Delay Defect Diagnosis Based Upon Statistical Timing Models – The First Step

Angela Krstic, Li-C. Wang, Kwang-Ting Cheng
Department of ECE, UC-Santa Barbara

Jing-Jia Liou
National Tsing-Hua University, Taiwan

Magdy S. Abadir
ASP High Performance Design, Motorola Inc.

Abstract — This paper defines a new diagnosis problem for diagnosing delay defects based upon statistical timing models. We illustrate the differences between the delay defect diagnosis and traditional logic defect diagnosis. We propose different diagnosis algorithms, and evaluate their performance via statistical defect injection and statistical delay fault simulation. With a statistical timing analysis framework developed in the past, we demonstrate the new concepts in delay defect diagnosis, and discuss experimental results based upon benchmark circuits.

A. Introduction

Process variations, manufacturing defects, and noise are major factors to affect timing characteristics of deep sub-micron designs [1, 2]. The delay effects from these factors are hard to predict [3] [4], and the traditional assumptions of discrete timing and delay models become inapplicable. These DSM factors should better be captured and simulated using statistical models and methods [5].

In today's industry, the single stuck-at fault model remains one of the most affordable and effective models for defect diagnosis. The stuck-at fault model does not contain timing information and hence, defect diagnosis is done purely on the logic domain. Logic defect diagnosis often relies on the construction of a *fault dictionary* that contains information to differentiate the good and faulty behavior in the presence of each stuck-at 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]. If we assume that the defects are from the single stuck-at faults, then it might be possible to identify the exact fault causing the problem depending on the existence of a test pattern set that can achieve the *maximal fault resolution* [6]. However, since defects are rarely single stuck-at faults, the diagnosis offers little guarantee. The best hope is usually that the diagnosis process can help to pin-point the location where defects might occur.

In delay defect diagnosis, the problem is fundamentally different in two ways: First, the exact delay configuration of the failing chip instance is unknown. Second, even with the single defect assumption, the size of delay defect can be a random variable. These two aspects prevent us from applying a traditional logic diagnosis algorithm to delay defect diagnosis.

In this paper, we define the problem of delay defect diagnosis based upon a statistical timing model. In this new diagnosis problem, statistical timing analysis serves as a predictor for the actual delay configuration of a given failing chip instance. Our diagnosis algorithms operate on the probabilistic space, instead of the logic space. Because of this, how to match the failing behavior to the probabilistic information contained in the fault dictionary becomes an interesting question. Moreover, since the delay defect size is a random variable, the criteria to determine the maximal fault resolution for a given pattern set become much more complicated.

Based on the statistical timing information, we propose a new diagnosis framework. We define the concept of diagnosis error function and study the performance of different error functions. Each function views the matching of the failing behavior from a different point of view and hence, may lead to different diagnosis results. We conduct experiments based upon statistical defect injection and fault simulation. At the end, we discuss future research directions for the new diagnosis problem.

B. Background and Motivation

Historically, the diagnosis problem was defined over the logic domain and no timing information was involved. To diagnose a logic defect, a fault model is usually assumed. The stuck-at fault model is widely used in many diagnosis algorithms that can often be classified into two types: an *effect-cause* approach and a *cause-effect* approach [6]. An effect-cause approach pre-computes faulty behavior based upon an assumed fault model and stores the information in a fault dictionary. Then, the behavior of a failing chip is compared with the fault dictionary and the most probable faults causing the faulty behavior are identified. In a cause-effect approach, the stuck-at fault model allows an ATPG to determine, from the failing behavior, if a particular line should be stuck-at. Then, by searching backward and matching to the input patterns, probable faults can be identified.

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 is to extend the basic diagnosis algorithm for the single stuck-at fault model to other defect types [7, 8, 9] or to multiple faults [15]. Even for diagnosing gate delay and path delay faults, most of the previous work is based purely on logic conditions for sensitizing the faults [13, 14].

A statistical diagnosis framework for delay defects has been proposed in [10]. However, this technique requires finding and storing all possible single and multiple *path delay faults* that can be logically sensitized under any of the failing vectors. Therefore, it is not practical for large designs.

In this work, our primary purpose is to carefully define the diagnosis problem for delay defects and understand what are the key aspects that make the problem different from traditional diagnosis. Then, we propose a new diagnosis framework based upon statistical timing analysis and develop various diagnosis algorithms. Through this study, we intend to answer the following questions:

(1). What are the differences between traditional logic fault diagnosis and the delay defect diagnosis in terms of their ATPG and fault simulation requirements? **(2).** With a good pattern set that gives us a good fault resolution in the logic domain, what is the remaining prob-

1530-1591/03 $17.00 © 2003 IEEE

lem(s) for delay defect diagnosis to solve in the timing domain? **(3).** Assuming that computing and storing logic information in fault dictionary is not an issue, how well can we do delay defect diagnosis? **(4).** What assumptions do we need to make to facilitate the development of a feasible delay defect diagnosis algorithm? Additionally, what tools do we need?

By using an ATPG operating on logic faults without timing, we temporarily avoid the complexity of the *timed ATPG* in this work. Similarly, by assuming that storing fault dictionary is possible, we temporarily avoid the problem of fault dictionary optimization. By isolating these two crucial issues within the logic domain, we can focus our work primarily on the aspects of the problem which uniquely belongs to the delay defect diagnosis. Moreover, we can study how effective the diagnosis will be when the test patterns are produced without considering the timing.

