Path Delay Test Generation Toward Activation of Worst Case Coupling Effects

Minjin Zhang, Huawei Li, *Senior Member, IEEE*, and Xiaowei Li, *Senior Member, IEEE*

***Abstract*—As the feature size scales down, crosstalk noise on circuit timing becomes increasingly significant. In this paper, we propose a path delay test generation method toward activation of worst case crosstalk effects, in order to decrease the test escape of delay testing. The proposed method performs transition-map based timing analysis to identify crosstalk-sensitive critical paths, followed by a deterministic test generation process. Using the transition map instead of the timing window to manage the timing information, the proposed method can identify many false coupling sites and thus reduce the pessimism in crosstalk-induced fault collection caused by inaccurate timing analysis. It can also efficiently calculate the accumulative crosstalk-induced delay, and find the sub-paths which cause worst case crosstalk effects during test generation. By converting the timing constraints of coupling lines into logic constraints, complex timing processing for crosstalk effect activation is avoided during test generation. In addition, the trade-off between accuracy and efficiency can be explored by varying the size of timescale used in the transition map.**

***Index Terms*—delay testing, test generation, timing analysis, signal integrity, crosstalk-induced delay, path delay fault**

I. INTRODUCTION

WITH the decrease of feature size into the deep sub-micron and nanometer era, coupling capacitance starts dominating circuit behavior and becomes a considerable contributor to signal integrity problems [1]. As a result, the delay of a line becomes increasingly dependent on the states of its adjacent lines.

If two lines are physically adjacent, which constitute a possible coupling site, the propagation delay of both lines can be speeded up/slowed down when the signals on them transit simultaneously or nearly simultaneously, in the same/opposite directions. Usually the line whose delay is concerned is called victim line, whereas the neighboring line affecting it is called aggressor line.

It has been reported that the crosstalk effect may cause up to 30% accumulative delay to a critical path [2]. In practice, various techniques including buffering, shielding [3] and optimal channel routing [4] have been used to minimize the crosstalk effect on delay variations. While these techniques can successfully control the crosstalk effect on path delays to be under 15% [2], this level of delay variation is still too high for high-performance designs which often cannot afford a high margin for tolerating delay variations.

If crosstalk effects are not properly considered during manufacturing testing, a non-trivial level of test escape may occur. As Figure 1 shows, a path with a little excess delay, such as a small delay defect, may cause timing violations when further affected by crosstalk, whereas it may not be detected by the ordinary structural delay test, since the delay of this path without activation of crosstalk is smaller than the clock period. So the crosstalk issues must be considered during manufacturing test to minimize test escape.

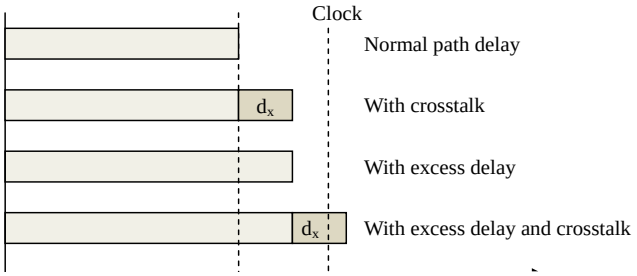

Figure 1. Path delay with crosstalk.

In addition, since path delay testing considering crosstalk can detect smaller delay defects than the standard path delay testing can [5], the worst case path delay under crosstalk effects, if not violating functional constraints, is in fact a more accurate performance indicator.

Several fault models and test generation techniques have been proposed to take into account crosstalk-induced delay. The common purpose of all these methods is to find the patterns causing maximum delay considering crosstalk-induced effects on the circuit under test (CUT).

Chen et al. [6] proposed a test pattern generation algorithm with a timing-oriented back-trace procedure targeting coupled transition faults (CTFs). Irajpour et al. [7] presented a timing-independent approach to generate tests for crosstalk-induced slow-down effects, while the huge number of targets makes it impractical for large circuits. Focused on all aggressor lines of a victim line, Ganeshpure et al. [8] proposed

Manuscript received Jan. 28, 2010; revised June 5, 2010. This paper was supported in part by National Natural Science Foundation of China (NSFC) under grant No. (60776031, 60633060, 60921002), and in part by National Basic Research Program of China (973) under grant No.(2011CB302501).

M. Zhang is with the Key Laboratory of Computer System and Architecture, Institute of Computing Technology, Chinese Academy of Sciences, and Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: zhangminjin@ict.ac.cn).

H. Li is with the Key Laboratory of Computer System and Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (corresponding author, phone: 86-10-62600719; fax: 86-10-62527489; e-mail: lihuawei@ict.ac.cn).

X. Li is with the Key Laboratory of Computer System and Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: lxw@ict.ac.cn).

a solution that combines an integer linear program with the traditional stuck-at fault ATPG. These three methods could not activate the worst case crosstalk-induced delay, since they consider testing of the crosstalk effect on a single victim line, similar to the transition delay testing, without considering accumulative delay defects or effects on a path.

Krstic et al. [9] presented a constrained path delay fault (CPDF) model as a combination of a critical path and a set of crosstalk noise sources interacting with the path. A genetic algorithm with a dynamic timing simulator was used to deal with timing information and justification of aggressor transitions to the primary inputs (PIs). This test generation method is a non-deterministic approach with respect to crosstalk activation. Paul et al. [10] proposed a timed ATPG method to generate critical paths and corresponding input vectors to sensitize these paths under crosstalk effects. It incorporated special timing processing techniques into ATPG algorithms and employed circuit-level timing simulation which is computationally expensive. These two methods are not scalable for large-scale circuits.

Ren et al. [11] used the timed-Boolean logic to characterize signal transitions in a time interval, and used Boolean satisfiability (SAT) to check the correlations between aggressor and victim transitions. Lee et al. [12] proposed a structural test pattern generation procedure to magnify parasitic crosstalk effects on delay-sensitive paths by inducing switching on nearby nets. These two methods can find the patterns efficiently by means of ignoring the timing of aggressors, but they couldn't guarantee meeting of the timing requirements for activation of the targeted crosstalk effects.

To generate deterministic test patterns for crosstalk-induced delay faults, timing information cannot be ignored. However, including timing information into an ATPG engine will significantly increase the complexity of the ATPG algorithm. Considering the timing of the aggressors is the main obstacle for efficient test generation.

In [13], we introduced a precise crosstalk-induced path delay fault (PCPDF) model, which is similar to CPDF, but consists of a critical path and sub-paths propagating transitions to the aggressor lines at certain times. Since the exact timing of signal switching is determined by the sub-path reaching the line, sensitizing specific sub-paths assures the aggressors switching at certain times. Based on the PCPDF model, a deterministic structural test generation method is proposed in [14], which is efficient in that it was extended from a conventional delay ATPG, without considering timing parameters explicitly.

Chun et al. [15] further proposed a test-generation method for critical paths considering multiple-aggressor crosstalk effects to maximize the noise of the victim lines. Physical and timing information are used to prune false aggressors, which is helpful for reducing the ATPG time cost. Specifically, timing false crosstalk effects are reduced based on static timing window analysis and recalculated timing windows using a delay test pattern of a victim path. This method has similar CPU efficiency to that of [13, 14]. Crosstalk-induced delays are not considered during critical path selection and the accumulative crosstalk-induced delay is not updated during test generation in this method, which may influence the final quality of delay testing.

This paper extends the method of [14] and introduces two different test generation methods with extensive experimental results. The main contributions of this paper include:

1) The proposed method performs a *transition-map* (TM) [16] based timing analysis to identify crosstalk-sensitive critical paths, followed by a deterministic test generation process for the PCPDF model. Timing and logic constraints are unified during test generation to deterministically activate crosstalk effects. Consequently, a Boolean satisfiability solver, or a structural ATPG tool can be efficiently applied for test generation. This method is efficient and scalable since it does not use a complicated timed ATPG algorithm.

2) Using a transition map (TM) instead of a timing window to manage all the likely transition arrival times of a line, the proposed method can identify many false coupling sites and thus reduce the pessimism in crosstalk-induced fault collection caused by inaccurate timing analysis. The worst crosstalk-induced delay on a path can be estimated based on TMs, and crosstalk-sensitive critical paths can be found efficiently. It is also effective to select proper sub-paths and dynamically update timing information for accurate fault activation. We can trade accuracy for efficiency by increasing the size of timescale used in the transition map, which can further increase the scalability of this method.

In the experiments of larger ISCAS'89 benchmark circuits, coupling capacitance parameters from layout are utilized in the test generation to target worst case coupling effects. The activation of crosstalk-induced delay is validated by the increase of path delays through HSPICE simulation.

This paper is organized as follows. Section II gives the notations. Section III describes the problem statement and the PCPDF model. Section IV gives a detail description of the transition map based timing analysis method, and how to find crosstalk-sensitive critical paths and select sub-paths based on TMs. In Section V we present two test generation methods. Experimental results on ISCAS'89 benchmark circuits are shown in Section VI. Limitations are discussed in Section VII and conclusions are given in Section VIII.

II. NOTATIONS

The following notations will be used in this paper.

a_i An aggressor line that affects the delay of a victim line when it switches.

v, v_i A victim line whose delay is affected by one or multiple aggressor lines.

$<V, A>$, $<v_i, a_i>$ A coupling pair, where V or v_i is the victim line, and A or a_i is the aggressor line.

