

Diagnosis-Based Post-Silicon Timing Validation Using Statistical Tools and Methodologies

Angela Krstic, Li-C. Wang, Kwang-Ting Cheng
Department of ECE, UC-Santa Barbara
{angela, licwang, timcheng}@ece.ucsb.edu

T. M. Mak
Intel Corporation
t.m.mak@intel.com

Abstract

This paper describes a new post-silicon validation problem for diagnosing systematic timing errors. We illustrate the differences between timing validation and the traditional logic defect diagnosis. The key difference between our validation framework and other traditional diagnosis methods lies in our assumptions of the statistical circuit timing and statistical distribution of the size of timing errors. Different algorithms are proposed and evaluated via statistical timing error injection and simulation. Due to the statistical nature of the problem, 100% diagnosis resolution often cannot be guaranteed. With a statistical timing analysis framework developed in the past, we demonstrate the new concepts in timing validation, and discuss experimental results based upon three types of systematic errors: timing correlation error, crosstalk, and single-site random-size delay perturbation.

1. Introduction

Process variation, manufacturing defects, and noise sources are major factors to affect the timing characteristics of deep sub-micron (DSM) designs [1, 2]. These DSM timing effects are often continuous in nature [3, 4] and the traditional assumptions of discrete timing models become less applicable. These DSM timing effects should better be captured and simulated using statistical timing analysis models and methods [5].

In the DSM domain, when a set of chips fails in timing, it might be due to various reasons. Usually, our first task is to decide if the failures represent *systematic behavior* or if they more resemble *random behavior*. The term "systematic behavior" is used to characterize the behavior from systematic errors that may come from problems in the design, the testing methodology, or the manufacturing process. For example, due to the mismatch between the delay models used in a timing analysis tool and the actual delay behavior on the silicon, we might misjudge the delays on some critical paths. Another example can be that our AC scan patterns may exercise the non-functional space and as a result, cause unexpected power consumption and long delays. A crosstalk-related timing problem can be viewed as a design error or as a problem in the physical analysis tools. In any of these cases, the problems are not due to defects and hence, should systematically exist on all silicon chips. On the other hand, the term "random behavior" is used to characterize random defects resulting from an

imperfect manufacturing process.

In this work, we focus our study on the diagnosis of systematic timing errors. We call it *post-silicon validation* to differentiate our work from the *diagnosis of random defects*. In post-silicon validation our goal is to validate the design model and the testing methodology, and rule out any systematic problems before applying the diagnosis of random defects. We assume that a set of *failing chip instances* are available for diagnosis, and our goal is to answer the following two questions:

1. Do they indicate any systematic error?
2. If yes, can we pinpoint the error source?

In our validation framework, we assume that a collection of error models is known. Also, for an error model, a set of candidate suspect errors can be derived. Then, given a set of failing chip instances, we compare their behavior against the behavior predicted by each model, and within each model, the behavior predicted by each candidate error suspect in order to answer the two questions above.

Our post-silicon timing validation methods are based upon statistical timing models and delay simulation methods, and can be classified into two approaches:

Diagnosis of Modeling Errors: These errors affect the timing on many places across the entire circuit. To diagnose a problem, we apply a *cause-effect* approach based upon the *collective* failing behavior of all chip instances. To demonstrate the methodology, we use the *timing correlation error* as our modeling error example. Timing correlation error assumes that in the timing analysis the delay model considers no correlation while in reality, delays from different circuit elements are actually correlated.

Diagnosis of Structural Errors: These errors affect the timing of a *fixed* single site (i.e. a spot error). To diagnose such a problem, we apply a combined *effect-cause* and *cause-effect* approach to each failing chip instance separately. Then, we decide the final validation answer based upon their collective diagnosis results. To demonstrate the methodology, we consider two examples: the *single-site random-size timing error* and the *crosstalk error*.

Our main findings include the following: Diagnosis of a modeling error is usually a more difficult problem. While deciding on the type of the error is possible, locating the exact problem source is very challenging. On the other hand, our

validation approach is usually effective for locating a single-site random-size timing problem (depending on the number of failing instances given). Often, a few tens of such instances are enough, to be statistically significant and for our diagnosis algorithm to be effective. For crosstalk error, we demonstrate that the real challenge lies in producing the meaningful test patterns, and our validation approach can pinpoint the exact location easily. Experimental results based upon statistical timing error injection and simulation are presented to explain these findings.

The rest of the paper is organized as follows. In Section 2 we illustrate the main concepts and methods in our timing validation framework. Section 3 discusses the reasons why for timing error diagnosis the information contained in the error dictionary should be probabilistic rather than deterministic. Section 4 describes the validation methodologies and their associated diagnosis algorithms for the two types of the errors. Section 5 summarizes the tools and methodologies for conducting the error injection and simulation experiments. In Section 6, we discuss how the clock setting may affect the accuracy of diagnosis, and present our methodology for selecting the clock. In Section 7, the concept of systematic behavior is illustrated through experimental results. Section 8 summarizes the diagnosis results for all three examples of errors under our study. Section 9 concludes the paper.

2. Logic Diagnosis vs. Timing Validation

Historically, the diagnosis problem was defined over the logic domain and no timing information was involved. Logic diagnosis algorithms are often classified into two types: an *effect-cause* approach and a *cause-effect* approach [6].

In an effect-cause approach, the set of potential suspects is decided by tracing backwards from the failing outputs. In a cause-effect approach, diagnosis relies on the construction of a *fault dictionary* that contains information to differentiate the good and faulty behavior in the presence of each fault. Then, for a given failing chip, the failing behavior is compared to the information in the fault dictionary, and the most probable fault is selected as the candidate for the defect source [6].

Diagnosis for systematic timing errors and diagnosis for logic defects can share many similar aspects. For example, in logic defect diagnosis, we need to assume a fault model. Similarly, in our work of timing error diagnosis, we utilize an *error model*. Otherwise, we would not know what we are looking for. However, in our timing validation, the problem is fundamentally different in four ways:

1. A systematic timing error does not necessarily mean that all failing chips would demonstrate consistent behavior. Hence, it is often required to decide if the failing behavior is due to a systematic error, or due to random defects.
2. Similar to the single-defect assumption commonly employed in the traditional logic diagnosis, for diagnosis of systematic timing error, we can also include the assumption of single error. However, a single error in the design model can easily affect the timing of the entire circuit and

manifest itself as timing errors on multiple sites.

3. For a given set of failing chip instances, the exact timing configurations can vary from one instance to another.
4. Even though all failing chip instances have their timing errors on the same site, the sizes of timing errors on each instance can still be different.

