A Test-Vector Generation Methodology for Crosstalk Noise Faults

Hamidreza Hashempour, Yong-Bin Kim, Naphill Park*
Department of Electrical and Computer Engineering, Northeastern University,
Department of Computer Science, Oklahoma State University*,
Email: hhashemp@ece.neu.edu, ybk@ece.neu.edu, npark@a.cs.okstate.edu

Abstract

This paper presents a new methodology to generate test vectors for crosstalk noise faults in deep sub micron devices. The methodology includes transition activation on aggressor and constant assignment on victim, transition time estimation on aggressor, noise characterization on victim, and propagating the noise to the primary outputs through the best paths. New approaches for transition time estimation and noise activation are proposed based on logic cell characterization already available in design library and solving a satisfiability problem. It is shown that test generation efficienty can be increased up to 18% and test generation time is decreased up to 30%.

1. Introduction

Recent advances in integrated circuit design has shown that crosstalk issues in deep sub micron can cause severe validation and test problems. The continuous advancement in design has resulted in high device densities, aggressive clocking schemes, and small signal transition times. Based on these design advancements, coupling effect between adjacent wires has become a very important issue. It has been shown that if this phenomenon is not carefully considered during design and validation, it can cause delay faults, crosstalk noise faults, and overshoot/undershoot faults. Delay and logic errors caused by crosstalk is one of the primary sources of functional failures in deep sub micron designs.

Crosstalk effect can also cause a finite energy pulse on one of the adjacent wires which is usually driven by a weaker driver than the aggressor wire. This is called crosstalk noise fault. A crosstalk noise fault on a clock signal can cause a wrong value to be clocked in a flipflop. Thus the state of a synchronous circuit can be changed erroneously and results in a device failure.

Crosstalk noise and delay faults have been one of the major research topics in testing deep sub micron devices. Bai et al. have shown an efficient high level crosstalk fault simulation technique for system-on-chip interconnects [1]. Fault modeling and simulation for crosstalk effect in system-on-chip interconnects was presented in [2]. A self-test methodology for crosstalk noise faults was presented in [3]. Another self-test technique for overshoot and undershoot faults was presented in [4]. Test generation for crosstalk noise faults have been studied in [5][6]. They considered crosstalk noise as a full voltage pulse induced on the victim line at a fault site, which is not true in general. A recent work has tried to generate test and predict output noise at the same time with some resolution [12]. This and similar works usually turn out to be complex and time consuming algorithms. In most of the cases, test generation and finding the effect of noise at primary outputs have been two different problems.

In this paper, we present a test vector generation methodology to detect crosstalk noise faults that are not detected during the early design phase. We also present a new approach for crosstalk noise activation which in practice is shown to be very efficient in terms of test generation. The

Proceedings of the 17th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT'02)
1063-6722/02 $17.00 © 2002 **IEEE**

IEEE COMPUTER SOCIETY

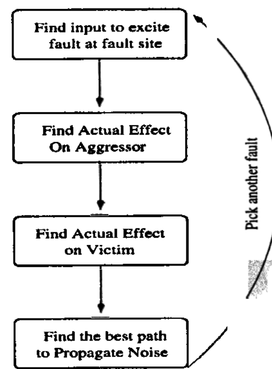

Figure 1. Overall flow for the proposed methodology

Value	Definition
0	Constant 0
1	Constant 1
X	Don't care
t_r	Rise Transition
t_f	Fall Transition
p_r	Negative Pulse
p_f	Positive Pulse

proposed algorithm finds the maximum noise effect at a fault site and choose the best path to propagate it. Once the vector set is known, HSPICE simulation is run for final characterization of noise at the primary outputs of the circuit. The rest of the paper is organized as follows: Section 2 presents our approach. Section 2.1 presents fault activation algorithms, and Section 2.2 presents fault characterization process. Sections 2.3 deals with propagation paths toward primary outputs and propagation path selection. Section 2.4 describes test generation. Section 3 presents experimental results followed by conclusions in Section 4.