$\uparrow$ / $\downarrow$ A rising / falling transition on a line, sometimes denoted as the superscript R/F of a variable representing the line.

p A logical path, which is a combinational path that starts from a primary input (PI) or the output of a flip-flop (called pseudo primary input, or PPI), and ends at a primary output (PO) or the input of a flip-flop (called pseudo primary output, or PPO), and has a transition ($\uparrow$

or ↓) at the PI/PPI.

sp-x A (logical) sub-path, which starts from a PI/PPI and ends at an internal line *x* in the circuit, and has a transition (↑ or ↓) at the PI/PPI.

sp-x/p Sub-path *sp-x*, which is on path *p*, where *x* is an internal line on *p*.

d( ) The delay of a sub-path or a path, depending on the argument.

L A line in the circuit.

$L.TM^{R}$ The transition map of line *L*, which is a bitmap structure that records the delays of all possible sub-paths that propagate a rising transition to *L* (see Definition 2).

$L.TM^{F}$ Similarly to $L.TM^{R}$, but the propagated transition to line *L* is a falling transition.

$L.TM^{R/F}$ Acronym of both transition maps $L.TM^{R}$ and $L.TM^{F}$. Similar forms of the superscript R/F are used as acronym of a pair of variables, the difference of which is only the transition direction.

I_j The j^{th} input line of a gate with *n* inputs, where $j=1, \ldots, n$.

$L.d_j^{R/F}$ The delay of propagating a rising / falling transition from the j^{th} gate input to the gate output *L*.

$L.d_w$ The wire delay of line *L*, where *L* is a gate output.

$L.t_{latest}^{R/F}$ The normal latest arrival time of line *L* (see Definition 3) without considering crosstalk effects.

$L.X_{MP}^{R/F}$ The maximum crosstalk-induced latency on line *L* considering the propagation of a rising/falling transition along a sub-path to *L* (see Definition 5).

$L.d_X$ The crosstalk-induced delay on line *L*, due to all possible aggressors coupling with *L*.

$v.t_{switch}^{F/R}$ The switching time of the falling / rising transition on a victim line *v* of path *p*, which is equal to *d*(*sp-v/p*). Note crosstalk-induced delay is considered in *d*(*sp-v/p*).

$v.t_p^{F/R}$ The normal arrival time of line *L* regarding path *p* (*L* is on *p*), without considering crosstalk effects.

III. Modeling Crosstalk- Induced Delay Fault

A. Timing Constraints of Crosstalk Activation

The problem of test generation targeting maximum path delay caused by crosstalk is to find a pattern that sensitizes the critical path and activates the worst case crosstalk-induced delay on it. Simply considering the logic constraints of a target circuit, the pattern should cause as many as possible aggressors rising or falling, and this max-satisfiability problem is NP-complete [17].

Apart from logical feasibility, the timing is also important, which is demonstrated with the example in Figure 2. Suppose each gate has a unit delay and each aggressor line can cause 1 unit crosstalk-induced delay on the victim only when the coupling pairs switch simultaneously. The possible switching time list of each line is shown in the bracket near the line.

First, suppose there are two sub-paths to a_1, and the delay values of them are 4 and 9 respectively. The methods ignoring timing may generate the pattern causing a switching on a_1. If the switching happens at 9, it will not affect the switching delay of v_1 at time 4.

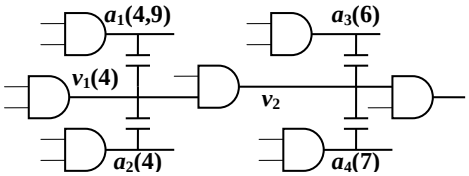

Figure 2. An example of multiple crosstalk effects

Second, if both of the aggressor lines a_1 and a_2 have inverted transitions to that of v_1 at time 4, causing 2 units' crosstalk-induced delay, the arrival time of v_2 is 4+2+1=7. Then the switching on a_4 could induce an extra delay on v_2 but the switching on a_3 cannot. If only one aggressor line to v_1 can be activated due to logic constraints, the arrival time of v_2 is 4+1+1=6. Then only the switching on a_3 could induce an extra delay on v_2.

To sum up, since the switching time of a victim line is affected by accumulative crosstalk-induced delay propagated from the anterior lines, the approach ignoring timing may not activate aggressors or select wrong aggressor lines for activation during test generation. Therefore, it cannot guarantee that generated patterns indeed activate the target crosstalk effects. So our object of test generation is not only making the aggressors switch but also making it switch at proper time. All in all, both timing and logic constraints should be considered in test generation.

B. Previous Fault Models and Test Generation

It is reasonable to consider testing of crosstalk-induced delay effects as an enhancement of conventional delay test. As we know, there are basically two delay fault models adopted in industry: transition fault (TF) and path delay fault (PDF).

Since the crosstalk effect is related to coupling lines, we can define a coupled transition fault (CTF) as a crosstalk-induced transition delay fault denoted by a cluster of lines <v, (a_1,a_2,…a_n)> and the crosstalk type. The crosstalk type can be one of the following four extreme situations: *SR* - simultaneously rising, *SF* - simultaneously falling, *VR* - *v* rising and a_i falling simultaneously, and *VF* - *v* falling and a_i rising simultaneously. We ignore other situations that some aggressor lines have rising transitions and others falling, because such coupling effects will reduce each other and the resulting delay effect is weaker than those of the extreme cases, when only capacitance coupling is considered. On the other hand, the failure of activating a certain aggressor during test generation is equivalent to removing the aggressor from the target cluster and considering the extreme situations of the reduced cluster. Thus, any possible combination of a_i and *v* corresponds to four possible CTFs.

The former two types of faults, simultaneously rising or falling, are crosstalk-induced speedup faults. The later two are crosstalk-induced slowdown faults. Figure 3 gives two CTFs: <C, (B, F)> and <H, F>, both with the type *VF*.

The CTF fault model has the same shortage as that of the TF fault model. Although the fault effect of a coupled transition

fault can be propagated along a longest path though the victim, it is still difficult to deal with cumulative effects of multiple aggressors coupled with some victims on the same longest path.

The disadvantage of TF testing can be reduced by augmenting some PDFs. For each combinational path in a circuit, there are two logical paths, thus two PDFs, corresponding to rising and falling transitions on the start (or end) point of the path. So a logical path, or a PDF, can be represented by its input transition direction, ↑ or ↓, and the lines along the path. In order to find the worst case situation of testing a long path, a constrained path delay fault (CPDF) is defined as the critical path and a set of interconnects coupled to it [9]. Critical paths refer to paths whose delay is longer than a given percentage (e.g. 90%) of the longest propagation delay in the circuit. For example, considering the longest path p (↓C-G-H-K) coupled with three aggressor lines shown in Figure 3, (p, {<C, B>, <C, F>, <H, F>}) is a candidate CPDF.

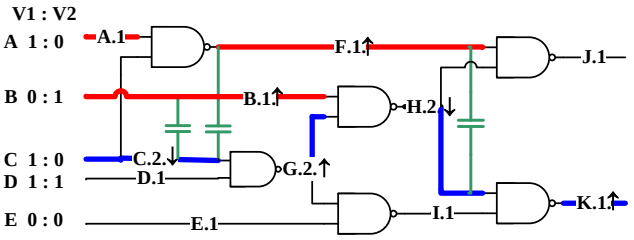

Figure 3. Examples of crosstalk-induced delay faults

Test pattern generation techniques of CTFs and CPDFs are not like conventional ATPG algorithms, since timing information should be considered again after target fault selection process. The summary of related works has been given in the first section of this paper. Ignoring timing constraints cannot guarantee that patterns indeed sensitize the faults under test. For methods considering timing during ATPG, the common feature of test generation for CTFs, and CPDFs is that they all need special techniques incorporated into ATPG algorithms to deal with necessary timing information for the activation of crosstalk-induced faults. It then includes much effort to revise a conventional ATPG algorithm for delay testing in order to generate tests for crosstalk-induced faults.

To sum up, timing and logic constraints are both the necessary requirements for crosstalk testing, and it is hard to process them together. If the timing and logic of circuit can be converted into uniform constraints during test generation, the patterns will be generated by employing conventional ATPG algorithms and the computational expensive timed ATPG will not be necessary. Next, we will discuss a delay fault model with time parameters to address this issue.

C. Precise Crosstalk-induced Path Delay Fault

The precise crosstalk-induced path delay fault (PCPDF) was defined in [13] to demonstrate the cause of crosstalk-induced delay effects on critical paths. Since the switching time of an aggressor is determined by the sub-path that propagates the transition to it, the sub-path sensitization is critical to the existence of crosstalk-induced delay faults. The PCPDF model is defined to include information of such sub-paths as follows.

***Definition* 1**: A precise crosstalk-induced path delay fault is a path delay fault on a logical path p constrained by one or more crosstalk-induced slow-down effects between multiple pairs $<v_i, a_i>$ with all the victims v_i on path p, such that for each aggressor a_i, there exists a sub-path sp-a_i, satisfying the following condition:

(1) a_i can have a transition reversely to that on v_i, which is propagated by the sub-path sp-a_i,

(2) The slowdown effect on v_i can be induced because:

$$\left|d(sp\text{-}v_i/p) - d(sp\text{-}a_i)\right| \le \delta, \qquad (1)$$

where δ is a specified small time interval related to the technology.

A PCPDF can be represented by

$$(p, \{sp\text{-}a_i, < v_i, a_i >\}), i = 1,2, \ldots, n.$$

δ is strongly depended on the switching delay of coupling lines, and can be determined by HSPICE simulation. Typically, δ is the delay of one or two gates [18].

The advantage of PCPDF model is that it gives exact information about which critical path should be sensitized, as well as which sub-paths should be sensitized to generate aggressor transitions. Thereby, if proper sub-paths are established (or selected) before test generation, a complicated procedure or technique to deal with timing information during ATPG will not be necessary, while it is necessary for ATPG based on CTF and CPDF fault models.