These four aspects prevent us from applying a traditional logic diagnosis approach to timing error diagnosis directly. In the past, much of the diagnosis research focused on two directions. One was to improve the efficiency of diagnosis by avoiding the computational expense of creating a large fault dictionary. The other was to extend the basic diagnosis algorithm for the single stuck-at fault model to other defect types (e.g., bridging faults) or to multiple faults. Even for diagnosing gate delay and path delay faults, most of the previous work was based purely on logic conditions for sensitizing the faults [8, 9]. A statistical diagnosis framework for delay defects was proposed in [10]. However, its technique requires finding and storing all possible single and multiple path delay faults logically sensitizable by all failing vectors and hence, is not practical.

2-1. Handling structural errors

Our validation methodology consists of three phases:

- 1. Effect-cause phase:** In this phase, a set of suspect errors are identified for each individual chip instance based purely on analyzing the logic conditions that cause the failing behavior.
- 2. Cause-effect phase:** For each chip instance, individually, we apply a diagnosis algorithm that operates on the probabilistic space, instead of the logic space to obtain a much smaller set of candidate suspect errors.
- 3. Voting phase:** Based upon the collective diagnosis results, we decide if a systematic error can be found. If yes, we calculate the confidence level that the behavior is due to a particular error source.

In phase 2, a statistical timing analyzer serves as a predictor for the delay configurations of the set of failing chip instances. By separating phase 2 from phase 1, we avoid the effort of processing, in phase 2, those suspect errors that can be decided impossible to cause the failing behavior using logic criteria. Hence, the effectiveness of our phase 2 diagnosis algorithm can be better observed.

While deciding which suspect errors should be kept after phase 1 is deterministic (because this decision is based purely on logic criteria), deciding which suspect errors should be used in phase 3 can only be probabilistic. Because of this, we redefine the concept of *diagnosis resolution* in timing validation and discuss various heuristics in the voting phase.

2-2. Handling modeling errors

For modeling errors, the effects may spread out across the entire circuit, and using an effect-cause approach up front

would not be effective. Hence, phase 1 can be skipped. Moreover, in the cause-effect diagnosis phase, we apply a different diagnosis algorithm based upon the *collective behavior* from all the failing instances. For each suspect error, the diagnosis results are interpreted collectively to decide if the failing behavior can be classified as "systematic." Then, we decide which error represents the most probable cause.

3. Probabilistic Error Dictionary

A key step in our diagnosis methods is to construct the *probabilistic error dictionary* for all known suspect errors. After that, the question becomes how to match the probabilistic information of simulated behavior contained in the error dictionary to the failing behavior.

In logic diagnosis, the circuit model used in the simulation is assumed to logically match the actual chips. In timing error diagnosis, this is not true due to the inclusion of statistical timing information. With a statistical model, each actual chip represents a single sample instance of all possible delay configurations intended to be modeled statistically by the timing tool.

Based on a statistical timing model, the diagnosis resolution is not the same as the fault resolution in the logic domain [7, 11]. This is because the criteria to determine if a pattern captures a timing fault, can be probabilistic. For example, with a statistical timing model, a *dynamic timing simulator* can calculate the circuit worst-case delay distribution achieved by a given test pattern [15]. Since the worst-case delay is characterized as a probability distribution, whether or not it exceeds a given clock will be characterized by a probability value as well. This probability value is called the *critical probability* [5].

For a given fault, two test patterns result in two critical probabilities. Hence, whether or not the two patterns can differentiate the fault should also be a probability value. Therefore, in timing error diagnosis, given a pattern v , our first task is to compute the probability that v detects a particular error (the critical probability of v with the error). This information is used to build the probabilistic error dictionary.

In this work, our primary goal is to evaluate the effectiveness of the timing validation methodology. By assuming that storing the fault dictionary is possible, we temporarily avoid the problem of fault dictionary optimization. Also, by using patterns that might not be the best diagnostic quality patterns, we temporarily avoid the complexity of considering timing in ATPG. By isolating these two crucial issues within the logic domain, we can focus our work primarily on the aspects of the problem which uniquely belong to the timing validation.

3-1. Matching simulated behavior with observed behavior

Consider the example (A) in Figure 1. Suppose that the failing behavior of a chip instance is characterized as a 0-1 matrix (1 means that an error is observed). Suppose we have a way to calculate (using simulation), for each candidate suspect error, a probability matrix P where p_{ij} represents the chance that a failure can be observed at primary output i during the application

of test vector j . Then, in this example, we are asking: which probability matrix (simulated behavior) is a better match to the failing behavior?

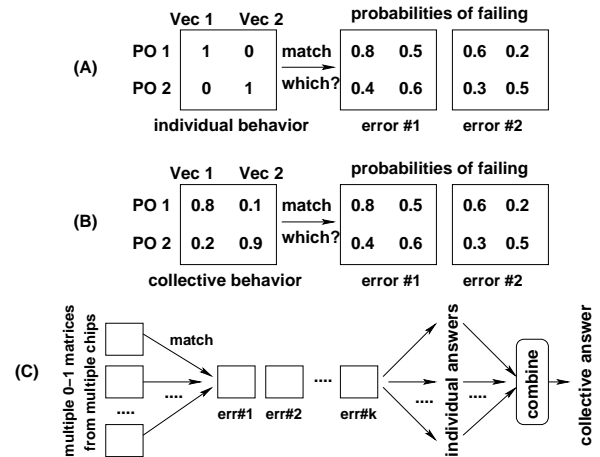


Figure 1: Conceptual examples of matching simulated behavior with observed behavior

In Figure 1-(A), if we focus on matching the "1" entries in the 0-1 matrix, we would say that error # 1 is a better match. However, if we focus on matching the "0" entries, error # 2 would be a better match. In general, depending on our view of what do we mean by a "better match", the diagnosis answer can be different. Hence, to define a diagnosis method, we need to first define carefully how the information in the probabilistic error dictionary should be matched to the failing behavior. We call such functions *diagnosis error functions* [7].

The matching becomes more interesting if we consider the collective behavior of all failing instances. In Figure 1-(B), instead of matching a 0-1 matrix to a probability matrix, we need to decide how to match a probability matrix (collective behavior) to another probability matrix. The collective behavior matrix is obtained by combining the individual behavior matrices for a collection of failing chip instances. The values in the collective behavior matrix represent the frequency of failure for a given output and a given vector in the observed set of failing instances.

In Figure 1-(C), the matching between the collective behavior and the simulated behavior is done differently. In (B), a probability matrix is built from multiple 0-1 matrices before matching, and the matching result represents the collective answer of diagnosis. In (C), each 0-1 matrix is matched individually. Then, their diagnosis answers are combined to form the collective answer for the diagnosis.

Both approaches illustrated in (B) and (C) produce diagnosis results based on collective behavior of multiple failing chips. Both approaches can be used to diagnose systematic errors. In our work, we discovered that the approach in (B) is more effective for diagnosis of modeling errors, and the approach in (C) is more effective for diagnosis of structural errors. For the approach in (C), how to combine individual answers to form the collective answer will be discussed in Section 4-3.

