Random Pattern Testable Logic Synthesis

Chen-Huan Chiang and Sandeep K. Gupta
University of Southern California, Los Angeles, CA 90089-2562

Abstract

Previous procedures for synthesis of testable logic guarantee that all faults in the synthesized circuits are detectable. However, the detectability of many faults in these circuits can be very low leading to poor random pattern testability. A new procedure to perform logic synthesis that synthesizes random pattern testable multilevel circuits is proposed. Experimental results show that the circuits synthesized by the proposed procedure tstfx *are significantly more random pattern testable and smaller than those synthesized using its counterpart* fast_extract (fx) *in* SIS. *The proposed synthesis procedure designs circuits that require only simple random pattern generators in built-in self-test, thereby obviating the need for complex BIST circuitry.*

1 Introduction

Built-in self-test (BIST) provides a framework to reduce the effort otherwise required to generate extensive test sets for large circuits. It also reduces/eliminates the need for sophisticated test equipments. However, extra hardware test pattern generators (TPGs) are needed and some performance penalty is sustained for implementing BIST. Typically linear feedback shift registers (LFSRs) or cellular automata (CA) are used as pseudo-random TPGs.

Increasingly, logic blocks in commercial circuits are synthesized using logic synthesis tools. The traditional objectives of logic synthesis procedures are minimization of circuit area and delay. Due to the importance of testing, testability is being considered during logic synthesis as well. The definition of testability used by most testable logic synthesis procedures is that all faults of interest (e.g. all single stuck-at faults) are testable.

Unfortunately, while all faults in the circuits designed by these procedures are detectable, many faults in the resulting circuits have few tests. Therefore, very long pseudo-random sequences are required to achieve desired fault coverage in BIST. Hence, to achieve high fault coverage, in reasonable test time, complex BIST TPG designs are needed thereby increasing the area and performance overheads. The objective of the proposed research is to enhance the notion of testability to *random pattern testability*. A synthesis procedure that ensures not only all single stuck-at faults in the synthesized circuit are detectable, but also *random pattern testable*, is proposed. If BIST is used, then circuits implemented by the proposed synthesis procedure will require only simple BIST TPGs. Even for external testing, high random pattern testability of faults decreases ATPG effort and test set size.

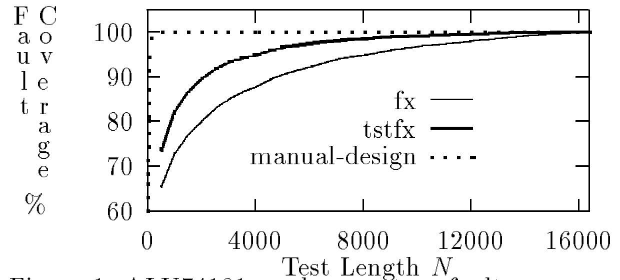


Figure 1: ALU74181 random pattern fault coverages.

2 Background

Motivation: It was observed that synthesized circuits required much longer random test sequences to achieve high fault coverage [9]. For example, the truth table of ALU 74181 (from the TTL handbook) was input to the testable synthesis procedure (*fx* [14] in *SIS* [6]) and synthesized version of the circuit was obtained. Figure 1 shows random pattern fault coverage as a function of test length for ALU 74181 (manual-design) and its automatically synthesized counterpart (fx). This and other similar results [9] indicate that circuits synthesized by typical synthesis procedures are not random pattern testable. Existence of random pattern testable implementations of the same Boolean functions shows that the presence of random pattern resistant faults is not due to the nature of the functions, but is a consequence of the structure of circuits generated by the synthesis procedure.

