STAX: Statistical Crosstalk Target Set Compaction

Shahin Nazarian, Massoud Pedram, Sandeep K. Gupta, Melvin A. Breuer
University of Southern California, Dept. of Electrical Engineering

ABSTRACT

This paper presents STAX, a crosstalk target set compaction framework to reduce the complexity of the crosstalk ATPG process by pruning non-fault-producing targets. In general, existing pruning techniques do not employ their processes in a cost-effective manner. Neither do they handle process variations properly. To address the first weakness, this paper presents a framework to determine a sequence of available analysis and pruning tool invocations to prune as many of the crosstalk targets as fast as possible. As a result, an initially enormous collection of crosstalk targets is usually reduced to a very small set of targets via a vectorless process. A statistical static timing analyzer is developed and embedded to address the second shortcoming of existing approaches. Experimental results on ISCAS'85 benchmark demonstrate that STAX greatly improves the runtime compared to other crosstalk target pruning methodologies, including ATPG, with no prior target set compaction.

KEYWORDS

ATPG, fault-producing target, compaction degree, pruning power, safe target, statistical static timing analyzer

1. INTRODUCTION

As the layout geometries in recent CMOS process technologies scales down to 65nm and below, increases in transistor packing density and operational frequency of VLSI circuits aggravate the noise effects, including crosstalk noise. This noise is caused by unwanted capacitive coupling between a pair of interconnect lines, referred to as a *crosstalk site*. Three types of crosstalk effects namely pulse (glitch,) slowdown, and speedup effects can be associated with each site [1]. This paper focuses on the slowdown target set compaction. When the signals on the interconnect lines of a crosstalk site make opposite transitions and their arrival times are close to each other, a slowdown effect occurs on both signal transitions. Each line can be considered as the victim line, while the other as the aggressor. Each of these lines is associated with two slowdown *targets*, namely the slowdown of the rising or the falling signal transition; hence there are four slowdown targets at each site. The slowdown of a signal transition can result in faulty circuit behavior, in which case the target is referred to as a *fault-producing target* (FT). Otherwise, the target is called a *safe target* (ST).

It seems necessary to generate test for FTs associated with each coupled interconnect line. However, a large VLSI circuit might contain a huge number of coupled interconnect lines, and thus a large population of crosstalk targets. In practice, only a small set of targets can result in faulty circuit behavior. Considering the high complexity of test generation for each target, it is reasonable to try and prune as many targets as possible and thereby reduce the size of target set that consists of the ones that should be considered during test generation. This procedure, which is referred to as *crosstalk target set compaction*, outputs a set of crosstalk targets that must include all of the FTs but may also contain some non-fault-producing targets. This could happen, for example, if a non-fault-producing target cannot be proven as safe by the available pruning tools.

Differences between identical features in a certain lithographic process are referred to as process variations, and such mask differences tend to rise rapidly as the technology scales down. In addition to these manufacturing-induced variations, environmental variations along with device/interconnect aging processes (e.g., hot electron effects and electro-migration,) tend to generate a rather large deviation of key circuit parameters from their designed values. These phenomena create parasitic and electrical parameter uncertainties for various elements in the circuit and cause significant timing variations. Consequently, highly sophisticated and robust crosstalk-aware performance analysis and optimization tools are needed to account for these variational effects. The impact of a crosstalk target on the correct operation of a circuit depends on logic values, signal arrival times and slews, and parasitic and electrical parameters. The timing and electrical parameters are strongly dependent on process variations.

Up until recently, corner-based timing analysis techniques, such as static timing analysis (STA), were used as relatively fast techniques to address the concerns related to various sources of variation in VLSI circuits. In general, corner-based techniques tend to overestimate circuit delay and noise effects. These techniques can also result in underestimation of circuit delay and noise because these metrics are non-monotone functions of some circuit parameters. Exacerbating the situation, it is nontrivial to find the worst-case value for each parameter that would result in the worst-case delay or noise. Therefore, crosstalk target set compaction using corner-based timing analysis tools will not be effective in future process technology nodes. Statistical analysis is thus viewed as an essential methodology for nanometer process technologies, which enables application of the actual statistics of the process technology parameters for accurate calculation of circuit characteristics such as gate and interconnect delay [2],[3].

