Fault Analysis in Networks with Concurrent Error Detection Properties

C. Bolchini, F. Salice, D. Sciuto
Politecnico di Milano – Dipartimento di Elettronica e Informazione
I-20133 Milano – Italy

Abstract

The design of Self-Checking circuits through output encoding finds a bottleneck in the realization of the network so that each fault produces only errors detectable by the adopted code. An analysis of an expected TSC network is proposed, based on the application of the weighted observability approach. The aim is the verification of the SC property of the encoded circuit (TSC fault simulation) and identification of critical areas for a consequent manipulation to achieve a complete fault coverage.

1. Introduction

The design of Totally Self-Checking (TSC) networks based on information redundancy provides the adoption of an encoding to the data flow associated with a synthesis methodology granting that each fault causes a detectable error. This allows achievement of a complete fault coverage. When all unidirectional errors detecting (AUED) codes are adopted ([1 2 3 4]), typically the applied synthesis process provides an inverter-free network. When dealing with the Parity Bit code, the synthesis constraint is such that each fault causes odd-cardinality errors only [5 6]. The challenge in TSC design is thus the realization of a network whose faults (single stuck-at fault model) cause errors covered by the selected encoding only. Sometimes costs and complexity prevent the application of such synthesis constraints. A fault analysis would allow the detection of gates which are not TSC.
By detecting TSC redundant faults [8], a subsequent local modification can be applied, thus implying the manipulation of the limited part of the network only.
The aim of this paper is to classify the faults of an expected TSC network with respect to the fundamental properties, fault-secureness and self-test. This will be accomplished by applying the *weighted observability* approach for TSC circuits, a TSC design methodology based on *fault observability* and *code weight* [7].

2. Basics

The **circuit model** representing m scalar Boolean functions $F(X)$, as reported in [5], is a DAG. The node set V consists of three disjoint subsets: the primary inputs set V_{in}, the internal node set V_g and the primary output set V_{out}. Each node $v_i \in V_g$, that is labeled by a variable, is associated with a scalar two level combinational Boolean function with primary inputs or internal nodes as support, while each edge $e_{i,j} \in E$ connects node v_i to node v_j.
Definition 1. The **Cofactor** of $f(X)$ with respect to variable x_i is $f_{xi} = f|_{xi=1} \equiv f(x_1,\dots,1,\dots,x_n)$. The Cofactor of $f(X)$ with respect to variable $!x_i$ is $f_{!xi} = f|_{xi=0} \equiv f(x_1,\dots,0,\dots,x_n)$.
Definition 2. The **Boolean derivative** of $f(X)$ with respect to variable x_i is:

$$\frac{\partial f(X)}{\partial x_i} \equiv f(X)|_{xi=1} \oplus f(X)|_{xi=0}.$$

3. TSC fault classification

The analysis of the network is performed by implicitly analyzing the errors that a fault will actually cause on the observable outputs, with respect to input configurations.

$$\frac{\partial y_i}{\partial z_j} = y_i|_{zj=0} \oplus y_i|_{zj=1} = 1 \qquad (1)$$

Expression (1) identifies the set of input configurations of F for which a change of the output z_j of vertex v_j results in a change of value of the output y_i. If the so determined set is empty, the primary output y_i is not reachable from vertex v_j, vice versa, vertex v_j is *observable* on output y_i.
A further step consists of the exam of how the affected primary output y_i changes value when the output z_j of vertex v_j changes. The following expressions are introduced:

$$\Delta_i^1(X) = \frac{\partial y_i}{\partial z_j} \wedge y_i \qquad (2) \qquad \Delta_i^0(X) = \frac{\partial y_i}{\partial z_j} \wedge \overline{y_i} \qquad (3)$$

representing the set of inputs such that a change on z_j causes output y_i to change value and to assume value 1 or 0.
The formalization of the conditions to be verified for identifying undetected faults on the inputs/output of a vertex (gate at logic level) is described by the following formula [7]:

$$\sum_{i=1}^{h} \left(\frac{\partial y_i}{\partial z_k} \wedge y_i - \frac{\partial y_i}{\partial z_k} \wedge \overline{w_i} \right) \times |w_i| = \pm 2\alpha \qquad (4)$$

where $h = n + r$, $\alpha \in \mathbb{N}^+$, $|w_i|$ is the weight associated with w_i.
To evaluate condition (4) it is necessary to algebraically add

the Boolean functions representing the number of outputs "at value 1" and those "at value 0" for each input configuration considering their weight, respectively producing the values $S^1 = s_{q-1}{}^1 \ldots s_1{}^1 s_0{}^1$ and $S^1 = s_{q-1}{}^0 \ldots s_1{}^0 s_0{}^0$, where variables s_i represent the bits of the binary number "counting" the number of outputs at value 1 or 0; $q = \lg_2(h + 1)$ is the number of bits necessary to "count" the outputs.

Other elements are introduced for the vertex classification.

$$\Psi = \prod_{i=0}^{q-1} (s_i{}^1 \overline{\oplus} s_i{}^0) \qquad (5)$$

Ψ identifies the set of input configurations causing a legal but erroneous codeword. Included in the obtained set there are also the input configurations for which no output changes value ($S^1 = S^0 = 0$), i. e., the vertex is *not observable* on the outputs (*redundant*).

