VLSI DESIGN
1996, Vol. 5, No. 1, pp. 49–57
Reprints available directly from the publisher
Photocopying permitted by license only

Improving Path Sensitizability of Combinational Circuits*

BHANU KAPOOR[a,†] and V. S. S. NAIR[b,‡]

[a]*Integrated Systems Laboratory, Texas Instruments Incorporated, P. O. Box 655474, MS 446, Dallas, TX 75265;* [b]*Dept. of Computer Science and Engg., Southern Methodist University, Dallas, TX 75275*

(Received 30 May 1994; In final form 14 October 1994)

The problem of modifying a synthesized circuit to improve its path sensitizability has been investigated. It is shown that a large number of paths, that cannot be sensitized using single-transition tests, are redundant paths and they can be removed by appropriate modification of the circuit. The effect of these modifications on area and performance of the circuit has been analyzed. For the paths which are neither redundant nor sensitizable using single-transition tests, it is shown that they can be sensitized using multiple-transition tests. Results obtained on some common benchmark examples suggest the validity and viability of this approach.

Keywords: Path sensitization, timing analysis, functional analysis, delay testing, ordered binary decision diagrams, logic synthesis

1. INTRODUCTION

Functional analysis of paths in combinational logic circuits has recently emerged as the key issue in circuit timing and testability analysis. In these problems, the question of interest is whether a path or a set of paths is capable of propagating a transition. Certain paths in a combinational circuit may never transmit a transition because of the logical functions computed by the gates and their propagation delays preclude these paths from being sensitized.

Circuits with sensitizable paths are attractive for many reasons: First, if a path is sensitizable then the delay of the path, as determined by a static timing analyzer [1], is also its true delay. This obviates any need for timing analyzers based on functional analysis [2]–[11]. Second, if a path is sensitizable then all the lines on the path are stuck-at testable. Further, such a path is also delay testable except under the conditions which result in a hazard on the path [12,13,18].

Given the usefulness of sensitizable paths, a natural question to ask is, "What are the penalties associ-

*Based on "Area, Performance, and Sensitizable Paths" by B. Kapoor and V. S. S. Nair which appeared in the *Proceedings of The Fourth Great Lakes Symposium on VLSI*, pp. 222–227, Mar. 1994. ©IEEE

†Phone: 214-995-7305. Fax: 214-995-6194. E-mail: kapoor@hc.ti.com.

‡E-mail: nair@seas.smu.edu

ated with the design of a circuit with sensitizable paths?" In this paper, we will investigate the cost of modifying a circuit to improve its overall path sensitizability, the cost being the change in area and delay of a circuit.

The relationship between area optimization and testability was explored in [19]. It was shown that redundancy removal, which is used as the first step in the area optimization algorithms, also improves stuck-at testability. The relationship between area optimization and multifault testability has been explored in [20]. It has been shown that algebraic restructuring techniques used in logic optimization retain the property of multifault testability. It is possible to improve single-fault testability as a result of these transformations.

While the relationship between area and testability has been known for some time, the relationship between performance and testability has been explored only recently [14,15]. Synthesis for robust path-delay-fault testability, a related topic, has been investigated [21]–[30] by several researchers. Robust testability is stronger a criterion than path sensitizability. However, robust tests do not exist for a large number of paths in most practical circuits [13]. Also, synthesis for complete robust testability can result in a large penalty in terms of area and performance of the circuit.

To enable the study of relationship between area, performance, and path sensitizability of a circuit, we need solutions to the following problems: (1) A method to determine the path sensitizability of a given path in the combinational circuit. (2) Once it is determined that a path is not sensitizable, a method to modify the given circuit to achieve higher path sensitizability. (3) Also, it is important to investigate the conditions in which these modifications increase or decrease the area and delay of the circuit. In this paper, we present some solutions to these problems.

This paper is organized as follows: In section 2, some of the key terms used in the paper are defined. Section 3 contains the derivation of conditions for which a path cannot be sensitized using single transition tests. Based on the conditions presented in Section 3, we present the possible circuit modifications in Section 4. Section 5 contains a short discussion on the use of OBDDs to determine the sensitization conditions. In Section 6, we present some results obtained on MCNC benchmark examples. Some limitations and suggestions are listed in Section 7. We conclude the paper with a brief summary in Section 8.