The concept of probabilistic error dictionary implies that an

optimal test set considering only the logical conditions may not be optimal for timing error diagnosis. A possible solution to obtain a good set of diagnostic delay patterns could be to use a *timed ATPG* [12] [13]. In our work, we do not consider a *deterministic* timed ATPG due to its high complexity. Instead, we use a path delay fault ATPG (based on logic sensitization conditions) as an approximation. Our patterns are generated based upon selected long paths [15].

4. Validation Methods

As described above, depending on which target error type we are considering, a separate diagnosis method is applied. For structural type of errors, the matching algorithm is applied to individual instances. For modeling type of errors, the matching algorithm is applied to the collective behavior of all instances.

4-1. Diagnosis method for structural errors

Given a failing n -output chip instance C_{in} and a set TP of m test patterns, suppose that the failing behavior is characterized in an $n \times m$ -matrix B . For $1 \leq i \leq n$, $1 \leq j \leq m$, each b_{ij} is "1" if an error is observed at output i while applying pattern j . Otherwise, $b_{ij} = 0$. We assume that B is obtained based on a fixed sampling clock.

Phase 1: Effect-Cause Analysis For each failing pattern we perform backward analysis from each failing output. We collect candidate suspect errors that fall on the logically sensitized paths (to the failing output) given by the patterns. At the end of this phase, we obtain a suspect error set $F = \{f_1, \dots, f_l\}$. Each error f_k (for $1 \leq k \leq l$) falls on at least one sensitized path to one failing output during the application of at least one pattern.

Phase 2: Cause-Effect Diagnosis Given F , for each candidate suspect error f_k , we construct an $n \times m$ -probability matrix E_k . Each e_{ij}^k represents the probability that an error can be observed at output i while applying pattern j for a given clock period if the error f_k is present. Suppose we can calculate these probability matrices for all errors in F using our statistical timing analysis tools, and obtain E_1, \dots, E_l [7, 11]. Then, the underlying question to ask is: which E_k (for $1 \leq k \leq l$) is the best match to the failing behavior matrix B ? To measure the accuracy of this "match," we utilize the *diagnosis error function* Err proposed in [7, 11]. In essence, $Err(B, E_k)$ measures the diagnosis error if f_k is selected as the answer of diagnosis.

The Diagnosis Error Function [11] To simplify the problem, we first assume that $n = 1$. Hence, $B = [b_1, \dots, b_m]$ and $E_k = [e_1^k, \dots, e_m^k]$ for error f_k . Then, we compute the *least-square error* between the expected results E_k when error f_k is assumed to be the defect, and the observed behavior B as the following:

$$Err(B, E_k) = Err_k = \|E_k - B\|^2 = \sum_{i=1}^m (e_i^k - b_i)^2 \quad (1)$$

Next, we compute Err_k for all f_k , $1 \leq k \leq l$, and we pick the minimum, i.e., pick the error source that minimizes this diagnosis error function $Err()$.

For multiple-output circuits ($n > 1$), Figure 2 demonstrates a simple view about the meaning of an error in the diagnosis. Under the equivalence checking model, an error in the diagnosis for a given pattern, is defined as *at least one output* produces a difference. In the figure, the delay configuration of the failing chip instance C_{in} is also unknown, and is modeled in the simulation with statistical timing in the circuit model C . What we know is the failing behavior matrix B .

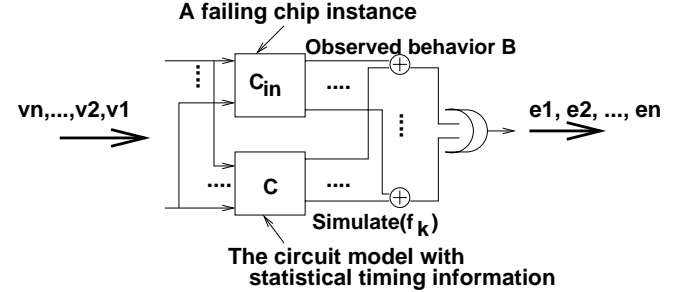


Figure 2: The view of diagnosis error under an equivalence model

What would be the ideal case? The ideal case, where no mismatch occurs, would be $e_1 = e_2 = \dots = e_n = 0$. However, this is impossible even if we have correctly guessed the error source f_k in the diagnosis process. The reason is that we still do not know the exact delay configuration of the failing chip instance C_{in} .

Now, suppose we have an algorithm to compute, for each suspect error f_k , the probability p_j^k that e_j is 1 ($1 \leq j \leq m$). In other words, the algorithm outputs a probability vector for each f_k as $P_k = [p_1^k, p_2^k, \dots, p_m^k]$. Then, since the ideal outcome we want to see is $\mathbf{0} = [0, 0, \dots, 0]$, we can measure the diagnosis error between the probability vector P_k and the ideal solution $\mathbf{0}$ as simply

$$Err_k = \sum_{j=1}^m (p_j^k)^2 \quad (2)$$

Equation (2) follows the same spirit as equation (1). Hence, we can use equation (2) to pick a candidate suspect error whose diagnosis error value is the minimum.

The complete flow of our diagnosis algorithm for structural errors is described below.

Algorithm 4.1 (The Matching Algorithm for Structural Errors)

Inputs A circuit with statistical timing model; a test pattern set TP ; a failing behavior matrix B ; and a set of suspect errors F given from phase 1 analysis; Moreover, a user-defined number K called the *diagnosis resolution*.

Outputs A set of *ranked* diagnosis error values $Err_1 < \dots < Err_K$ which indicate the K most likely error sources causing the failing behavior.

Steps The clock period clk is used to observe the failing behavior matrix B .

1. For each error f_k in F , calculate the probability matrix E_k . The tools and methodologies used in this calculation will be summarized in Section 5. From E_k and B , we use the method described in [7] to calculate the probability vector $P_k = [p_1^k, p_2^k, \dots, p_m^k]$, where each p_j^k is the probability that e_j is 1 (there is mismatch between the observed and simulated results at least at one output) if f_k is present, as shown in Figure 2.

2. Calculate $Err_k = \sum_{j=1}^m (p_j^k)^2$ as described above to measure the diagnosis error.
3. After we finish the calculation for all suspect errors in F , we have $\{Err_1, Err_2, \dots, Err_l\}$. We rank them in such a way that $Err_{j_1} \leq Err_{j_2} \leq \dots \leq Err_{j_l}$ and output the first K error sources as the diagnosis answer.

End of Algorithm

4-2. Diagnosis method for modeling errors

For diagnosing modeling type of errors, the failing behavior of all instances is collectively characterized in the $n \times m$ -matrix B . Hence, for $1 \leq i \leq n$, $1 \leq j \leq m$, the value of each b_{ij} represents the probability that an error is observed at output i while applying pattern j based upon all the sampled instances. The matching algorithm for modeling errors follows the same flow as the algorithm for structural errors, except that the diagnosis error function is defined differently.

