

Using Hierarchy in Design Automation: The Fault Collapsing Problem

Raja K. K. R. Sandireddy¹ and Vishwani D. Agrawal²

Abstract

Although the problem of fault collapsing is not considered to be too complex, the time of collapsing faults in large circuits can be several hours or more. Large circuits are efficiently described using hierarchy, which significantly helps the architectural design, verification and physical design. We add fault collapsing to that list. We do not flatten the circuit and the collapsed fault sets computed once for sub-circuits are reused for all instances of those sub-circuits. The CPU time for collapsing faults in a flattened 128-bit array multiplier, which is about 8 hours, can be brought down to 40 seconds by using multiple levels of hierarchy. Additionally, by applying the exponential-complexity functional fault collapsing only to smaller sub-circuits, hierarchical collapsing in large circuits results in collapse ratios lower than those obtained with structural collapsing of flattened circuits. Using functional collapsing for a few small library cells, we hierarchically collapse faults in the 128-bit multiplier to sets of 480,757 equivalence and 265,824 dominance collapsed faults. In comparison, the flattened circuit collapses into 712,208 and 534,284 equivalence and dominance collapsed fault sets, respectively. We observe that the CPU time for fault collapsing for Boolean circuit by conventional programs grows as the square of the circuit size. A closer to linear time complexity can be expected for hierarchical fault collapsing.

Keywords (Index): CAD algorithms, fault collapsing, fault modeling, testing.

1. Introduction

Use of hierarchy allows us to solve complex automation problems like synthesis, verification and physical design. A static or structural analysis can easily benefit from hierarchy, while a dynamic analysis like simulation or test generation often flatten the hierarchy. Because fault collapsing is not dynamic, the use of hierarchy is proper choice. We show that benefits are two-fold, in reduced CPU time and in greater reduction in faults. Fault Collapsing is widely used to reduce the number of target faults for test generation and fault simulation. It has been shown [Goel (1980)] that the cost of test generation has higher than linear complexity as the number of faults increases. Hence, any technique reducing the size of the collapsed fault set would improve the performance of test development procedure for VLSI circuits. Structural fault collapsing is discussed

¹ Intel Corporation, Hillsboro, OR 97124, USA; Email: raja.sandireddy@intel.com.

² Auburn University, Dept. of ECE, 200 Broun Hall, Auburn University, AL 36849, USA; Email: vagraval@eng.auburn.edu.

in any text book on digital testing [Abramovici, Breuer and Friedman (1999), Bushnell and Agrawal (2000)].

Fault collapsing is classified into two types – equivalence collapsing and dominance collapsing. Two faults are called equivalent if and only if the corresponding faulty circuits have identical output functions [Bushnell and Agrawal (2000)]. Equivalent faults are indistinguishable because they cannot be distinguished at the primary outputs by any input vector. The set of all faults in a circuit can be partitioned into equivalence sets, such that all faults in a set are equivalent to each other. Equivalence collapsing is the process of partitioning faults into equivalence sets and selecting one fault from each set. The selected faults are called an equivalence collapsed set. Another form of collapsing that can further reduce the fault set size is dominance fault collapsing. If all tests of fault f1 detect another fault f2, then f2 is said to dominate f1. The two faults are also called “conditionally” equivalent with respect to the test set of f1 [Bushnell and Agrawal (2000)]. In dominance collapsing, all dominating faults in an equivalence collapsed set are left out retaining their respective dominated faults.

In Section 2, we discuss the background of fault collapsing. The hierarchical fault collapsing technique is described in Section 3. Results are discussed in Section 4. Recent publications [Sandireddy (2005), Sandireddy and Agrawal (2005a), Sandireddy and Agrawal (2005b)] describe related work.

