Center for
Reliable
Computing

SYNTHESIS OF MAPPING LOGIC FOR GENERATING TRANSFORMED PSEUDO-RANDOM PATTERNS FOR BIST

Nur A. Touba and Edward J. McCluskey

Center for Reliable Computing
Stanford University

ON-CHIP TEST PATTERN GENERATION

☞ **On-Chip Test Pattern Generation**

- **Requirement for BIST**
- **Provide High Fault Coverage**
- **Reasonable Test Length**

☞ **Linear Feedback Shift Register (LFSR)**

- **Simple Structure ⇒ Small Area**
- **Dual Purpose: Test Pattern Generator**
Output Response Analyzer
- **May Not Give High Enough Fault Coverage**

PROBLEM

Given: LFSR Doesn't Provide High Enough Fault Coverage

☞ **Choose Different Seed or Characteristic Polynomial**

- **Less than Factor 10 Reduction in Test Length**

☞ **Improve Detection Probabilities**

- **Insert Test Points or Redesign Circuit**

☞ **Augment LFSR with Additional Logic**

- **Improve Patterns Generated**

PREVIOUS APPROACHES

☞ **Mixed-Mode**

- **Use Deterministic Patterns to Detect Missed Faults**
- **Deterministic Patterns On-Chip ⇒ Large Overhead**

☞ **Multiple Seeds and/or Reconfigurable LFSR**

- **Logic to Reseed or Change Polynomial of LFSR**
- **Seeds/Polynomials Must Be Stored On-Chip**

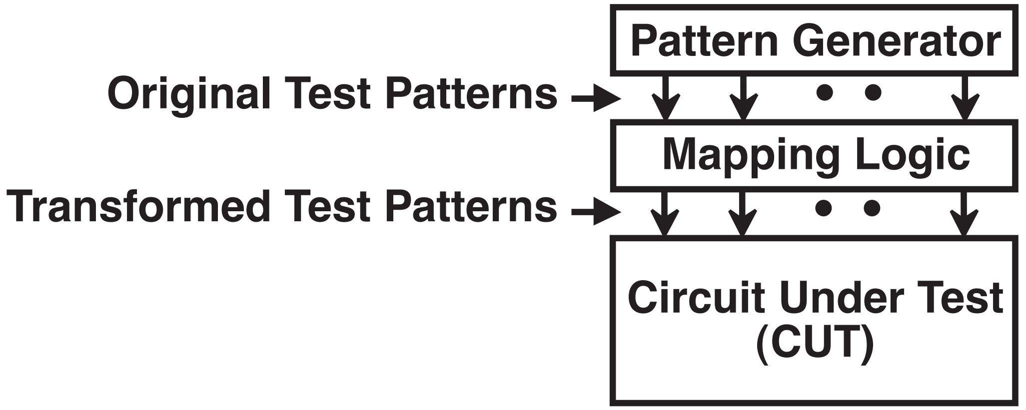
☞ **Weighted Patterns**

- **Bias Patterns to those that Detect Hard Faults**
- **Multiple Weight Sets Often Required**

TRANSFORMED PSEUDO-RANDOM PATTERNS

☞ **Transform Output of Pseudo-Random Pattern Generator**

- **Mapping Logic**
- **Combinational or Sequential**
- **Want to Minimize Overhead**



MAPPING LOGIC

☞ **Combinational Mapping Logic - Fixed Transformation**

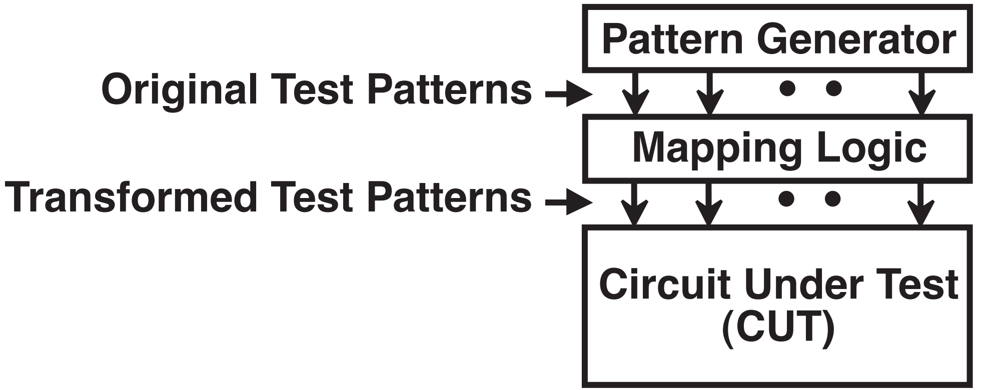
- **Single Weight Set**
- **Cube Mapping, Touba & McCluskey, 1995.**
- **Combinational Units, Chatterjee & Pradhan, 1995.**
- **LFSROM, Dufaza et al., 1995.**