Review: Early research in area of testable logic design concentrated on the identification and elimination of redundancies from combinational circuits [3, 4]. It has been shown that all single stuck-at faults in any prime and irredundant two level realization of a circuit are fully testable. Procedure for synthesis of circuits in which all single and multiple stuck-at faults are testable has been reported in [7]. In recent years, testability preserving logic synthesis transformations have been studied. It is shown in [10] that algebraic transformations preserve testability of multiple stuck-at faults. Hence, if all multiple stuck-at faults in a given circuit are testable, then all multiple stuck-at faults in a circuit obtained by application of algebraic transformations to the circuit are also testable. It has been demonstrated in [11, 12] that careful assignment of don't cares of functions can improve random pattern testability of the circuit. In [14], a logic synthesis procedure based on concurrent extraction of single and double cube divisors and their compelments has

Permission to copy without fee all or part of this material is granted, provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

© 1994 ACM 0-89791-690-5/94/0011/0125 $3.50

been presented. The procedure has been proven to preserve the testability of single stuck-at faults. (This procedure is available as *fast_extract* (*fx*) within *SIS*.) Finally, a logic synthesis procedure to reduce deterministic test length has been presented in [8].

The objective of this research is to develop a logic synthesis procedure that enhances random pattern testability of the circuit while preserving testability of single stuck-at faults. An approach similar to ours (using algebraic transformations) has been independently developed [17].

Definitions and Notations: In the following, some definitions are presented. The reader is referred to [6, 14] for the formal definitions.

A literal is either an input or intermediate variable z_i or its compliment $\overline{z_i}$. A cube is a product (AND) of one or more literals. Each variable appears at most once in a cube either in complemented or uncomplemented form. An expression is a sum (OR) of one or more cubes. An expression E is said to be *cube-free* if the only cube that can divide E is 1. An expression g is called an algebraic divisor of expression f if $f = gq + r$ where q and r are expressions and $q \neq 0$. If divisor g is a single cube, then g is called a single cube divisor. If divisor g is a sum of two cubes, then g is said to be a double-cube divisor. Subsets of double-cube divisors are represented by $D_{x,y,s}$, where x ($\leq y$) is the number of literals in the first cube, y is the number of literals in the second cube and s is the total number of distinct variables that appear in the double cube. Subsets of single-cube divisors are denoted by S_k, where k is the number of literals in the single-cube divisor.

Detectability Profile: Random pattern testability is difficult to quantify in a manner that can be incorporated into logic synthesis procedures. However, the detectability profile can be used to guide logic synthesis. Detectability profile has been used to estimate random pattern test sequence length required to attain desired fault coverage for a given circuit [2, 13, 15, 18].

Given an n-input circuit, the number of tests for any fault in the circuit can vary between 0 (undetectable) and 2^n. Let h_k be the number of faults in a circuit which have exactly k tests. Then $H = \{h_0, h_1, \ldots, h_{2^n}\}$ is called the detectability profile of the circuit. In the following, some simple empirical observations [13, 15, 18] will be presented. They are needed because complex relationship between the detectability profile and random test sequence length can not be easily used to direct the logic synthesis.

1. The faults with fewest tests (k_{min}) are the hardest to detect. Empirically, the faults with $k_{min}, k_{min+1}, \ldots, 2 * k_{min}$ tests (called hard-to-detect faults) require long pseudorandom test sequences for their detection. Hence, logic synthesis procedure should improve the detectability of these faults.
2. Among the hard to detect faults, the lower the number of tests for a given fault, the higher is its impact on the random pattern test length. Hence, maximization of k_{min} is desirable.
3. If k_{min} cannot be increased, then $h_{k_{min}}$ should be minimized. If that cannot be achieved, then $h_{k_{min}+1}$ should be minimized, and so on.

These observations are used in the following to define a cost function that directs the proposed synthesis procedure.

3 Proposed Synthesis Procedure

Multilevel logic synthesis procedure can be described using three steps. First, a set of possible divisors is computed. Next, a cost which reflects the merit of selecting the given divisor for extraction is assigned to each divisor. Third, the candidate with the lowest cost is extracted. After each extraction (in some cases after a number of extractions), the set of possible divisors and their costs are recomputed and the procedure is repeated.