C. Logic Diagnosis Vs. Delay Diagnosis

In logic diagnosis, the circuit model used in the simulation is assumed to logically match to the chip instance. In delay diagnosis, this is not true due to the inclusion of statistical delay information. The failing chip represents only a single instance of all possible delay configurations intended to be modeled statistically by the CAD tools.

Suppose the single stuck-at fault model is used in logic diagnosis. Let $\{f_1, \ldots, f_n\}$ be the n faults that belong to n different fault equivalence classes. Suppose a pattern set is available to achieve the maximal fault resolution, i.e. for any pair of faults f_i, f_j, there exists a pattern in the set to differentiate these two faults (detect one but not the other). Then, in theory, given the failing behavior resulting from a single stuck-at defect the diagnosis algorithm can conclude exactly which fault is the cause. On the other hand, if the pattern set does not achieve the maximal fault resolution, then depending on the resolution, an algorithm can conclude a subset of the faults as the potential causes. Exactly which one is unknown. Based upon these observations, we can say that the resolution of the diagnosis in the logic domain is the same as the fault resolution.

Take the single transition fault model as an example. In the model, no delay information is involved. Therefore, the above statement for the stuck-at fault model is also true in this case. However, if delay information is involved, then the resolution of the diagnosis is not the same as the fault resolution in the logic domain. Figure 1 illustrates the reasons. In the figure, output arrival times are characterized as probability distributions.

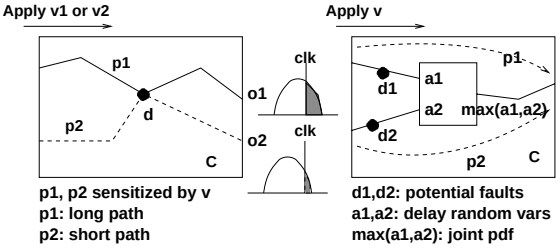

Figure 1: The Impact of Delays in Diagnosis

In the first case, for a fault d, suppose two patterns v_1, v_2 are available. In logic domain, both patterns detect d and can differentiate between d and d'. However, depending on the timing length of the sensitized path ($p1$ or $p2$), the critical probability (shaded area) resulting from each pattern can be different. If a pattern detects a fault through a short path (like v_2), then it is possible that with a small delay defect size, the pattern does not detect the defect at all. Consequently, v_2 can differentiate the two faults in the logic domain but cannot do so by considering the delays (it may detect none).

In the second case, a pattern v detects both faults d_1, d_2, logically through sensitized paths p_1, p_2, respectively. Suppose the two paths merge at a 2-input cell and the arrival time random variables at the two inputs are denoted as a_1, a_2. The output arrival time random variable of the cell is the joint pdf random variable $\max(a_1, a_2)$. Then, suppose $Prob(a_1 > a_2) = 1$. Then, it is possible that p_1 always dominates the output delay (or vice verse). Hence, the pattern v can differentiate the two faults. As it can be seen, even though logically the pattern v does not differentiate the two faults, timing-wise it may.

Due to the above two reasons, in general, whether or not a test pattern can differentiate two given faults *should be characterized as a probability value that depends on the given clock period clk*. Therefore, in delay defect diagnosis, given a pattern v, our first task is to compute the probability that v detects a particular fault. This information is used to build the *probabilistic* fault dictionary, and our algorithm will use the dictionary to guess which fault is the most probable one to be the cause of failure.

C-1. Probabilistic Fault Dictionary

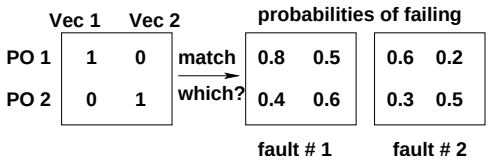

Figure 2: Illustration of The Key Problem

The probabilistic nature of the fault dictionary raises an interesting question. Consider the example in Figure 2. Suppose the failing behavior of a chip instance is characterized as an 0-1 matrix (1 means that an error is observed). Suppose we have a way to calculate (in the simulation), for each candidate suspect fault, a probability matrix P where p_{ij} represents the chance that a failing output is observed at primary output i during the application of test vector j. Then, in the example, the underlying question to ask is: which probability matrix is a better match to the failing behavior?

If we focus on matching the "1" entries in the 0-1 matrix, we would say that fault # 1 is a better match. However, if we focus on matching the "0" entries, fault # 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, from this example it is clear that in order to develop an accurate diagnosis algorithm, our first task is to define carefully how to match the information in the probabilistic fault dictionary to the failing behavior. We call such functions the *diagnosis error functions*. In this paper, we propose different diagnosis error functions and compare their performance.

The concept of probabilistic fault dictionary also implies that an optimal test set considering only the logical conditions may not be optimal for delay defect diagnosis. In this work, we do not consider using a timed ATPG due to its high complexity. Instead, we use a path delay fault ATPG as an approximation.

D. Problem Definition

In this section, we define the statistical timing framework and the delay diagnosis problem. We begin with a sequence of definitions regarding the circuit, statistical timing models, and the defect models.