An example of a PCPDF of Circuit C17 is also shown in Figure 3. Suppose each line has the same delay to propagate a rising and falling transition. The delay of each line is shown near the line based on the fan-out weighted delay model. Path p(↓C-G-H-K) is one of the critical paths in the circuit. Then, line C has a falling transition at time 2. Given a falling transition at time zero to primary input A, line F may have a rising transitions at time 2, which may be an aggressor transition to the falling transition on line C. Given a rising transition at time zero to primary input B, line B may have a rising transition at time 1. Suppose δ =1,

$$| d(\text{C-C}) - d(\text{B-B}) | = |2\text{-}1| = 1.$$
$$| d(\text{C-C}) - d(\text{A-F}) | = |2\text{-}2| = 0.$$
$$| d(\text{C-C}) - d(\text{C-F}) | = |2\text{-}3| = 1.$$
$$| d(\text{C-H}) - d(\text{A-F}) | = |6\text{-}2| > 1.$$
$$| d(\text{C-H}) - d(\text{C-F}) | = |6\text{-}3| > 1.$$

Therefore, both sub-path ↓A-F and sub-path ↓C-F can induce slowdown effect on line C of path p, but either of them cannot induce slowdown effect on line H of path p. Then we get two candidate PCPDFs: (p, {↓A-F, < C, F >; ↑B-B, < C, B >}) and (p, {↓C-F, < C, F >; ↑B-B, < C, B >}). Obviously, the less the time difference is, the more the crosstalk-induced delay will be. So the first one (with sub-path ↓A-F) activates more additional delay on path p and thus is a better target fault for path p.

From the above analysis, it is important to find all possible switching times of target aggressors of the path under test (PUT), and proper sub-paths to activate slowdown effects as large as possible. The PCPDF model includes such timing information and is used for fault collection and test generation in our work.

The flow of the proposed method is summarized in Figure 4. Firstly, using the extracted capacitance and timing information, timing analysis is performed with three steps: transition map computation, followed by crosstalk-sensitive critical path selection and target coupling sites selection. Secondly, tests for the selected critical paths are generated considering all related target coupling sites for sensitization of aggressive sub-paths, using either a structural ATPG method, or a SAT solver. More detail about the flow can be found in the following Sections.

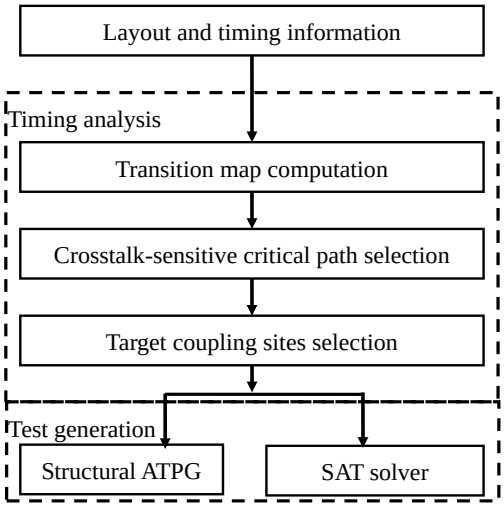

Figure 4. Flow diagram for crosstalk-oriented path delay test

IV. Transition Map based Timing Analysis

An aggressor may switch at some time, and usually the timing of transitions on aggressors is captured by timing windows [15]. Since the circuit structure does not always allow signal switching at arbitrary time within the continuous timing window, more accurate timing analysis method is needed to capture switching times, for example, Chen et al. proposed the discontinuous timing window [19].

Furthermore, there may be many sub-paths feeding to an aggressor line, and it is necessary to present an efficient method to select sub-paths with required path delays for activation of target coupling effects.

Focusing on these two objectives, we proposed to analyze coupling effects using the transition map based timing analysis method.

A. Transition Map

Transition map is a bitmap structure for characterizing the timing of rise and fall transitions, in which a bit, called a time slot, denotes a time interval with certain size [16]. Two transition maps (TM^{R} and TM^{F}, the superscript R or F denotes the rising or falling transition respectively) are used to record possible switching times of each line, instead of a single continuous window. So, the arrival times of all the possible transitions on a line are recorded by TMs. The timescale (denoted as *Timescale* later), which is the size of a time slot, is selected beforehand, and all the delay values are denoted by integer times of this scale in timing analysis.

***Definition* 2**: The transition maps of line L, denoted as $L.TM^{\mathrm{R}}$ and $L.TM^{\mathrm{F}}$, are bitmap structures that record the delays of all possible sub-paths that propagate a transition to L. Each bit in $L.TM^{\mathrm{R/F}}$ is a Boolean variable defined as follows:

$$L.TM^{\mathrm{R/F}}[i] = \begin{cases} 1, & \text{if } \exists sp\text{-}L, d(sp\text{-}L) = i \ \textit{Timescale} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

where *Timescale* is a given time unit, and the bit-width of $L.TM^{\mathrm{R/F}}$ is equal to the maximum circuit delay in units of *Timescale*. □

All the lines of the CUT are visited in their topological order as in the conventional static timing analysis (STA) method. The two transition maps are recursively calculated for a line L using the following equations.

1) If L is a primary input or the output of a flip-flop,

$$L.TM^{\mathrm{R/F}}[1] = 1, L.TM^{\mathrm{R/F}}[i] = 0 \text{ for } i > 1. \quad (3)$$

The arrival times of all PIs/PPIs are assumed to be the same, since inputs driven by storage elements are usually controlled by the same clock. This assumption is not required in our method and different initial values can be assigned if necessary.

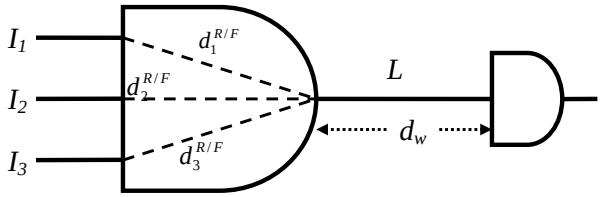

Figure 5. Mean delay model of a gate

2) If L is an output of gate G having n inputs lines ($I_1, I_2, \dots, I_n$), and its timing information includes n rising gate delay values ($L.d_1^{\mathrm{R}}, L.d_2^{\mathrm{R}}, \dots, L.d_n^{\mathrm{R}}$), n falling gate delay values ($L.d_1^{\mathrm{F}}, L.d_2^{\mathrm{F}}, \dots, L.d_n^{\mathrm{F}}$), and the wire propagation delay $L.d_w$, as shown in Figure 5. The transition maps of L can be obtained from the transition maps of gate inputs $I_1, I_2, \dots, I_n$, through *adding* the corresponding gate delay and wire delay to each transition map, and *combining* them together:

$$L.TM^{\mathrm{R/F}} = \begin{cases} \cup_{j=1}^{n} \left(I_j.TM^{\mathrm{F/R}} \gg \left(L.d_j^{\mathrm{R/F}} + L.d_w \right) \right), & \text{NAND/NOR/NOT} \\ \cup_{j=1}^{n} (I_j.TM^{\mathrm{R/F}} \gg (L.d_j^{\mathrm{R/F}} + L.d_w)), & \text{AND/OR} \end{cases} \quad (4)$$

where $\cup$ is the bit "OR" operator and ">>" is the right shift operator to accomplish the accumulation of gate and wire propagation delay at the output of a gate. It should be noted that since TM is used to record all the possible arrival times, it is assumed during static timing analysis that any input transition can be propagated to the output as long as there is a connecting path between the input and the output. □

For simplicity, we use the mean delay to calculate the timing of the CUT in this paper, and the mean gate delay model in Figure 5 does not handle the situation when the gate output is a stem and the interconnect delay for each branch is different. Since we use 180nm technology for experiments in this paper, the difference of branch delays can be ignored. One can use more accurate or complex delay models, and translate the possible arrival times into transition maps, which is not the focus of this work.

Transition maps can be compacted into some integer structures and be operated by bit operations. It makes our

approach have very efficient memory usage and small time penalty, which are comparable to those of timing window based analysis when *Timescale* is not too small.

The timing of the CUT and of crosstalk effects are mutually dependent. To solve the mutual dependence, a possible way is by iteration [20]. First, all the transiting maps are computed based on static delay values without considering crosstalk effects. Second, the transition maps of aggressors are used to compute crosstalk-induced delay on victim lines and fix all the lines' transition maps. Then, the new values of transition maps are used to update transition maps again. This process is iterated until we find a converged solution.

The iteration method can be used for worst-case full-circuit crosstalk-induced delay estimation. However, not all slow-down effects can be activated and many such effects are don't-care delays during test generation. So the iteration method will not be used during static timing analysis since it is not known which effects are activated. As for test generation, we only focus on activating maximum crosstalk-induced delay on a critical path. The crosstalk effects which are not on the victim path are ignored in our test generation method and the iteration process is thus simplified.

B. Computation of Crosstalk-induced Delay

Considering crosstalk-induced delay will improve the accuracy in static timing analysis. Several methods [21][22][23] have been proposed to calculate the impact of crosstalk on interconnect delay. Any of these delay calculation methods could be used in our approach. We employ a simulation based method in this paper.

Before timing analysis, HSPICE simulation is done with various RC parameters and skews of two arrival times to obtain the crosstalk-induced delay. Then, we use these delay values to establish a lookup-table, whose values are indexed by signal skews and coupling capacitances.

The accumulative crosstalk-induced delay due to all aggressors is computed based on a linear summation of each aggressor's nonlinear impact on the victim's delay. The results of [24] shows, in 130nm technology, the largest error of linear summation to HSPICE simulation is less than 8%.

