM testing

- Three modes of testing
  - Single-port from FPGA
    - March LR with BDS
    - VDHL Implementation – 203 PLBs
    - Isolate AVR from data SRAM
  - Single-port from AVR
    - March LR with BDS
    - Requires two phases with stack relocation
    - Implementation in C language
  - Dual-port from both FPGA and AVR
    - March s2pf and March d2pf w/o BDS
    - Requires two phases with stack relocation
    - Implementation in C language
- Total of five configurations
  - Reduced to three by combining single-port tests with dual-port tests
  - Downloaded to test data SRAM in AT94K10 and AT94K40 devices
# Summary of Memory tests for Atmel SoCs

<table>
<thead>
<tr>
<th>Testing resource</th>
<th>Configuration</th>
<th>BIST exec time (sec)</th>
<th>Dwell time (ms)</th>
<th>Total test time (ms)</th>
<th>TPG PLBs</th>
<th>ORA PLBs</th>
<th>Clock Speed (Hz)</th>
<th>Speed Up</th>
</tr>
</thead>
<tbody>
<tr>
<td>Free RAM (using FPGA)</td>
<td>Dual-port</td>
<td>147 µ</td>
<td>500</td>
<td>500.14</td>
<td>66</td>
<td>960</td>
<td>17.7 M</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Single-port sync</td>
<td>124 µ</td>
<td>500</td>
<td>500.12</td>
<td>123</td>
<td>1152</td>
<td>12.3 M</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Single-port async</td>
<td>40 µ</td>
<td>500</td>
<td>500</td>
<td>18</td>
<td>1152</td>
<td>21.4 M</td>
<td></td>
</tr>
<tr>
<td>Free RAM (using AVR&amp;FPGA)</td>
<td>All modes</td>
<td>8.8 m</td>
<td>600</td>
<td>608.8</td>
<td>14</td>
<td>1152</td>
<td>20 M</td>
<td>2.46</td>
</tr>
<tr>
<td>Free RAM (using AVR)</td>
<td>All modes</td>
<td>20 m</td>
<td>180</td>
<td>200</td>
<td>14</td>
<td>1152</td>
<td>20 M</td>
<td>7.5</td>
</tr>
<tr>
<td>Data SRAM</td>
<td>Single-port</td>
<td>32.7 m</td>
<td>375</td>
<td>407.7</td>
<td>210</td>
<td>16</td>
<td>18.5 M</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(using FPGA)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Single-port + dual-port</td>
<td>657 m</td>
<td>375</td>
<td>1032</td>
<td>30</td>
<td>8</td>
<td>20 M</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>(with stack relocation)</td>
<td>95 m</td>
<td>375</td>
<td>470</td>
<td>30</td>
<td>8</td>
<td>20 M</td>
<td></td>
</tr>
</tbody>
</table>
Portability of VHDL approach

- **VHDL approach**
  - FPGA-independent approach
    - Portable to most FPGAs
  - Reduces BIST development time
    - Minimal modifications needed due to architectural changes
  - Needs support from synthesis tool for placement of logic and RAM resources to implement diagnosis
  - March algorithm changes with the type of RAM and technology used
    - RAMBISTGEN tool- generates VHDL code for any single-port *march* algorithm
  - Flexibility of this approach assessed by testing embedded RAMs in Xilinx FPGAs and SoCs
RAMBISTGEN tool

Algorithm | Input File Format
--- | ---
\( \uparrow(w0); \) | d w 0
\( \uparrow(r0;w1;r1); \) | u r 0 , w 1 , r 1
\( \downarrow(r1;w0;r0); \) | d r 1 , w 0 , r 0
\( \uparrow(r0); \) | u r 0

- **Notation**
  - U – Upward Addressing
  - D – Downward Addressing
  - w – write
  - r – read

- **Implemented in Tcl/Tk**
  - 400 lines of code
  - Compatible with Windows and Unix environments
Xilinx FPGAs Architecture