2. TERMINOLOGY

A combinational logic circuit is represented as a labeled, directed, acyclic graph (DAG) $G = (V, E)$ where V and E are the vertex set and the edge set, respectively. The gates form the vertices and connections form the edges.

A *path* P in a combinational circuit from the vertex g_0 to the vertex g_{k+1} is a list of alternating vertices and lines $\{g_0, l_0, g_1, l_1 \ldots, l_k, g_{k+1}\}$. The line l_i connects the gate g_i to the gate g_{i+1}. These lines will be referred to as the *path-inputs* of P. All other lines connected to the gates on path P are called *side-inputs*.

A logic value is the *controlling* value C to a gate if and only if the logic value at an input to the gate independently determines the value at the output of the gate. For example, the logic value 0 is a controlling value for an AND gate. The *non-controlling* value nC to a gate is the value $\bar{C}$.

A *single-transition* test is pair of vectors $< V_1, V_2 >$ such that if V_2 is applied after V_1, it provokes a transition to propagate its effect along a single path to a primary output. The test vectors $V_{1 \text{ and } V_2}$ differ in only one bit position, the bit position corresponding to the primary input at which the path originates. A *multiple-transition* test pair differs in multiple bit positions.

To model *gate delay*, we use a simple linear delay model. The model characterizes the delay from an input pin i to the output pin of a gate g using a linear equation of the form $\alpha_{i,g} + \beta_{i,g}\gamma$. The coefficient $\alpha_{i,g}$ models the intrinsic delay through the gate; coefficient $\beta_{i,g}$ models the load dependent delay: γ is the capacitive load at the output of the gate, which is estimated by looking at the output connections of the

gate. The model can distinguish between rise and fall delays. Gate-delay models can be more complex than this linear model, however, only a simple linear model is needed to validate the claims made in this paper.

3. PATH SENSITIZATION

A transition can be propagated along a path if the side-inputs on the path assume non-controlling values for the gates on the path. The sensitization condition for a path can be derived as follows: Let P be a path originating at the primary input variable x_i, s_i's be the side-inputs of the path P, and

$$\hat{s}_i = \begin{cases} s_i & \text{if } s_i \text{ is connected to an AND/NAND gate} \\ \overline{s_i} & \text{if } s_i \text{ is connected to an OR/NOR gate} \end{cases}$$

Then the sensitizing function $\Gamma(P)$ for the path P is given by:

$$\Gamma(P) = \prod_{i=1}^{i=M} \hat{s}_i \tag{1}$$

where Π denotes the AND operation and M is the number of side-inputs to the path P.

Let us now consider the conditions for stable sensitization with respect to single-transition tests. The transition occurs at the primary input x_i on the path P under consideration. The term *stable* refers to a valid sensitization condition for the path P such that it is independent of the value at the primary input x_i. A stable sensitization condition can be derived as follows: Note that a condition satisfying $\Gamma(P) = 1$ does not guarantee a stable sensitization condition. In general, $\Gamma(P)$ can be expressed as:

$$\Gamma(P) = x_i.A + \bar{x}_i.B + C$$

where A, B, and C are Boolean functions independent of x_i. Let $\Lambda(P)$ be the Boolean function which needs to be satisfied in order to satisfy $\Gamma(P)$ independent of x_i. Define

$$\Lambda(P) = C + A.B$$

Boolean function $\Lambda(P)$ has the following properties: (i) It is independent of x_i. (ii) $\Lambda(P) = 1 \Rightarrow \Gamma(P) = 1$ Rewriting $\Lambda(P)$ in terms of $\Gamma(P)$:

$$\Lambda(P) = \Gamma(P)|_{x_i=1}.\Gamma(P)|_{x_i=0} \tag{2}$$

A satisfying condition for $\Lambda(P)$ results in a stable sensitization condition for the path. From Equation 2, the conditions under which $\Lambda(P)$ cannot be satisfied are given as follows:

$$\Lambda(P) = 0 \text{ if } \begin{cases} \Gamma(P)|_{x_i=1} = 0 \text{ or} \\ \Gamma(P)|_{x_i=0} = 0 \text{ or} \\ \Gamma(P)|_{x_i=1} \neq 0, \Gamma(P)|_{x_i=0} \neq 0, \\ \text{and } \Gamma(P)|_{x_i=1}.\Gamma(P)|_{x_i=0} = 0 \end{cases} \tag{3}$$

We use these conditions to guide the modification of the circuit to achieve improved path sensitizabilty for the circuit. This is described in the next section.