Definition D.1 *(Circuit Model) A circuit is a 5-tuple* $C = (V, E, I, O, f)$*, where* V *is a set of vertices,* E *is a set of arcs,* I, O *are two subsets of* V *with* $I \cap O = \phi$*, and* f *is a function on* E *where* $\forall e_i \in E$*,* $f(e_i)$ *is a random variable defined over* $[0, +\infty]$.

This view of the circuit is consistent with the statistical timing model defined based upon cell-based pin-to-pin delay random variables proposed in [5]. In essence, the f function characterizes the pin-to-pin delay random variables while each vertex corresponds to a

cell. We note that in this circuit model, the delay random variables can be correlated, i.e. $f(e_i)$ can be correlated with $f(e_j)$ for any $i \neq j$. This circuit model is supported in our false-path-aware statistical timing analysis framework [17].

Definition D.2 *(Circuit Instance) A circuit instance is a 5-tuple* $C_{in} = (V, E, I, O, f_{in})$*, where* V *is a set of vertices,* E *is a set of arcs,* I, O *are two subsets of* V *with* $I \cap O = \phi$*, and* f_{in} *is a function on* E *where* $\forall e_i \in E$*,* $f_{in}(e_i)$ *is a constant value* $\in [0, +\infty]$.

Unlike a circuit model, on a circuit instance, the pin-to-pin delay are all fixed values. For convenience, we use C to denote a circuit model, and C_{in} to denote a circuit instance. In essence, C is used in our CAD tools as the predictor for each C_{in} manufactured.

D-1. Terminology in Statistical Timing Analysis

Below we define the terminology used in our statistical timing analysis framework [17]. The diagnosis problem will be formulated using these terms.

A *path* p on C (or C_{in}) is defined as a path starting from a vertex in I and ending with a vertex in O. Let $p = \{e_1, \ldots, e_i\}$, the *Timing Length* of p, denoted as $TL(p)$ is a random variable characterized by the joint distribution $Sum = f(e_1) + \cdots + f(e_i)$. For each vertex $o_i \in O$, the *arrival time* denoted as $Ar(o_i)$ is a random variable characterized by the joint distribution $Max = \max\{p_1, \ldots, p_j\}$ where each p_l, $1 \leq l \leq j$, is ending at o_i.

The *circuit delay* of C is defined as a random variable characterized by the distribution $\Delta(C) = \max\{Ar(o_1), \ldots, Ar(o_{|O|})\}$.

Definition D.3 *(Induced Circuit) Given path set* P*, the induced circuit of* P *on* C *(or* C_{in}*), denoted as* $induced(P)$*, is a subcircuit* C' *where any arc not on a path in* P *is removed from* C.

Definition D.4 *(Sensitized Paths) Given a test pattern* v *for a circuit* C*, we define the sensitized set of paths by* v *as* $Sen(v) = Path_v$ *that contains all paths in* C *sensitized by the test pattern* v*. Similarly, given a test pattern set* TP *for a circuit* C*, the Sensitized Path Set* $Sen(TP) = Path_{TP}$ *is the set of paths* $\{p_1, \ldots, p_j\}$ *such that for all* i*,* $1 \leq i \leq j$*,* $p_i \in Path_v$ *and* $v \in TP$ *for some test pattern* v.

Definition D.5 *(Static and Dynamic Timing Simulations) Given a circuit* C*, in static timing simulation, the goal is to compute the random variable* $\Delta(C)$*. To compute* $\Delta(C)$*, the simulator will compute* $\{Ar(o_1), \ldots, Ar(o_{|O|})\}$ *as well. With a test pattern set* TP*, a dynamic timing simulator computes the random variables* $\Delta(Induced(Path_v))$ *for each* $v \in TP$ *and consequently, computes* $\Delta(Induced(Path_{TP}))$.

In [17], we developed a false-path-aware statistical timing analysis tool that provides an approximation for $\Delta(Induced(Path_{TP}))$ where TP consists of all possible patterns. The analysis was done implicitly without a pattern set TP.

Since all delays are calculated as random variables, we introduce the notion of *critical probability* below.

Definition D.6 *(Critical Probability) Given a delay random variable* A *and a cut-off period* clk*, the critical probability of* A*, denoted as* crt_A *is the probability* $Prob(A > clk)$.

Definition D.7 *(Error Vectors) Given a circuit* $C = (V, E, I, O, f)$ *(or a circuit instance* C_{in}*), a pattern set* TP*, and a cut-off period* clk*, the error (probability) vector for a test pattern* $v \in TP$ *is* $Err(C, v, clk) = [crt_1, \ldots, crt_{|O|}]$*, where each* crt_i*, for* $1 \leq i \leq |O|$*, is the critical probability* $Prob(Ar(o_i) > clk)$ *in the induced circuit* $Induced(Path_v)$ *for* $o_i \in O$*. Subsequently, we define the error (probability) matrix as an* $|O| \times |TP|$*-matrix* $Err_M(C, TP, clk) = [Err^T(C, v_1, clk), \ldots, Err^T(C, v_{|TP|}, clk)]$*. The* Err^T *denotes the transpose vector of* Err*. We note that for any output* o_j *that is not included in* $Induced(Path_v)$*,* $crt_j = 0$ *by default.*

If C is a circuit model, the error matrix characterizes the probabilities of error behavior predicted by the model. If C_{in} is a circuit instance, the error matrix characterizes the error behavior observed on the particular circuit by applying the test pattern set TP. Later, we will illustrate these points in the development of our diagnosis algorithms. In the following, we define the diagnosis problem.