The equivalence model described above is suitable for individual instance diagnosis because B is a 0-1 matrix. If B is a probability matrix, we utilize a different function $Err_c(B, E_k)$. Let $B = [B_1, B_2, \dots, B_m]$, where each B_i is a column vector $[b_{1i}, b_{2i}, \dots, b_{ni}]^T$. Similarly, let $E_k = [E_1^k, \dots, E_m^k]$, where each E_i^k is also a column vector $[e_{1i}, e_{2i}, \dots, e_{ni}]^T$. Then, we have

$$Err_c(B, E_k) = Err_k = \sum_{i=1}^m \|E_i^k - B_i\|^2 \quad (3)$$

In other words, we first calculate the diagnosis error based upon the distance between the two probability vectors for each pattern, and then sum up the results for all patterns. The outcome of the algorithm is a ranking among all possible error sources based upon the Err_c calculated values.

4-3. Interpretation of diagnosis results

Both diagnosis algorithms will report a ranking among all the suspect candidate errors. Next, we need to interpret the diagnosis results in order to decide if a systematic error has been identified.

In the case of individual diagnosis (i.e., structural errors), the results are interpreted based upon a majority-based voting scheme, where a vote is obtained from the diagnosis results of each failing instance.

Answer by majority of the votes: Given a K (the diagnosis resolution), suppose for an instance i the suspect ranking is $\{f_1, f_2, \dots, f_K\}$. Then, each f_j , for $1 \leq j \leq K$, will receive 1 vote from instance i . After the voting from all instances, the suspect error that receives the maximum number of votes will be reported as the answer for the systematic error. The confidence of this answer is given as a percentage $V/(K * N)$, where V is the total number of votes received by the answer and N is total number of failing instances participating in the voting.

The decision whether or not the voting result represents a systematic error depends on the application. For example, we may say that for $K = 1$, if the confidence level is greater than 50% (50% of the failing instances give their votes to the same answer), then we would consider it as a systematic error.

In the case of diagnosis against collective behavior (i.e., modeling errors), the interpretation will be made by plotting the diagnosis error values. The details will be illustrated in Section 7.

5. Tools and Methodologies for Experiments

The key tools to realize the proposed validation framework include a statistical timing analysis tool and a dynamic timing simulator. Moreover, to measure the effectiveness of our validation method, we need to perform statistical timing error injection and simulation.

5-1. Statistical Timing Analysis

In the statistical timing analysis framework, the delays of cells/interconnects are modeled as correlated random variables with known probability density functions (pdf's). Given cell/interconnect delay functions and a cell-based netlist, the statistical framework can derive the pdf's of signal arrival times for both internal signals and primary outputs. The random variables of cell/interconnect delays and pdf's of signal arrival times can be derived either analytically [21] or using Monte Carlo simulations [22, 5]. In this paper, our focus is on validating the new diagnosis methodologies rather than on the issues of statistical modeling and analysis.

In our experiments, we use a cell-based statistical timing analysis framework [5]. It requires pre-characterization of cells, i.e., building libraries of pin-to-pin cell delays and output transition times (as random variables). We use a Monte-Carlo-based SPICE (ELDO) [23] to extract the statistical delays of cells for a 250nm, 2.5V CMOS technology. The input transition time and output loading of the cells are used as indices for building/accessing these libraries. Each interconnect delay is also modeled as a random variable and is pre-characterized once the RCs are extracted.

5-2. Dynamic timing simulation

With a given set of test patterns the statistical timing analysis framework can be used to perform statistical dynamic timing simulation to obtain the pdf's of internal signals and primary outputs for the given set of test patterns. These pdf's are obtained by simulating a large number of circuit instances (ex. 1000-5000) with different cell/interconnect delay assignments.

5-3. Timing error injection and simulation

As described earlier, we consider three types of timing errors: the timing correlation error, single-site random-size timing perturbation, and the crosstalk error.

Timing Correlation Error In this model, we assume that the failing behavior is caused by specifying an incorrect correlation in the timing model. For example, we may assume that during the design phase, our timing analysis tool considers no correlation among the delays of circuit elements (correlation 0). However, when the design is manufactured as real chips,

there exists an unknown correlation affecting all circuit elements (correlation > 0). To model this type of error, a correlation factor is used in the statistical simulation model among all the delay random variables. Figures 3 and 4 illustrate the effect of this correlation factor.

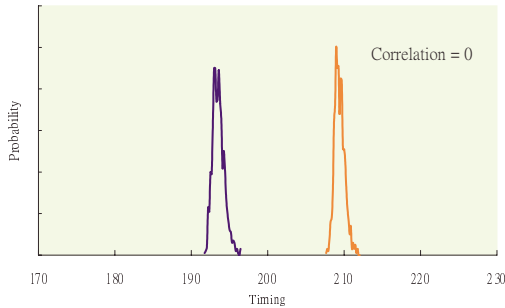


Figure 3: Delay distribution without correlation, s35932

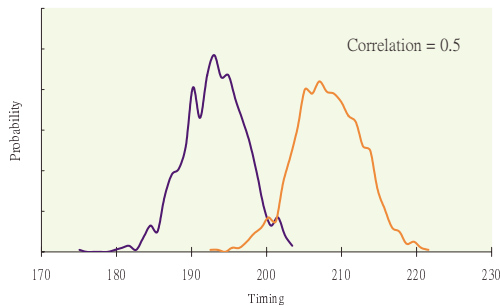


Figure 4: Delay distribution with correlation= 0.5, s35932

In each figure, two timing distributions for the circuit are shown. The distribution on the right-hand side is obtained without false-path analysis. The distribution on the left-hand side is obtained after false path removal using the techniques proposed in [15].

On each figure, if we draw a vertical line as the clock period, then the area beyond the clock line can be viewed as the probability that a chip instance of the circuit would exceed the clock. For example, in Figure 4, if the clock line is set at 210, then the left-hand side pdf tells that no instance should exceed the clock while the right-hand side pdf indicates that there is a significant chance of this happening.

From these two figures, the effect of a non-zero correlation can be easily observed. In Figure 4, because delays of circuit elements are correlated, the overall circuit delay tends to spread out into a wider range. Hence, although the mean delay remains the same, the variance will significantly increase. If we set the clock at 200, then without the correlation, no instance should fail the clock. However, with correlation 0.5, this is not true (from the left-hand side pdfs).

Single-Site Random-Size Timing Perturbation In this model, we assume that for all failing instances, the error happens at the same location but with different delay sizes. For the delay size, we adopt an exponential distribution $\lambda e^{-\lambda x}$ where x is the defect size and λ is a constant. We use $\lambda = 0.04$ in our experiments.