2. Background

There has been considerable work in the area of fault collapsing. Several authors [Clegg (1973), Grüning, Mahlsdedt and Koopmeiners (1991), Hartanto, Boppana and Fuchs (1996), Lioy (1991), McCluskey and Clegg (1971), Shertz and Metzger (1972), Veneris, Chang, Abadir and M. Amiri (2004)] concentrate on finding the fault equivalences, while others [Agrawal, Prasad and Atre (2003), Lioy (1992), Prasad, Agrawal and Atre (2002)] deal with fault dominance relations. Recent papers also give methods to find fault equivalences using ATPG [Grüning, Mahlsdedt and Koopmeiners (1991), Veneris, Chang, Abadir and M. Amiri (2004)] and simulation [Al-Asaad and Lee (2002)]. Fault equivalence identification can be based on redundancy information [Amyeen, Fuchs, Pomeranz and Boppana (2001)], and hence test generation can prove equivalence [Hartanto, Boppana and Fuchs (1996)]. There are techniques to find the relations between the faults on a fan-out stem, its branches and the reconvergence points [Abramovici, Miller and Roy (1992), Lioy (1993), Nadjarbashi, Navabi and Movahedin (2000), To (1973), Vaaje (2006)]. It can be shown [Goundan (1978), Ibarra and Sahni (1975)] that identifying fault equivalence between two arbitrary faults in a combinational circuit is an NP-complete problem because it requires proving the equivalence of the two faulty functions. So, functional collapsing is not recommended for large circuits. Hence, most ATPG programs [Cheng and Chakraborty (1989), Kelsey, Saluja and Lee (1993), Lee and Ha (1993), Niermann and Patel (1991)] use only structural collapsing. Hierarchical fault collapsing seems to be an alternative that allows some functional collapsing. Hierarchical fault collapsing results in collapse ratios lower than those obtained by structural collapsing alone, but not as low as would be obtained if a complete functional collapsing was possible.

Collapse ratio is the fraction of total faults in the collapsed set [Bushnell and Agrawal (2000)].

The increasing complexity of the circuits can be efficiently handled using a hierarchical design process. The hierarchical description of a circuit at the highest level consists of an interconnection of few blocks, which are described at a lower level of hierarchy as interconnects of other blocks and gates [Wittmann, Seiss and Antreich (1993)]. A block or a sub-circuit C_j in a circuit C_i is called an instance of C_j . The circuit defined by hierarchical description can be obtained by an expansion process referred to as flattening. The hierarchy of a circuit description can be used in test generation [Lee and Patel (1996), Weening and Kerkhoff (1991), Zhongliang (2003)] and fault simulation [Kundu (2004), Motohara, Murakami, Urano, Masuda and Sugano (1988), Rogers and Abraham (1985)]. To the best of the authors' knowledge, there have been few papers [Agrawal, Prasad and Atre (2003), Hahn, Krieger and Becker (1994), Prasad, Agrawal and Atre (2002), Sandireddy and Agrawal (2005a), Sandireddy and Agrawal (2005b)] that have used hierarchy of the circuit to collapse faults. There are advantages of using hierarchy to collapse faults. A reduced fault set, computed once for a sub-circuit, can be reused for all instances of the sub-circuit. If functional fault collapsing techniques are used for smaller sub-circuits, we can achieve better collapse ratios.

Figure 1. Dominance graph of an OR gate.

Table 1. Dominance matrix of OR gate.

	a_0	a_1	b_0	b_1	c_0	c_1
a_0	1	0	0	0	1	0
a_1	0	1	0	1	0	1
b_0	0	0	1	0	1	0
b_1	0	1	0	1	0	1
c_0	0	0	0	0	1	0
c_1	0	1	0	1	0	1