When $\alpha \neq 0$ (Parity code), condition (4) is characterized by the following expression

$$\Theta = (s_0{}^1 \overline{\oplus} s_0{}^0) \qquad (6)$$

Θ identifies the set of input configurations such that a change of value, caused by a fault, on output z_j of vertex v_j produces an even number of output bits changing, thus causing a legal but erroneous codeword.

Another information can also be extracted from the determined relation, to provide a classification of the node:

$$\Phi_0 = \bigvee_{i=0}^{l-1} s_i{}^0 \qquad (7) \qquad \Phi_1 = \bigvee_{i=0}^{l-1} s_i{}^1 \qquad (8)$$

They provide the set of input configurations indicating whether the number of changing bits from 1 to 0 (Φ_0) and from 0 to 1 (Φ_1) is zero. The Boolean constraint $\Phi_0 = 1$ ($\Phi_1 = 1$), or more simply Φ_0, represents the set of inputs such that the number of changing bits is greater than zero while $\Phi_0 = 0$ ($\Phi_1 = 0$) represents the set of input configurations such that the number of changing bits is zero.

Given Θ, Ψ, Φ_0 and Φ_1, it is possible to classify a vertex.

Set definition	Identification
$\Phi = (\Phi_0 \vee \Phi_1)$	OBSERVABLE
$!\Phi$	NOT_OBSERVABLE
$\Psi \wedge \Phi$	UNDETECTED – AUED
$\Theta \wedge \Phi$	UNDETECTED – Parity
$!\Psi \wedge \Phi$	DETECTED – AUED
$!\Theta \wedge \Phi$	DETECTED – Parity
$(\Phi_0 \wedge \Phi_1) \wedge !\Psi$	DETECTED_BID – AUED
$(\Phi_0 \oplus \Phi_1) \wedge !\Psi$	DETECTED_UNI – AUED

By analyzing for each node the resulting sets Ψ, Θ, Φ_0 and Φ_1 (Φ) and their cardinality (empty or not), it is possible to identify the following situations for the analysis of the fault that may affect the node:

1. $\Psi(\Theta) = \Phi = \varnothing \rightarrow$ A change of value of the vertex never produces a change of value on the outputs (*redundant*);
2. $\Psi(\Theta) = \varnothing$; $\Phi \neq \varnothing \rightarrow$ A change of value on the vertex produces either the correct output codeword or an out-of-code, and there is *at least* an input configuration such that it produces an out-of-code word (*TSC*);
3. $\Psi(\Theta) \wedge \Phi \neq \varnothing \rightarrow$ A change of value on the node produces the correct codeword, an out-of-code word *at least* and *at least* an erroneous codeword (*not fault secure*);
4. $\Psi(\Theta) \vee !\Phi \neq \varnothing \rightarrow$ A change of value on the node either produces the correct codeword or an erroneous codeword, never an out-of-code word (*TSC redundant*) [8].

In the first case the node is redundant and could be removed from the network. In the second case, the network implementation guarantees that any fault affecting the node will be detected on the encoded outputs, since no erroneous codeword is ever produced. If an error is generated, an out-of-code word is produced. Cases 3 and 4 need a manipulation of the network to prevent undetected errors.

4. Conclusions

The paper presents a new approach to the analysis of Self-Checking circuits based on the adoption of output encoding. A smaller number of modifications constitute the advantage of the technique.

5. References

[1] J. E. Smith, G. Metze, "Strongly Fault Secure Logic Networks," *IEEE Trans. Comput.*, vol. C-27, No. 6, pp. 491-499, June 1978.

[2] Jha and Wang, "Design and Synthesis of Self-checking VLSI Circuits," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 12, No. 6, pp. 879-887, June 1993.

[3] F. Busaba, P. K. Lala, "Self-checking Combinational Circuit Design for Single and Unidirectional Multibit Error," *Journal of Electronic Testing: Theory and Applications*, n. 5, pp. 19-28, 1994.

[4] V. V. Saposhnikov, A. Morosov, VI. V. Saposhnikov, M. Gössel, "Design of Self-Checking Combinational Circuits with Low Area Overhead," Proc. Int. On-Line Testing Workshop'96, San-Jean De-Luz, F., pp. 56-67, 1996.

[5] N. A. Touba, E. J. McCluskey, "Logic Synthesis Techniques for Reduced Area Implementation of Multilevel Circuits with Concurrent Error Detection", Proc. of Int. Conf. on Computer-Aided Design (ICCAD), pp. 651-654, 1994.

[6] K. De, C. Natarajan, D. Nair, and P. Banerjee, "RSYN: A System for Automated Synthesis of Reliable Multilevel Circuits," *IEEE Trans. on Very Large Scale Intergration (VLSI) Systems*, vol. 2, n. 2, pp. 186-195, June 1994.

[7] C. Bolchini, F. Salice, D. Sciuto, "Designing Networks with Error Detection Properties through the Fault-Error Relation," Proc. *DFT'97*, Parigi, F, 1997, pp. 290-297.

[8] C. Bolchini, F. Salice, D. Sciuto, "Redundant Faults in TSC Networks: Definition and Removal," Proc. *DFT '96*, Boston, U.S.A., 1996, pp. 277-285.