The above describes the basic outline of the proposed procedure as well. The candidates are single cube divisors with two literals and double cube divisors used in *fx* [14]. This decision was based on three main considerations. Firstly, a procedure based on these candidates is guaranteed to preserve testability of all single stuck-at faults [14]. This guarantees that the proposed procedure synthesizes circuits with no undetectable single stuck-at faults, provided all single stuck-at faults in the input circuits are testable. Secondly, as shall be seen in the following, the simplicity of these candidates allows comprehensive analysis of the impact of extraction of each divisor on the detectabilities of various faults in the modified circuit. Lastly, the number of candidates (objects of size two) is always in polynomial domain. Also great cost reduction, in terms of the number of literals, have been obtained by their use in traditional logic synthesis.

However, since the objective of the proposed algorithm is to generate logic that is random pattern testable, the cost assignment to the various single and double cubes is different from the cost that is used in *fx* (number of literals saved). In this section, we will concentrate on the description of cost that reflects improvement in random pattern testability resulting from the extraction of each single/double cube divisor.

Hard to Detect Faults: Based on the empirical observations discussed earlier, only the impact of various extractions on the hard to detect faults will be considered. Hence, if the extraction of a particular divisor only improves the detectabilities of faults that already have many tests, then the divisor is assigned a lower cost.

Let k_{min} be the number of tests for the stuck-at fault with the fewest test vectors. Let $k_2 > k_{min}$ be such that $h_{k_2} \neq 0$ and $h_{k_{min}+1} = \ldots = h_{k_2-1} = 0$. In this work, all faults that have $k_{min}, k_2, \ldots, k_p$ tests are classified as hard-to-detect faults ($HTDF$) where $k_p = const * (k_2 - k_{min})$.

Impact of Extraction on Detectabilities: Any extraction changes the structure of the circuit. A change in the structure of the circuit changes the test sets for various faults. Also, some lines in the circuit are eliminated while new lines are created. To compute the cost of a single or double cube divisor, it is essential to understand the exact impact of the extraction on the detectabilities of the (hard to detect) faults. In the following, the impact of single cube extraction on

testability of various faults will be discussed to illustrate the computation of the cost.

Consider the subcircuit shown in Figure 2 (a). Let ab be the single cube divisor being considered for extraction. A_i $(1 \leq i \leq l)$ are cubes of primary inputs and/or intermediate variables. Let $T(x_s)$ denote the test set for the fault line-x stuck-at-s, where $s \in \{0,1\}$. Figure 2 (b) shows the subcircuit after the extraction of the cube ab. The cube extraction eliminates lines $a^1, a^2, \ldots, a^l$ and $b^1, b^2, \ldots, b^l$. Instead, a and b are inputs to the *AND* gate implementing the extracted cube $x^0 = ab$. The output x^0 fans out $x^1, x^2, \ldots, x^l$ to the *AND* gates (which now have one less input than in the original circuit).

Consider the test sets of the faults in the circuit shown in Figure 2 (b). They can be expressed in terms of the test sets for the faults in the original circuit (Figure 2 (a)) as:

$$T(x_0^i) = T(a_0^i) = T(b_0^i)\ , \forall 0 \leq i \leq l. \quad (1)$$

The concepts of fault equivalence and dominance [1] have been used to derive all these results which are stated here without proof (see [9] for details). Hence, the detectability of the stuck-at-0 faults on lines x^i is the same as the detectability of the stuck-at-0 faults on lines $a^i(b^i)$ in the original circuit, for all $0 \leq i \leq l$. On the other hand, the stuck-at-1 fault in the newly added lines x^i has a larger test set than either a^i or b^i as shown by following relations:

$$T(x_1^i) \supset T(a_1^i)\ , \forall 0 \leq i \leq l,$$

and

$$T(x_1^i) \supset T(b_1^i)\ , \forall 0 \leq i \leq l.$$

Hence the test set of x^i stuck-at-1 is given by

$$T(x_1^i) \supset T(a_1^i) \cup T(b_1^i)\ , \forall 0 \leq i \leq l. \quad (2)$$

This implies that the stuck-at-1 faults at the newly created lines x^i $(0 \leq i \leq l)$ are easier to detect than stuck-at-1 faults in the lines a^i (and b^i) $(0 \leq i \leq l)$ in the original circuit. Furthermore, since $T(a_1^i) \cap T(b_1^i) = \phi$, $|T(x_1^i)| \geq |T(a_1^i)| + |T(b_1^i)|$ for all $0 \leq i \leq l$. The test sets of the remaining faults in this subcircuit do not change.