☞ **Sequential Mapping Logic - Transformation Changes after some Num. of Patterns**

- **Multiple Weight Sets**
- **3-Weight Method, Pomeranz and Reddy, 1992.**
- **Fixed-Biased Method, AlShaibi and Kime, 1994.**

CHOOSING MAPPING FUNCTION

- ☞ **Many Functions Satisfy Fault Coverage Requirement**
- ☞ **Amount of Logic to Implement Each Function Varies**
- ☞ **Problem: Choose Function that Minimizes Area**



SPECIFYING MAPPING FUNCTION

GIVEN: Pattern Generating Circuit (LFSR Configuration)
Test Length and Fault Coverage Requirement

1) Simulate Pattern Generating Circuit for Test Length
   - Determine Original Pattern Set

2) Perform Fault Simulation
   - Keep Track of Patterns that Drop Faults

3) Do ATPG for Undetected Faults
   - Leave Unspecified Inputs as ‘X’ to Form Test Cubes

SPECIFYING MAPPING FUNCTION

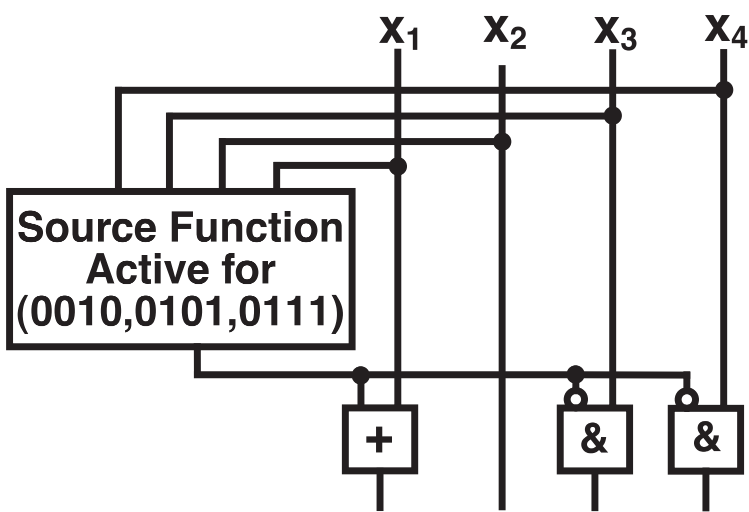
	Original Patterns $x_1 x_2 x_3 x_4 x_5 x_6$		Transformed Patterns $x_1 x_2 x_3 x_4 x_5 x_6$
Patterns That Drop Faults	0 1 0 0 1 1	→	0 1 0 0 1 1
	0 1 1 0 0 0	→	0 1 1 0 0 0
	1 0 1 1 0 1	→	1 0 1 1 0 1
	0 1 0 1 1 0	→	0 1 0 1 1 0
	1 0 1 1 0 1	→	1 0 1 1 0 1
Patterns Assigned to Test Cubes	? ? ? ? ? ?	→	0 0 1 0 X 1
	? ? ? ? ? ?	→	X 0 1 1 0 0
	? ? ? ? ? ?	→	1 X X 0 1 X
Unassigned Patterns	1 0 0 1 1 0	→	X X X X X X
	0 0 1 1 0 1	→	X X X X X X
	0 1 0 1 1 1	→	X X X X X X
	1 1 0 1 0 0	→	X X X X X X
	1 0 0 0 0 1	→	X X X X X X
	1 1 1 0 0 1	→	X X X X X X
	0 0 1 0 1 0	→	X X X X X X

BIT-FIXING

☞ **Example: 0010 mapped to 1000**

- **Fix First Bit to a 1**
- **Fix Third Bit to a 0**

Original Patterns x_1 x_2 x_3 x_4		Test Cubes x_1 x_2 x_3 x_4	B-Matrix x_1'	x_2'	x_3'	x_4'	x_1	x_2	x_3	x_4
1 1 1 1	→	0 0 1 1	1*	1*	0	0	0	0	1	1
0 0 1 0	→	X 0 0 0	1	1	1*	1	1	0	0	0
0 1 0 1	→	1 X X 0	0	1	1	1*	1*	1	1	0
0 1 1 1	→	1 1 0 X	0	0	1*	1	1*	1	0	1



MINIMIZING MAPPING FUNCTION

☞ **Assign Original Patterns to Test Cubes to Minimize Rectangle Cover of B-Matrix**