D-2. The Delay Diagnosis Problem

Definition D.8 *(Problem Definition - Diagnosis for Delay Defects) Given a circuit model* $C = (V, E, I, O, f)$*, a circuit instance* $C_{in} = (V, E, I, O, f_{in})$*, a test pattern set* TP*, a cut-off period* clk*, and an unknown defect distribution function* D_{in}*, the problem of diagnosis is to find a defect distribution function* D *such that the following margin of error is minimized:*

$$\varepsilon = Mar(Err_M(D_{in}(C_{in}), TP', clk), Err_M(D(C), TP', clk)) \quad (1)$$

where TP' *is any other arbitrary pattern set.*

We need to emphasize several key aspects in the above definition of the diagnosis problem:

(1). The *Margin of Error* in essence is the *diagnosis error function* mentioned earlier in Section C-1, which is to measure the accuracy of diagnosis. Depending on the definition of such a margin function, the problem of diagnosis can be different and consequently, can lead to different views about what diagnosis algorithm is the optimal one. **(2).** By using an arbitrary pattern set TP' in the margin of error function, we ensure that the diagnosis results can be generalized. This avoids the problem of identifying a defect function that is specialized to the pattern set TP when TP is small. **(3).** In the definition, we assume that the circuit model is correct. In general, the model C may not be correct. However, in this paper we do not extend the discussion to that situation. **(4).** There are three unknown things in the definition: the delay function f_{in} in the circuit instance and the defect functions D_{in} and D. For f_{in}, we can use f in the model as a predictor. For defect function D_{in}, the most intuitive approach is to assume it has a certain structure in order to simplify the search for D.

D-3. Defect Distribution

Since D essentially alters the circuit delay of C, in this work we adopt a simple segment-oriented assumption for the structure of D.

Definition D.9 *(Segment Oriented) Given a circuit model* $C = (V, E, I, O, f)$*,* D *is a function defined on* E*, where* $D(e_i) = (\delta_i, \rho_i)$*,* ρ_i *is a random variable characterizing the probability of a defect occurrence on* e_i*, and* δ_i *is a random variable characterizing the delay defect size. Usually, we can assume that* δ_i *and* ρ_i *are independent.*

For simplicity, we further assume that δ_i and δ_j are independent for $i \neq j$. Similarly, we assume ρ_i and ρ_j are independent.

Definition D.10 *(Single Defect Model) Given a circuit* $C = (V, E, I, O, f)$*,* D_s *is a function defined on* E *such that* $D_s(e_i) = (\delta_i, \rho_i)$*. Let* $m = |E|$*.* D_s *is a single-defect model if* $\rho_i \in \{0, 1\}$ *and* $(\Sigma_{i=1}^{m} \rho_i) = 1$*. Again,* ρ_i *is the probability of defect occurrence and* δ_i *is the random variable of the defect size.*

With the single defect assumption, the defect function D in the problem definition D.8 is assumed to be of the form D_s. Then, during the diagnosis, the only unknown parameters in D_{in} which needs to be determined from the error behavior Err_M is the *defect vector* $\rho^{vec} = (\rho_1, \rho_2, \ldots, \rho_m)$.

E. The Initial Algorithm

The matrix defined in definition D.7 serves as the basis for our diagnosis algorithm. We use it M_{crt} to denote $Err_M(C, TP, clk)$ as the matrix of critical probabilities based upon the cut-off period clk. Each crt_{ij} in M_{crt} is the critical probability of the arrival time random variable $Ar(o_i)$ at output o_i during the dynamic simulation of pattern v_j in TP. In addition, we use E_{crt} to denote $Err_M(D_s(C), TP, clk)$ as the matrix of critical probabilities when a particular defect function D_s is applied to the circuit C.

Definition E.1 *(Signature Probability Matrix) Given M_{crt} and E_{crt}, the signature probability matrix is defined as the difference between the two matrix:* $S_{crt} = E_{crt} - M_{crt} =$

$$\begin{vmatrix} err_{11} - crt_{11} & \cdots & err_{1|TP|} - crt_{1|TP|} \\ err_{21} - crt_{21} & \cdots & err_{2|TP|} - crt_{2|TP|} \\ \cdots & & \\ err_{|O|1} - crt_{|O|1} & \cdots & err_{|O|\times|TP|} - crt_{|O|\times|TP|} \end{vmatrix} \quad (2)$$

We note that $\forall i, j$, $err_{ij} \geq crt_{ij}$. Hence, for each s_{ij} in the signature probability matrix, $s_{ij} \geq 0$. If we use a very large clock *clk*, then M_{crt} can be very sparse, i.e. most of the entries are zeros. In fact, we can always make *clk* large enough so that $M_{crt} = \mathbf{0}$. In that case, $S_{crt} = E_{crt}$. Also note that each s_{ij} characterizes the "additional contribution" to the critical probability at output o_i from the defect function D_s. For example, suppose that without a defect $crt_{ij} = 0.1$. It means that with 0.1 probability, o_i's delay would exceed *clk* when applying pattern v_j. With defect D_s, that probability would increase to $err_{ij} = 0.3$. Hence, the contribution of the defect to the critical probability is $0.3 - 0.1 = 0.2$.