C. Crosstalk-Sensitive Critical Path Selection

Due to different accumulative delays induced by crosstalk on each path, the delay of a path is quite different under different input patterns. Which paths are more critical is thus correlative with not only the gate/interconnect delays but also the crosstalk-induced delay. So crosstalk-sensitive critical paths should be considered in delay testing.

Similar to the definition of a critical path, a crosstalk-sensitive critical path is a path whose delay under activated crosstalk effects can be longer than a given percentage (e.g. 90%) of the longest propagation delay in the circuit. Therefore, it is necessary to calculate crosstalk-induced delays in timing analysis, as it will affect the set of paths selected for test generation. As mentioned earlier, this requires analyzing the worst crosstalk-induced delay of the circuit, and it usually need many iterations.

The accumulative crosstalk-induced delays on the whole path are considered for crosstalk-sensitive critical path selection. We introduce a path based worst crosstalk-induced delay estimation method to get the static worst delay on a path, which is the upper bound of the path delay under crosstalk effects. The logic constraints are ignored in this process and will be dealt with in test generation.

***Definition* 3**: The normal latest arrival time of line *L*, denoted as $L.t_{\text{latest}}^{\text{R/F}}$, is the maximum rising/falling transition arrival time on *L* without considering crosstalk effects. □

***Definition* 4**: The crosstalk-induced latency on line *L*, regarding sub-path *sp-L*, is the accumulative crosstalk-induced delays on *sp-L*, while each crosstalk-induced delay is caused by a crosstalk-induced slowdown effect between a coupling pair whose victim line is on *sp-L*. □

On sub-path *sp-L*, if there are multiple aggressors that couples to the lines on *sp-L*, a transition propagated on *sp-L* may be delayed by multiple crosstalk-induced slowdown effects. The sum of all these crosstalk-induced delays is recorded by the crosstalk-induced latency on *L*, regarding *sp-L*.

***Definition* 5**: Considering the propagation of a rising/falling transition to line *L*, the maximum crosstalk-induced latency on *L*, denoted as $L.X_{\text{MP}}^{\text{R/F}}$, is the maximum of crosstalk-induced latencies on *L* regarding any sub-paths ending at *L*. □

All the lines are visited in the same order as used in the TM computation. The latest arrival time considering crosstalk for line *L*, i.e., the sum of $L.t_{\text{latest}}^{\text{R/F}} + L.X_{\text{MP}}^{\text{R/F}}$, is recursively calculated using the following equations.

1) If *L* is a primary input or the output of a flip-flop,

$$L.t_{\text{latest}}^{\text{R/F}} = 1 \tag{5}$$

$$L.X_{\text{MP}}^{\text{R/F}} = 0 \tag{6}$$

2) If *L* is an output of gate G having *n* inputs lines ($I_1, I_2, \dots, I_n$), and its timing information includes *n* rising gate delay values ($L.d_1^{\text{R}}, L.d_2^{\text{R}}, \dots, L.d_n^{\text{R}}$), *n* falling gate delay values ($L.d_1^{\text{F}}, L.d_2^{\text{F}}, \dots, L.d_n^{\text{F}}$), the wire propagation delay $L.d_w$, and the crosstalk-induced delay $L.d_X$ due to all possible aggressors coupling with *L*.

$$L.t_{\text{latest}}^{\text{R/F}} = Max_j(I_j.t_{\text{latest}}^{\text{R/F}} + L.d_j^{\text{R/F}}) + L.d_w \tag{7}$$

$$L.X_{\text{MP}}^{\text{R/F}} = Max_j\left(I_j.t_{\text{latest}}^{\text{R/F}} + L.d_j^{\text{R/F}} + I_j.X_{\text{MP}}^{\text{R/F}} + L.d_X\right) + L.d_w - L.t_{\text{latest}}^{\text{R/F}} \tag{8}$$

If the gate G is of negative polarity, the $L.t_{\text{latest}}^{\text{R/F}}$ and $L.X_{\text{MP}}^{\text{R/F}}$ can be obtained by changing all the superscripts R/F to F/R in Equation (7) and (8). □

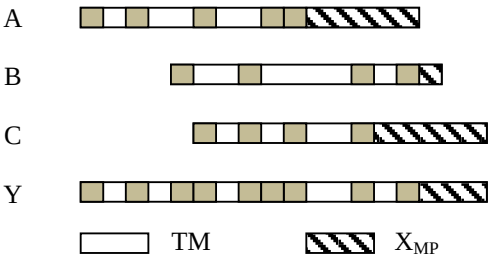

Figure 6. An example of latest arrival time estimation

An example of estimation of latest arrival time considering crosstalk is shown in Figure 6. A, B, C are the three inputs of a gate and Y is the output. Their TMs are demonstrated. For simplicity, all the $d_j^{\text{R/F}}$, d_w and d_X are assumed as zero. It can be

seen from Figure 6, that:

$$\mathrm{Y}.\, t_{\text{latest}}^{\text{R/F}} = \mathrm{B}.\, t_{\text{latest}}^{\text{R/F}},$$
$$\mathrm{Y}.\, X_{\text{MP}}^{\text{R/F}} = \mathrm{C}.\, X_{\text{MP}}^{\text{R/F}} + \mathrm{C}.\, t_{\text{latest}}^{\text{R/F}} - \mathrm{B}.\, t_{\text{latest}}^{\text{R/F}}.$$

At last, we modify the computation of the slack of a line or a path using the following Equation (9):

$$\text{slack} = t_{\text{required}}^{\text{R/F}} - t_{\text{latest}}^{\text{R/F}} - X_{\text{MP}}^{\text{R/F}} \quad (9)$$

As STA methods usually do, a line whose slack is less than a given threshold is a critical line, and critical paths are constituted by critical lines.

D. Overlapping Analysis

After all transition maps and crosstalk-induced delay have been calculated, we select the target coupling pairs using TM overlapping analysis. As mentioned above, if the difference of transition times of a coupling pair is not more than a specified time interval (δ), the coupling pair is a target coupling site. In our experiments, δ is specified to half of the rising delay.

Consequently, for a coupling site <*v*, *a*> and a victim path *p*, a slow-down effect on *v* is induced if the following condition is satisfied.

$$\exists a.TM^{\text{R/F}}[i] = 1, \left(v.t_{\text{switch}}^{\text{F/R}} - \delta\right) \le i \le (v.t_{\text{switch}}^{\text{F/R}} + \delta) \quad (10)$$

where $v.t_{\text{switch}}^{\text{F/R}}$ is the switching time of *v*, which is equal to *d*(*sp*-*v*/*p*), the sum of normal arrival time of *v* (which is denoted as $v.t_p^{\text{F/R}}$) and the crosstalk-induced latency on *v*. As mentioned before, crosstalk effects which are not on the victim path are ignored.

Actually, the activated crosstalk-induced delay effects in test generation are usually less than what are estimated in timing analysis. So we can use more weak constraints in target site selection:

$$(v.t_p^{\text{F/R}} - \delta) \le i \le (v.t_p^{\text{F/R}} + v.X_{\text{MP}}^{\text{F/R}} + \delta) \quad (11)$$

Based on Formula (11), we can select the target coupling sites. For example, Figure 7 demonstrates the transition maps of a victim line and its three aggressor lines. As the figure shows, only a_2 and a_3 can cause the slow-down effects on the victim line.

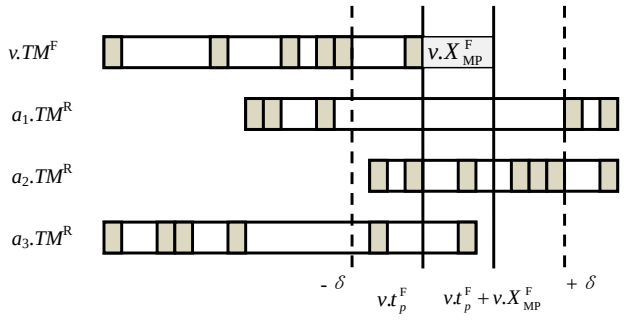

Figure 7. Transition map overlapping analysis

E. Sub-path Selection

Relied on transition maps, it is easy to find a proper sub-path which makes the aggressor line switch at $v.t_{\text{switch}}^{\text{F/R}}$ or nearest to $v.t_{\text{switch}}^{\text{F/R}}$. We use the process given in Table I to select a proper sub-path to the aggressor line.

Usually, there are many sub-paths to an aggressor whose delay is around $v.t_{\text{switch}}^{\text{F/R}}$, especially when *Timescale* is large. We use two selection strategies for structural and SAT-based test generation methods respectively.

For structural ATPG, we sequentially select and sensitize the target sub-paths which are sorted by the difference of the sub-path delay and $v.t_{\text{switch}}^{\text{F/R}}$ in the ascending order. When the current selected sup-path cannot be sensitized, this process is also used to select the next one.

For SAT-based test generation, we select a set of sub-paths using a backtrace process. In the beginning, we find all anterior A_j of the aggressor line a, which satisfies $a.TM^{\text{R}}[t] = 1$ and $A_j.TM^{\text{R}}\left[t - L.d_j^R - L.d_w\right] = 1$. And then we find recursively the anterior lines of A_j, until A_j is a PI or PPI. All these lines compose a tree, and a path from the root (the aggressor) to a leaf node (PI/PPI) is a proper sub-path to the aggressor line.