2. Test Generation Algorithm

The proposed methodology includes several steps as shown in Fig. 1. In this paper, we assume preliminary fault list is previously compiled and made available for test generation process. In order to have the compiled fault list, layout information should be analyzed to locate potential fault sites based on interconnect wire spacing information [7]. Usually potential fault sites are analyzed during design and fixed during re-design and signal re-routings. However, due to process variations, there are still chances that these sites cause crosstalk noise faults. These faults can only be detected by application of special test vectors generated for crosstlak noise faults. A crosstalk fault is identifed by two properties. The first one is the location of the *aggressor* and *victim* lines in the circuit, and the second one is its direction, i.e., rise or fall transition on the aggressor line. Once the transition on the aggressor is specified, the victim value is implied to be the inverse of that. A rising transition can cause a crosstalk noise logic "0" level on the victim line and vice versa.

The first step as identified in Fig. 1, is to partially identify two-vector sets to create the transition and constant values on the aggressor and victim lines. A two-vector set includes two assignments of input values. They must be applied one after another so the required transition is built at fault site. For simplicity, we use a logic value system that includes transition and pulse values as summarized in Table 1. For each vector in a two-vector set, some of the bits may be unknown (X).

1063-6722/02 $17.00 © 2002 **IEEE**
IEEE COMPUTER SOCIETY

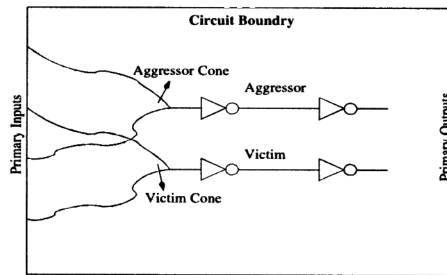

Figure 2. Aggressor and Victim cones.

A two-vector set to create a transition on aggressor is represented by a single vector using the new logic value system because transition bits can be represented by t_r or t_f. From now on, whenever we name a vector, we refer to a two-vector set that is represented as a single-vector using the new logic value system. A typical vector would be $(0, 1, t_r, X)$. This vector has the first two bits identifed, the third bit as a rise transition, and the last bit as unknown. So we can apply $(0, 1, 0, X)$ followed by $(0, 1, 1, X)$ to detect a fault. A vector does not include pulse values. They are only to model crosstalk noise fault on victim line.

Once the vector set is partially known and faulty conditions are setup at fault site, the fault must be characterized. This is the second step in Fig. 1. The vector set must be simulated so that the exact waveform at fault site is found. The third step in Fig. 1. is to apply one of the crosstalk noise models to the fault site to characterize how crosstalk noise affects victim line. The actual crosstalk pulse on the victim line is modeled and characterized. Finally, similar to the classical test generation for stuck-at-faults, there is a last step dealing with the propagation of noise to at least one primary output. In propagation, a finite energy transient pulse is being passed through gates and it may be attenuated or amplified depending on the gate characteristics. So it is important to characterize propagation paths and pick the path that can propagate the noise effect to a primary output without attenuating but amplifying.

2.1. Crosstalk Noise Activation

To activate the crosstalk noise, all vectors that their application cause a transition on the aggressor and keeps a constant value on the victim must be found. Fig. 2. shows two circuit cones that must be searched to find these vectors. This problem can be mapped onto two *satisfiability* problems. One is to set victim line to its constant value and another is to set aggressor to the required transition. Solutions to this problem are available [8][9].

These two vector sets are intersected so that the set of all vectors that can activate the fault are identified. The objective of intersection is to maximize noise effect at the fault site. It has been shown that crosstalk noise energy is almost constant and its amplitude is proportional to the ratio of driving strength of aggressor and victim lines, coupling capacitance between them, and the transition time of the aggressor [10].