The idea of applying a crosstalk target set compaction tool prior to using ATPG was first introduced in [1]. A qualitative and detailed discussion of the target identification ideas was then given in [4]. Following this work, a pruning method was proposed in [5] for crosstalk target identification in sequential circuits. A different method was proposed in [6] to prune redundant crosstalk

faults in sequential and combinational circuits. Unfortunately, none of these techniques address the problem of how to utilize the filtering resources in a cost-effective manner. In addition, none considers the effect of process variations. Departing from this practice, the XIDEN target identification framework for crosstalk pulse [7] and slowdown [8] showed how to find effective sequences of available timing and parameter extraction tools and filtering tools that drastically reduce the cost of pruning and overall testing. On the weak side, process variations were handled by employing corner-based analysis techniques.

In this paper we present STAX, a STatistical Xtalk (crosstalk) target set compaction methodology to evaluate crosstalk slowdown in the existence of process variations and to efficiently filter as many targets as possible. A number of timing analysis, extraction, and filtering tools with different qualities and runtime complexities are incorporated in the STAX framework. The first question that comes to mind is: why not use the best timing analyzer, extractor, and filtering tools available to prune as many targets as possible. The answer is that we can achieve the same level of pruning in much shorter CPU times by not restricting ourselves to only these tools. The general idea is to use less accurate and fast tools in the initial stages to process a large initial set of targets and prune as many targets as possible, and then use the more accurate, but computationally more expensive tools, on the remaining (much smaller) set of targets in later stages.

Target set compaction is dependent on the ordering of pruning tools. The ratio of the number of remaining targets after a sequence of tool invocations to the number of initial crosstalk targets is referred to the *compaction degree*. Different tool sequences with identical compaction degrees can have computational costs that can differ by orders of magnitude. Therefore, the goal is to find effective sequence(s) of tool invocations to provide the highest compaction degree in the least amount of CPU time.

Table 1: Notation and descriptions

Symbol	Description	Symbol	Description
V	Victim	E^i	Extractor *i*
A	Aggressor	F^i	Filter *i*
δ	AT(V)−AT(A)	T^i	Timing analyzer *i*
C_c	Coupling capacitance	$AT_{\mu-3\sigma}(V)$	Earliest arrival time at victim
PO	Primary output	$AT_{\mu+3\sigma}(V)$	Latest arrival time at victim
R(V)	Victim's required time	$SD_{max\text{-}Cc}$	Maximum slowdown by C_c

The remainder of this paper is organized as follows. In section 02 the variation-aware crosstalk model, extraction and statistical timing analysis tools used in STAX are reviewed. The crosstalk slowdown filters are discussed in section 3. In section 4 we review how statistical timing analyzers, extractors, and filters are placed into an efficient order. Sections 5 and 6 explain some of our experimental results and summary, respectively. Notation to be used throughout this paper is summarized in Table 1.

2. MODELING, EXTRACTION, ANALYSIS

2.1 Coupled Interconnect Characterization/Modeling

The distributed RC-π model of Figure 1(a) is used to model a pair of capacitively-coupled interconnect lines while considering the local variations of physical parameters, such as line width and thickness. In this circuit, each RC-π stage represents an interconnect segment of predefined length, L_{seg}. The coupling between two interconnect lines along segment *i* is captured by the coupling capacitance C_{ci}. Moreover, the self capacitance and resistance of the victim line in segment *i* are denoted by C_{vi} and R_{vi}, respectively. Although lengths of all segments are identical, due to process variations, the parameters (i.e., the value of different elements) of the corresponding electrical circuit are different. The variation of physical parameters, such as interconnect width and thickness, along the interconnect line is due to IC manufacturing defects, neighboring metal lines, optical proximity, chemical mechanical polishing (CMP) metal process, etc.

The following procedure is used for calculating electrical parameters of the distributed model. First, complete physical outlines of the coupled interconnect lines are generated, including information about their width, height, and interlayer dielectric thickness along their length. This physical outline is used to calculate resultant electrical parameters for each interconnect segment by using a scheme similar to that introduced in [9].