2.1. Graph Model

We use a graph model described in the literature [Akers, Joseph and Krishnamurthy (1987), Prasad, Agrawal and Atre (2002)]. The fault equivalence and dominance relations are represented by a directed graph. In this graph each

fault is represented by a node. If fault f_1 dominates fault f_2 then this is represented by a directed edge from node f_2 to f_1 . This edge indicates that any test for f_2 must detect f_1 . Clearly, the presence of edges $f_1 \rightarrow f_2$ and $f_2 \rightarrow f_1$ indicates that the two faults f_1 and f_2 are equivalent. Fault dominance graph, or simply a dominance graph, represents the dominance relations among the faults of a circuit. Figure 1 shows the dominance graph for all faults of an OR gate. The subscript fault notation has been used, that is, a_0 means that the fault is on line named 'a' and is s-a-0. The dominance graph is conveniently represented by its dominance matrix shown in Table 1. A 1 entry at the intersection of a row and a column means that the fault corresponding to the column dominates the fault corresponding to the row. For example, the 1 in the second row and the last column indicates that c_1 dominates a_1 . Equivalence of two faults is expressed by two 1's placed at both intersections of the rows and columns of those faults. Since there is also a 1 in the last row and second column indicating that a_1 dominates c_1 , it can be said that a_1 and c_1 are equivalent. This dominance matrix is used in hierarchical fault collapsing technique to represent all the dominance relations between the faults.

3. Hierarchical Fault Collapsing

Typically, faults in a hierarchically described circuit are collapsed after flattening the hierarchy [Cheng and Chakraborty (1989)]. In our method, we do not flatten the circuit to collapse the faults. Hierarchical fault collapsing is based on the following theorem [Goundan and Hayes (1980), Hahn, Krieger and Becker (1994)]:

Theorem: If two faults are functionally equivalent in a combinational sub-circuit M that is embedded in a circuit N , then they are also functionally equivalent in N . ■

Functional equivalence here means diagnostic equivalence, as defined in a recent paper [Sandireddy and Agrawal (2005a)]. Such faults are called intrinsically equivalent in M [Goundan and Hayes (1980)]. This is demonstrated using the circuit in Figure 2, where the s-a-1 faults on lines x and y are intrinsically equivalent in M . There is another kind of equivalence, called extrinsic equivalence [Goundan and Hayes (1980)]. If two faults f_i and f_j located in a sub-circuit, M , are equivalent in the circuit N , but are not equivalent in the sub-circuit, then they are called extrinsically equivalent in M . In Figure 2, the s-a-1 faults on lines p and s are not equivalent in sub-circuit M but become equivalent in the complete circuit. The extrinsically equivalent fault pairs form a subset of functionally equivalent fault pairs. Such relationships are not detected by hierarchical fault collapsing.

The collapsing starts at the topmost level of hierarchy. Structural equivalence collapsing is done at this level using a line oriented structural fault collapsing technique as explained by Nadjarbashi *et al.* [Nadjarbashi, Navabi and Movahedin (2000)]. Next, the dominance relations between the faults in the equivalence collapsed set are found using the algorithm in the next section. For every s-a-0 on any input of an OR or NOR gate, the fault among the equivalent set that dominates it is found by setting b as 0 in the algorithm. Then a 1 is introduced in the dominance matrix to indicate this dominance using the update algorithm [Dave (2004), Dave, Agrawal and Bushnell (2005)], which computes the transitive closure of the dominance matrix. A similar procedure is followed for s-a-1 faults on the inputs of AND and NAND gates.

3.1. Algorithm to Find the Dominance Matrix and its Transitive Closure

1. Consider a stuck-at-b fault (f1) in the equivalence collapsed set at the input of a Boolean gate.
2. If the gate is of inverting type (NOT, NOR, NAND), then invert b.
3. If the equivalent set has s-a-b on this gate output, say f2, then place a 1 at the intersection of the row corresponding to f1 and column corresponding to f2 and use update [Dave (2004), Dave, Agrawal and Bushnell (2005)] for computing transitive closure. Go to step 1 until all faults in the equivalence collapsed set that are on the inputs of Boolean gates are considered.
4. Move one gate forward toward the primary output and go to step 2.

Figure 2. A circuit demonstrating extrinsic equivalence.