Let $S_{single_cube(c)}$ denote the set of s-a-0/1 faults whose detectabilities change due to the extraction of single cube $c = ab$. Note that a_0^i and b_0^i are replaced by x_0^i which has the same testability. However, two lines are replaced by one, thereby impacting the detectability profile. Also, a_1^i and b_1^i are replaced by x_1^i which is easier to detect. Hence, $S_{single_cube(c)} = \{a_s^i, b_s^i | 1 \leq i \leq l, s = 0, 1\}$ is the set of faults whose detectabilities are improved in one way or other by the extraction of single cube c. The faults that should be considered for the evaluation of the cost of the single cube extraction are given by $S_{single_cube(c)} \cap HTDF$. The cost associated with the extraction of single cube c is hence given by:

$$cost(c) = \sum_{f \in S_{single_cube(c)} \cap HTDF} weight(f),$$

where f is any single stuck-at fault and

$$weight(f) = k_p - |T(f)|.$$

Similarly cost functions for double-cube divisor in single output, multiple outputs, $D_{2,2,2}$ and $D_{2,2,3}$ can be computed [9].

The above describes the proposed procedure and the method to compute the costs for various single/double cube extractions. Computation (and recomputation) of single and double cube divisors can be carried out using the procedure used by *fx*. However, computation of $T(f)$ at the beginning and its recomputation after each extraction needs new procedures.

Computation of $T(f)$: The input to the proposed multilevel logic synthesis is a two-level fully testable *AND-OR* circuit. For two level circuits, $T(f)$ can be computed for all faults using the procedure outlined in [9]. The procedure can be easily applied to most *PLA* benchmark circuits [5]. However, after each extraction, the circuit structure and the detectabilities of some faults change. Cost must be recomputed for various candidates (the set of candidates is also recomputed). Computation of exact $|T(f)|$ values for faults in an arbitrary multilevel circuit is difficult. However, the values of $|T(f)|$ can be recomputed approximately using the results shown above. Consider the case of single cube extraction. Equations 1 and 2 can be used to compute lower bounds on the values of $|T(x_0^i)|$ and $|T(x_1^i)|$ $(0 \leq i \leq l)$. Similar equations have been derived for other divisors and can be used to compute lower bounds on $|T(f)|$ after each extraction. Note that after the first few extractions, $|T(f)|$ values are inaccurate (but they are lower bounds on the actual detectabilities) and may influence future extractions adversely. Procedures to accurately recompute $|T(f)|$ are being developed.

4 Experimental Results

The proposed procedure for random pattern testability directed single and double cube divisor extraction has been implemented as *tstfx* within *SIS 1.0*. As outlined above, the procedure is analogous to *fx*. The proposed cost function replaces literal savings as the divisor selection criterion. Algorithms to compute test set size for all single stuck-at faults have been implemented. Also, procedures to update test sets for stuck-at faults after each extraction have been implemented. In the following experiments, the inputs to *tstfx* and *fx* are two-level prime and irredundant implementations of the benchmark functions (obtained by using *espresso -do single_output*).

Table 1 compares the characteristics of the resulting circuits obtained by the application of these two procedures. (For better measure the merit of *fx*, we ran *simplify -m nocomp; resub -a; fx; eliminate 0* [16] on input circuits.) Surprisingly, the circuits obtained by application of *tstfx* have fewer literals in most cases. Table 2 compares the random-pattern testability of circuits obtained by using *tstfx* and *fx*. The random testability of the resulting circuits are obtained by using LFSR generated pseudo-random sequences. Data presented is the average of fault coverages for several different

Table 1: Area comparison of *tstfx* and *fx*.