Crosstalk Error The cell/interconnect delays in the presence of crosstalk are functions of cell/interconnect parameters such

as interconnect resistance and capacitance, coupling capacitance, rise/fall times of the signal on the interconnect, etc. Given these parameters as random variables, the random variables of cell/interconnect delay in the presence of crosstalk can be computed using any of the known delay calculation methods considering coupling [16, 17, 18, 19, 20] through Monte Carlo sampling and simulations. In our experiments, we use the model proposed in [20] due to its simplicity.

6. The Impact of The Clock Period

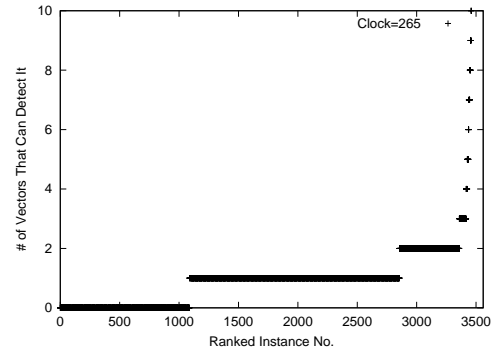


Figure 5: Number of patterns (y-axis) detecting an instance (x-axis), clock=265

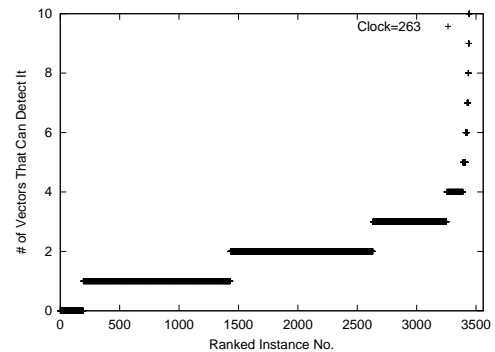


Figure 6: Number of patterns (y-axis) detecting an instance (x-axis), clock=263

In timing validation, the clock setting can affect the resolution of diagnosing a timing error. On one hand, we would want to have a long clock period to improve the diagnosis resolution. This is because with a long clock period, many errors that cause small-size delay problems may not appear in the failing behavior. Hence, in a way we can use the clock to differentiate between small-size delay errors and large-size delay errors, and reduce the number of suspect errors to be considered. On the other hand, we would want to have a short clock period because a short clock period would result in more failing instances and increase the sample size. With a long clock period, we may have only a few failing chip instances whose collective behavior can be statistically too noisy to make a systematic decision. These two opposite concerns present a trade-off for choosing the clock period in timing validation.

To illustrate the concept, Figures 5 and 6 compare the results of two slightly different clocks. These results were obtained on benchmark s5378 with 3650 instances and more than 10000

transition fault patterns simulated. Each instance was injected with a single-site random-size error with $\lambda = 0.04$. As it can be seen, for clock=265, there are many failing instances that could be detected by only one pattern. Many could not be detected by any pattern. For clock=263, many instances that could be detected by only one pattern before now can be detected by two patterns. Many that could not be detected before now can be detected by one pattern.

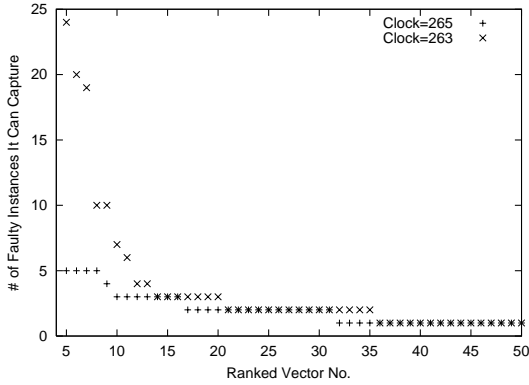


Figure 7: Number of instances (y-axis) detected by a test pattern (x-axis)

Figure 7 presents a reverse plot. As the clock decreases and more instances appear to be faulty, the number of instances that can be detected by a pattern increases as well.

In principle, a longer clock period reduces the number of useful patterns and the number of faulty instances. Hence, it reduces the search space for diagnosis. However, to obtain a statistically significant sample size, a clock cannot be too long. In our experiments, we utilize our statistical timing analysis framework to set the clock. We first generate the probability distribution (with false-path-aware analysis) for the circuit as that shown in Figure 3 without considering any injected error. The clock is set as $\mu + 3\sigma$ where μ is the mean and σ is the standard deviation of the distribution.

7. The Systematic Behavior of Correlation Error

In this section, we discuss how to observe the systematic behavior from the diagnosis results of a timing correlation error. From the timing correlation model, we derive ten candidate suspects $C = \{0.1, 0.2, \dots, 1.0\}$. For example, the suspect marked "0.1" represents a correlation factor of 0.1 among all cell/interconnect delays. In each experimental run, we inject a correlation error ≥ 0.5 and collected 100 failing chip instances. Note that we do not consider injecting an error < 0.5 because for a small correlation, the error effect is too small to produce the 100 failing instances within a few hundreds simulated instances. Note that the selection of the number "100" was arbitrary and was chosen to reflect the fact that in an industrial environment, it is often not economical to manufacture a large number of test chips.

Figure 8 shows the diagnosis error values for each of the four injected errors (0.5, 0.6, 0.8, 1.0) across the 10 suspects in set C based upon the calculation method discussed in Section 4-2 above. The circuit used is s1196 and the number of patterns is

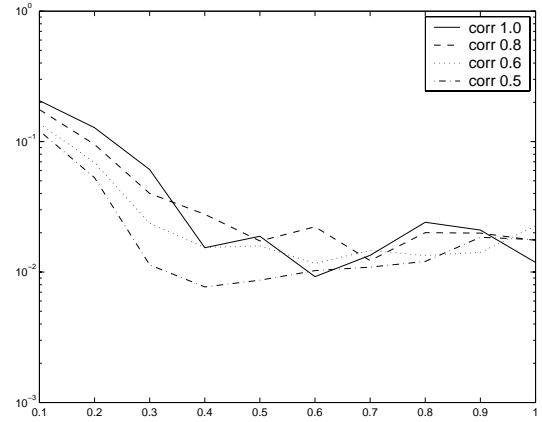


Figure 8: Plot of diagnosis error values for each injected correlation error

20. The patterns are generated for the 20 longest non-robustly testable paths reported by [15]. In the figure, the x-axis denotes the correlation suspect candidates and the y-axis denotes the diagnosis error values.

The best match in each case will be shown in Table I later. In this figure, our goal is to demonstrate the consistent trend observed in all cases. We note that with an injected correlation error, the suspect error 0.1 receives the highest diagnosis error values. The diagnosis error values decrease as the suspect correlation error size increases. This trend can be used to determine that the failing behavior has demonstrated the systematic behavior of a timing correlation error.

In contrast, Figure 9 demonstrates the cases where single-site defects are present, with and without a timing correlation error. In Figure 9-(a), no correlation is considered. As it can be seen, after we apply the diagnosis algorithm for structural faults on the ten correlation suspects, we observe an entirely opposite trend: The diagnosis error values increase with the increase of the correlation error size.