Next, the processing of instances of the sub-circuits is done. If the sub-circuit has a library file containing its collapsed information, that file is used to introduce new faults and relations into the dominance matrix. If there is no such library file, a library file is created dynamically as follows. If the sub-circuit does not have any user-defined blocks, namely, instances of other sub-circuits, in its description and has less than a pre-specified number of gates, say 15, we can use the functional collapsing algorithms [Sandireddy and Agrawal (2005a)]. Otherwise, the sub-circuit is processed recursively in the same way as the top level description is processed. This continues until all sub-circuits are processed. The collapsed fault set description is written to a library file for future use.

To obtain the equivalence collapsed set, the fault set is processed by algorithm equivalence [Prasad, Agrawal and Atre (2002)]. This step removes all extra faults that were added to assist the hierarchical fault collapsing. If dominance collapsing is required, then algorithm dominance [Prasad, Agrawal and Atre (2002)] is used.

4. Results

We will demonstrate the hierarchical method of collapsing by two example cases, namely, ripple-carry adders and array-multipliers. A functionally collapsed cell library is used. In each case, collapsing is done two ways: (1) Flat – Structure is flattened to gate level and faults are structurally collapsed in the traditional way; (2) Hierarchical – Design hierarchy is followed from the lowest level (level 0) cells for which functional collapsing results are available from the library data. For each sub-circuits containing only library cells (level 1), faults are collapsed using the transitive closure algorithms discussed in the

previous section. The collapse data thus generated is added to the library to be used in case a level 1 sub-circuit is used at higher levels. Two things should be mentioned:

- (a) Although collapsing in sub-circuits is structural, it contains the benefit of functional collapsing at the cell level. Hence, the collapsed sets are smaller than those obtained from a completely flattened sub-circuit.
- (b) When a sub-circuit (level > 0) is added to the library, its dominance matrix contains two types of faults, i.e., faults in the collapsed set, and the input-output (I/O) faults that are not included in the collapsed set. These I/O faults provide connectivity in the next level dominance matrix and are eliminated during collapsing at that level.

4.1. Cell Library

This cell library contains XOR (made of four NAND gates), HA (half-adder made of XOR and AND), and FA (full adder made of HA and OR). Each cell is flattened and complete functional collapsing is done to generate its collapsed dominance matrix. The method used is ATPG based [Sandireddy and Agrawal (2005a)]. The cell library data is summarized in Table 2. Logic gates are n-input Boolean gates. For these structural and functional collapsing are identical. For these library circuits, CPU times for structural collapsing are negligible and those for functional collapsing using the Hitec ATPG program [Niermann and Patel (1991)] on a Sun Ultrasparc 5_10 (360MHz, 128MB) are shown in Table 2. Although not evident from these small circuits, these times increase rapidly as the circuit becomes larger. The collapsed set sizes are minimal and do not have the extra I/O faults included in the dominance matrix, as mentioned earlier.

Table 2. Cell Library used for hierarchical fault collapsing.

Cell name	Cell characteristics				Collapsed fault set sizes				Func coll. CPU (s)
	No. of inputs	No. of outputs	No. of gates	Total faults	Structural		Functional		
					Equ	Dom	Equ	Dom	
Logic gate	n	1	1	2n	n+2	n+1	n+2	n+1	-
XOR	2	1	4	24	16	13	10	4	7.9
HA	2	2	5	30	20	16	15	6	9.1
FA	3	2	11	60	38	30	26	12	15.7

4.2. Ripple-Carry Adders

Details of the results in this subsection can be found elsewhere [Sandireddy (2005), Sandireddy and Agrawal (2005b)]. A two-bit adder sub-circuit is first built using two 1-bit adder cells from the library. This sub-circuit is added to the library. Then, a hierarchy of four blocks per level is used to construct larger adders. We constructed a 4-bit adder, i.e., 4-bit by 4-bit adder, by connecting four one-bit adder cells. An 8-bit adder was constructed using four 2-bit adder sub-circuits. A 16-bit adder then was built using four 4-bit adder sub-circuits. A 64-bit adder was built using four 16-bit adder sub-circuits.