TABLE I
Sub-path selection

step	operations
1	For each coupling site (*v*, falling, *a*, rising), Calculate $v.t_{\text{switch}}^{\text{F}}$;
2	If $\exists i, a.TM^{\text{R}}[i] = 1, (v.t_{\text{switch}}^{\text{F}} - \delta \le i \le v.t_{\text{switch}}^{\text{F}} + \delta)$ goto step 3; Else, exit and report that *a* is a false aggressor;
3	Find time *t* that satisfies $a.TM^{\text{R}}[t] = 1$, and minimizes $\left\lvert t - v.t_{\text{switch}}^{\text{F}} \right\rvert$;
4	Set $L = a$;
5	If the input gate type of *L* is AND or OR, search the TM^{R} of each anterior line of *L*, and find the j^{th} anterior line A_j satisfying the condition: $A_j.TM^{\text{R}}\left[t - L.d_j^R - L.d_w\right] = 1$; Else if the gate type of *L* is NAND or NOR or NOT, search the TM^{F} of each anterior line of *L*, and find the j^{th} anterior line A_j satisfying the condition: $A_j.TM^{\text{F}}\left[t - L.d_j^F - L.d_w\right] = 1$;
6	Set $t = t - L.d_j^{R/F} - L.d_w$; $L = A_j$;
7	If *L* is a PI or PPI, the sub-path is found. Else, goto Step 5.

F. Timescale

By controlling the size of *Timescale*, the trade-off between the accuracy and efficiency can be explored, which makes this approach highly scalable. Using a smaller *Timescale* will reduce more pessimism in coupling site identification and improve the accuracy of accumulative crosstalk-induced delay calculation, but it will result in more time and memory penalty.

First, the smaller *Timescale* is, the fewer number of sub-paths to a line in the same time slot are. So, we can find the sub-path to an aggressor whose delay is much closer to $v.t_{\text{switch}}^{\text{F/R}}$ of a victim *v*.

Furthermore, employing smaller scale helps to calculate more accurate crosstalk-induced delay, since we use the signal skew to calculate crosstalk-induced delay.

From the above analysis, the effectiveness of transition map strongly depends on the size of *Timescale*. We use *density*, which is defined as the ratio of number of bit ones to the total number of time slots in the TM, to measure how effective the transition map is [19]. If the density is close to 0, the TM tends to be very spare, and this method can cut down most of the pessimism in the target fault identification.

It should be mentioned that there is a bottom line for the *Timescale* of a CUT, so that the accuracy of crosstalk analysis will no longer increase after this bottom line. Such bottom line is the smallest *Timescale* that should be used in delay

calculation when considering crosstalk effects.

V. Test Pattern Generation

After timing analysis, necessary timing information has been collected. Then, the objective of test generation is to sensitize the critical path and propagate transitions to aggressors coupling to the path at required times.

The value of a side-input of a target path is usually restricted to some non-controlling value implied by a two-vector delay test (V1, V2), which is the logic constraint during test generation, and generally called path sensitization criterion. There are mainly three classes of path sensitization criterion employed in path delay test generation: robust test, non-robust test, and functional sensitization [25]. We use robust test in this paper.

Since both critical paths and sub-paths to aggressors are selected based on timing analysis, as discussed in Section IV, timing constraints are mapped to on-path signal requirements of selected sub-paths and PUT during test generation. In this way, timing and logic constraints are unified during test generation to explicitly activate crosstalk effects.

We propose two distinct methods for path delay test generation toward activation of worst case coupling effects, which are described in the following sub-sections.

A. Structural Test Generation

It is easy to convert a conventional ATPG system for path delay faults to an enhanced ATPG system targeting crosstalk constrained path delay faults. In this sub-section, the robust test method for single coupling effect we proposed in [32] is applied to structural test generation considering multiple coupling effects. The extra thing to do is sub-path sensitization, which includes: 1) adding constraint transition signals on lines of each sub-path, and 2) adding constraint signals on lines of side-inputs of the sub-path according to certain path sensitization criterion.

For robustly testing a PCPDF, the path p should be robustly sensitized, and the sub-paths (sp-a_i) are relaxed to functionally sensitized to increase the possibility of generate aggressor transitions. The path sensitization criterion for the path and sub-paths is shown in Table II, while 10-value logic is used in test generation [26].

TABLE II
The value requirement at the side-input

Type	On-input of path p		On-input of sub-path sp-a	
	falling	rising	falling	rising
AND/NAND	S1	X1	UX1	X1
OR/ NOR	X0	S0	X0	UX0

The 10 values include 4 basic values and 6 composite values as follows.

S0, a static 0 value,
S1, a static 1 value,
U0, a signal whose final value is 0,
U1, a signal whose final value is 1,
X0 = {S0, U0},
X1 = {S1, U1},
UX = {U0, U1},
UX0 = {S0, UX},
UX1 = {S1, UX},
XX = {X0, X1}.

A path delay fault ATPG engine can be used to find the pattern sensitizing a critical path p and some sub-paths simultaneously. The structural test generation method is an expansion of the conventional path delay ATPG engine, which is shown in Figure 8.

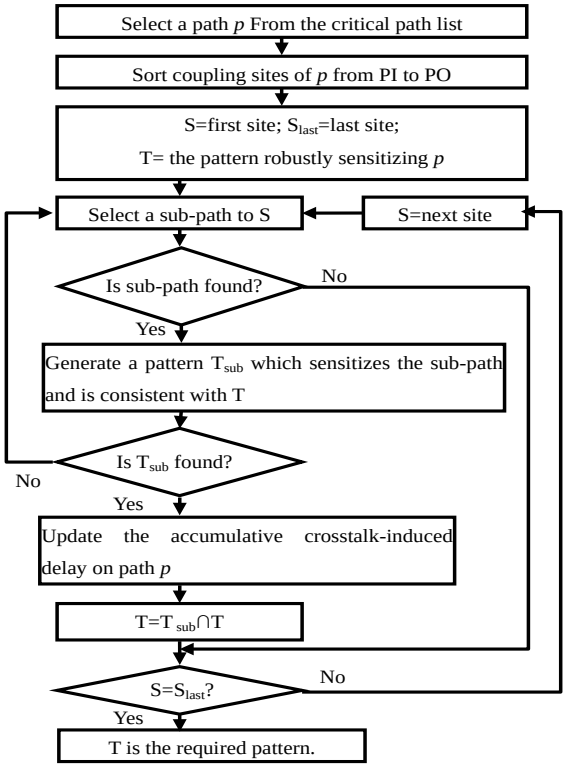

Figure 8. Structural test generation flow

In the proposed structural test generation flow, aggressors are considered from the primary input to the primary output, and the sub-paths are selected using the steps shown in Table I and sensitized one by one. In this way, the accumulative crosstalk-induced delay on a critical path is updated after each coupling site is indeed activated by sensitizing any sub-path to the aggressor. So this method can guarantee that the generated pattern indeed activates the target crosstalk-induced delay if the given timing of the CUT is accurate.

This method is easy to be implemented and has enough accuracy, but in some case, it cannot activate maximum crosstalk-induced delay because it does not consider a good selection order of aggressors during test generation. For example, if the sensitization criterions of two aggressors conflict and the posterior aggressor has larger coupling capacitance, this sequential method may activate the aggressor with smaller coupling capacitance first and then fail to activate the more significant one.

B. SAT Problem Based Test Generation

Aiming to activate maximum crosstalk-induced delay on a

critical path, in the second method, we convert test generation into a Boolean Satisfiability problem, and then solve it by a normal SAT solver.

Solving an SAT problem is to find a solution of $f(x_1, x_2 \ldots, x_n) = 1$, where $f()$ is usually expressed in conjunctive normal form (CNF), i.e., product of sums. A CNF has a set of m clauses $\{C_1, C_2, ..., C_m\}$, each of which is a disjunction of a set of literals (a variable or its negation).

The key job of applying an SAT solver to test generation, is to model the test generation problem in CNF. A combinational net-list can be modeled in CNF [27] as an input for an SAT solver. In addition, a test for a PCPDF consists of two input vectors, (V_1, V_2), which propagates the transition signal along a sub-path to each aggressor and sensitizes the victim critical path. We use two CNF copies of the CUT to represent the circuit state under the first and second patterns respectively. For example, the CNF of a gate Y=AND (A, B) is:

$$(A_1 + \overline{Y_1})(B_1 + \overline{Y_1})(\overline{A_1} + \overline{B_1} + Y_1)$$
$$*(A_2 + \overline{Y_2})(B_2 + \overline{Y_2})(\overline{A_2} + \overline{B_2} + Y_2).$$

Path Sensitization

The path sensitization criterions need also be modeled in CNF. The criterion on the critical path can be converted to CNF formulas as that introduced in [28], but the criterions on the sub-paths cannot be simply converted. Since there may be many sub-paths to an aggressor that satisfy the timing requirement, we need a clever way to convert and combine all of them into CNF formulas.

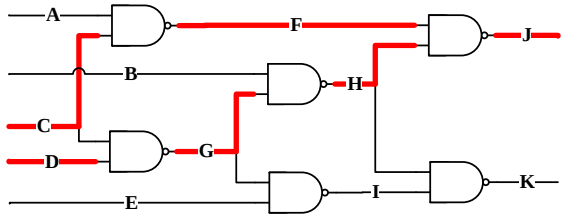

Figure 9. An example of the sub-paths to an aggressor line

Since an aggressor is activated when any proper sub-path to it is sensitized, we convert criterions of all the sub-paths to one aggressor into one CNF. An example of two sub-paths to J is shown in Figure 9.

Sub-path 1) ↑C-F-J.

Sub-path 2) ↓D-G-H-J.

The functional sensitization conditions are:

1) *C* (01), *F* (10), *J* (01), *A* (x1);

2) *D* (10), *G* (01), *H* (10), *J* (01), B (x1).

So the Boolean constraints for sub-path 1) and 2) are:

1) $\overline{C_1} C_2 F_1 \overline{F_2}\ \overline{J_1} J_2 A_2 = 1$;

2) $D_1 \overline{D_2}\ \overline{G_1} G_2 H_1 \overline{H_2}\ \overline{J_1} J_2 B_2 = 1$.