4. CIRCUIT MODIFICATION

For any given Boolean function F and a path P in F, the relationship between the path sensitization function $\Gamma(P)$ and the Boolean function F can be expressed as follows:

$$F = G + \Gamma(P).x_i^P \tag{4}$$

where x_i^P represents the primary input variable x_i associated with path P, and G is a Boolean function associated with F. In order to see this, consider the combinational circuit shown in Figure 1.

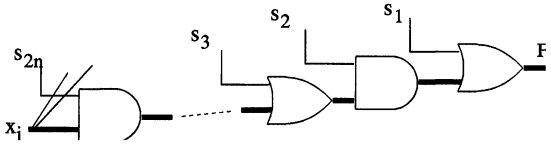

FIGURE 1 A path in a combinational circuit

Without loss of generality, we can assume that a path in a combinational circuit i.e. the path shown using bold lines in Figure 1, is a sequence of $2n$ alternating AND and OR gates. Let s_{2i+1} and s_{2i+2} be the side input functions associated with OR and AND gates, respectively, for $i = 0, 1,\ldots, n-1$.

$$F = s_1 + s_2.s_3 + s_2.s_4.s_5 + \ldots. + s_2.s_4.\ldots.s_{2n}.x_i^P \quad (5)$$

This can be rewritten as:

$$F = s_1 + s_2.s_3 + \ldots. + s_2.s_4.\ldots.s_{2n}.\bar{s}_1.\bar{s}_3.\ldots.\overline{s_{2n-1}}.x_i^P \quad (6)$$

Note that Equation 5 can be obtained from Equation 6 by recursive application of a Boolean algebra rule, namely, $p + \bar{p}.q = p + q$. Equation 6 can now be rewritten as:

$$F = s_1 + s_2.s_3 + \ldots. + \Gamma(P).x_i^P \quad (7)$$

which is of the same form as Equation 4. Equation 4 holds true for a general combinational circuit, even though the example shown here consists of only AND and OR gates. Now, depending upon the Boolean function $\Gamma(P)$, which may be a function of x_i, it can be decided if the line x_i^P can be replaced by a constant.

LEMMA 1 *If $\Gamma(P)|_{x_i}=0 = 0$ ($\Gamma(P)|_{x_i}=0 = 0$) then the path P is a redundant path and it can removed by assigning a constant value of 1(0) to the line connected to the primary input x_i. If the path contains an intermediate fanout node then the path can be first unfolded by replicating the gates along the path and then removed by assigning a constant value to the line.*

Proof: If $\Gamma(P)|_{x_i}=0 = 0$ then $\Gamma(P) = x_i.A$. In this case, $F = G + x_i.A.x_i^P$, which can be simply expressed as $F = G + x_i.A$. This is equivalent to assigning x_i^P a value of 1. Similarly, If $\Gamma(P)|_{x_i}=0 = 0$ then $\Gamma(P) = \bar{x}_i.B$. In this case, $F = G + x_i.B.x_i^P$, which can be simply expressed as $F = G$. This is equivalent to assigning x_i^P a value of 0. □

It must be noted that the substitution of a constant value is only with respect to the path P. In the presence of fanout nodes on the path, the portion of path between the primary input and the farthest fanout node on the path must be replicated before carrying out the substitution. Figure 2 shows an example circuit to illustrate this point. Consider the path shown using bold lines in the figure.

For this path,

$$\Gamma(P) = x.\bar{z}.(\bar{x} + \bar{z}).\overline{(x.\bar{y} + y.\bar{x})}$$

It is easy to see that $\Gamma(P)|_{z=1} = 0$. Since the path contains a fanout node, the path is first unfolded by replicating the inverter connected to z and then the line connected to z is assigned a value 0 (from Lemma 1) as shown in Figure 3(a). The resulting circuit, which is equivalent to the original circuit, is shown in Figure 3(b).

As a result of replication of a portion of a path, the area of the circuit may increase. However, if a path does not have an intermediate fanout node, then under the conditions given in Lemma 1, the area of the circuit decreases as a result of this modification. This leads us to the following result about circuits with such redundant paths.

LEMMA 2 *If there exists a fanout-free redundant path P in a combinational circuit C then there exists an equivalent circuit Ć such that:*

(1) Ć does not contain path P, and
(2) Area(Ć) < Area(C).