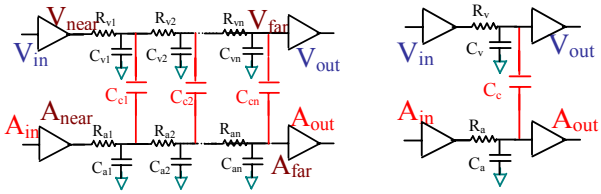


Figure 1: (a) Distributed RC-π model of a crosstalk site, (b) Lumped RC-π model of the crosstalk site

The heuristic explained below is used as a model order reduction technique to construct the variational circuit model of coupled interconnect lines as a variational coupled single RC-π model as depicted in Figure 1(b). In this RC-π model, the mean value of each quantity of interest (i.e., R_v, C_v, R_a, C_a, C_c) is calculated as the summation of the mean values of all the coupled segments in the distributed circuit model of Figure 1. The variance of each quantity is however calculated as weighted summation of the variances of the coupled segments. In particular, the weights are designed to monotonically decrease from the near-end of the coupled line toward the far-end. This is because we have empirically observed that the effect of segment variations on the output delay at the far-end of the coupled lines decreases as one visits segments starting from the near-end toward the far-end.

The key advantage of the proposed modeling approach is the ability to locally capture the effect of process

variations on each interconnect segment. This is done by directly calculating the corresponding values of local resistance and capacitance of the RC-π model based on the exact information about the actual geometry of the interconnect lines in each segment.

To achieve convergence in the desired statistical properties of the output variables, Monte Carlo simulation is performed, where we calculate the mean and variance of a collection of samples, each comprising of a large number of units in the population under study. According to our experiments, a *sample size* of 2500 is suitable to use, i.e., the population generation and electrical parameter extraction steps are iterated 2500 times to achieve convergence in the desired statistical properties for each sample. The number of samples (or *sample count*) is then selected so that a 98% confidence level with 1ps error in the estimates of mean and variance of interconnect delay is achieved. (Recall that the effectiveness of Monte Carlo simulation technique is based on the fact that, regardless of the population distribution, the sample distribution becomes normal; hence, a well-defined stopping criterion exists from which the confidence interval for the final estimate can be calculated.) There is a tradeoff between the level of accuracy and complexity of closed-form expressions. As the number of input parameters increase, lower order models such as modeling of the delay random variable as a linear function of sources of variation becomes more suitable. According to our experimental setup, we have found that the 2nd order modeling is the most successful for capturing the distribution properties of crosstalk-affected delay on the victim line:

$$\text{mean(delay)} = \sum_{\text{parameter } i} (A_i x_i^2 + B_i x_i) \quad (1)$$

$$\text{variance(delay)} = \sum_{\text{parameter } i} (C_i x_i^2 + D_i x_i) \quad (2)$$

where x_i is a physical parameter, such as wire width or length, and A_i to D_i are regression coefficients found by using statistical analysis and curve fitting techniques.

2.2 Parameter Extraction/Estimation

To determine the quantitative value of crosstalk-affected delay of a crosstalk target, knowledge of the parametric electrical values associated with the aggressor and victim lines is required. The fastest way to estimate the value of an electrical parameter is to keep track of pre-computed upper and lower bounds for the parameter, given the CMOS process manufacturing technology and the circuit. It may be sufficient to rule out a target as a fault by using these bounds for some parameters, and approximate and/or extract values for other parameters. In this case, there is no need to determine more accurate values of the parameters associated with the crosstalk site. However, it is more accurate to estimate the value of a parameter by extraction. But the cost of extraction is higher than bound approximation. An extractor is a tool that determines (estimates) the values of a set of parameters within a certain degree of accuracy. The list of extractor models utilized in STAX is reported in Table 2. The total cost of extraction is approximated by a cost per site of the utilized extractor tool multiplied by the number of crosstalk sites in the input set of targets, which is passed to the extractor.