As mentioned above, an aggressor with many sub-paths is activated when constraints of any one sub-path to the aggressor are satisfied. Therefore the constraint of an aggressor is a Disjunctive Normal Form (DNF). Before using the SAT solver, we should translate DNFs to CNFs. Set two intermediate variables g_1 and g_2, the CNF of the above constraints can be represented as following:

$$(g_1 + g_2)(\overline{C_1} + \overline{g_1})(C_2 + \overline{g_1})(F_1 + \overline{g_1})(\overline{F_2} + \overline{g_1})$$
$$* (\overline{J_1} + \overline{g_1})(J_2 + \overline{g_1})(A_2 + \overline{g_1})$$
$$* (C_1 + \overline{C_2} + \overline{F_1} + F_2 + J_1 + \overline{J_2} + \overline{A_2} + g_1)$$
$$* (D_1 + \overline{g_2})(\overline{D_2} + \overline{g_2})(\overline{G_1} + \overline{g_2})(G_2 + \overline{g_2})(H_1 + \overline{g_2})$$
$$* (\overline{H_2} + \overline{g_2})(\overline{J_1} + \overline{g_2})(J_2 + \overline{g_2})(B_2 + \overline{g_2})$$
$$* (\overline{D_1} + D_2 + G_1 + \overline{G_2} + \overline{H_1} + H_2 + J_1 + \overline{J_2} + \overline{B_2} + g_2).$$

Weighted Max-SAT problem

To activate coupling latency on a critical path as large as possible, we define the Max-crosstalk-induced-delay problem as follows:

Given the CNF of a circuit with n nodes, $F(x_1, x_2, \ldots, x_n)$, find a variable assignment $X \in \{0,1\}^n$ such that X satisfies $F()$ and maximizes W in Equation (12) by satisfying logic constraints of sub-paths (i.e., c_{ij}) as many as possible.

$$W = \sum_{i=1}^{p} w_i (\bigvee_{j=1}^{m_i} c_{ij}) \quad (12)$$

Where the critical path has p aggressors ($a_1, a_2, \ldots a_p$), each aggressor a_i has m_i sub-paths, c_{ij} is the constraint of the sub-path $sp\text{-}a_{ij}$, and w_i is the weight of a_i (i.e., the coupling capacitance).

This problem can be represented as a Weighted-Max-SAT problem, whose goal is to find an assignment of values to variables, if one exists, where all clauses are satisfied, or if none exists, find an assignment which maximizes the sum of the weights of satisfied clauses. After combining CNFs of the CUT, of sensitizing the critical path, and of sensitizing the sub-paths (i.e., Equation (12)), we employ an SAT solver to find the solution.

This method simultaneously considers more sub-paths than the sequential method in the structural ATPG does, which brings more efficiency in targeting the maximum delay effects. It avoids the shortcoming of sequential sub-path sensitizing, but it may introduce the inaccuracy problem. As can be seen, the main difference of the two methods is the timing information used when sub-paths are selected: dynamically updated during test generation, or statically obtained before test generation. For the structural ATPG, sub-paths are selected during test generation, and accumulative extra delay on the PUT is updated when an aggressor is activated. However, for the SAT-based test generation, static timing information is used for sub-path selection. Since accumulative extra delay strongly depends on whether the anterior crosstalk effects are activated, crosstalk estimation in static timing analysis causes inaccuracy for sub-path selection. If the actual activated rate is low, the accuracy of the SAT based method will not be assurable. However, the SAT based method is more efficient than the sequential sub-path selection method especially when the activated rate is high.

Based on the above ideas, we add a preprocessing phase before sub-path selection. In this phase, the aggressors whose sensitizing constraints conflict with those of the critical path under test are deleted from the list of target coupling sites. This way ensures that this method has enough activated rate and accuracy.

VI. Experimental Results

In this section, the experimental results for the six largest full-scanned ISCAS89 benchmark circuits are given. We use

Cadence Encounter for physical design and parasitic extraction under 180-nm, 1.2-V, 6-metal technology. The test generation framework described above has been implemented in C++ and performed on a Pentium IV 2.66GHz desktop with 768MB memory. For these experiments, the coupling capacitances which form more than 20% of the total wire capacitance were selected as the initial coupling sites, which is also called spatial pruning of false aggressors in related works [15]. So the results in the section focus on temporal pruning and test generation.

A. Time Scale

Timescale, the size of a time slot, is the most important factor of the transition map based method. In the first experiment, we compare the coupling site identification by various *Timescale* values using circuit S5378, which has 3042 lines and 13615 coupling sites whose coupling capacitance is larger than 1fF.

TABLE III
Timescale effect on target site selection

Timescale (ps)	*No. of Target Sites*	*Select Rate (%)*	*Memory (byte)*	*Density %*	*CPU Time (s)*
50	3587	26.3	21K	26.6	3.2
10	2772	20.4	38K	19.1	3.2
5	2464	18.1	71K	12.5	3.5
2	1897	13.9	171K	6.6	3.9
1	1692	12.4	338K	3.7	4.2
0.5	1412	10.3	672K	2.0	4.2
0.1	1241	9.1	3M	0.41	18.6
0.05	1226	9.0	6M	0.26	29.8
TW	6338	46.6	48K	1	3.1

The comparison is shown in Table III. This experiment is similar to [19], but we compare the results under smaller *Timescale*. The last row marked with "TW" is the results of conventional timing window based method. Column 1 shows the size of a time slot which are from 50ps to 0.05ps. Column 2 shows the number of target fault sites identified. The "Select Rate" in Column 3 indicates the percentage which is equal to the number of target fault sites divided by the total number of coupling sites (13615). In Column "Memory", we list the memory to store the TMs of all the lines. It needs almost 48K byte memory to build the two "long" structure used for timing window. Column 5 shows the TM density defined in Section IV. Obviously, the density of timing windows is 1.

In comparison with the timing window based method, the TM based method can prune much more false target sites, up to 37.6% improvement (against the total number of coupling sites after spatial pruning) at the smallest scale. It shows that the proposed TM based STA technique can highly improve the accuracy of identifying coupling sites, and thus reduce much pessimism caused by inaccurate timing analysis.

The complexity of the TM based method is similar to that of the timing window based method. As shown in Table III, as *Timescale* continuously decreases, the TM based method can prune more pessimism at the cost of more time and memory penalty on timing analysis. The trade-off between accuracy and efficiency can be explored by controlling the size of a time slot. If there is high accuracy requirement, a smaller *Timescale* can be used. If the CPU efficiency is more important for larger circuits, a larger *Timescale* will be preferred. So the proposed TM based timing analysis method is highly scalable.

B. Critical Path Selection

The conventional static timing analysis ignoring crosstalk will result in inaccurate path delays. Tayade et al. [29] proposed a method to estimate the maximum path delay in the presence of coupling noise considering both logic and timing constraints, and showed that ignoring aggressor constraints can result in missing some real critical paths.

In our method, path selection is a preliminary phase of test generation, so only timing constraints of aggressors are considered in path selection. Due to assuming that all aggressors identified by TM based timing analysis can be activated, the crosstalk-induced latency is more pessimistic than in real conditions. But the object is not to miss possible critical paths during path selection, and since TM based timing analysis has reduced much pessimism caused by inaccurate timing analysis, we can rely on it to select target paths for PCPDFs and leave the work of solving logic constraints to ATPG.

TABLE IV
Critical path selection

Circuit Names	*S5378*	*S9234*	*S13207*	*S15850*	*S38417*	*S38584*
Considering crosstalk	*462*	*21190*	*84572*	*382670*	*3995*	*48493*
Not considering	*314*	*20181*	*29234*	*263755*	*1393*	*19964*
No. of Different Paths of top 100	*4*	*6*	*9*	*5*	*11*	*9*

Table IV shows that the number of critical paths considering crosstalk or not in critical path selection. In this experiment, a critical path is a path whose calculated path delay is larger than 90% of the threshold, which is defined as the max path delay without considering crosstalk. Path ranking is also important when the top K critical paths are to be selected for test [30]. Row 4 shows the number of paths missed due to incorrect ranking, when top 100 critical paths are selected for CUT.

As the results shows, considering the crosstalk effect, the increase ratio of number of target paths varies from 5% to 2 times, and there exists on average 6% differentiation in top 100 path ranking. As the trend to have many near critical paths with tight timing constraints during synthesis, the differentiation may become more prominent. So it is important to consider crosstalk in static timing analysis and critical path selection.

C. Test Generation

Experimental results of test generation are shown in Table V. In this experiment, only 100 robustly testable longest paths are selected, and 100 test patterns are generated for these 100 PCPDFs respectively. On various logical paths, a physical coupling site is denoted by various logical sites. In this experiment, all sites are logical sites, whereas the coupling sites

in the first experiment are physical sites.

Due to lack of a Weighted-Max-SAT solver, we expand the capability of a conventional SAT solver to find solutions. Considering the number of main aggressors (which dominate the coupling capacitance) of most clusters are less than 5, we transform the Weighted-Max-SAT problem to some SAT problems according to the combinations of aggressors. This method will decrease the efficiency but can be improved by some heuristic technique. By using a Weighted-Max-SAT solver in the future, more accuracy and efficiency can be achieved. We employ the Mini-SAT solver [31] to find solutions, and the maximum number of sub-paths to an aggressor is limited to 16.