Proof: According to Lemma 1, path P can be removed by assigning a constant value to the line connected to the primary input on path P. Since the path

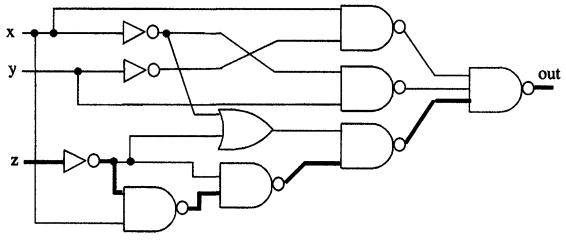

FIGURE 2 Example of a redundant path

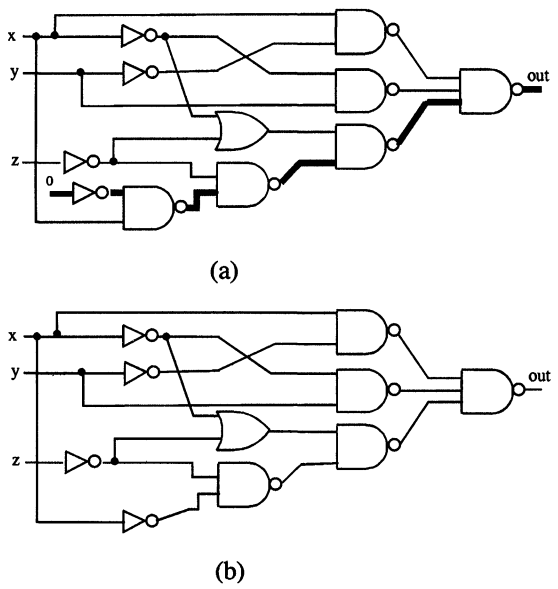

FIGURE 3 Removal of a redundant path

is fanout-free, the resulting circuit $\acute{C}$ has at least one line less than the original circuit C. Hence, the resulting circuit has a smaller area.

Let us consider the effect of such modifications on the delay of a given circuit. In the case of removal of redundant paths, it can be shown that irrespective of the presence of fanout nodes on a path, the delay of the resulting circuit cannot be greater than the delay of the original circuit.

LEMMA 3 *If there exists a redundant path P in a combinational circuit C then there exists an equivalent circuit $\acute{C}$ such that:*

(1) $\acute{C}$ does not contain path P, and
(2) Delay($\acute{C}$) $\leq$ Delay(C).

Proof: The correctness of Lemma 3 can be reasoned as follows: When a path is modified based on Lemma 1, there are two possible scenarios with respect to critical path of the circuit. Either the critical path shares at least one gate with the gates on the path being modified or it shares none. In the second case, the delay of the circuit remains unchanged as a result of modification.

However, in the first case, the delay may either decrease or remain the same. There are two situations in which the delay may decrease. First, if at least one of the shared gates is a fanout node then as a result of duplication of nodes along the modified path, the fanout of the shared nodes will decrease. A decrease in the fanout of a node reduces the capacitive load of that node. This results in smaller delay for the critical path. Second, if the path being modified is the critical path then, as a result of constant substitution, it is removed from the circuit. As a result, the new critical path has a equal or smaller delay than original critical path, which no longer exists. This establishes the correctness of Lemma 3. □

Now consider the third condition in Equation 3. For these paths, the path sensitization function $\Gamma(P)$ has the following form:

$$\Gamma(P) = x_i.A + \bar{x}_i.B$$

with A and B satisfying the following conditions:

$$A \neq 0, B \neq 0, \textit{and } A.B = 0$$

This is possible only when there exists at least one primary input, common to Boolean functions A and B, that requires conflicting values to satisfy $A = 1$ and $B = 1$, simultaneously. Hence, by assigning different values to these primary inputs during test vectors $V1$ and $V2$, a stable sensitization conditions can be maintained at the side-inputs. This requires that at least one primary input Λ other than the path input change its value during the test. Hence, multiple-transition tests are required to sensitize these paths.

An example of such a path is shown with bold lines in Figure 4. For this path $\Gamma(P) = z.\bar{x}.\bar{z}.x.\bar{y}.y.\bar{x}$, which is equivalent to $x.y.z + \bar{x}.\bar{y}.z$. $A = y.z$ and $B = \bar{y}.z$ satisfy the conditions mentioned above.