How to proceed with the diagnosis? The diagnosis information we need for a chip instance of C_{in} can be characterized by a 0-1 matrix as the following:

$$B = \begin{vmatrix} b_{11} & b_{12} & \cdots & b_{1|TP|} \\ b_{21} & b_{22} & \cdots & b_{2|TP|} \\ \cdots & & & \\ b_{|O|1} & b_{|O|2} & \cdots & b_{|O|\times|TP|} \end{vmatrix} \quad (3)$$

where $b_{ij} = 1$ if output o_i fails pattern v_j and, $b_{ij} = 0$ otherwise.

We call B the (failing) behavior matrix. It corresponds to the observed behavior of the failing chip. Given a behavior matrix, our goal is to, under the single defect assumption, uncover the defect vector ρ^{vec}. We note that under the single defect assumption, $(\Sigma_{i=1}^{|E|}\rho_i) = 1$.

Algorithm E.1 *(Diagnosis Algorithm Alg_{sim})*

Inputs A circuit $C = (V, E, I, O, f)$; a pattern set TP; a behavior matrix B; and an unknown defect function D_s under the single defect assumption. Moreover, a user-defined number K.

Outputs A set of *ranked* defect vectors $\rho_1^{vec}, \ldots, \rho_K^{vec}$ each indicating an unique location of potential defect occurrence.

Steps Use the same cut-off period *clk* to observe the matrix B.

1. Find a set of suspect faults $S \subset E$ such that each fault in S is *logically* sensitized to a faulty output by at least one pattern. This step follows a *cause-effect* approach in the logic domain to first prune down the number of suspect faults.
2. For each fault $i = 1 \ldots \mid S \mid$, perform the steps 3-7 to calculate the probability that $\rho_i = 1$, i.e., the suspect fault i is the cause of behavior described by matrix B. Denote this probability as $\wp_i$.
3. Assume $\rho_i = 1$ and $\rho_j = 0$ for all $j \neq i$.
4. With statistical dynamic timing simulation, calculate the matrix M_{crt} and the matrix E_{crt}. As a result, we can obtain the signature probability matrix S_{crt}.

   Let $S_{crt} = [S_1, \ldots, S_{|TP|}]$, where each S_j is a column vector, for $1 \leq j \leq \mid TP \mid$.
5. For each j, $1 \leq j \leq \mid TP \mid$, calculate $P_j =$

$$\begin{bmatrix} b_{1j}s_{1j} + (1-b_{1j})(1-s_{1j}) \\ b_{2j}s_{2j} + (1-b_{2j})(1-s_{2j}) \\ \cdots \\ b_{m|O|j}s_{|O|j} + (1-b_{|O|j})(1-s_{|O|j}) \end{bmatrix} = \begin{bmatrix} p_{1j} \\ p_{2j} \\ \cdots \\ p_{|O|j} \end{bmatrix}$$

   Probability p_{ij}, $i = 1 \ldots \mid O \mid$ represents the probability that the behavior at the output i under vector j is consistent with the observed behavior on the failing chip.
6. From P_j, calculate $\phi_j = \Pi_{k=1}^{|O|}(p_{kj})$. Treat ϕ_j as an approximation to the probability that B_j is caused by $\rho_i = 1$, i.e., the behavior of all the primary outputs under vector j matches the observed behavior.
7. For all $1 \leq j \leq \mid TP \mid$, after calculating all ϕ_j, we calculate $\wp_i$, the overall probability that the faulty behavior B is caused by ρ_i (defect i is the cause). Essentially, each method below implies a different diagnosis error function (how each S_{crt} matches to B).

   Method I $\wp_i = 1 - \Pi_{j=1}^{|TP|}(1-\phi_j)$. $(1-\phi_j)$ is the probability that the defect cause $\rho_i = 1$ is not consistent with the output behavior by test pattern v_j. Hence, $\Pi_{j=1}^{|TP|}(1-\phi_j)$ is the probability that $\rho_i = 1$ is not consistent with the output behavior by all test patterns. Consequently, $\wp_i$ is the probability that $\rho_i = 1$ (defect i) is the cause for the behavior of at least one test pattern in TP.

   Method II $\wp_i = \frac{\Sigma_{j=1}^{|TP|}\phi_j}{|TP|}$. In this way, $\wp_i$ is the average probability that $\rho_i = 1$ is the cause for the output behavior of a test pattern in TP.

   Method III $\wp_i = \Pi_{j=1}^{|TP|}(\phi_j)$. $\wp_i$ is the probability that the defect $\rho_i = 1$ is the cause (or is consistent with) the output behavior of all test patterns in TP.

   As it can be seen, each method has a different way to decide which defect is the most probable cause for the faulty behavior.
8. Let $\wp = [\wp_1, \wp_2, \ldots, \wp_{|S|}]$. To diagnose the faulty behavior, we rank them as $\wp_{j_1} \geq \wp_{j_2} \geq \cdots \geq \wp_{j_{|S|}}$ and output the first K defects $\{\rho_{j_1}^{vec}, \ldots, \rho_{j_K}^{vec}\}$ as the answer.