Table 2: Extractors modeled in STAX

E	(Extracted parameters and extraction accuracies)	Cost (Sec/Site)
E^1	{(C_c, 45%), (C_v, 45%), (R_v, 45%)}	0.005
E^2	{(C_c, 45%), (C_v, 30%), (R_v, 35%)}	0.02
E^3	{(C_c, 30%), (C_v, 30%), (R_v, 30%)}	0.05
E^4	{(C_c, 15%), (C_v, 15%), (R_v, 15%)}	0.18
E^5	{(C_c, 10%), (C_v, 10%), (R_v, 10%)}	0.45

2.3 Statistical Timing Analysis Tools

Logical level statistical timing analysis is an effective way of modeling IC manufacturing process variations. It is also an important aspect of determining whether or not a coupling capacitance subjected to process variations can transform a crosstalk target into a crosstalk fault. Using *statistical static timing analysis* (SSTA) for all signal transitions in a circuit, one can determine statistical characteristics such as the mean and variance of key design attributes at intermediate circuit nodes, such signal *arrival times* and *required times*.

The required time R(V) associated with a line V is the maximum (latest) time at which a transition can occur at this line, yet propagate to all primary output lines before the end of a clock period (while satisfying the flip-flop set-up time.) In principle, the upper bound (μ+3σ) on the arrival time at victim node V, i.e., $AT_{\mu+3\sigma}(V)$, should be less than the required time at V, namely R(V).

2.3.1 SSTA Tools

The accuracy of SSTA depends on the delay models used for logic cells and crosstalk sites. We have incorporated a SSTA tool in STAX that exploits the variation-aware modeling technique discussed in Section 2.1 for the coupled interconnects, and relies on the non-linear delay modeling of [2] for logic cells. In this way, the SSTA tool can calculate the arrival time distributions of each line through a forward traversal process. Required times are computed using a backward traversal procedure as in [10].

Similar to trade-offs between computation time and accuracy employed for extraction, the SSTA tool can operate at different levels of accuracy and run-time complexity. In this paper, the run-time complexity of the SSTA tools is considered as part of the cost of the filter(s) that employ them. (The STAX filters and their characteristics will be explained in Section 3.)

In general, SSTA computes the μ±3σ bounds for the timing parameter of interest, which are needed by filters. Based on the computed bounds, the filters determine whether to prune a target. Details are provided below.

SSTA T^1: Arrival time calculation considering the crosstalk effects

T^1 calculates the (variational) arrival times (i.e., μ±3σ values) of all circuit lines by a forward traversal algorithm. It is executed on a site-by-site basis for each site associated

with the set of crosstalk targets which are passed to T^1 by some filter. The arrival times are computed by considering the slowdown effect of the crosstalk target that is under consideration. This is a CPU-intensive task that requires iterative calculation of the maximum slowdown of the victim output as a function of the overlap between the arrival time ranges of the victim and the aggressor. Each time T^1 is called to process a crosstalk target, the *crosstalk-affected arrival times* (denoted by CT) of the victim and aggressor lines as well as all nodes in the fanout cones of the victim and aggressor lines are computed.

SSTA T^2: Required time calculation ignoring the crosstalk effects

T^2 calculates the (variational) required times (i.e., $\mu\pm3\sigma$ values) of all circuit lines through a backward traversal approach. To do this, T^2 must first calculate arrival times by a forward traversal. In this case, however, the arrival times are calculated without considering the slowdown effect of any crosstalk target in the circuit. As a result, given the current (undecided) set of crosstalk targets in the circuit, T^2 is executed once to calculate the required times for all nodes in the circuit.

3. FILTERING

A *filter* is a tool to assess a sufficient set of conditions that can confirm whether or not a crosstalk target is non-fault-producing. The conditions can be related to circuit parameters, such as the coupling capacitance value of a target, or physical dimensions, such has the coupled interconnect line width. For example, if it is known through a pre-processing that the coupling capacitance values, C_C, of all FTs is equal to or greater than a threshold value, say $C_{C\text{-}th}$, then a filter can be devised that simply states the following: "For each target T_i in the input target set, if $C_C < C_{C\text{-}th} \Rightarrow$ target is safe and is pruned."