Column 2 shows the numbers of total logical coupling sites on target paths. Column 4 indicates the numbers of target sites which are selected by TM overlapping analysis. Column 5 shows the number of singly testable sites based on S-PCPDF [32]. An S-PCPDF untestable site means that there is no pattern can robustly sensitize the critical path through the victim line v and cause the aggressor line to switch at $v.t_{\text{switch}}^{\text{F/R}}$ simultaneously. As the results shows, 30%~50% target sites are S-PCPDF testable, and only these sites are considered in fault coverage in PCPDF based test generation.

The comparison of the results of structural ATPG based method and SAT-solver based method is shown in the last 8 columns. "No. of Activated Sites" is the number of activated sites in 100 PCPDFs by two approaches. The column "Activated rate of sites" is the ratio of number of activated sites to the number of S-PCPDF testable sites. The column "Activated rate of capacitance" is the ratio of capacitance of activated sites to the total capacitance of testable sites. In practice, the second ratio is the better indicator of test quality. To our knowledge, this is the first time to use activated coupling capacitance to evaluate the test quality for testing crosstalk-induced effects. The column "CPU time (s)" gives the total CPU time including both timing analysis and test generation.

Since there are 4 testable sites on average in a target PCPDF fault using the bigger scale (5ps), the activated rate of our method is correspondingly high. Using the smaller timescale, the number of target sites is much less, and then we can achieve higher activated rate with some time penalty (spending less time for test generation but more time for target fault selection).

Furthermore, it is observable that the SAT-solver based method gains more efficiency than the structural ATPG method does, since the SAT based method is good at searching bigger solution space. Due to considering not only more sub-paths together but also the maximization of coupling capacitance in the SAT based test generation, the activated rate of capacitance is also improved by 3.8%.

It is difficult to compare the ATPG results directly with related works, since the crosstalk-induced faults are strongly related to physical and timing information of the circuit under test. We roughly compare the CPU time as follows.

In [6], the average CPU time of test generation for 100 critical paths in which each critical path has five coupling sites on average is around 1/4 hour for larger circuits of ISCAS89. The method employing complicated timed ATPG algorithm proposed in [8] takes almost 4 hours to deal with large ISCAS85 circuits. The method proposed in [15] achieves a CPU performance similar to our structural ATPG method (we know this because it compared with our previous work in [32]).

It was reported that the average reduction ratio of false-aggressor candidates in [15] is 91.15% using spatial pruning, timing pruning, and test generation on 100 randomly selected aggressors. As shown in Table V, the average ratio of No. of testable sites (Column 5) to No. of total sites (Column 2)

TABLE V
Results of test generation

Circuit Names	*No. of Total Sites*	*Time Scale (ps)*	*No. of Candidate Sites*	*No. of Testable Sites*	*Structural ATPG based*				*SAT based*			
					No. of Activated Sites	**Activated rate of Sites (%)**	**Activated rate of Capacitance (%)**	**CPU Time (s)**	**No. of Activated Sites**	**Activated rate of Sites (%)**	**Activated rate of Capacitance (%)**	**CPU Time (s)**
S5378	4943	5	1570	662	633	95.6	90.4	4.7	640	96.7	96.9	3.2
		0.5	782	423	415	98.1	93.0	6.6	403	95.3	98.2	4.4
S9234	4404	5	1389	584	539	92.3	86.4	41.5	539	92.4	94.7	13.9
		0.5	669	331	309	93.3	94.5	69.5	319	96.4	95.4	17.2
S13207	2150	5	809	250	241	96.4	91.2	59.7	237	94.8	97.7	19.4
		0.5	542	229	225	98.3	89.5	88.7	225	98.2	99.3	26.4
S15850	3426	5	761	148	148	100	100	298.8	148	100	100	54.8
		0.5	459	95	95	100	100	525.1	95	100	100	93.2
S38417	2342	5	879	389	308	79.2	77.5	260.1	315	80.9	85.5	56.3
		0.5	388	203	186	91.6	85.4	408.6	194	95.6	94.3	84.2
S38584	3210	5	696	302	300	99.3	99.7	291.8	299	99.1	99.7	59.1
		0.5	435	195	195	100	100	429.6	195	100	100	74.5
Avg.	3413	5	1017	389	361	92.8	90.9	--	363	93.3	95.7	--
		0.5	545	246	238	96.8	94.0		238	96.9	97.8	

is 9.39%, while No. of total sites equals the number of aggressor candidates after spatial pruning. It indicates that even after spatial pruning, our method can still achieve an additional 90.61% reduction of false-aggressor candidates by timing and ATPG pruning. Considering spatial pruning itself can reduce 50%-95% false-aggressor candidates, our method totally can achieve a 95.3%-99.95% reduction of false-aggressor candidates. As stated before, the timing pruning method in [15] is basically based on timing window analysis, while our results in Table III show transition map based timing analysis can identify more false target sites (e.g., 37.6% improvement) in comparison with the timing window based method. This explains why our method can achieve a larger reduction ratio of false-aggressor candidates.

Furthermore, [15] didn't consider crosstalk-induced delay on critical path selection and didn't update the accumulative crosstalk-induced delay during test generation. So our approach is more effective for generating patterns targeting maximum path delay with multiple crosstalk effects.

D. HSPICE Simulation

In the fourth experiment, HSPICE simulation was performed on S5378 to validate the test generation results. Table VI compares the path delays using conventional PDF patterns for five testable critical paths in s5378, with those using patterns generated by our method, each of which has 5 activated coupling sites. Row 1 lists the path ID. Row 2 reports the average path delay of 50 conventional PDF patterns when the don't-care bits are randomly filled with 1/0. Row 3 shows the path delay of the pattern generated by the proposed method, and row 4 gives the increase ratio of the path delay in comparison with conventional PDF patterns.

It can be seen that the path delay has a notable increase (9.6% on average) under the deterministically generated pattern targeting worst case coupling effects, which validate the effectiveness of our method on crosstalk effect activation.

TABLE VI

Path delay increase by the generated patterns

S5378 (ns)	*Path1*	*Path2*	*Path3*	*Path4*	*Path5*
Conventional	2.66	2.80	2.78	2.63	2.49
Proposed	2.91	3.04	3.05	2.89	2.75
Inc. (%)	**9.4**	**8.6**	**9.7**	**9.9**	**10.4**

VII. LIMITATIONS

With an accurate delay modeling of the CUT, the proposed PCPDF test generation method is also applicable to delay fault diagnosis since the target crosstalk effects are explicitly activated. However, the dependence on accurate delay modeling is against the reality of process variations. To relieve this limitation, the min-max delay model, instead of the mean delay model used in this paper, can be adopted in the transition map based timing analysis to deal with delay variations to some extent. For simplicity, suppose any delay falls into an interval [d_{min}, d_{max}], there are two situations when applying the TM-based analysis method to the min-max delay model:

1) If $d_{max} - d_{min} \leq Timescale/2$, there is no difference between using the mean delay model and the min-max delay model, because the proposed method has a natural toleration of delay variation regarding the specified size of a time slot.

2) Else if $\left\lceil \frac{d_{max}-d_{min}}{Timescale} \right\rceil = k$, where k is an integer no less than 1, the transition map of line L can be calculated as follows:

$$\cup_{i=0}^{k} \left(L.TM^{\mathrm{R/F}}(d_{min}) \gg i \right), \tag{13}$$

where $L.TM^{\mathrm{R/F}}(d_{min})$ is the $L.TM^{\mathrm{R/F}}$obtained using d_{min}'s in the mean delay model, $\cup$ is the bit "OR" operator and ">>" is the right shift operator.

Using the min-max delay model only influences the calculation of transition maps, which will increase the density of transition maps, bringing more aggressor candidates. There is basically no change on test generation. So it only causes an increased time penalty when more candidates of aggressors and sub-paths need to be considered.

In summary, the application of the proposed methodology to smaller technologies depends on the accuracy of the obtained timing model. *Timescale* is an adjustable parameter to tolerate delay variations including process variations, or temperature and voltage related variations, but only to some extent. When the total delay variation is large, the min-max delay model can be used. However, changing to the min-max delay model may lead to overly pessimistic coupling delay analysis results (like timing window based methods), which would lead to targeting a large number of unnecessary aggressors during test generation.

Another concern of testing considering crosstalk, is the

contradiction between test escape and over-testing. With more logic constraints on aggressors, it is at higher risk that the CUT is over-tested in delay testing. To reduce this risk, some pseudo functional testing techniques [33][34] can be borrowed to constrain the PCPDF test generation for only generating legal functional tests. For instance, functional constraints (e.g., on illegal states, or from high-level specifications) can be translated into virtual circuits [35] of the CUT, so that over-testing vectors will be removed if they violate the logic constraints of the virtual circuits.

VIII. Conclusions

This paper proposes a test generation technique based on the PCPDF model, which finds patterns activating the worst case crosstalk effects and in turn maximizing the delay of critical paths. The presented method processes the necessary timing information to identify crosstalk-sensitive critical paths, followed by an ordinary path delay ATPG engine or an SAT solver to generate patterns under unified timing and logic constraints. Using a transition map instead of a timing window in timing analysis, our method can trade accuracy for efficiency by controlling the size of a time slot used in transition map, with a smaller *Timescale* for accurate crosstalk-induced delay computation and a larger one for rapid test generation of lager scale circuits, which makes this approach highly scalable.

Experimental results on ISCAS89 benchmark circuit show that this method incurs relatively small processing time by means of avoiding complex timing processing in the ATPG algorithm. It shows transition map based timing analysis can identify more false target sites (e.g., 37.6% improvement) in comparison with the timing window based method. We also demonstrate that, for the benchmarks implemented in a 180nm technology lib, crosstalk will affect selection results of critical paths and may contribute to 9.6% path delay increase under the generated patterns targeting the worst case coupling effects.