Another interesting thing to observe from the figure is that as the single-site delay error size becomes large (5 or 10), not only the absolute diagnosis error values increase significantly, but also the trends disappear (become flat). In those cases, the effects of the single-site timing errors become significant and dominate the delays on the failing instances. Therefore, we would not be able to diagnose successfully for the timing correlation error. However, if the effect from random delay error is not dominating, then as long as the trend is there, we can still decide that a timing correlation error has occurred.

7-1. Observing the systematic behavior from a correlation error in the presence of other timing errors

To further study the effects of random timing errors on top of the timing correlation errors, consider Figure 10. The size of the random errors follows the exponential distribution assumption with $\lambda = 0.04$, as before. In every failing instance, the locations of the injected random error(s) are fixed. We consider injecting simultaneously up to five errors on an instance.

In Figure 10, it can be seen that if the timing error sizes are

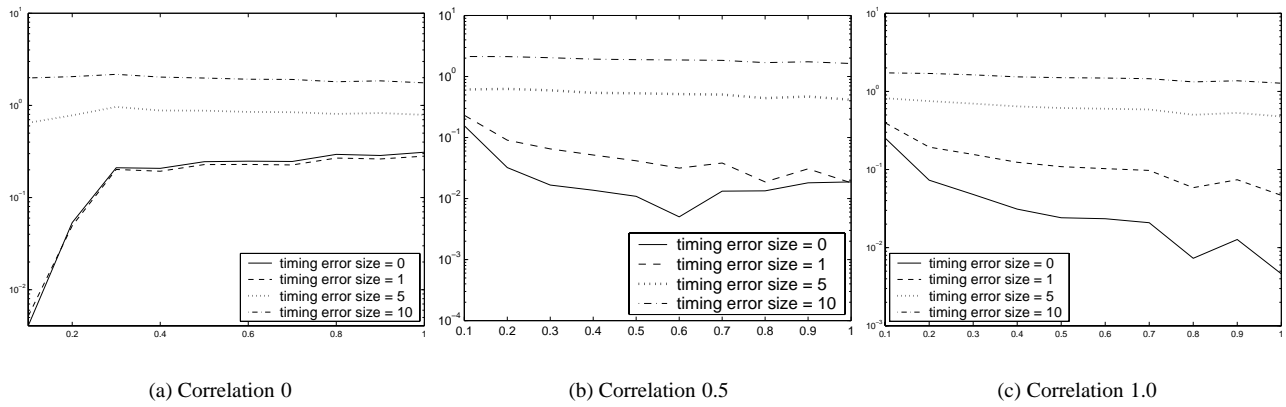


Figure 9: Plots of diagnosis error values for one random-site timing error whose size is fixed, s1196

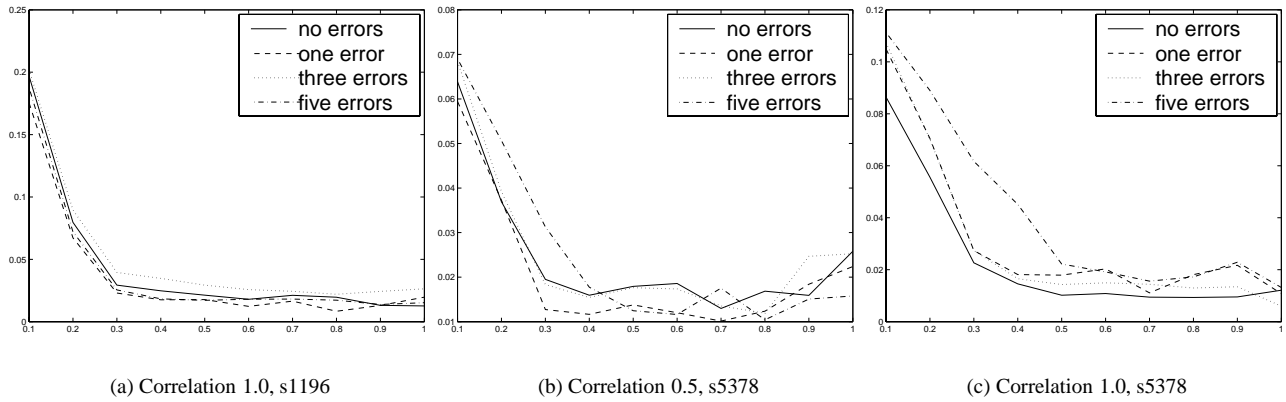


Figure 11: Additional plots of diagnosis error values for s1196, and s5378 in the presence of random timing errors

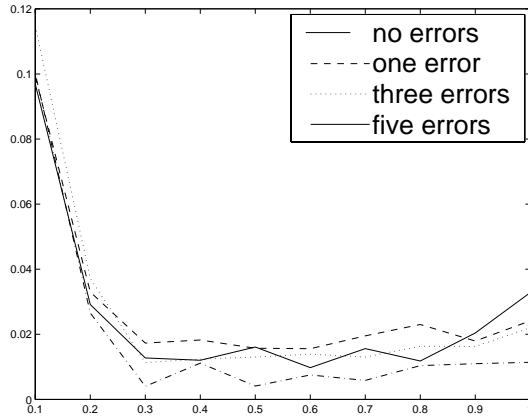


Figure 10: Plot of diagnosis error values for s1196 in the presence of random timing errors (correlation 0.5)

random, then even with five errors, the effects of random timing errors will not dominate the effects of the timing correlation error 0.5. Figure 11 shows additional three plots for s1196 and s5378, by considering another timing correlation error 1.0. These additional results show similar behavior. From these results, we summarize the following two points:

(1). With the presence of a large-size random-site timing error on every failing instance, even though the location of each instance's error is random, the collective effect of all these er-

rors can mask the effect of a large timing correlation error (in the collective behavior of the failing instances). Recall that in the collective diagnosis algorithm, the diagnosis is done based upon the collective behavior, not upon an individual behavior. Hence, large-size spot errors can significantly affect the collective behavior of failing chip instances. If there are 100 instances each with a large-size spot error located at a different site, in the collective behavior, it will look like 100 sites are simultaneously affected by a smaller-size spot timing error.

(2). With the presence of a random-size single-site timing error, since the exponential distribution inherently assumes more small-size errors, the collective behavior on all failing instances is still dominated by the effect of an injected timing correlation error. This is true even with up to five random-size errors injected on each instance. This result indicates that the effectiveness of our validation methodology for the systematic modeling type of errors can be immune to the noise effect in the presence of other structural types of errors, to some degree.