The *pruning power* of a filter is defined as the ratio of the number of targets pruned to the number of targets in the input set of targets passed to the filter. The pruning power of a filter is dependent on both the effectiveness of its pruning conditions and the set of targets passed to it. For example, if the smallest set of targets (generated as a result of the strongest pruning possible) is passed to a filter, then the pruning power would be zero, because no more pruning is possible. To compare the relative pruning power of filters, the same set of targets should be passed to them. We say that filter F^x is *dominated* by filter F^y with respect to initial target set S if S_y is contained in S_x where S_x (S_y) is the set of remaining targets in S after application of F^x (F^y).

The value of circuit and timing parameters required by a filter are determined by extractor(s) and timing analyzer(s). After a filter has been applied to a set of targets, the remaining set of targets can be passed to another non-dominated filter or one with relatively higher pruning power. Alternatively, a new extractor or timing analysis tool can be run that determines at least one circuit or timing parameter value of higher accuracy than previously known. Then some filter (even the same one as before) can be applied to achieve additional pruning.

The cost of a filter is a function of the number of targets in the input set passed to it. Typically, a filter with higher cost has a higher pruning power. This is because more elaborate conditions must be checked to increase the pruning power of the filter. At each stage of target set compaction, the relative pruning power of the filters as well as the filter cost must be known in order to decide which filter is the best one to use next. The pruning power of a filter, however, in going from one stage to the next, is a function of the actual pruning tools that have been previously executed. It also varies from one circuit to the next. This variability of the filter pruning power is one of the key factors that make the formulation of target set compaction difficult.

Next we describe the main filters that are used in STAX to process crosstalk slowdown targets.

Filter F^1: Required time-based pruning based on looked-up maximum slowdown values

Figure 2 depicts the crosstalk-affected signal arrival time at the victim, $AT_{\mu+3\sigma}(V)$, of a crosstalk site as a function of the input skew, δ. Let $SD_{max\text{-}Cc}$ be the highest possible slowdown that can be generated by the crosstalk site with coupling value C_C. Filter F^1 checks whether or not the maximum arrival time plus the worst-case slowdown of the victim can violate its minimum required time:

$$\text{For target } i,\ \{AT_{\mu+3\sigma}(V) + SD_{max\text{-}Cc} \le R_{\mu-3\sigma}(V)\}$$

$$\Rightarrow \text{Target } i \text{ is safe and can be pruned.} \qquad (3)$$

F^1 uses the required times computed by T^2. This filter has two modes, depending on whether the value of C_c is known or not. If an extractor is used to extract C_c, then we can utilize the slowdown vs. input skew lookup tables (such as the one which is graphically depicted in Figure 2) to determine $SD_{max\text{-}Cc}$; On the other hand, if C_c is unknown, a worst case value will be assumed for looking up the table and fetching $SD_{max\text{-}Cc}$.

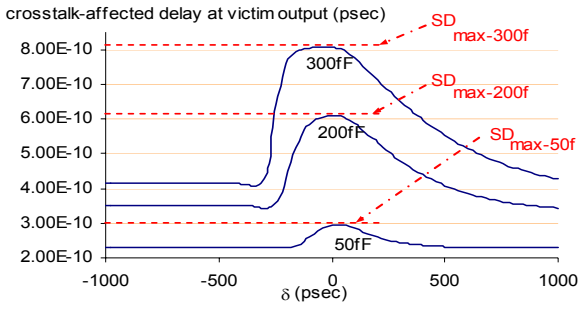


Figure 2: Slowdown curves as a function of skew for different C_C

Filter F^2: Required time-based pruning based on crosstalk-affected arrival times

For each target under consideration, this filter uses the crosstalk-affected arrival times of the victim and the aggressor as computed by T^1 as well as the required time of the victim as calculated by T^2. The target is pruned if the

$\mu+3\sigma$ value of the victim's arrival time does not violate the $\mu-3\sigma$ value of its required time.