Table 3 shows the result. As defined in Section 2, collapse ratios are the fraction of total faults in the collapsed sets [Bushnell and Agrawal (2000)]. Collapsing in all cases is done by the same program implementing dominance matrix and transitive closure techniques [Prasad, Agrawal and Atre (2002)]. In the first case the hierarchy is flattened to the gate level and the collapse ratios are exactly the same as obtained by any traditional structural collapsing program. The CPU time is recorded on a Sun Ultrasparc 5_10 (360MHz, 128MB) computer. For hierarchical collapsing, the lowest level sub-circuits use library data and their collapsed dominance matrices are saved in library. Next level sub-circuits are then processed and the matrices are saved in library. The process continues until the top level dominance matrix is generated. The final collapsing then produced the collapse ratios shown in Table 3. The CPU times, for the same computer, include the effort of creating all intermediate level sub-circuit data for library.

Table 3. Fault collapsing of hierarchical ripple-carry adders.

Addersize in bits	No. of logic gates	Total faults	Flattened circuit collapsing			Hierarchical collapsing		
			Collapse ratio		CPU (s)	Collapse ratio		CPU (s)
			Equiv.	Dom.		Equiv.	Dom.	
4 × 4	44	234	0.62	0.49	0.02	0.42	0.21	0.01
8 × 8	88	466	0.62	0.49	0.03	0.42	0.21	0.02
16 × 16	176	930	0.62	0.48	0.04	0.42	0.21	0.05
32 × 32	352	1858	0.62	0.48	0.10	0.41	0.21	0.07
64 × 64	704	3714	0.62	0.48	0.24	0.41	0.21	0.10
128 × 128	1408	7426	0.62	0.48	0.75	0.41	0.21	0.24
256 × 256	2816	14850	0.62	0.48	2.49	0.41	0.21	0.49
512 × 512	5632	29698	0.62	0.48	9.38	0.41	0.21	1.05
1024 × 1024	11264	59394	0.62	0.48	39.9	0.41	0.21	2.31
2048 × 2048	22528	118786	0.62	0.48	166.4	0.41	0.21	4.80
4096 × 4096	45056	237570	0.62	0.48	674.1	0.41	0.21	16.6
8192 × 8192	90112	475138	0.62	0.48	2676.0	0.41	0.21	55.0

We observe that the functional collapsing in the small library cells produced significantly reduced collapse ratios for the hierarchical approach. The CPU time of hierarchical collapsing, although much lower than that of the flat collapsing, still has quadratic growth with gates, the same as for the flat collapsing. As discussed elsewhere [Sandireddy (2005), Sandireddy and Agrawal (2005b)], the circuit connectivity structure processing dominates the complexity of both algorithms. The flat algorithm complexity is quadratic in the number of gates and that of hierarchical algorithm with re-used sub-circuits is quadratic in the number of I/O signals. For ripple-carry adders, the numbers of gates and those of I/O terminals are both proportional to the size of the adder.

4.3. Multipliers

We constructed large integer multipliers using a divide and conquer technique [Parhami (2000)]. For an even N, the N-bit Integer arguments, A and B, are separated into N/2 high-end bits, A_H and B_H , and N/2 low-end bits, A_L and B_L .

Four $N/2 \times N/2$ multipliers generate, $a_0 = A_H \times B_H$, $a_1 = A_H \times B_L$, $a_2 = A_L \times B_H$, and $a_3 = A_L \times B_L$. The final product is then generated by bit shifting and adders, as $A \times B = 2^n a_0 + 2^{n/2}(a_1 + a_2) + a_3$. Initially, a 2×2 multiplier is constructed just using 14 Boolean gates. Larger multipliers were then built hierarchically using four smaller multipliers and HA and FA cells. Although the faults in the 14-gate 2×2 multiplier can be functionally collapsed and it can be included in the functional cell library, that was not done for the present results, which were obtained using the cell library of Table 1.