The driving capability of a gate is also affected by the location of the inputs in terms of the distance from the gate output. Assuming only one input is changing, Fig. 3 shows HSPICE simulation of output waveform when the only switching input is applied to different inputs whose distances are different from the output. As it is seen in Fig. 4, the location of switching input can weaken driving strength as much as 30% per transistor. This behavior has not been considered previously in test generation algorithms for crosstalk noise faults. If layout information is available, it can be used to find better tests.

Considering a certain threshold for the ratio of driving strength of aggressor to victim, we limit input search space. Logic level simulation for the part of the circuit including aggressor and victim cones is performed to mark vectors with driving strengths ratio higher than a certain threshold. These vectors are used for later refinement.

1063-6722/02 $17.00 © 2002 IEEE
IEEE COMPUTER SOCIETY

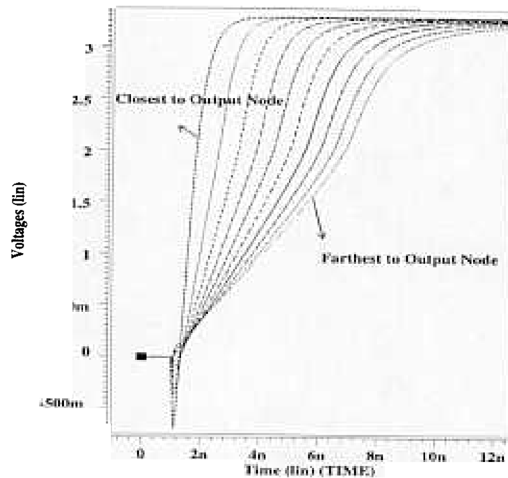

Figure 3. Driving strength versus location of switching input

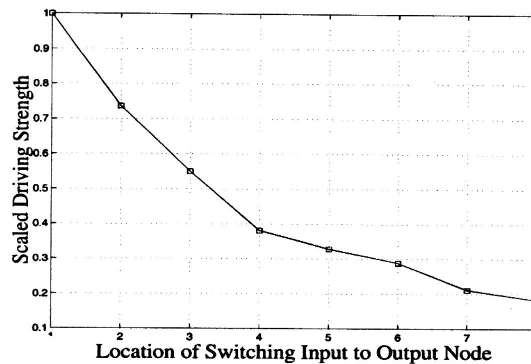

Figure 4. Driving strength vs. location of switching transistor

2.2. Crosstalk Evaluation

The next step as outlined in Fig. 1, is to characterize analog behavior of the crosstalk noise at fault site and to estimate how crosstalk noise is actually induced on victim line. To do so, a crosstalk model of the aggressor and victim line must be present. There has been previous models as reported in [10][11]. All these models require the rise or fall time of the transition on aggressor. To obtain the transition time of aggressor, the perfect method would be to perform HSPICE simulation of the aggressor cone of the circuit and measure transition time on fault site. This is a very expensive approach. A better approach would be to dynamically collapse each gate to its equivalent inverter gate and then use these simplified models to accelerate simulation [12].

In this paper we propose a new approach to obtain transition time of aggressor. Since all the logic cells are characterized in the design library, those information are used to develop a table to estimate transition time at the output of a gate according to its load capacitance and input conditions. Therefore a fast simulation of the circuit is possible for the purpose of transition time estimation on aggressor line at fault site. The simulation is done for all the vectors passed the previous stage. These models can be implemented in hardware description language such as VHDL-

1063-6722/02 $17.00 © 2002 **IEEE**
IEEE COMPUTER SOCIETY

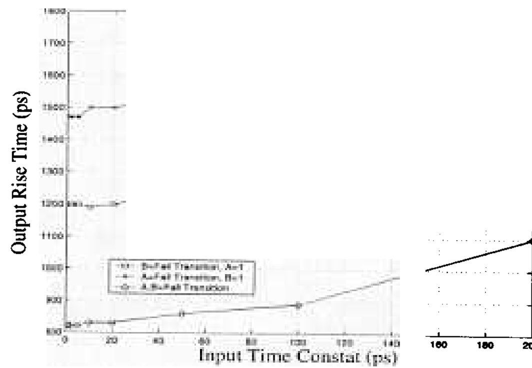

Figure 5. Output rise time for a NAND2 gate versus input fall time constant

AMS, and by binding circuit netlist to these models we have access to a transition time simulation. Simulation results for a NAND2 gate is shown in Fig. 5.