8. Complete Experimental Results

Table I presents the results for correlation timing error. The target correlation is the correlation error used to obtain the 100 failing instances. The best-match correlation is the one whose diagnosis error value is the minimum by applying the diagnosis in Section 4-2. As it can be seen, locating the exact correlation number is not an easy task. Although in almost all cases, the

answer is close enough to the target, there is no guarantee that it can always pinpoint the right answer.

Ckt	Target		Best Match		Ckt	Target		Best Match	
	Corr	Corr	Corr	Corr		Corr	Corr	Corr	Corr
s1196	1.0	1.0	s1238	1.0	0.9				
	0.9	0.9		0.9	0.8				
	0.8	0.7		0.8	1.0				
	0.7	0.5		0.7	0.8				
	0.6	0.8		0.6	0.7				
0.5	0.5	0.5	0.5	0.8					
s1423	1.0	1.0	s1488	1.0	1.0				
	0.9	1.0		0.9	1.0				
	0.8	0.7		0.8	0.7				
	0.7	0.9		0.7	0.8				
	0.6	0.6		0.6	0.5				
0.5	0.5	0.5	0.5	0.7					
s5378	1.0	0.8	s9234	1.0	0.8				
	0.9	0.6		0.8	0.8				
	0.8	0.5		0.7	0.6				
	0.7	0.6		0.6	0.7				
	0.6	0.9		0.5	0.5				

TABLE I: VALIDATION ACCURACY ON BENCHMARK EXAMPLES

We emphasize that since a timing correlation error affects the delays on all circuit elements, the sample size to derive the collective failing behavior is crucial. In practice, it may not be economical to collect thousands of samples and hence, statistical noise in the collective failing behavior will always be there. By using the correlation error as an example, we demonstrate that for this type of errors, perhaps it is more important to decide if such an error exists, than to decide exactly which error occurs. Therefore, results in Table I should be used as a starting point to fix a design modeling error, not necessarily be used as the golden answer to a timing problem.

8-1. Single-site random-size timing errors

Ckt	k	Defect			Ckt	k	Defect		
		#1	#2	#3			#1	#2	#3
s1196	1	0.72	0.52	0.74	s1423	3	0.33	0.42	0.39
	3	0.82	0.68	0.88		5	0.58	0.52	0.51
	5	0.89	0.83	0.96		10	0.75	0.84	0.83
	10	0.97	1.0	1.0		14	0.96	0.97	0.97
s1238	4	0.69	0.78	0.61	s1488	2	1.0	0.18	0.2
	5	0.71	0.83	0.67		8	1.0	0.63	0.84
	10	0.93	0.94	0.76		10	1.0	0.93	0.97
	13	1.0	0.98	0.92		14	1.0	1.0	1.0
s5378	1	0.84	0.73	0.67	s9234	6	0.8	0.66	0.85
	3	0.90	0.78	0.75		10	0.93	0.88	0.92
	5	0.90	0.92	0.83		14	0.95	0.92	0.92
	10	1.0	1.0	1.0		17	1.0	0.94	0.94

TABLE II: VALIDATION ACCURACY FOR SINGLE-SITE RANDOM-SIZE TIMING ERRORS

Table II presents the diagnosis accuracy for single-site random-size timing errors using the algorithm presented in Section 4-1. The failing sample size in these runs ranges from 10 to 50 instances, depending on how easy an error effect can be observed. The table shows three columns of results for each circuit. Each column represents a fixed-location of timing error on all failing instances. As mentioned before, the diagnosis algorithm is applied to each instance individually.

The accuracy of diagnosis is measured in two ways: 1) In the algorithm, if the user-defined diagnosis resolution number K has a value (refer to Section 4-1), then the accuracy is a binary value, *success* or *failure* depending on whether the answer matches the injected error or not. 2) If the user-defined $K > 1$,

then if the injected error is *contained* in the suspect error set obtained by the algorithm, it is counted as a *success*; otherwise, it is a failure. Then, we calculate the success rate as the accuracy measurement by averaging over the results from all N instances. Clearly, the larger the K value is, the higher the success rate will be.

To produce the patterns used in diagnosis, for each error, we find a set of statistically "long" paths through the fixed site and generate path delay tests for them *without* considering timing. These long paths are derived using the false-path aware static statistical timing analysis tool proposed in [15]. Then, robust or non-robust patterns for testing these paths are produced.

In Table II, as expected, the rates of success increase for larger K . The number of applied diagnostic patterns is in the range of a few tens of patterns, depending on the circuit. The results in the table can be interpreted in the following way: For example, for s1196, error # 1 and $K = 1$, 72% of the failing chip instances can be diagnosed with the correct error location. For $K = 3$, three possible error locations are answered for each failing instance. For 82% of the failing instances, the correct error location is one of the three suggested locations.

These numbers for K should be compared to the number of remaining suspect errors after the phase 1 effect-cause analysis. Usually, hundreds of suspect error locations would still remain. In other words, our diagnosis algorithm would select a few locations out of the hundreds of suspects as the potential answers for the timing error location.

Ckt	k	Defect			Ckt	k	Defect		
		#1	#2	#3			#1	#2	#3
s1196	1	0.79	0.58	0.75	s1423	1	0.38	0.25	0.36
s1238	1	0.35	0.47	0.39	s1488	1	0.50	0.62	0.64
s5378	1	0.84	0.73	0.67	s9234	1	0.18	0.19	
	2	0.55				2	0.55		

TABLE III: RESULTS GIVEN BY THE MAJORITY OF THE VOTES

Table III tries to interpret the diagnosis results in Table II using the "Vote By Majority" scheme described in Section 4-3. In each case, we show the smallest K where the right answer receives the majority of the votes and also its associated confidence level. With a majority vote, we can usually pinpoint the right location with a very small K . However, the confidence level can be quite low, meaning that in those cases, only a small portion of the failing instances agree on finding a systematic timing error.

8-2. Validation results for crosstalk errors

Ckt	number of faults			Diagnosis Results $K = 1$
	with overlapping timing windows	satisfying logic cond.	satisfying timing cond.	
s1238	20	6	2	100%
s1423	18	10	2	100%
s5378	32	13	4	100%

TABLE IV: VALIDATION ACCURACY FOR CROSSTALK ERRORS

Table IV shows results on the accuracy of diagnosis for crosstalk errors. These results are obtained using 10 failing instances in each case.

To narrow down the candidate suspect crosstalk location, we follow a 3-step process proposed in [11]. For example, for s1238, best- and worst-case statistical analysis found 20 potential crosstalk problems. However, only 6 of them could be sensitized due to logic constraints and only 2 error sources could cause any delay effects due to the timing constraints.