The results for multipliers are shown in Table 4. We again observe that the use of the functional collapsing for the library cells produces more collapsing. Had we included the functional collapsing for the 2×2 multiplier in the cell library, the collapse ratios would be even lower. CPU times in Table 4 are for Sun Ultrasparc 5_10 (360MHz, 128MB). With increasing size of the multiplier, the number of gates grows quadratically but the number of I/O signals grows linearly. For larger multipliers in Table 4, as the size doubles, gates quadruple, and the time for flat collapsing increases about 16-fold. The time for hierarchical collapsing increases only four times; hierarchy offers greater advantage for deeper circuits. This observation has been explained by an analysis using the Rent's rule [Sandireddy (2005), Sandireddy and Agrawal (2005b)].

Table 4. Fault collapsing of hierarchical multipliers.

Multiplier size in bits	No. of logic gates	Total faults	Flattened circuit collapsing			Hierarchical collapsing		
			Collapse ratio		CPU (s)	Collapse ratio		CPU (s)
			Equiv.	Dom.		Equiv.	Dom.	
2×2	14	84	0.67	0.52	0.01	0.55	0.33	0.01
4×4	139	726	0.62	0.47	0.02	0.44	0.25	0.05
8×8	737	3762	0.61	0.46	0.31	0.42	0.23	0.15
16×16	3325	16842	0.61	0.46	4.60	0.41	0.23	0.59
32×32	14069	71034	0.60	0.45	98.57	0.41	0.23	2.28
64×64	57829	291546	0.60	0.45	1679.17	0.41	0.23	9.25
128×128	234437	1181082	0.60	0.45	27645.4	0.41	0.23	40.28

5. Conclusion

We have shown that the benefit of smaller collapse ratios achieved by functional collapsing for small sub-circuits is passed on to the larger hierarchical circuits. A smaller collapse ratio may reduce the fault simulation effort and also the number of test vectors. The time for collapsing a hierarchical circuit is less than that for the corresponding flat circuit. The time for flat circuits is proportional to the square of the circuit size, while for hierarchical circuits, the time for collapsing could be reduced closer to linear complexity. It has been observed that as the number of levels of hierarchy increase, the time for collapsing decreases [Sandireddy and Agrawal (2005b)]. Care is needed in using dominance collapsed set since there can be instances where the dominated fault is redundant and the dominating fault (not included in the collapsed set) is testable. For fault diagnosis, we may only use the equivalence collapsed set.