- Coarse-grained architecture
- PLBs
  - Two to four slices
    - Two 4-input LUTs
      - RAM mode
        - Distributed memory
      - Shift Register mode
  - Multiplexers
  - 2 Flip-Flops
  - Carry Logic

Slice Architecture
Block RAM Features

- Embedded memory cores in Xilinx Spartan and Virtex FPGAs
- Number, size and organization varies with device
  - Virtex I and Spartan II
    - 4K bits
    - Five operational modes: 4Kx1 – 256x16
  - Virtex II, Virtex II pro and Spartan III
    - 18K bits
    - Six operational modes: 16Kx1 – 512x36
- True Dual-port
  - Independent control of clock and other control signals

Block RAM Architecture
Block RAM Features (cont...)

- **Write Modes**
  - **Write-First**
    - Cannot read and write to same location
  - **Read-First**
    - Supports read and write to same location
  - **No-Change**
    - Cannot read and write to same location

- **Parity bits**
  - One parity bit for every byte in Virtex II, Virtex II Pro and Spartan III devices

- **Synchronous set/reset of output latches**
Block RAM Testing

- Single-port mode testing
  - Five configurations for Virtex I and Spartan II
  - Six configurations for Virtex II, Virtex II Pro and Spartan III
  - Active levels of control signals, active edge of clock and all write modes tested in different configurations
- VHDL approach
  - TPG
    - RAMBISTGEN tool used
    - March LR used in all configurations
    - March LR with BDS with highest data width
    - March LR w/o BDS in other configurations
    - Reduces testing time
  - ORA
    - Circular comparison – increases diagnostic resolution at edges
    - Modified MULTICELLO used for diagnosis
- Placement of RAMs from VDHL

<table>
<thead>
<tr>
<th>RAM BIST Algorithm</th>
<th>TPG slices</th>
<th>ORA slices</th>
</tr>
</thead>
<tbody>
<tr>
<td>March LR w/o BDS</td>
<td>62</td>
<td></td>
</tr>
<tr>
<td>March LR with BDS (16-bit)</td>
<td>110</td>
<td>NxD x2</td>
</tr>
<tr>
<td>March LR with BDS (36-bit)</td>
<td>174</td>
<td></td>
</tr>
</tbody>
</table>

T = # of block RAMs

D = # of data bits

The VLSI Design & Test Seminar Series
March 23, 2005
Block RAM Testing (cont...)

- **Dual-port testing**
  - Only tested with highest data width
  - Two BIST configurations
  - TPG
    - March S2pf and March D2pf w/o BDS
    - Same VHDL code used for testing data SRAM in Atmel SoCs
- **ORA**
  - Expected data comparison with Read-First mode
  - Simpler diagnostics

**Devices used for testing**
- Spartan II – 2S50 - 8 Block RAMs
- Spartan II – 2S200 – 14 Block RAMs
- Virtex II Pro – 2VP30 - 136 Block RAMs

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Data Width</th>
<th>TPG Count (slices)</th>
<th>ORA Count (slices)</th>
</tr>
</thead>
<tbody>
<tr>
<td>March s2pf</td>
<td>D=16</td>
<td>49</td>
<td>N<em>2</em>D</td>
</tr>
<tr>
<td>March d2pf</td>
<td>D=16</td>
<td>76</td>
<td>N<em>2</em>D</td>
</tr>
<tr>
<td>March s2pf</td>
<td>D=36</td>
<td>64</td>
<td>N<em>2</em>D</td>
</tr>
<tr>
<td>March d2pf</td>
<td>D=36</td>
<td>113</td>
<td>N<em>2</em>D</td>
</tr>
</tbody>
</table>

\[ N = Number of Block RAMs \quad D = Data width \]
LUT RAM Testing

- Two LUTs/slice
  - Operational modes
    - 16 x 1 single-port – single LUT
      - Not tested
    - 16 x 2 single-port – two LUTs
      - Tested with March Y w/o BDS – 9 slices
    - 32 x 1 single-port – two LUTs
      - Tested with March Y w/o BDS – 10 slices
    - 16 x 1 dual-port – two LUTs
      - Tested with March DPR – 40 slices
  - Three BIST configurations
    - 16x1 single-port not needed
    - Two phases of testing required for each mode of testing
      - Half the LUTs for TPG and ORA logic
      - Remaining half form RUTs
    - Different ORA design to accommodate ORA logic in one slice
      - Diagnostic resolution limited to a slice instead of a LUT