CKT	I/O	Nodes		Lit(fac)			Levels	
		fx	tstfx	fx	tstfx	%imp	fx	tstfx
b10	15/11	197	121	670	554	17.31	10	11
b4	33/23	109	95	365	395	-8.22	8	13
c181	14/8	781	332	2533	1519	40.03	8	15
chkn	29/7	200	99	699	509	27.18	9	15
gary	15/11	220	126	728	612	15.93	11	11
in2	19/10	265	142	861	556	35.42	9	12
in4	32/20	370	190	1132	711	37.19	11	15
in5	24/14	133	107	442	373	15.61	10	13
in6	33/23	109	95	366	405	-10.65	8	13
in7	26/10	80	60	242	174	28.10	9	13
rckl	32/7	132	102	430	329	23.49	10	12
vg2	25/8	139	67	461	283	38.62	7	14
x1dn	27/6	148	148	440	449	-2.04	8	17

Table 2: Random pattern testability comparison.

CKT	2 level input k_{min}	Test Length At 99% Fcov			Final Data		
		fx	tstfx	%imp	Test Len	FCOV% fx	FCOV% tstfx
b10	2	11264	10240	9.09	2^{15}	22912*	22912*
b4	2^{18}	71552	40960	42.75	-	122176*	122176*
c181	1	13760	9470	31.18	2^{14}	16288*	16096*
chkn	32	-	-	-	2^{22}	95.30	97.60
gary	2	11776	11264	4.35	-	22912*	22912*
in2	64	10752	5632	47.62	2^{19}	31744*	17920*
in4	4096	686080	111616	83.73	2^{20}	99.57	99.96
in5	1024	17920	8192	54.28	2^{18}	99.73	38912
in6	2^{18}	71552	40960	42.75	-	122176*	122176*
in7	512	205632	20480	90.04	2^{18}	99.11	99.94
rckl	1	-	-	-	2^{22}	79.07	80.62
vg2	64	605696	263168	56.55	2^{20}	99.83	888576*
x1dn	96	-	743552	-	2^{20}	97.85	99.87

* indicates the test length at 100% stuck-at fault coverage.

sequences, generated using LFSRs with different feedback polynomials and seeds (typically 5-10 different sequences are used). Since some of the circuits have large number of inputs, pseudo random sequences with lengths only a fraction of exhaustive test length could be used due to the large number of simulations necessary. The data clearly shows that a large decrease in random pattern test length results for 99% fault coverage. The reduction in test length varied from 4% to 90%. In most cases, the random pattern test length was decreased by half. Considering the fact that this was accomplished with a simultaneous decrease in circuit area shows that this approach to random pattern testable logic synthesis is very effective. However, it should be noted that the circuits synthesized by *tstfx* have more levels than those synthesized by *fx*. The impact of this on circuit delay, and modifications to the procedure to avoid increase in circuit depth are being investigated.

Figure 1 clearly shows both the improvement obtained by the proposed *tstfx* on ALU74181 and further improvements that may be possible.

5 Conclusion

A logic synthesis procedure *tstfx* that synthesizes circuits that are random pattern testable has been presented. The procedure is analogous to *fast_extract* (*fx*) in *SIS*. The procedure analyzes the improvements in the detectabilities of hard to detect faults due to each possible extraction. The overall improvement in the detectability of hard to detect faults is used to direct the multilevel logic synthesis procedure. Experimental results indicate that the random pattern testability of circuits synthesized by the proposed *tstfx* are significantly higher than those synthesized by *fx*. In most examples the decrease in random pattern test length is accompanied by reduction in circuit size. Hence, the proposed logic synthesis procedure can greatly simplify BIST hardware design and improve fault coverage.

Currently, procedures to enhance the quality of the detectability estimates ($|T(f)|$) for faults in the circuit are being developed. Simplification procedures that preserve/enhance random pattern testability of the resulting circuits are being studied. They will enable iterative application of *tstfx* to further reduce circuit size and enhance its random pattern testability.

References

[1] M. Abramovici, M. A. Breuer, and A. D. Friedman. *Digital Systems Testing and Testable Design.* Computer Science Press, New York, N.Y., 1990.

[2] P. H. Bardell, W. H. McAnney, and J. Savir. *Built-In Test for VLSI: Pseudorandom Techniques.* John Wiley & Sons, 1987.