Original Patterns x_1 x_2 x_3 x_4		Test Cubes x_1 x_2 x_3 x_4	B-Matrix x_1'	x_2'	x_3'	x_4'	x_1	x_2	x_3	x_4
1 1 1 1	→	0 0 1 1	1*	1*	0	0	0	0	1	1
0 0 1 0	→	X 0 0 0	1	1	1*	1	1	0	0	0
0 1 0 1	→	1 X X 0	0	1	1	1*	1*	1	1	0
1 0 1 1	→	1 1 0 X	0	0	1*	1	1	1*	0	1

☞ **Replace 1011 with 0111**

Original Patterns x_1 x_2 x_3 x_4		Test Cubes x_1 x_2 x_3 x_4	B-Matrix x_1'	x_2'	x_3'	x_4'	x_1	x_2	x_3	x_4
1 1 1 1	→	0 0 1 1	1*	1*	0	0	0	0	1	1
0 0 1 0	→	X 0 0 0	1	1	1*	1	1	0	0	0
0 1 0 1	→	1 X X 0	0	1	1	1*	1*	1	1	0
0 1 1 1	→	1 1 0 X	0	0	1*	1	1*	1	0	1

PROCEDURE FOR SELECTING MAPPING FUNCTION

INPUT: Test Cubes for Undetected Faults
Original Patterns that Don't Drop Faults

1) Form B-Matrix
2) Initial Assignment of Original Patterns
   - Based on Minimizing Number of Bit Differences
3) Let Each Stared Entry in B-Matrix be in R
4) EXPAND(R)
5) IRREDUNDANT(R)
6) Reassign Original Patterns to Eliminate Rectangles in R
7) REDUCE(R)
8) If Size of R Decreased, Loop Back to Step 4

SYNTHESIZING BIT-FIXING LOGIC

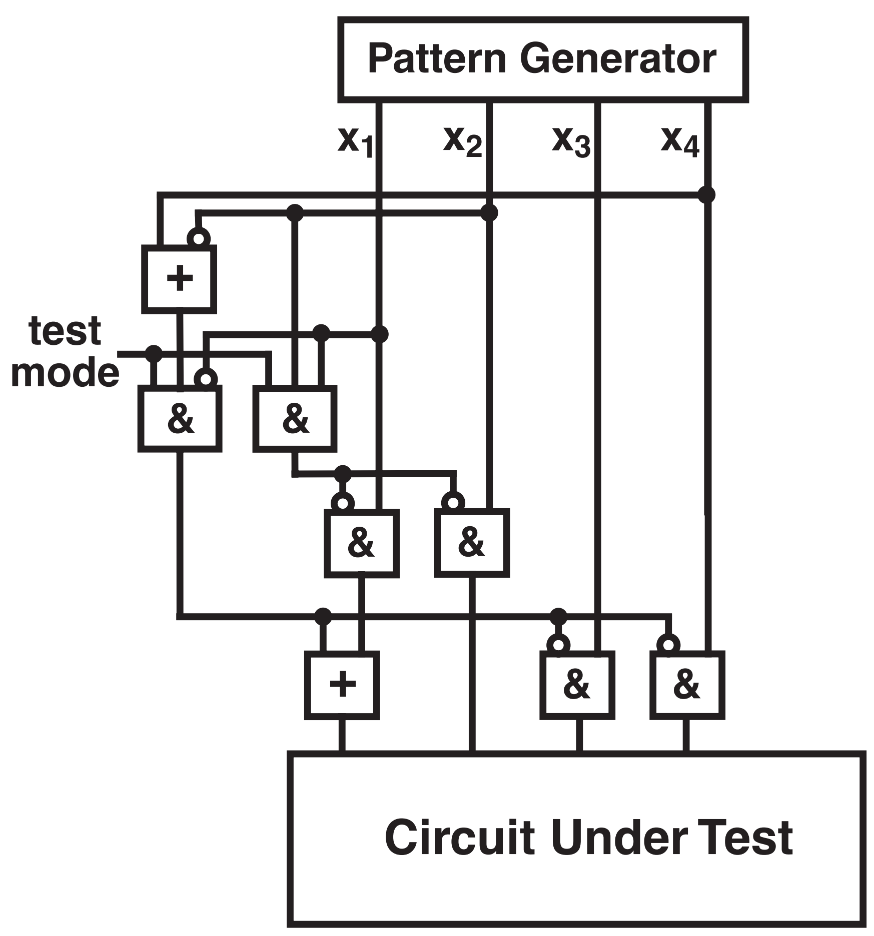
☞ **Determine Source Function for Each Rectangle in R**

- **On-Set = Assigned Patterns to Transform**
- **Off-Set = Patterns that Drop Faults**
  + Other Assigned Patterns