To have t_r on output of this gate, at least one input must be t_f. If only one input has t_f, the other must be constant 1. So there are three cases to have a rising transition on output. In Fig. 5, we show three curves. They show rise time of output versus time constant of inputs. Curves are shown for three cases, the first case is when input A and B both have t_f, the second case is when input A has t_f and B=1, and finally when input B has t_f and A=1. Input A is closer to output node and input B is farther from output node. From Fig. 5 we can write:

$$Transition_{Time} = \alpha \times Input_{\tau} + \beta \tag{1}$$

α and β for the NAND2 gate for each case are reported in Table 2. $Input_{\tau}$ is time constant of input. If multiple inputs are switching at the same time, we consider the fastest switching input. If multiple inputs are switching at different times, we consider the earliest arriving input.

To have t_f on the output of the NAND2 gate, at least one of the inputs must be t_r and the other must be at constant 1. This is rather complex as the pull-down network is discharging the output and the intermediate capacitances cannot be easily lumped into the output node capacitance as the previous case. As we are macro-modeling the analog behaviour of the gate for the purpose of transition time estimation, we do not consider the pull-down network drain/source capacitances, though they can be lumped into the output capacitance using Elmore technique as in [12]. Instead we attempt to complete our look-up table for different cases that can happen when pull-down network is discharging output. For the NAND2 gate, there are 3 different cases, when either input switches or when both inputs are switching.

If there are multiple inputs switching at the same time, we consider the slowest one. This is different than the pull-up case where the fastest switching was considered. The reason is because pull-down network transistors are connected in series and the slowest one determines the overal discharge time. In comparison, the pull-up network transistors are parallel and the fastest switching one determines the overal charge time. If multiple inputs are switching at different times, we consider the latest arraiving input. In general when we have n inputs, there are $2 \times (2^n - 1)$ rows in the look-up table. Majority of information in these tables are taken from parametrized design library, otherwise simulation can be done to compile the remainder of the table.

Following this step, we have a set of vectors, and for each vector there is a transition time on aggressor associated. All units are in ps.

2.3. Path Selection and Noise Propagation

Once the fault is activated and we have characterized noise on victim line, we have to propagate the noise to at least one primary output. There are two different issues to be addressed.

IEEE COMPUTER SOCIETY

Case	α	β
A changing, B constant	1.5	1183
B changing, A constant	1.4	1470
A,B changing	1.33	807

Table 2. Coefficients for NAND2 fast rise time simulation

One is to deal with selecting a path that amplifies the crosstalk noise as much as possible toward a primary output or at least propagates it with minimum attenuation. Another issue would be how to propagate the noise from fault site toward primary outputs. More specifically, an efficient and accurate propagation technique is required so that the output and noise level at output is known for future reference and validation.

Once a path is found, the propagation can be simulated using HSPICE. It is efficient even for large circuits as there is only one best path. In order to find the actual noise effect, only that identified path is to be HSPICE simulated given a test vector. If there are multiple paths with propagation cost in a certain suitable range and HSPICE simulation is found to be time-consuming, other approaches such as equivalent gate collapsing can be used. As we are interested in finding the test vectors, we do not focus on finding noise waveform at a primary output in this work.