For target i, $\{CT_{\mu+3\sigma}(V) \leq R_{\mu-3\sigma}(V)\}$

$\Rightarrow$ The target is safe and can be pruned. (4)

Filter F^3: PO arrival time-based pruning

For a given crosstalk target, this filter uses T^1 to compute the crosstalk-affected arrival time at the primary outputs (PO) of the circuit. The target is pruned if the $\mu+3\sigma$ value of the any of the PO arrival times violates clock cycle time, D.

For target i, $\{\forall PO_j: CT_{\mu+3\sigma}(PO_j) \leq D\}$

$\Rightarrow$ Target i is safe and can be pruned. (5)

A cost-per-target value is associated with each filter accounting for the computation cost of the timing tool that it uses. For F^1, this cost is approximately 50µsec/target on Sun Blade 1000 machine. The cost for F^2 and F^3 is approximately 35 and 55msec/target, respectively.

With respect to a random set of sites, filter F^3 usually shows the highest pruning power since it uses T^1 to compute accurate distribution of arrival times and a forward traversal to determine whether or not the additional delay at a site actually violates the clock sampling. F^1 is the fastest of the filters described, but also the most pessimistic. F^2 is faster than F^3 because in F^2, the backward traversal of T^2 to calculate the required time of circuit lines is done only once and subsequently used for all targets, whereas in F^3, the forward traversal of additional delay must be repeated for each site.

4. PROBLEM STATEMENT AND SOLUTION

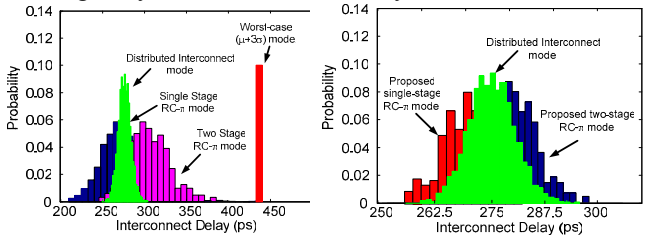
Assume a CMOS VLSI circuit with an initial set of crosstalk targets, Set_0. There are n filter tools, F^1, ... , F^n, m extractor tools, E^1, … , E^m, and p statistical timing analyzers T^1, … , T^p readily available. For a given circuit, there exists an optimal sequence of extractors, filters and timing analyzer tools to execute to find a compact set of targets in the minimal amount of CPU time. The problem is to find that optimal sequence that consists of (a subset of) the available tools that provides the best pruning possible (or a desired amount of pruning). Unfortunately, this sequence is usually different for each circuit.

Three factors help in identifying a good pruning sequence. First, there is a partial ordering among many of the tools. For example, once some set of extractors are executed, the only extractors that can be subsequently executed are those that compute at least one parameter value to a higher degree of accuracy compared to any previously executed extractor. Similarly, if a filter is executed, then a dominated filter cannot be executed unless the accuracy of at least one variable is improved. The second factor is that a good sequence based upon a suite of circuit benchmarks is often a good sequence for a new circuit. Finally, the application of one tool might imply the application of another one. For example, T^2 is required to run at least once prior to running F^1. Additionally, it is known that F^3 does not require T^2. Reference [7] shows how the subset property (i.e., every subset of a frequent set is frequent) and the corresponding association rules among targets sets can drastically reduce the complexity of finding good sequences.

5. EXPERIMENTAL RESULTS

5.1 Statistical Analysis Tool in STAX

To show the need for a statistical approach to compute the necessary timing information, first the statistical model based on the distributed RC-π circuit is compared against the conventional corner-based approach. A coupled global interconnect pair, each 1000µm long, is used to study the effect of line width and height variations on the crosstalk-affected output delay of the victim line. From Figure 3(a), the corner-based value of the victim delay shows more than 46% pessimism compared to that in the statistical model. We also substituted our distributed model with lumped RC-π and 2RC-π models and performed statistical analysis. The mean delay was found to be close to that for the distributed model. The $\mu+3\sigma$ value for the single RC-π (2RC-π) shows about 13% (8%) pessimism. We next repeated the experiment by using the RC values found by the variational lumped RC-π model construction heuristics described in Section 2.1. The results are shown in Figure 3(b). Compared to Figure 3(a), the overestimation is drastically reduced (e.g., for the case of 2RC-π model from 7.9% to 4% error for $\mu+3\sigma$ value and from 3.2% to 1.4% error for the mean value.) The intuitive explanation is that in case of summation of the distributed parameters to a single value, the variations tend to cancel each other and thus the pessimism of the conventional approach for the extraction of a single component is reduced. Using lumped models greatly increases the efficiency of STAX.