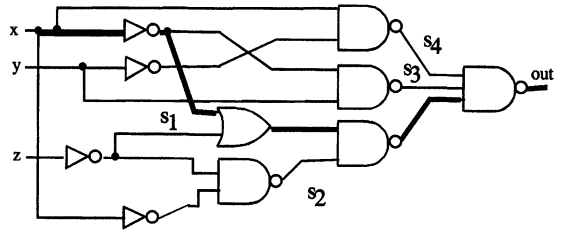

FIGURE 4 Example of an irredundant path not sensitizable by single-transition tests

5. PATH SENSITIZATION USING OBDDS

Ordered binary decision diagrams (OBDD) [16, 31] have been shown to be a practical way of representing Boolean functions. OBDDs have been used in logic synthesis, design verification, design correction, and to carry out static sensitization to avoid costly backtracking operations used in path analysis programs. OBDDs have been used to determine the path sensitization conditions given in Equation 3. There are two main advantages of using OBDDs for this purpose: (1) There exists an efficient OBDD representation for a large number [16] of Boolean functions. (2) Once the OBDDs for the internal nodes of the circuit have been created, the task of finding path sensitization conditions reduces to applying Boolean operations over these OBDDs. As a result, each path under investigation can be examined efficiently.

The derivation of path sensitization conditions is achieved using a sequence of *APPLY* and *RESTRICT* [16] operations over OBDDs for various side-inputs on the path. At the outset, the OBDDs for each node in the circuit is created. Note that the path sensitization conditions may require either the Boolean function at a side-input or its negations. The OBDD for negated function is simply obtained by swapping the 0 and 1 node of the given OBDD.

As an example, consider the boldfaced path shown in Figure 4, the side-inputs are s_1, s_2, s_3, and s_4. The OBDDs for $\hat{s}_1$, $\hat{s}_2$, $\hat{s}_3$, and $\hat{s}_4$ are shown in Figures 5(a), 5(b), 5(c), and 5(d), respectively. The bold lines represent an assignment of 1 to the node variable and the dotted lines represent an assignment of 0.

Three *APPLY* operations over these OBDDs results in the OBDD for the sensitization function of the boldface path.

$$Q = (APPLY\ AND\ \hat{s}_1\ \hat{s}_2)$$

$$R = (APPLY\ AND\ Q\ \hat{s}_3)$$

$$\Gamma(P) = (APPLY\ AND\ R\ \hat{s}_4)$$

The OBDD for the sensitization function ($\Gamma(P)$) for the path is shown in 6(a). In order to compute $\Gamma(P)|_{x=1}$ and $\Gamma(P)|_{x=0}$, we apply two *RESTRICT* operations on this OBDD with respect to the variable x. This results in OBDDs shown in Figure 6(b) and 6(c). $\Lambda(P)$ is obtained by using another *APPLY* operation over these two OBDDs with AND as the operator for *APPLY* function. Figure 6(d) shows $\Lambda(P)$ for the path under consideration. Hence, for this path, the third condition in Equation 3 is satisfied.

As a result of the path modification, some new nodes may be created along the path. The corresponding OBDDs are created by using a sequence of *APPLY* and *RESTRICT* operation over existing OBDDs. For each path, we require $O(d)$ *APPLY* operations and $O(1)$ *RESTRICT* operations over the OBDDs for the various nodes in the circuit, where d is the depth of the circuit.

6. EXPERIMENTAL RESULTS

We have studied the results of circuit modification on a set of standard logic synthesis benchmark examples. These example were first subjected to multilevel logic optimization, followed by a technology mapping [17] step using the MCNC Library 1. These synthesized circuits form the starting point for the experimentation. The method described in this paper has

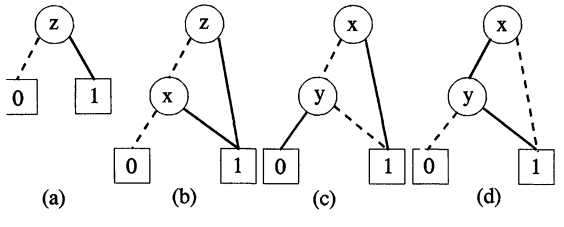

FIGURE 5 OBDDs for side-input functions

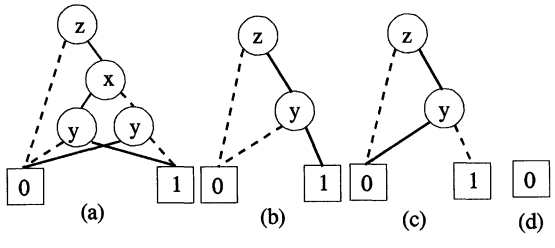

FIGURE 6 OBDDs for sensitization functions

been implemented in Common Lisp, and the indicated CPU times are given in seconds for a SPARC-Station 2. As a final check for the correctness of the procedure described in this paper, OBDDs were used to verify the functional equivalence of the original and the corresponding modified circuits obtained as a result of applying this procedure.