[3] K. A. Bartlett, R. K. Brayton, G. D. Hachtel, R. M. Jacoby, C. R. Morrison, R. Rudell, A. Sangiovanni-Vincentelli, and A. R. Wang. Multi-Level Logic Minimization using Implicit Don't Cares. *IEEE Trans. on CAD*, 7(6):732–740, June 1988.

[4] D. Brand. Redundancy and Don't Cares in Logic Synthesis. *IEEE Trans. on Computer*, C-32(10):947–952, October 1983.

[5] R. K. Brayton, G. D. Hachtel, C. McMullen, and A. Sangiovanni-Vincentelli. *Logic Minimization Algorithms for VLSI Synthesis.* Kluwer Academic Publishers, Boston, MA, 1984.

[6] R. K. Brayton, R. Rudell, A. Sangiovanni-Vincentelli, and A. R. Wang. MIS: A Mutiple-Level Logic Optimization System. *IEEE Trans. on CAD*, CAD-6(6):1063–1080, November 1987.

[7] R. Dandapani and S. Reddy. On the Design of Logic Networks with Redundancy and Testability Considerations. *IEEE Trans. on Computer*, C-23(11):1139–1149, November 1974.

[8] K. De and P. Banerjee. Logic Partition and Resynthesis for Testability. In *Proceedings IEEE International Test Conference*, pages 906–915, 1991.

[9] S. K. Gupta and C.-H. Chiang. Random Pattern Testable Logic Synthesis. Technical Report CENG 94-08, University of Southern California, 1994.

[10] G. Hachtel, R. Jacoby, K. Keutzer, and C. Morrison. On Properties of Algebraic Transformations and the Multifault Testability of Multilevel Logic. In *Proceedings IEEE International Conference on Computer-Aided Design*, pages 422–425, 1989.

[11] A. Krasniewski. Can Redundancy Enhance Testability. In *Proceedings IEEE International Test Conference*, pages 483–491, 1991.

[12] A. Krasniewski and A. Albicki. Random Testability of Redundant Circuit. In *Proceedings IEEE International Conference on Computer Design*, pages 424–427, 1991.

[13] A. Majumdar and S. Sastry. On the Distribution of Fault Coverage and Test Length in Random Testing of Combinational Circuit. In *Proceedings IEEE-ACM Design Automation Conference*, pages 341–346, 1992.

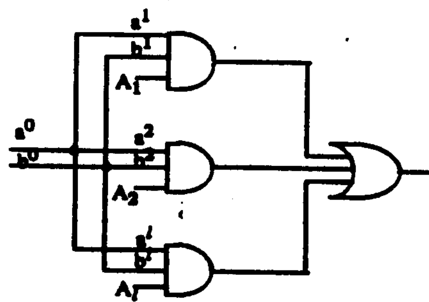
[14] J. Rajski and J. Vasudevamurthy. The Testability-Preserving Concurrent Decomposition and Factorization of Boolean Expressions. *IEEE Trans. on CAD*, 11(6):778–793, June 1992.

[15] J. Savir and P. H. Bardell. On Random Pattern Test Length. *IEEE Trans. on Computer*, C-33(6):467–474, June 1984.

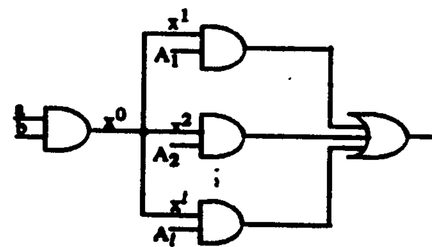
[16] H. Savoj. *Don't Cares in Multi-Level Network Optimization.* PhD thesis, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, 1992.

[17] N. A. Touba and E. J. McCluskey. Automated Logic Synthesis of Random Pattern Testable Circuits. In *Proceedings IEEE International Test Conference*, 1994.

[18] K. D. Wagner, C. K. Chin, and E. J. McCluskey. Pseudorandom Testing. *IEEE Trans. on Computer*, C-36(3):332–343, March 1987.



(a) single cube divisor *ab*



(b) circuit after extraction

Figure 2: Extraction of a single cube divisor *ab*.