Path selection is to find a path in the cone of victim line at fault site to primary output(s). A path includes all gates from the gate immediately after victim line to the gate that drives a primary output. Assuming a cost function is defined for each gate in the path, the cost function of the whole path is simply multiplication of costs of all gates in the path. Cost is a measure of how a crosstalk noise effect penetrates a gate. If gate A has a lower cost than a gate B, then gate A is a better choice to make a path toward a primary output because noise effect can better penetrate gate A and still maintains its effect. The cost function is related to load capacitance at output of a gate and its driving strength. With same driving strength, the load capacitance affects the pulse amplitude: the larger the load capacitance, the smaller the noise amplitude. With same load capacitance, the driving strength affects the pulse amplitude: the larger the strength, the larger the noise amplitude because same output capacitance is charged more easily. So the cost function is defined as:

$$Cost_{Function} = \frac{C_{Load}}{DrivingStrength} \quad (2)$$

Load capacitances are available from layout analysis. Driving strength of gates are available by estimating effective β of each logic gate. For each gate, strength of pullup and pulldown networks are calculated and compiled for future references. This cost calculation is statically performed and then cost of all paths from victim line to primary outpus is calculated. A path with best cost is picked up though it might be useful to pick all paths with cost function higher than a certain threshold value.

2.4. Test Generation

Our test generation begins with solving the two SAT problems and then intersecting input vector sets to activate fault. This intersection reduces the set of vectors. During intersection, two criterions are used to limit search space further. The first criterion is the ratio between the aggressor driving strength and victim driving strength. The second is the aggressor transition time. Once the fault is successfully activated, we pick the best path and find the set of vectors that maximize noise strength from the victim toward outputs. The latter vector set is intersected with the vector set of solutions obtained by SAT problems. Finally, we have a reduced set of vectors that can activate a fault and propagate it. A fault is reported as undetectable if either satisfiability problem used for its activation returns null. If there are vector sets to activate transition on aggressor and assign constant value to victim but intersection is null, there again the fault is undetectable. If intersection exists but during strength characterization, no vector could achieve a driving strength ratio higher than a certain threshold, again the fault is reported as undetectable.

IEEE COMPUTER SOCIETY

Circuit	Detected	Undetected	Time	Coverage
c2670	385	115	983s	0.77
c5315	446	54	1219s	0.89
c880	365	135	872	0.73
c1908	311	189	1649	0.62
c3540	218	282	4960	0.44
7552	383	117	4076	0.76
Experimental results using the proposed methodology				
Circuit	**Detected**	**Undetected**	**Time**	**Coverage**
c2670	308	192	1090s	0.66
c5315	368	132	1671s	0.83
c880	270	230	707	0.60
c1908	256	244	2298	0.61
c3540	123	377	6182	0.44
7552	291	209	3552	0.66
Experimental results using W. Chen et al.				

Table 3. Experimental Results

3. Experimental Results

We used a public domain satisfiability solver to implement the first step [13]. A code is written for intersection of vector sets and is called whenever intersection is required throughout the approach. Basic logic gates up to 5 inputs NOR and NAND gates plus NOT and BUF are characterized for the purpose of transition time simulation. To prune a vector set in crosstalk fault activation step, we used Verilog model of the circuit. Experimental circuit is converted to a Verilog model of the circuit. Required communications between Verilog simulator and C codes are implemented through VPI which is a C interface to Verilog simulation engine. We have tried two ISCAS85 benchmarks circuits. Load capacitances on the output nodes of gates in these circuits, are set as $\alpha \times Fanout$, where α is a constant capacitance taken from an inverter gate and *Fanout* is number of gates being driven by this gate. We consider 500 pairs of lines. Because these pairs are not listed based on any layout analysis and currently no information on layout of the circuits is available, we expect no test vector to be available for many of these pairs. The fault type, i.e., rise pulse or fall pulse is also set randomly. The threshold for driving strength was set to 2, i.e., aggressor driving strength must be at least twice that of victim. As opposed to previous works, we do not have ***aborted*** faults. Aborted faults are defined in [5][6] as faults that backtrack limit is reached during test generation, and test generation is aborted. It is not known if they are really detectable or not. Table 3 shows experimental results for these benchmark circuits versus results in [12]. Test generation efficiency is higher than [12]. A large portion of our running time is due to Verilog simulation, though, running time is less than [12] the case where crosstalk noise propagation is also performed during test generation. Coverage is not a measure of these circuits testability versus crosstalk noise faults. If faults were compiled based on layout information and not picked randomly, it would represent to some extent circuit testability against crosstalk noise faults. This factor may vary for the same circuit based on layout and technology used for implementation.