ORA Design
BIST Implementation

- Boundary-scan interface used for BIST control
- Device and package independent I/O interface
- Devices used for testing are:
  - Spartan II 2S50 - 16 x 24 PLBs
  - Spartan II 2S200 – 28 x 42 PLBs
  - Virtex II pro 2vp30 – 80 x 46 PLBs

### Function of JTAG pins

<table>
<thead>
<tr>
<th>JTAG pin</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRCK1</td>
<td>CLK</td>
</tr>
<tr>
<td>SEL2</td>
<td>RESET</td>
</tr>
<tr>
<td>TDI</td>
<td>SHIFT</td>
</tr>
<tr>
<td>TDO1</td>
<td>SCANOUT</td>
</tr>
</tbody>
</table>

The VLSI Design & Test Seminar Series

March 23, 2005
Implementation Issues

- Increase in number of RAMs is not linear with the Array Size
Implementation Issues (cont...)

BIST circuitry did not fit it four devices
Implementation Issues (cont...)

- Solutions for testing smaller devices
  - Use a less complex *March* algorithm
    - Decreases fault detection capability
  - Increase fan-out limit to fit in some devices
    - Reduces speed of testing
  - Test RAMs in two phases with half the RAMs each stage
    - Increases test time considerably
  - Use ORAs similar to those used for LUT RAMs
    - Decreases diagnostic resolution
    - Best solution
Multiplier cores

- 18-bit multipliers in Virtex II pro and Spartan III devices
  - Associated with each Block RAM
  - Produces 36-bit output from two 18-bit inputs
  - Modes of operation: combinational and registered
  - Uses modified booth algorithm

- Three configurations
  - Combinational mode
  - Registered mode

- Tests clock enable and reset levels in two configurations
- VHDL approach used

### Multiplier BIST details

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Mode</th>
<th>TPG slices</th>
<th>ORA slices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count [10]</td>
<td>combinational</td>
<td>8</td>
<td>$N \times 36$</td>
</tr>
<tr>
<td>Modified count</td>
<td>registered</td>
<td>10</td>
<td>$N \times 36$</td>
</tr>
<tr>
<td>Modified count</td>
<td>registered</td>
<td>10</td>
<td>$N \times 36$</td>
</tr>
</tbody>
</table>

$N = Number of Multiplier Cores$
Conclusions

- Two approaches presented
  - VHDL approach
    - Portable
    - Reduces BIST development time
    - Used for testing embedded memories in two different devices
    - Similar approach can be used for testing other regular structure cores
  - Processor Approach
    - Reduces test-time significantly
    - Can be used in SoCs with partial-reconfiguration capability
  - A total of 18 BIST configurations developed for testing memory cores in Atmel and Xilinx devices
Observations

- Architecture of FPGA has significant impact on BIST approach
  - Fine-grained: better control over placement and routing required
  - Boundary-scan interface makes I/O interface device-independent
  - Read-back capability can save some test time
    - Xilinx FPGAs have read-back capability
      - Frame-level instead of Byte-level
- Processor BIST
  - Can result in better speed-up if number of BIST configurations increase
  - May not offer significant speed-up for smaller devices
Future Research

- Processor BIST for Virtex II pro SoCs
  - May result in better speed-up
  - Test Block RAMs with program stored in Block RAMs
- Extension of RAMBISTGEN for multi-port memories
- BIST for processors
  - Processor assumed to be fault-free
- VDHL approach for Logic and other resources
  - With support from synthesis tools
Publications