Table I contains the initial statistics for the example circuits. Columns 2 and 3 show the number of inputs and number of outputs for various circuits. Column 4 contains the area measured as the equivalent number of two-input NAND gates. Column 5 indicates the percentage of primary-input-to-primary-output paths in the circuit that are sensitizable. Column 6 indicates the delay of the circuit measured using the delay model for MCNC benchmark Library 1. The last column indicates whether the critical path of the circuit is sensitizable with respect to single-transition test or not.

Table II gives the area, delay, and the percentage of sensitizable paths of the circuits after the removal of redundant paths in the circuit. For the circuits sao2, apex5, and misex3, the increase in area, due to the removal of certain redundant paths with fanout nodes, dominates the decrease due to the removal of fanout-free redundant paths. Area does not change for some of the circuits. For these circuits, the paths which are not sensitizable by single-transition tests satisfy the third condition in Equation 3.

TABLE I Initial Statistics for the Circuits

circuit	in	out	area	% sen	delay	cp
clip	9	5	316.0	88.92	9.0	√
rd53	5	3	110.0	97.75	6.4	√
rd73	7	3	299.0	98.17	11.0	√
rd84	8	4	568.0	68.74	15.0	X
duke2	22	29	866.5	88.68	14.8	√
sao2	10	4	204.0	83.88	9.4	X
apex3	54	50	3353.5	73.81	47.4	X
apex4	9	19	5553.5	76.32	69.6	X
apex5	117	88	1505.0	98.00	25.6	X
apex7	49	37	294.5	85.63	13.0	√
misex3	14	14	3003.5	75.47	37.2	X
alu4	14	8	2241.5	69.44	27.2	X

TABLE II After Redundant Path Removal

example	area	% sen	delay	cpu	cp
clip	300.5	97.20	9.0	40	√
rd53	110.0	97.75	6.4	6	√
rd73	299.0	98.17	11.0	68	√
rd84	510.5	100.0	13.2	397	√
duke2	804.0	98.15	14.8	73	√
sao2	211.5	86.38	9.4	20	√
apex3	3080.5	95.97	39.2	816	√
apex4	5056.5	93.00	57.0	2441	√
apex5	1574.5	98.80	25.2	213	√
apex7	276.5	100.0	13.0	101	√
misex3	3343.5	89.91	33.6	2128	√
alu4	2028.5	98.56	26.6	2107	√

However, for every circuit, the delay of the final circuit is no greater than the original circuit. This is consistent with Lemma 3. As shown in the last column, as a result of modifications, each circuit now contains a critical path which is sensitizable with respect to a single-transition test.

For most of the circuits, we see an improvement in the percentage of paths that are sensitizable with respect to single-transition tests. The remaining paths, which cannot be sensitized using single-transition test, are sensitizable with respect to multiple-transitions test. As a result of the modifications presented in this paper, the delay of the modified circuits, as determined by a static timing analyzer, is also the true delay.

7. FURTHER COMMENTS

The algorithmic complexity of the method presented is dependent on the size of the OBDDs for various nodes in the circuit. For each path, $O(d)$ *APPLY* operations are used over these OBDDs, where d is the depth of the circuit. Each *APPLY* requires $O(G_1 \cdot G_2)$ work, where G_1 and G_2 are the sizes of the OBDDs over which an *APPLY* operation is used. The efficiency of the algorithm will gain significantly through the use of an efficient variable ordering [32, 33] method. Note that creation of node OBDDs is a one time process, however, $O(d)$ *APPLY* operations are used for each path.

Typically, there exists a large number of paths in combinational circuits. For large circuits, it may be impractical to consider each path for modification purpose. However, to guarantee that the critical path is sensitizable, the paths can be modified in order of decreasing criticality until the critical path becomes sensitizable. Also, since the only penalty of this process may come from increased area, the circuit can be modified until a maximum area limit is reached. In this scenario, fanout-free paths should be considered first as the modification of such paths results in decreased area as well.