References

- [1] M. Abramovici, M. A. Breuer and A. D. Friedman (1990), *Digital Systems Testing and Testable Design*. Piscataway, New Jersey: IEEE Press, 1990.
- [2] M. Abramovici, D. T. Miller and R. K. Roy (1992), "Dynamic Redundancy Identification in Automatic Test Generation," *IEEE Trans. Computer-Aided Design*, vol. 11, no. 3, pp. 404–407, Mar. 1992.
- [3] V. D. Agrawal, A. V. S. S. Prasad and M. V. Atre (2003), "Fault Collapsing via Functional Dominance," *Proc. International Test Conf.*, 2003, pp. 274–280.
- [4] S. B. Akers, C. Joseph and B. Krishnamurthy (1987), "On the Role of Independent Fault Sets in the Generation of Minimal Test Sets," *Proc. International Test Conf.*, 1987, pp. 1100–1107.
- [5] H. Al-Asaad and R. Lee (2002), "Simulation-Based Approximate Global Fault Collapsing," *Proc. International Conf. on VLSI*, 2002, pp. 72–77.
- [6] M. E. Amyeen, W. K. Fuchs, I. Pomeranz and V. Boppana (2001), "Fault Equivalence Identification using Redundancy Information and Static and Dynamic Extraction," *Proc. 19th IEEE VLSI Test Symp.*, 2001, pp. 124–130.
- [7] M. L. Bushnell and V. D. Agrawal (2000), *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. Boston: Springer, 2000.
- [8] W. T. Cheng and T. J. Chakraborty (1989), "Gentest: An Automatic Test Generation System for Sequential Circuits," *Computer*, vol. 22, no. 4, pp. 43–49, April 1989.
- [9] F. W. Clegg (1973), "Use of SPOOF's in the Analysis of Faulty Logic Networks," *IEEE Trans. Computers*, vol. 22, no. 3, pp. 229–234, Mar. 1973.
- [10] K. K. Dave (2004), "Using Contrapositive Rule to Enhance the Implication Graphs of Logic Circuits," Master's thesis, Rutgers University, New Brunswick, May 2004.
- [11] K. K. Dave, V. D. Agrawal and M. L. Bushnell (2005), "Using Contrapositive Law in an Implication Graph to Identify Logic Redundancies," *Proc. 18th International Conf. VLSI Design*, Jan. 2005, pp. 723–729.
- [12] P. Goel (1980), "Test Generation Costs Analysis and Projections," *Proc. 17th IEEE/ACM Design Automation Conf.*, June 1980, pp. 77–84.
- [13] A. Goundan (1978), "Fault Equivalence in Logic Networks," PhD thesis, University of Southern California, Los Angeles, Feb. 1978.
- [14] A. Goundan and J. P. Hayes (1980), "Identification of Equivalent Faults in Logic Networks," *IEEE Trans. Computers*, vol. C-29, no. 11, pp. 978–985, Nov. 1980.
- [15] T. Grüning, U. Mahlsdedt and H. Koopmeiners (1991), "DIATEST: A Fast Diagnostic Test Pattern Generator for Combinational Circuits," *Proc. International Conf. Computer-Aided Design*, Nov. 1991, pp. 194–197.
- [16] R. Hahn, R. Krieger and B. Becker (1994), "A Hierarchical Approach to Fault Collapsing," *Proc. European Design & Test Conf.*, 1994, pp. 171–176.
- [17] I. Hartanto, V. Boppana and W. K. Fuchs (1996), "Diagnostic Fault Equivalence Identification Using Redundancy Information and Structural Analysis," *Proc. International Test Conf.*, Oct. 1996, pp. 294–302.
- [18] O. H. Ibarra and S. K. Sahni (1975), "Polynomially Complete Fault Detection Problems," *IEEE Trans. Computers*, vol. C-24, pp. 242–247, Mar. 1975.
- [19] T. P. Kelsey, K. K. Saluja and S. Y. Lee (1993), "An Efficient Algorithm for Sequential Circuit Test Generation," *IEEE Trans. Computers*, vol. 42, no. 11, pp. 1361–1371, Nov. 1993.
- [20] S. Kundu (2004), "Pitfalls of Hierarchical Fault Simulation," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 2, pp. 312–314, Feb. 2004.
- [21] H. K. Lee and D. S. Ha (1993), "On the Generation of Test Patterns for Combinational Circuits," Technical Report 12 93, Dept. of Electrical Eng., Virginia Polytechnic Institute and State University, 1993.