(a) (b)

Figure 3: (a) Comparison of distributed, RC-π , and 2RC-π models (b) accuracy improvement using our heuristic

5.2 Target Set Compaction in STAX

Our experiments on different sequences show that the sequences with highest compaction degrees for different circuits are similar to each other. For example, all sequences end with the highest quality tools available. Therefore, we use a training procedure where the most efficient sequences for pruning the targets in a number of (training) circuits are found. Table 3 shows the sequences with highest compaction degree found for each of the training circuits. As mentioned in the previous section,

these sequences are also effective for other circuits. All sequences end by executing the most accurate extractor (E^5) and the filter with highest pruning power (F^3.)

Table 3: Sequences with highest compaction degrees for training circuits

Circuit	Sequence
C17	S_1=$E^1T^2\mathbf{F^1}E^2T^2\mathbf{F^1}T^1T^2\mathbf{F^2F^3}E^3T^1\mathbf{F^3}E^4T^1\mathbf{F^3}E^5T^1\mathbf{F^3}$
C432	S_2=$E^1T^2\mathbf{F^1}E^2T^2\mathbf{F^1}T^1T^2\mathbf{F^2F^3}E^4T^1\mathbf{F^3}E^5T^1\mathbf{F^3}$
C499	S_3=$E^1T^2\mathbf{F^1}E^2T^2\mathbf{F^1}T^1T^2\mathbf{F^2}E^4T^1\mathbf{F^3}E^5T^1\mathbf{F^3}$
C880	S_4=$E^1T^2\mathbf{F^1}E^2T^2\mathbf{F^1}T^1\mathbf{F^3}E^5T^1\mathbf{F^3}$

The sequences shown in Table 3 were applied to five benchmark circuits, namely C1355, C1908, C3540, C5315, and C7552. None of these circuits were included in the training set. To demonstrate the efficacy of tool/filter sequences found by STAX, we generated several semi-random sequences and ran them on those circuits. The reason for "semi-random" selection is that the last pruning tool is the one with highest compaction degree, i.e., **SP**≡$E^5T^1F^3$. Thus, all sequences produce the same final set of potential crosstalk faults. Table 4 shows the execution times for all STAX generated sequences (S_1 to S_4), and a few semi-random ones (SR_1 to SR_4) as well as for SP. We see that for each of the five circuits, each of the four training sequences results in about the same computation time. Sequence SR_4 is very similar to S_4 and performs fairly well, but the rest require up to one order of magnitude more time to generate the final set of targets.

To compare our framework with previous work on target identification [1],[4]-[8], we assume that the highest quality filter and most accurate extractor used in these earlier works are the same as ours. This means that they use something equivalent to SP. Hence, for C7552, their system would that run more than 49 times slower than the sequences produced by our framework.

Table 4: Efficiency results of STAX

(a) List of the semi-random sequences

Sequence	Sequence Elements
SR_1	$E^4T^2\mathbf{F^1}T^1T^2\mathbf{F^2F^3}E^5T^1\mathbf{F^3}$
SR_2	$E^1T^2\mathbf{F^1}E^1T^2\mathbf{F^1}E^1T^1T^2\mathbf{F^2}E^5T^1\mathbf{F^3}$
SR_3	$E^2T^1T^2\mathbf{F^2}E^4T^1T^2\mathbf{F^2}E^4T^1\mathbf{F^3}E^5T^1\mathbf{F^3}$
SR_4	$E^1T^1\mathbf{F^1}E^2T^1\mathbf{F^1}T^1T^2\mathbf{F^2}E^5T^1\mathbf{F^3}$
SP	$E^5T^1\mathbf{F^3}$

(b) Results of using "best" sequences on large circuits as well as the rest of the sequences in Table 2(a)