8. SUMMARY

Based on the path sensitization conditions derived in this paper, we have presented a method for path-based modification of combinational circuits to improve their path sensitizability. The effects of such modifications on area and performance of the circuits were discussed. We have shown that a large number of paths, which cannot be sensitized using single-transition tests, are redundant paths and they can be removed by appropriate modifications of the circuit. For the paths which are neither redundant nor sensitizable using single-transition tests, it has been shown that they can be sensitized using multiple-transition tests. We have demonstrated the validity of this approach through results obtained on some common benchmark examples.

Acknowledgment

The authors are grateful to Dr. J. Brock Barton of Texas Instruments for suggesting very significant improvements to the paper. Thanks to Dr. C. Robert Hewes and Dr. Pallab K. Chatterjee of Texas Instruments for providing the support for this work.

References

[1] R. B. Hitchcock, Sr., G. L. Smith, and D. D. Cheng, "Timing Analysis of Computer Hardware," *IBM J. Res. Develop.*, pp. 100–105, Jan. 1982.

[2] D. Brand and V. S. Iyengar, "Timing Analysis Using Functional Analysis," in *IEEE Trans. Comp.*, pp. 1309–1314 Oct. 1988.

[3] P. McGreer and R. Brayton, "Provably Correct Critical Paths," in *The Proceedings of the Deccenial Caltech VLSI Conference*. 1989.

[4] S. Devadas, K. Keutzer, and S. Malik "Delay Computation in Combinational Logic Circuits: Theory and Algorithms," in *Proc. of the ICCAD*, pp. 176–179, Nov. 1991.

[5] H-C. Chen and D. H-C. Du "Path Sensitization in Critical Path Problem," in *IEEE Trans. on Computer-Aided Design*, pp. 196–207, Feb. 1993.

[6] H-C. Chen, D. H-C. Du, and L-R. Liu "Critical Path Selection for Performance Optimization," in *IEEE Trans. on Computer-Aided Design*, pp. 87–101, Jan. 1992.

[7] P. McGreer and R. Brayton, "Efficient Algorithms for Computing the Longest Viable Path in a Combinational Circuit," in *Proc. Design Automation Conf.*, pp. 561–567, 1989.

[8] S. Perremans, L. Classen, and H. DeMan, "Static Timing Analysis of Dynamically Sensitizable Paths," in *Proc. Design Automation Conf.*, pp. 568–573, 1989.

[9] W. K. C. Lam, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "Circuit Delay Models and Their Exact Computation Using Timed Boolean Functions," in *Proc. Design Automation Conf.*, pp. 128–134, 1993.

[10] H. Chang and J. Abraham, "An Efficient Vigorously Sensitizable Path Extractor," in *Proc. Design Automation Conf.*, pp. 112–117, 1993.

[11] Y-C. Ju and R. Saleh, "Incremental Techniques for the Identification of Statically Sensitizable Critical Paths," in *Proc. Design Automation Conf.*, pp. 541–546, 1992.

[12] G. L. Smith, "A Model for Delay Fault Based on Paths," in *Proc. Int. Test Conf.*, pp. 342–349, 1985.

[13] C. J. Lin and S. M. Reddy, "On Delay Fault Testing in Logic Circuit," in *IEEE Trans. Computer-Aided Design*, vol. CAD-6, pp. 694–703, Sept. 1987.

[14] K. Keutzer, S. Malik, A. Saldhana, "Is Redundancy Necessary to Reduce Delay?" in *Design Automation Conf.*, June 1990, pp. 228–234.

[15] A. Saldhana, R. K. Brayton, A. L. Sangiovanni-Vincentelli, "Circuit Structure Relations to Redundancy and Delay: The KMS Algorithm Revisited" in *Design Automation Conf.*, pp. 245–248, 1992.

[16] R. E. Bryant, "Graph-Based Algorithms for Boolean Function Manipulation," in *IEEE Trans. Comp.*, vol. 6, pp. 677–691, Aug. 1986.

[17] K. Keutzer, "DAGON: Technology Mapping and Local Optimization," in *Proc. of Design Automation Conf.*, pp. 341–347, 1987.

[18] J. Savir and W. H. McAnney, "Random Pattern Testability of Delay Faults," in *IEEE Trans. Comp.*, vol. 17, pp. 291–300, Mar. 1988.