The key in the above algorithm lies in Step 5. In that step, we calculate $p_{kj} = b_{kj}s_{kj} + (1-b_{kj})(1-s_{kj})$ for $1 \leq k \leq |O|$. Then, we calculate the quantity $\phi_j = \Pi_{k=1}^{m}(p_{kj})$ in Step 6. We treat ϕ_j as the probability that B_j is caused by fault i. The meaning can better be explained through an example.

Example E.1 *Suppose $O = \{o_1, o_2, o_3\}$, i.e., we have a 3-output circuit. By applying a test pattern v_j, suppose we observe $B_j^T = [0,1,1]$. Also assume that the jth column in the signature probability matrix S_j is $S_j^T = [0.4, 0.3, 0.1]$. Now we compute p_{kj} as the following.*

$$p_{1j} = 0 \times 0.4 + (1-0) \times (1-0.4) = 0.6$$
$$p_{2j} = 1 \times 0.3 + (1-1) \times (1-0.3) = 0.3$$
$$p_{3j} = 1 \times 0.1 + (1-1) \times (1-0.1) = 0.1$$

As we can see, we basically "flip" the probability (meaning that we get $1-p$) if the corresponding entry in B_j is 0 (no error). Otherwise, we keep the probability for that entry. Therefore, we obtain $P_j = [0.6, 0.3, 0.1]$ as the probability vector such that given $\rho_i = 1$, the circuit behavior on pattern v_j would match that specified in B_j. If we need to convert this into a single probability number, then we have

$$\phi_j = 0.6 \times 0.3 \times 0.1 = 0.018 \text{ (all three match)}$$

Therefore, with a probability 0.018, ρ_i will lead to the outcome B_j.

F. Defining An Explicit Diagnosis Error Function

The above algorithm is based upon the principle of selecting the most probable cause. Therefore, for each candidate defect i, the larger the probability $\wp_i$ is, the most likely the defect is the cause. The diagnosis error function in those methods is never explicitly defined. In this section, we propose a diagnosis error function explicitly and use the function to decide which fault is the most probable cause.

F-1. An Error Function Based Upon Euclidean "Distance"

Now, suppose our choice for D is limited, i.e. we can only pick the answer as one of the defect functions from the set $\{D_1, D_2, \ldots, D_{|E|}\}$, then how can we define which is the best choice?

Suppose we have an algorithm such that for each defect function D_i, our algorithm is able to compute, for a single output circuit, an n-dimensional (n is the number of total patterns, $|TP|$) probability vector $P_i = [p_{i1}, p_{i2}, \ldots, p_{in}]$. Each p_{ij} is the probability that output $y = 1$ for pattern v_j if D is actually D_i. Suppose the observed behavior for this output y is indeed the vector $y = [y_1, y_2, \ldots, y_n]$. Then, it seems that the Euclidean *distance* between the expected results P_i and the true outputs can actually be measured by

$$Err_i = || P_i - y ||^2 = \Sigma_{k=1}^{n} (p_{ik} - y_k)^2 \quad (4)$$

Then, we compute Err_i for all D_i, $1 \leq i \leq | E |$, and we can simply pick the minimum, i.e. pick the defect function that minimizes the error function defined in equation (4).

F-2. Error Under An Equivalence Checking Model

For multiple-output circuits, Figure 3 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 true defect function D is unknown. Moreover, the delay configuration of the failing chip instance is also unknown. What we know is only the behavior matrix B.

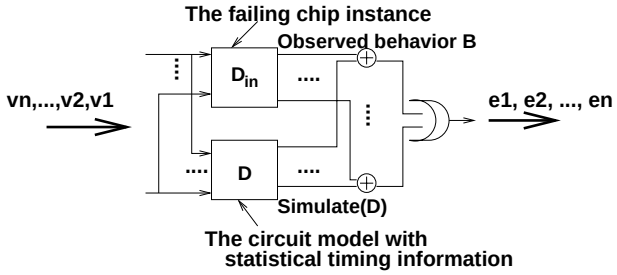

Figure 3: Error Under An Equivalence Checking Model

What is the ideal case? The ideal case, where no mismatch occurs, is that $e_1 = e_2 = \cdots = e_n = 0$. However, this is impossible even though we have correctly guessed the defect function in the diagnosis process. The reason is that we still do not know the exact delay configuration of the chip instance. Hence, we can only rely on the statistical timing model in order to make the most probable guess.

Now, suppose we have an algorithm to compute, for each potential defect function D_i, the probability p_{ij} that e_j is 1. In other words, the algorithm outputs an answer for D_i as $P_i = [p_{i1}, p_{i2}, \ldots, p_{in}]$. Then, since the ideal outcome we want to see is $\mathbf{0} = [0, 0, \ldots, 0]$, we can measure the Euclidean "distance" between the probability vector P_i and the ideal solution $\mathbf{0}$ as simply

$$Err_i = \Sigma_{j=1}^{n} (p_{ij})^2 \quad (5)$$

Equation (5) follows the same spirit as equation (4), both of them use the Euclidean distance to measure the error. Then, it is clear that we can use equation (5) to pick a defect function whose error is the minimum.

F-3. The Revised Algorithm

We revise the simple algorithm presented earlier in order to minimize the error function defined in equation (5).

Algorithm F.1 *(Revised Diagnosis Algorithm Alg_{rev})*

Inputs and Outputs Same as Alg_{sim}.