	C1355 (sec)	C1908 (sec)	C3540 (sec)	C5315 (sec)	C7552 (sec)
S_1	105	50	380	507	656
S_2	100	47	384	563	631
S_3	103	41	402	492	651
S_4	98	46	383	627	624
SR_1	752	767	2725	5025	8055
SR_2	211	189	1655	1772	1467
SR_3	905	878	2281	4121	671
SR_4	105	51	625	882	1015
SP	2425	2677	9463	16323	32067

To show the effectiveness of this framework as the first phase of a test generation system, similar to [8], we assume that an ATPG system requires at least 40 seconds, which translates to 1,450,000 seconds to process all initial targets in C7552. Using **SP**, followed by ATPG would take about 35,296 seconds (33,056 for **SP** and 2,240 for an ATPG to process 14 sites). Using STAX and ATPG together would cost around 2,950 seconds (710 for STAX, and 2,240 for ATPG to process 14 sites.) The ratios of these three times are 491:15:1.

Finally, compared to previous work in [8], an average of 20% more pruning was observed in ISCAS85 benchmarks, e.g., for the case of the C7552 circuit, more than 33% improvement in pruning efficacy was achieved (going from 21 targets in [8] to 14 in STAX.)

6. SUMMARY

We presented STAX, a framework for statistical crosstalk slowdown target set compaction, which incorporates an efficient sequence of filters, statistical timing analyzers, and extraction tools. A variation-aware coupled interconnect modeling was used to consider the local process variation effects along coupled interconnects. Experimental results confirm that STAX can significantly improve the efficiency of identifying crosstalk targets to be considered for test generation.

REFERENCES

[1] W.Y. Chen, S.K. Gupta, M.A. Breuer, "Analytic models for crosstalk delay and pulse analysis for non-ideal inputs," *Proc. Int'l Test Conf.* (*ITC*), pp. 809-818, 1997.

[2] H. Chang, V. Zolotov, S. Narayan, C. Visweswariah, "Parameterized block-based statistical timing analysis with non-gaussian parameters, nonlinear delay function," *Proc. Design Automation Conf.* (*DAC*), 71-76, June 2005.

[3] S. Abbaspour, H. Fatemi, and M. Pedram, "VITA: Variation-aware interconnect timing analysis for symmetric and skewed sources of variation considering variational ramp input," *Proc. Great Lakes Symposium on VLSI*, pp. 426-430, 2005.

[4] M.A. Margolese, F.J. Ferguson, "Using temporal constraints for eliminating crosstalk candidates for design and test," *Proc. VLSI Test Sym.* (*VTS*), pp. 80-85, 1999.

[5] H. Takahashi, K.J. Keller, K.T. Le, K.K. Saluja, Y. Takamatsu, "A method for reducing the target fault list of crosstalk faults in synchronous sequential circuits," *IEEE Trans. on CAD, Vol. 24, Issue 2, pp. 252-263, Feb. 2005.*

[6] A.D. Sathe, M.L. Bushnell, V.D. Agrawal, "Analog macromodeling of capacitive coupling faults in digital circuit interconnects*," Proc. Int'l Test Conf. (ITC),* pp. 375-383, 2002.

[7] S. Nazarian, H. Huang, S. Natarajan, S. K. Gupta, and M.A. Breuer, "XIDEN: Crosstalk target identification framework," *Proc. Int'l. Test Conf.* (*ITC*), pp. 365-374, 2002.

[8] M.A. Breuer, S.K. Gupta, S. Nazarian, "Efficient identification of crosstalk induced slowdown targets," *Proc. Asian Test Symp.* (ATS), pp. 124-131, Nov. 2004.

[9] J.H. Chern, J. Huang, L. Arledge, P.C. Li, P. Yang, "Multilevel metal capacitance models for CAD design synthesis systems," *IEEE Electron Device Letters*, Vol. 13, Issue 1, pp. 32-34, Jan. 1992.

[10] D. Blaauw, V. Zolotov, S. Sundareswaran, "Slope propagation in static timing analysis," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, pp. 1180-1195, 2002.