[19] K. Bartlett, R. Brayton, G. Hactel, R. Jacoby, C. Morrison, R. Rudell, A. Sangiovanni-Vincentelli, and A. Wang "Multilevel Logic Minimization Using Implicit Don't Cares," in *IEEE Trans. on Computer-Aided Design*, pp. 723–740, June 1988.

[20] G. Hachtel, R. Jacoby, K. Keutzer and C. Morrison, "On Properties of Algebraic Transformations and the Multifault Testability of Multilevel Logic," in *Proc. of the ICCAD*, Nov. 1989.

[21] S. Kundu and S. M. Reddy, "On Design of Robust Testable CMOS Combinational Logic Circuits," in *Proc. Fault Tolerant Computing Symp.*, pp. 220–225, 1988.

[22] S. Devadas and K. Keutzer, "Synthesis of Robust Delay-Fault-Testable Circuits: Theory," in *IEEE Trans. on Computer-Aided Design*, pp. 87–101, Jan. 1992.

[23] A. K. Pramanick, S. M. Reddy, S. Sengupta, "Synthesis of Combinational Logic Circuits for Path Delay Fault Testability," in *Proc. Int. Symp. Circuits and Systems*, pp. 3105–3108, 1990.

[24] S. Devadas and K. Keutzer, "Synthesis of Robust Delay-Fault-Testable Circuits: Practice," in *IEEE Trans. on Computer-Aided Design*, pp. 277–300, Mar. 1992.

[25] S. Devadas and K. Keutzer, "Synthesis for Testability: A Brief Survey," in *Proc. Int. Sympp. on Circuits and Systems*, pp. 3097–3100, Jan. 1992.

[26] B. Kapoor, "Synthesis and Analysis of Delay Fault Testable Digital Circuits," in *Ph.D. Thesis*, Southern Methodist University, Dallas, Texas, 1994.

[27] P. Ashar, S. Devadas, and K. Keutzer, "Testability Properties of Multilevel Logic Networks Derived from Binary Decision Diagrams," in *Conf. on Advanced Research in VLSI*, Santa Cruz, CA, Apr. 1991.

[28] N. K. Jha and S. Kundu, "Testing and Reliable Design of CMOS Circuits," Norwell, MA: Kluwer, 1990.

[29] I. Pomeranz and S. M. Reddy, "Achieving Complete Delay Fault Testability by Extra Inputs," in *Proc. Int. Test Conf.*, 1991.

[30] B. Kapoor and V. S. S. Nair, "Area, Performance, and Sensitizable Paths," in *Fourth Great Lakes Symposium on VLSI*, pp. 222–227, 1994.

[31] K. S. Brace, R. L. Rudell, and R. E. Bryant, "Efficient Implementation of a BDD Package," in *Proc. Design Automation Conf.*, pp. 40–45, 1990.

[32] S. Malik, A. R. Wang, R. K. Brayton, and A. Sangiovanni-Vincentelli, "Logic Verification using Binary Decision Diagram in a Logic Synthesis Environment," in *Proc. of ICCAD*, pp. 6–9, 1989.

[33] P-Y Chung, I. N. Hajj, and J. H. Patel, "Efficient Variable Ordering Heuristics for Shared ROBDD," in *Proc. Int. Symp. Circuits and Systems*, pp. 1690–1693, 1993.

Authors' Biographies

Bhanu Kapoor received the B. Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur, and the M.S. and Ph.D. degrees in computer science from the Southern Methodist University, Dallas. Since 1987, he has been a Member of Technical Staff with the Integrated Systems Laboratory of Texas Instruments, Dallas. His research focuses on computer-aided design of VLSI, with emphasis on application of graph theoretic algorithms to VLSI CAD, logic synthesis, VLSI testing, low power design, and system-level design tools for DSP applications. He is a member of IEEE and ACM.

V. S. S. Nair received his B.Sc. Engg. in Electronics and Communication Engineering from the University of Kerala in 1984. He received his M.S. and Ph.D. in Electrical and Computer Engineering from the University of Illinois at Urbana in 1988 and 1990, respectively. Currently, he is an Assistant Professor in the Computer Science and Engineering Department at the Southern Methodist University at Dallas where he holds a J. Lindsay Embree Trustee Professorship in Engineering. His research interests include fault-tolerant computing and communication, VLSI synthesis and design for testability, and high-performance architectures. He is a member of the IEEE and ACM.