Steps Same as Alg_{sim}, except that we change step 7, and step 8

7. (Revised) For all j, $1 \leq j \leq | TP |$, after calculating all ϕ_j, we calculate $\wp_i^y = [\wp_{i1}, \wp_{i2}, \ldots, \wp_{i|TP|}]$ where each $\wp_{ij} = (1 - \phi_j)$. Hence, each $\wp_{ij}$ is the probability that *at least one* output has a mismatch (diagnosis error).

We then calculate $\wp_i = \Sigma_{j=1}^{|TP|} (\wp_{ij})^2$

8. (Revised) After we finish the calculation for all possible defect functions, we have $\wp = [\wp_1, \wp_2, \ldots, \wp_{|S|}]$. We rank them such that $\wp_{j_1} \leq \wp_{j_2} \leq \cdots \leq \wp_{j_{|S|}}$ and output the first K defects $\{\rho_{j_1}^{vec}, \ldots, \rho_{j_K}^{vec}\}$ as the answer.

End of Algorithm

G. Pattern Generation for Delay Fault Diagnosis

Our diagnosis algorithms still need "good" diagnostic patterns. However, generating good patterns for delay faults with timing consideration is a complex task. This is because a given fault can be sensitized through short or long paths and also because the delay of a particular path depends on the applied patterns. To ensure good diagnostic patterns for defects resulting from small-size delay faults (e.g., faults caused by crosstalk, bridging faults or by resistive opens or shorts) the first task is to select long paths to sensitize the faults. Next, a path delay fault test generator needs to produce a test such that it actually results in a long path delay for the given fault.

Due to the complexity reasons, most conventional path delay fault test generators do not take timing information into account and generate tests based purely on logic path sensitization conditions. Thus, the tests might not always exercise the worst-case timing scenarios for the given path. One possible solution for generating such tests could be a timed ATPG technique in which the timing information is used to guide each step in the test generation process. One such technique has been proposed in [12] for generating tests for detecting crosstalk induced delay faults. However, due to its complexity, timed ATPG might not be widely applicable in practice.

Another possibility could be to use Genetic Algorithm based ATPG techniques that can generate tests resulting in longer path delays based on a fitness function [11]. After assigning the mandatory values to sensitize a given path, usually there are still many unspecified values at the primary inputs. Different assignments of these unspecified values can result in different path delays.

H. Tools and Methodologies for Experiments

The key tools to realize the proposed diagnosis algorithms include a statistical timing analysis tool and a dynamic timing simulator. Moreover, to measure the effectiveness of each diagnosis method, we need to perform statistical defect injection and fault simulation.

H-1. Statistical Timing Analysis

In statistical timing analysis framework, the delays of cells/interconnects are modeled as correlated random variables with known probability density functions (pdf's). These pdf's can be obtained using a Monte-Carlo-based SPICE simulator. 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 using Monte-Carlo based simulation technique.

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-pin cell delays and output transition times (as random variables). We use a Monte-Carlo-based SPICE (ELDO) [18] to extract the statistical delays of cells for a 0.25μm, 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.

H-2. Dynamic Timing Simulation

With a given set of test patterns the statistical timing analysis framework can be used to perform statistical dynamic timing simulations 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 with different cell/interconnect delay assignments.

H-3. Defect Injection and Simulation

We experimented based upon the single defect model in definition D.10. This model can be used to represent small delay faults resulting from manufacturing defects, resistive opens and shorts, crosstalk or bridging faults.

H-4. Pattern Generation

For the injected fault and circuit instance, we find a set of "longest" paths through the fault site and generate path delay tests for them. The longest paths are derived using false-path aware static statistical timing analysis [17]. Path are tested with robust or non-robust patterns derived without considering timing.

I. Experimental Results

To measure the accuracy of a method, our approach is the following. For each circuit model C and a defect model D_s, we produce N circuit instances with different delay configurations. On each instance, we inject a delay defect of which both location and size are drawn randomly according to the model D_s. We then apply a diagnosis method to each instance. The accuracy of the diagnosis is measured in two ways: 1) In the algorithm, if the user-defined K value is 1 (refer to Algorithm E.1 above), then the accuracy is a binary value *success* and *failure* depending on if the answer matches the injected defect or not. 2) If the user-defined $K > 1$, then if the injected defect is *contained* in the potential defect set answered by the algorithm, then it is counted as a *success*; otherwise, it fails. 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.

	K	Alg_{sim}(%) I	Alg_{sim}(%) II	Alg_{rev} (%)		K	Alg_{sim} (%) I	Alg_{sim} (%) II	Alg_{rev} (%)
	1	0	5	10		1	0	15	20
s1196	3	0	30	30	s1238	2	5	25	25
	7	5	35	60		7	25	65	65
	1	10	15	10		1	5	5	5
s1423	2	30	35	35	s1488	3	35	30	30
	9	50	60	65		5	55	60	65
	1	15	25	25		2	25	30	30
s5378	2	30	40	45	s9234	5	40	50	50
	7	80	85	90		11	60	75	70
	1	10	20	20		1	10	10	10
s13207	5	30	50	60	s15850	2	30	30	30
	13	70	70	80		9	40	35	45

TABLE I: DIAGNOSIS ACCURACY ON BENCHMARK EXAMPLES

Table I shows results on the accuracy of diagnosis for different methods and different values for K for several benchmark examples. As expected, the rates of success increase for larger K. From our experiments, Method III of Algorithm D.1 seems to be too restrictive since it requires that none of the primary outputs for none of the vectors and none of the circuit delay configurations shows a behavior different than the observed behavior. Otherwise, $\wp_i = 0$ for fault i.