☞ **Use Logic Synthesis Tool to Generate Implementation**

EXAMPLE OF IMPLEMENTATION

- ☞ **Source Function 1:** $x_1 \; x_2$ **Fix Bits:** x_1', x_2'
- ☞ **Source Function 2:** $x_1' \, (x_2' + x_4)$ **Fix Bits:** x_1, x_3', x_4'



COMPARING WITH PRIOR METHODS

☞ **Three Important Factors for Comparing Methods:**

- **Test Length**
- **Fault Coverage**
- **Hardware Area: Flip-Flops plus Gate Equivalents**

☞ **Assume Parallel Application ("Test Per Clock")**

☞ **Gate Equivalents (GE's)**

- **(0.5)(n) GE's for n-input NAND or NOR**
- **(2.5)(n-1) GE's for n-input XOR**
- **1.5 GE's for 2-to-1 MUX**

MULTIPLE WEIGHT SETS

- ☞ **Weight Sets from [Bershteyn, ITC 93] Used**
- ☞ **Assume No Extra Stages Added to Avoid Correlation**
- ☞ **Assume 4 two-input NAND/NOR to Generate Weighted Signals for Each Input**
- ☞ **One 2-to-1 MUX per Input for Each Weight Set**

FF's = $\log_2$(Number of Weight Sets)

GE's = [4 + (1.5)(Number of Weight Sets)](Number of Inputs)

COMPARISON WITH WEIGHTED PATTERN TESTING

☞ **Weight Sets from [Bershteyn, ITC 93] Used**

Circuit Name	Random TLen	Multiple Weight Sets				Proposed Method		
		TLen	WS	FF	GE	TLen	FF	GE
s420	1.1M	532	4	≥2	350	500	0	37
		1.8K	2	≥1	245	1K	0	28
s641	1.0M	593	3	≥2	459	500	0	22
						10K	0	12
s838	>99M	893	5	≥3	770	850	0	86
		17K	2	≥1	469	10K	0	37
C2670	4.6M	1.3K	9	≥4	4078	1K	0	218
		12K	3	≥2	1981	7K	0	121
C7552	>99M	2K	12	≥4	4554	10K	0	286
		69K	5	≥3	2380	50K	0	139

3-WEIGHT METHOD

☞ **Proposed by Pomeranz and Reddy, 1992.**

☞ **While Random Patterns Applied**
- **3-Gate Modules Fix Certain Inputs**
- **Forms "Expanded Tests"**

☞ **Extra Flip-Flops Control Which Expanded Test Applied**

☞ **Logic for 3-Gate Modules Depends on Fan-in**

FF's = $\log_2$(Number of Expanded Tests)

GE's = (Number of 3-Gate Modules)(1 + Average Fan-In)

COMPARISON WITH 3-WEIGHT METHOD

☞ **Proposed by Pomeranz and Reddy, 1992.**

Circuit Name	Random TLen	3-Weight Method			Proposed Method		
		TLen	FF	GE	TLen	FF	GE
C2670	4.6M	19K	5	1507	1K	0	218
		30K	5	1316	7K	0	121
C7552	>99M	47K	6	3003	10K	0	186
		72K	6	2475	50K	0	139

FIXED-BIASED METHOD

☞ **Proposed by AlShaibi and Kime, 1994.**

☞ **Use Weighted Bit Stream and Fix Value of Some Bits**

☞ **Configuration Sequences Periodically Loaded from ROM**

- **Assume ROM is Off-Chip for Comparison Sake**

☞ **17-Stage LFSR used to Generate Weighted Bit Stream**

☞ **Each Fixed Bit Requires 1 FF, 4 MUXes, and a NAND**

FF's = 17 + Number of Fixed Bits
GE's = [(4)(1.5) + 1] (Number of Fixed Bits)

COMPARISON WITH FIXED-BIASED METHOD

☞ **Proposed by AlShaibi and Kime, 1994.**

Circuit Name	Random TLen	Fixed-Biased			Proposed Method		
		TLen	FF	GE	TLen	FF	GE
s420	1.1M	5K	18	>7	500 1K	0 0	37 28
s641	1.0M	19K	20	>21	500 10K	0 0	22 12
s838	>99M	86K	19	>14	850 10K	0 0	86 37
C2670	4.6M	19K	54	>259	1K 7K	0 0	218 121
C7552	>99M	191K	111	>658	10K 50K	0 0	186 139

CONCLUSIONS

☞ **Select Mapping Function**

- **Iterative Procedure Involving Global Operations**
- **Less Overhead than Other Methods**

☞ **Advantages:**

- **Easy to Insert into Existing Design**
- **Fully Compatible with BILBO Registers**
- **Easy to Trade Off: Test Time
Fault Coverage
Hardware Area**
- **No Additional Sequential Logic Required**
- **Very Simple Control (Only Test Mode Line)**