Searching for effective patterns to activate the crosstalk errors is a complex task due to the need to satisfy stringent timing and logic conditions on the victim and aggressor wires. However, while pattern generation can be a difficult problem, locating the crosstalk sites seems to be simpler. For example, in all our test cases the sensitizable crosstalk induced timing errors could be uniquely diagnosed with $K = 1$ for all instances. This result can be due to two reasons: 1) the suspect candidate set is very small, and 2) the patterns to activate each crosstalk error are very unique because they need to satisfy strict logic as well as timing constraints. Based upon our results, we therefore conclude that the challenge of timing validation against a crosstalk problem lies in the test generation of good patterns. If good patterns can be obtained, then since the patterns are unique to their respective crosstalk errors, diagnosis of the crosstalk locations is much easier. In this work, we use Genetic Algorithm based ATPG [25] to produce patterns for the crosstalk error. This approach seems to have ability to strike a good balance between the quality of produced test patterns and complexity.

9. Conclusions and Future Work

In this paper, we define and study the timing validation problem with respect to systematic errors. We utilize three error models to illustrate the important concepts and concerns in timing validation. For modeling type of errors that affect the timing of the entire circuit, the validation results are obtained using a new diagnosis algorithm based upon collective failing behavior. We conclude that for this type of errors it is difficult to pinpoint the exact problem source although deciding that such a systematic error exists is possible. In this case, the systematic error can be decided even in the presence of other single-site random-size timing errors.

For structural type of errors, we utilize a different diagnosis algorithm applied to the individual failing instances. The individual results are then combined using voting methods to obtain the final validation answer. We conclude that for validation against single-site random-size timing errors, the accuracy cannot be guaranteed due to the statistical nature of the problem and the limitation of the failing sample size. However, in most cases, our validation methodology remains quite effective and can pinpoint a small number of candidate error locations containing the correct answers.

Our work brings up two interesting research problems in timing validation for the future: 1) How to collect a set of meaningful error models? and 2) How to use the clock setting to improve the diagnosis accuracy? In general, timing validation is a very challenging problem. Results presented in this paper may inspire many new research directions in the area.

Acknowledgement: The authors would like to thank Professor D. M. H. Walker at Texas A&M University for his valuable comments and suggestions to revise the final paper.

References

- [1] K. Baker, G. Gronthoud, M. Lousberg, I. Schanstra, , and C. Hawkins. Defect-Based Delay Testing of Resistive Vias-Contacts, A Critical Evaluation. *ITC*, pp. 467-476, Sep. 1999.
- [2] M. A. Breuer, C. Gleason, and S. Gupta. New Validation and Test Problems for High Performance Deep Sub-Micron VLSI Circuits. *Tutorial Notes, VTS*, 1997.
- [3] R. C. Aitken, "Nanometer Technology Effects on Fault Models for IC Testing", *Computer*, Nov. 1999, pp. 46-51.
- [4] K-T Cheng, S. Dey, M. Rodgers, and K. Roy Test Challenges for Deep Sub-Micron Technologies, *DAC* 2000.
- [5] J.-J. Liou, A. Krstić, K.-T. Cheng, D. Mukherjee, and S. Kundu. Performance Sensitivity Analysis Using Statistical Methods and Its Applications to Delay Testing. *ASP DAC*, Jan. 2000, pp. 587-592
- [6] M. Abramovici, M. A. Breuer, and A. D. Friedman, Ch. 12: Logic Level Diagnosis, *Digital Systems Testing and Testable Design*, W.H.Freeman, 1990.
- [7] A. Krstic, L.-C. Wang, K.-T. Cheng, and J.-J. Liou. Diagnosis of Delay Defects Based Upon Statistical Timing Models — The First Step. *DATE*, pp. 328-333, March 2003.
- [8] P. Girard, C. Landrault, and S. Pravossudovitch, A Novel Approach to Delay-Fault Diagnosis. *DAC*, June, 1992.
- [9] P. Pant, and A. Chatterjee, Path-Delay Fault Diagnosis in Non-Scan Sequential Circuits with At-Speed Test Application. *ITC*, pp. 245-252, Oct. 2000.
- [10] M. Sivaraman, and A. J. Strojwas, Path Delay Fault Diagnosis and Coverage - A Metric and an Estimation Technique. *TCAD*, pp. 440-457, Mar. 2001.
- [11] A. Krstic, L.-C. Wang, K.-T. Cheng, and J.-J. Liou. Delay Defect Diagnosis Using Statistical Timing Models. *IEEE VTS*, pp. 339-344, 2003.
- [12] W.-Y. Chen, S. K. Gupta, and M. A. Breuer, Test Generation for Crosstalk-Induced Delay in Integrated Circuits. *ITC*, pp. 191-200, Oct. 1999.
- [13] Y-M. Jiang, A. Krstic, K.-T. Cheng, Estimation for Maximum Instantaneous Current Through Supply Lines for CMOS Circuits. *IEEE Tran. on VLSI*, Vol. 8 No. 1, Feb, 2000. pp. 61-73
- [14] J.-J. Liou, K.-T. Cheng, and D. Mukherjee. Path Selection for Delay Testing of Deep Sub-Micron Devices Using Statistical Performance Sensitivity Analysis. *VTS*, pp. 97-104, Apr. 2000.
- [15] J.-J. Liou, A. Krstic, L.-C. Wang, and K.-T. Cheng. False-Path-Aware Statistical Timing Analysis and Efficient Path Selection for Delay Testing and Timing Validation. *DAC*, pp. 566-569, June 2002.
- [16] T. Sakurai, Closed-Form Expressions for Interconnect Delay, Coupling and Crosstalk in VLSI's. *Trans. on Electron Devices*, vol. 40, no. 1, pp. 118-124, Jan. 1993.
- [17] F. Dartu, L. T. Pileggi, Calculating Worst-Case Gate Delays Due to Dominant Capacitance Coupling. *ITC*, pp. 809-818, Oct. 1997.
- [18] H. Kawaguchi, T Sakurai, Delay and Noise Formulas for Capacitively Coupled Distributed RC Lines. *ASP DAC*, pp. 35-43, Jan. 1998.
- [19] A. B. Kahng, S. Muddy, D. Vidhani, Noise and Delay Uncertainty Studies for Coupled RC Interconnects. *Int'l ASIC/SOC Conf.*, pp. 3-8, Sep. 1999.
- [20] P. Chen, D. A. Kirkpatrick, K. Keutzer, Miller Factor for Gate-Level Coupling Delay Calculation. *ICCAD*, pp. 68-74, Nov. 2000.
- [21] M. Orshansky and K. Keutzer, A General Probabilistic Framework for Worst Case Timing Analysis. *DAC*, pp. 556-61, Jun. 2002.
- [22] R. Hitchcock, Timing Verification and The Timing Analysis Program. *DAC*, 1982.
- [23] Anacad. Eldo v4.4.x User's Manual. 1996.
- [24] D. E. Goldberg, and R. Burch. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, MA. 1989.
- [25] A. Krstic, Y.-M. Jiang and K.-T. Cheng. Pattern Generation for Delay Testing and Dynamic Timing Analysis. *TCAD*, vol. 20, (no. 3), pp. 416-425, March 2001.