In these experiments $N = 20$. The number of applied test patterns used for diagnosis is usually smaller than 20. This is because in step (1), we have applied the *cause-effect* approach to prune down the fault candidate set. Then, the average number of suspect faults (faults logically sensitized by the paths ending at primary outputs that are observed as failing) varies per circuit and is in the range of 100 to 600. The random variable corresponding to the injected defect size has a mean that is in the range of 50% to 100% of a cell delay and we assume 3σ is 50% of the mean.

As discussed earlier, the accuracy of diagnosis depends on the set of test patterns and the quality of the test patterns applied in our experiments could probably be further increased by including timing information into the test generation process.

J. Conclusion

Alg_{rev} seems to be the best so far. This demonstrates the effectiveness of using an explicit error-function-driven approach. Therefore, our conclusion is that to develop a good diagnosis algorithm in the future, we need to search for a good error function first.

Future research includes many possible directions: 1) enhance the diagnosis test pattern quality, 1) improve the dynamic statistical timing simulator for more accurate delay fault simulation, 2) develop heuristics to select K automatically, 3) relax the restriction of the single defect assumption and see how that impacts the performance of the diagnosis algorithms, 4) reduce the expense of computing and storing the probabilistic fault dictionary, and 5) develop new error functions that are more consistent with the error definition in problem definition D.8, and develop new diagnosis algorithms accordingly.

References

[1] K. Baker, G. Gronthoud, M. Lousberg, I. Schanstra, , and C. Hawkins. Defect-Based Delay Testing of Resistive Vias-Contacts, A Critical Evaluation. *Proceedings of IEEE International Test Conference*, pages 467–476, September 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, IEEE VLSI Test Symposium*, April 1997.

[3] Robert C. Aitken, "Nanometer Technology Effects on Fault Models for IC Testing", *IEEE Computer*, November 1999, pp. 46-51

[4] K-T Cheng, S. Dey, M. Rodgers, and K. Roy, Test Challenges for Deep Sub-Micron Technologies, *ACM/IEEE Design Automation Conference* 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. *Proceedings of Asian South Pacific Design Automation Coneference*, pages 587–592, January 2000.

[6] Miron Abramovici, Melvin A. Breuer, and Arthur D. Friedman, Chapter 12: Logic Level Diagnosis, *Digital Systems Testing and Testable Design*, W.H.Freeman, 1990.

[7] Irith Pomeranz, and Sudhakar M. Reddy, Location of Stuck-At Faults and Bridging Faults Based on Circuit Partitioning *IEEE Trans. on Computers*, 1998, pp. 1124-1135

[8] D. B. Lavo, B. Chess, T. Larrabee, and F. J. Ferguson, Diagnosing Realistic Bridging Faults with Single Stuck-At Information. *IEEE Transactions on Computer-Aided Design*, pp. 255-268, March, 1998.

[9] J. Ghosh-Dastidar, and N. A. Touba, Diagnosing Resistive Bridges Using Adaptive Techniques. *IEEE Custom Integrated Circuits Conference*, pp. 79-82, 2000.

[10] M. Sivaraman, and A. J. Strojwas, Path Delay Fault Diagnosis and Coverage - A Metric and an Estimation Technique. *IEEE Transactions on Computer-Aided Design*, pp. 440-457, March, 2001.

[11] A. Krstic, J.-J. Liou, Y.-M. Jiang, and K.-T. Cheng, Delay Testing Considering Crosstalk-Induced Effects. *IEEE International Test Conference*, pp. 558-567, October, 2001.

[12] W.-Y. Chen, S. K. Gupta, and M. A. Breuer, Test Generation for Crosstalk-Induced Delay in Integrated Circuits. *IEEE International Test Conference*, pp. 191-200, October, 1999.

[13] P. Girard, C. Landrault, and S. Pravosssudovitch, A Novel Approach to Delay-Fault Diagnosis. *Design Automation Conference*, pp. 357-360, June, 1992.

[14] P. Pant, and A. Chatterjee, Path-Delay Fault Diagnosis in Non-Scan Sequential Circuits with At-Speed Test Application. *IEEE International Test Conference*, pp. 245-252, October, 2000.

[15] Hiroshi Takahashi, Kwame Osei Boateng, Yuzo Takamatsu, A New Method for Diagnosing Multiple Stuck-at Faults using Multiple and Single Fault Simulations, *Proceedings of 17th IEEE VLSI Test Symposium*, 1999.

[16] J.-J. Liou, K.-T. Cheng, and D. Mukherjee. Path Selection for Delay Testing of Deep Sub-Micron Devices Using Statistical Performance Sensitivity Analysis. *Proceedings of IEEE VLSI Test Symposium*, pages 97–104, April 2000.

[17] 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. *Proceedings of Design Automation Conference*, June 2002.

[18] Anacad. Eldo v4.4.x User's Manual. 1996.