Acknowledgment

The authors would like to thank Prof. K.-T. Cheng for his valuable suggestions on this work. They are grateful to the reviewers for their detailed review that helped them to improve the presentation and experiments of this work.

References

[1] K.-T. Cheng, S. Dey, M. Rodgers, and K. Roy, “Test challenges for deep sub-micron technologies,” in *Proc. Design Automation Conf.*, 2000, pp.142-149.

[2] A. Tetelbaum, “Statistical STA: Crosstalk Aspect,” in *Proc. IEEE International Conference on Integrated Circuit Design and Technology*, 2007, pp.1-6.

[3] T. Zhang, S. S. Sapatnekar, “Simultaneous shield and buffer insertion for crosstalk noise reduction in global routing” *IEEE Trans. Very Large Scale Integ. Sys.*, vol.15, no.6, pp.624-636, June 2007.

[4] S. S. Sapatnekar, “A timing model incorporating the effect of crosstalk on delay and its application to optimal channel routing,” *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol.19, no.5, pp.550-559, May 2000.

[5] B. Kruseman, A. Majhi, and G. Gronthoud, “On performance testing with path delay patterns,” *in Proc. VLSI Test Symp.*, 2007, pp.29-34.

[6] W.-Y. Chen, S. K. Gupta, and M. A. Breuer, “Test generation for crosstalk-induced delay in integrated circuits,” in *Proc. Int. Test Conf.*, 1999, pp.191-200.

[7] S. Irajpour, S. K. Gupta, and M. A. Breuer, “Timing-independent testing of crosstalk in the presence of delay producing defects using surrogate fault models,” in *Proc. Int. Test Conf.*, 2004, pp.1024-1033.

[8] K. P. Ganeshpure, S. Kundu, “On ATPG for multiple aggressor crosstalk faults in presence of gate delays,” in *Proc. Int. Test Conf.*, 2007, p. 2.1.

[9] A. Krstic, J.-J. Liou, Y.-M. Jiang and K.-T. Cheng, “Delay testing considering crosstalk-induced effects,” in *Proc. Int. Test Conf.*, 2001, pp.558-567.

[10] B. C. Paul, K. Roy, “Testing cross-talk induced delay faults in static CMOS circuit through dynamic timing analysis,” in *Proc. Int. Test Conf.*, 2002, pp.384-390.

[11] Y. Ran, A. Kondratyev, K. H. Tseng, Y. Watanabe, M. Marek-Sadowska, “Eliminating false positives in crosstalk noise analysis,” *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol.24, no.9, pp.1406-1419, Sept. 2005.

[12] J. Lee and M. Tehranipoor, “A novel pattern generation framework for inducing maximum crosstalk effects on delay-sensitive paths,” in *Proc. Int. Test Conf.*, 2008, p. 33.2.

[13] H. Li and X. Li, “Selection of Crosstalk-induced Faults in Enhanced Delay Test”, *Journal of Electronic Testing: Theory and Applications*, vol.21, no.2, pp.181-195, April 2005.

[14] M. Zhang, H. Li, X. Li, “Multiple coupling effects oriented path delay test generation,” in *Proc. VLSI Test Symp.*, 2008, pp.383-388.

[15] S. Chun, T. Kim, and S. Kang, “ATPG-XP: test generation form maximal crosstalk-induced faults,” *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol. 28, no.9, pp. 1401-1413, Sept. 2009.

[16] M. Zhang, X. Li, “Test generation for crosstalk glitches considering multiple coupling effects,” in *Proc. Asian Test Symp.*, 2007, pp.259-264.

[17] A. Glebov, S. Gavrilov, D. Blaauw, S. Sirichotiyakul, Oh. Chanhee, V. Zolotov, “False-noise analysis using logic implications,” in *Proc. Int. Conf. Comp.-Aided Des.*, 2001, pp.515-521.

[18] H. Takahashi, K. J. Keller, K. T. Le, K. K. Saluja and Y. Takamatsu, “A method for reducing the target fault list of crosstalk faults in synchronous sequential circuits,” *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol.24, no.2, pp.256-263, Feb. 2005.

[19] P. Chen, Y. Kukimoto, and K. Keutzer, "Refining switching window by time slots for crosstalk noise calculation," in *Proc. Int. Conf. Comp.-Aided Des.*, 2002, pp.583-586.

[20] H. Zhou, "Timing analysis with crosstalk is a fixpoint on a complete lattice," *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol.22, no.9, pp.1261-1269, Sept. 2003.

[21] M. Kuhlmann, S. S. Sapatnekar," Exact and efficient crosstalk estimation," *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol.20, no.7, pp.858-866, July 2001.

[22] D. Li, D. Blaauw, P. Mazumder, "Accurate crosstalk noise modeling for early signal integrity analysis," *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol.22, no.5, pp.627-634, May 2003.

[23] T. Sato, Y. Cao, K. Agarwal, D. Sylvester, H Chenming "Bidirectional closed-form transformation between on-chip coupling noise waveforms and interconnect delay-change curves," *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol.22, no.5, pp.560-572, May 2003.

[24] K. Tseng, M. Horowitz, "False coupling exploration in timing analysis," *IEEE Trans. Comp..-Aided Des. Integ. Cir. Sys.*, vol.24, no.11, pp.1795-1805, Nov. 2005.

[25] K.-T. Cheng and H.-C. Chen, "Classification and identification of nonrobust untestable path delay faults," *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol.15, no.8, pp.845-853, Aug. 1996.

[26] K. Fuchs, F. Fink, and M. H. Schulz, "DYNAMITE: an efficient automatic test pattern generation system for path delay faults," *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol.10, no.10, pp.1323-1335, Oct. 1991.

[27] T. Larrabee, "Test pattern generation using boolean satisfiablity," *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol.11, no.1, pp.4-15, Jan. 1992.

[28] S. Y. Lu, M. T. Hsieh, J. J. Liu, "An efficient SAT-based path delay fault ATPG with an unified sensitization model," in *Proc. Int. Test Conf.*, 2007, p. 10.3.

[29] R. Tayade, J. A. Abraham, "Critical path selection for delay test considering coupling noise," in *Proc. Euro. Test Symp.*, 2008, pp.119-124.

[30] Q. Wangqi and D. Walker, "An efficient algorithm for finding the k longest testable paths through each gate in a combinational circuit," in *Proc. Int. Test Conf.*, 2003, pp.592-601.

[31] N. Een and N. Sorensson, "An extensible SAT-solver," *Lecture Notes in Computer Science*. Heidelberg Springer, 2004, 2919502-518

[32] H. Li, P. Shen and X. Li, "Robust test generation for precise crosstalk-induced path delay faults," *in Proc. VLSI Test Symp.*, 2006, pp.34-39.

[33] Y.-C. Lin, F. Lu, and K.-T. Cheng, "Pseudo-functional Testing," *IEEE Trans. Comp.-Aided Des. Integ. Cir. Sys.*, vol.25, no.8, pp.1535-1546, August 2006.

[34] F. Yuan, Q. Xu, "Compression-aware Pseudo-Functional Testing," in *Proc. Int. Test Conf.*, 2009, p. 9.1.

[35] R. S. Tupuri, A. Krishnamachary and J. A. Abraham, "Test Generation for Gigahertz Processors Using an Automatic Functional Constraint Extractors," in *Proc. Design Automation Conf.*, 1999, pp.647-652.

Minjin Zhang received his B.S. and M.S. degrees from Wuhan University, Wuhan, China, in 2002 and 2004, respectively. He received his Ph.D. degree from Institute of Computing Technology, Chinese Academy of Sciences (CAS) in 2009. He is currently an assistant professor at Institute of Automation, CAS. His research interests include signal integrity, delay test and test generation.

Huawei Li (M'00-SM'09) received her B.S degree in computer science from Xiangtan University, Xiangtan, China, in 1996, and M.S. and Ph.D. degrees from Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), in 1999 and 2001, respectively.

She is currently a professor at ICT, CAS. Her research interests include VLSI/SoC design verification and test generation, delay test, and dependable computing.

Prof. Li serves as Secretary General of CCF (China Computer Federation) Technical Committee on Fault Tolerant Computing since 2008. She also served as program chairs of IEEE Asian Test Symposium (ATS) 2007, and IEEE Workshop on RTL and High Level Testing (WRTLT) 2003. She is a senior member of IEEE.

Xaowei Li (M'00-SM'04) received his B.Eng. and M.Eng. Degrees in computer science from Hefei University of Technology, Hefei, China, in 1985 and 1988, respectively, and his Ph.D. degree in Computer Science from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), in 1991.

From 1991 to 2000, he was an Assistant Professor and an Associate Professor (since 1993) in the Department of Computer Science, Peking University, Beijing, China. He joined the ICT, CAS as a Professor in 2000. He is now the deputy director of the Key Lab. of Computer System and Architecture, CAS. His research interests include VLSI Testing, design for testability, design verification, software testing, dependable computing, and wireless sensor networks. He has co-published over 150 papers in academic journals and international conference, hold 21 patents and 29 software copyrights.

Prof. Li served as Chair of CCF (China Computer Federation) Technical Committee on Fault Tolerant Computing since 2008. He served as IEEE Asian Pacific Regional TTTC (Test Technology Technical Council) Vice Chair since 2004. He served as the Steering Committee Vice-chair of IEEE Asian Test Symposium (ATS) since 2007; he also served as the Steering Committee Chair of IEEE Workshop on RTL and High Level Testing (WRTLT). In addition, he serves on the Technical Program Committee of several IEEE and ACM conferences, including VTS, DATE, ASP-DAC, PRDC, etc. He also serves as member of editorial board of JCST, JOLPE, JETTA, etc.