4 Conclusions

In this paper, we present an approach for test generation for crosstalk noise faults in deep sub micron devices. The fundamental steps included in the approach are, transition activation on aggressor and constant assignment on victim, transition time estimation on aggressor, noise characterization on victim, and finally picking the best path and propagating the noise to the primary outputs. Additionally, we applied a new fault activation technique based on SAT problem solving, and also used a new fast transition time estimation. We showed that driving strength of gates are not only

IEEE COMPUTER SOCIETY

affected by the number of switching inputs but also by distance of switching inputs to output node of a gate. We drop complex calculation required for propagation of noise and paid full attention to picking the best path for propagating a noise to a primary output. This approach enhances the efficiency of the algorithm.

References

[1] X. Bai, S. Dey, "High-level Crosstalk Defect Simulation for System-on-Chip Interconnects" *in Proc. 19th IEEE VLSI Test Symposium, L.A., CA, USA,* April 2001

[2] M.Cuviello, S.Dey, X.Bai, Y.Zhao, "Fault Modeling and Simulation for Crosstalk in System-on-Chip Interconnects" *in Proc. IEEE/ACM Intl. Conf. on Computer Aided Design (ICCAD), San Jose, California,* November 1999

[3] X.Bai, S.Dey, J.Rajski, "Self-Test Methodology for At-Speed Test of Crosstalk in Chip Interconnects" *in Proc. 37th Design Automation Conference, Los Angeles, California,* June 2000

[4] A. Attarha, M. Nourani, "Built-In-Chip Testing of Voltage Overshoots" *in Proc. 19th IEEE VLSI Test Symposium, L.A., CA, USA,* April 2001

[5] A. Rubio, N. Itazaki, X. Xu, K. Kinoshita, "An approach to the analysis and detection of crosstalk faults in digital VLSI circuits", *IEEE Trans. On Computer Aided Design of Integrated Circuits and systems, Vol. 13, pp. 387-394,* March 1994

[6] K. T. Lee, C, Nordquist, J. A. Abraham, "Automatic test pattern generation for crosstalk glitches in digital circuits", *Proc. VTS,* 1998

[7] V. S. Subramanian, C. P. Ravikumar, "Estimating Crosstalk From VLSI Layouts", *in Proc of 14th International Conference on VLSI Design,* Jan. 2001

[8] P. Stephan, R. K. Brayton, and A. Sangiovanni, "Combinational Test Generation Using Satisfiability", *IEEE Trans. on Computer aided Design of Integrated Circuits and Systems, vol. 15,* 1996

[9] J. P. Marques-Silva, K. A. Sakallah, "GRASP: A Search Algorithm for Propositional Satisfiability", *IEEE Transactions on Computers, vol. 48, pp. 506-521,* 1999

[10] W. Chen, S. Gupta, M. A. Breuer, "Analytic Models for Crosstalk Delay and Pulse Analysis Under Nin-Ideal Inputs", *Proc. Of Intl. Test Conf.,* 1997

[11] A. K Goel, "High-Speed VLSI interconnections: Modeling, Analysis, and Simulation", *John Wiley and Sons Inc.* 1994

[12] W. Chen, S. Gupta, M. A. Breuer, "Test Generation in VLSI circuits for Crosstalk Noise", *Proc. Intl. Test Conf.,* 1998

[13] H. Zhang, "SATO: An efficient propositional prover", *Proc. Intl. Conf. Automated Deduction,* 1997

Proceedings of the 17th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT'02)
1063-6722/02 $17.00 © 2002 **IEEE**

IEEE COMPUTER SOCIETY