- [22] J. Lee and J. H. Patel (1996) "Hierarchical Test Generation under Architectural Level Functional Constraints," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 9, pp. 1144–1151, Sept. 1996.
- [23] A. Lioy (1991), "Looking for Functional Fault Equivalence," *Proc. International Test Conf.*, Oct. 1991, pp. 858–863.
- [24] A. Lioy (1992), "Advanced Fault Collapsing," *IEEE Design & Test of Computers*, vol. 9, no. 1, pp. 64–71, Mar. 1992.
- [25] A. Lioy (1993), "On the Equivalence of Fanout-Point Faults," *IEEE Trans. Computers*, vol. 42, no. 3, pp. 268–271, Mar. 1993.
- [26] E. J. McCluskey and F. W. Clegg (1971), "Fault Equivalence in Combinational Logic Networks," *IEEE Trans. Computers*, vol. C-20, no. 11, pp. 1286–1293, Nov. 1971.
- [27] A. Motohara, M. Murakami, M. Urano, Y. Masuda and M. Sugano (1988), "An approach to fast hierarchical fault simulation," *Proc. 25th Design Automation Conference*, 1988, pp. 698–703.
- [28] M. Nadjarbashi, Z. Navabi and M. R. Movahedin (2000), "Line Oriented Structural Equivalence Fault Collapsing," *Proc. IEEE Workshop on Model and Test*, 2000.
- [29] T. M. Niermann and J. H. Patel (1991), "HITEC: A Test Generation Package for Sequential Circuits," *Proc. European Design Automation Conference*, Feb. 1991, pp. 214–218.
- [30] B. Parhami (2000), *Computer Arithmetic: Algorithms and Hardware Designs*, New York: Oxford University Press, 2000.
- [31] A. V. S. S. Prasad, V. D. Agrawal and M. V. Atre (2002), "A New Algorithm for Global Fault Collapsing into Equivalence and Dominance Sets," *Proc. International Test Conf.*, Oct. 2002, pp. 391–397.
- [32] W. A. Rogers and J. A. Abraham (1985), "CHIEFS: A Concurrent, Hierarchical and Extensible Fault Simulator," *Proc. International Test Conf.*, March 1985, pp. 710–716.
- [33] R. K. K. R. Sandireddy (2005), "Hierarchical Fault Collapsing for Logic Circuits," Master's thesis, Auburn University, Auburn, AL, May 2005.
- [34] R. K. K. R. Sandireddy and V. D. Agrawal (2005a), "Diagnostic and Detection Fault Collapsing for Multiple Output Circuits," *Proc. Design, Automation and Test in Europe Conf.*, March 2005, pp. 1014–1019.
- [35] R. K. K. R. Sandireddy and V. D. Agrawal (2005b), "Use of Hierarchy in Fault Collapsing," in 14th IEEE North Atlantic Test Workshop, May 2005.
- [36] D. R. Schertz and G. Metze (1972), "A New Representation for Faults in Combinational Digital Circuits," *IEEE Trans. Computers*, vol. C-21, no. 8, pp. 858–866, Aug. 1972.
- [37] K. To (1973), "Fault Folding for Irredundant and Redundant Combinational Circuits," *IEEE Trans. Computers*, vol. C-22, no. 11, pp. 1008–1015, Nov. 1973.
- [38] A. Vaaje (2006), "Theorems for Fault Collapsing in Combinational Circuits," *Journal of Electronic Testing: Theory and Applications*, vol. 22, no. 1, pp. 23–36, Feb. 2006.
- [39] A. Veneris, R. Chang, M. S. Abadir and M. Amiri (2004), "Fault Equivalence and Diagnostic Test Generation Using ATPG," *Proc. IEEE International Symposium on Circuits and Systems*, May 2004, pp. 221–224.
- [40] E. C. Weening and H. G. Kerkhoff (1991), "A New Hierarchical Approach to Test-Pattern Generation," *Proc. 4th IEEE International ASIC Conference and Exhibit*, Sept. 1991, pp. P6–1/1–4.
- [41] H. C. Wittmann, B. H. Seiss and K. J. Antreich (1993), "Using Circuit Hierarchy for Fault Simulation in Combinational and Sequential Circuits," *Proc. 4th European Conference on Design Automation*, Feb. 1993, pp. 432–436.
- [42] P. Zhongliang (2003), "Hierarchical Test Generation using Neural Networks for Digital Circuits," *Proc. International Conference on Neural Networks and Signal Processing*, Dec. 2003, pp. 245–248.