Post-Silicon Diagnosis of Segments of Failing Speedpaths due to Manufacturing Variations

Lin Xie, Azadeh Davoodi, and Kewal K. Saluja
Department of Electrical & Computer Engineering
University of Wisconsin - Madison
Email: {lxie2,adavoodi,saluja}@ece.wisc.edu *

ABSTRACT

We study diagnosis of segments on speedpaths that fail the timing constraint at the post-silicon stage due to manufacturing variations. We propose a formal procedure that is applied after isolating the failing speedpaths which also incorporates post-silicon path-delay measurements for more accurate analysis. Our goal is to identify segments of the failing speedpaths that have a post-silicon delay larger than their estimated delays at the pre-silicon stage. We refer to such segments as "failing segments" and we rank them according to their degree of failure. Diagnosis of failing segments alleviates the problem of lack of observability inside a path. Moreover, root-cause analysis, and post-silicon tuning or repair, can be done more effectively by focusing on the failing segments. We propose an Integer Linear Programming formulation to breakdown a path into a set of non-failing segments, leaving the remaining to be likely-failing ones. Our algorithm yields a very high "diagnosis resolution" in identifying failing segments, and in ranking them.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**]: Design Aids

General Terms

Algorithms, Design

Keywords

Post-Silicon Diagnosis, Process Variations

1. INTRODUCTION

Post-silicon validation involves operating prototype chips to verify their correct behavior. In the past, incorrect behavior was mainly due to logical errors impacting the functional behavior of the chip. However, the presence of electrical failures, has now turned into a major hurdle. Electrical failures can be due to a combination of deep submicron effects such as power droop and crosstalk. In addition, manufacturing variations can be classified as a cause of electrical failures.

*This research is supported by National Science Foundation under awards CCF-0811082 and CCF-0811467.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
DAC 2010, June 13-18, 2010, Anaheim, California, USA.
Copyright 2010 ACM ACM 978-1-4503-0002-5 ...$10.00.

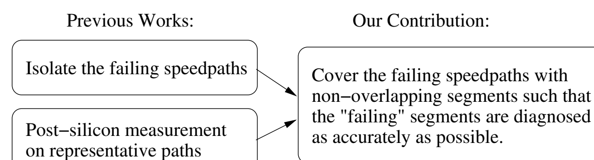

Figure 1: Overview of Our Approach

Among different types of electrical failures, timing failure is perhaps the most important over many application sectors. This type of failure occurs when a chip does not operate at a desired frequency due to a combination of various electrical problems such as power droop, crosstalk noise, or a "slower" manufacturing imprint.

Recent research has focused on isolating failures. In [8], a technique is proposed to dynamically isolate failures (including timing failures) at the micro-architecture level in which the isolation is at the block-level. In [1], a statistical learning approach is proposed to predict the failing speedpaths by measuring the delays of a small set of representative paths at the post-silicon stage. To help identify such representative paths, in [3], a technique is proposed which relies on defining a set of basic features (e.g., the number or types of logic gates) to rank the target speedpaths. In [5], a manually-guided technique is used to isolate failing speedpaths, via combination of techniques including clock shrinkage and using a debug tester. In [7], a branch-and-bound technique for isolation of speedpaths is proposed which is based on parameterized static timing analysis.

The focus of the above research has been on isolating the failing speedpaths. However, the major challenge after identifying a failing speedpath is to understand the cause of failure in order to effectively predict the remaining ones, or apply repair or tuning if such infrastructure is in place. This task is particularly challenging due to lack of observability, and inability to analyze different segments of a speedpath.

In this paper, we propose a formal procedure to identify segments of failing speedpaths. Our objective is to form segments with post-silicon delays larger than their pre-silicon ones, and then rank them according to the degree of their delay deviations. We refer to such segments as "failing segments". Identifying failing segments helps towards addressing the above challenges to drive a more focused analysis (e.g., the identified segments can help find layout patterns that are timing-sensitive).

As shown in Fig. 1, we assume failing speedpaths are first isolated, and timing measurements are made on representative paths at the post-silicon stage (for example using [5] or [9]) to incorporate actual delays to enhance the prediction.

More specifically, our contributions are listed below:

1. We propose a novel ILP formulation which breaks down a failing speedpath into provably non-failing segments, leaving the remaining edges on the path to be potentially-failing candidates. We show a very high "diagnosis resolution" in identifying the failing edges.
2. We provide a ranking of the failing edges based on the degree of delay deviations from the pre-silicon models, and show this ranking to be highly accurate.

In this work we assume the cause of a timing failure is manufacturing variations; i.e., dependency on specific instance of parameter variations in a manufactured chip. We also inject a random noise representing unknown silicon behavior in our analysis. While dynamic factors such as crosstalk or power-grid noise are also important causes of failure, they are not considered here. We believe focusing on manufacturing variations is still an important cause of some of the failures which we consider as a first step.

2. PRELIMINARIES

Given a timing-graph, let us assume the nodes represent logic gates and directed edges represent their interconnections. We assume each edge has a delay which captures the delay of its corresponding interconnect and the logic gate from where it initiates. We also assume primary input and output nodes have zero delay. We define a segment to be a sequence of connected edges on this graph.

In the presence of manufacturing variations, we assume $\mathbf{X}$ is a vector of the varying parameters such as channel lengths or threshold voltages of devices. We describe the delay of segment s using the following linear expression:

$$d_s = \mu_s + \mathbf{a}_s^T \mathbf{X}, \tag{1}$$

where μ_s is the average delay and $\mathbf{a}_s$ is the sensitivity vector with respect to parameter variations $\mathbf{X}$. In the special case when a segment is an edge, we can still get a similar linear expression. Also, the delay of a path in the timing-graph can be written as a summation of the delay expressions of its edges. This linear modeling is done using a standard procedure as discussed in [2] and is shown to be very accurate.

Please note that in this work we do not model the circuit delay which requires statistical maximum operations on arrival times under variability. We only need the above linear modeling of segments or paths which has been shown to be much less prone to error than circuit delay.

At the post-silicon stage, the instance of parameter variations (i.e., $\mathbf{X}$) are unknown. For a fabricated chip, the degree of variation might be such that the delays of some of the paths exceed their timing requirements. We assume such paths are isolated and their delays are measured as discussed in [1], [3], [5], [7], [9]. Moreover, we assume the delays of additional speedpaths are also measured. In this paper, we refer to these measured paths as representative paths.

Let us assume n representative paths $\mathcal{P} = \{p_1, ..., p_n\}$ are measured at the post-silicon stage. The *actual* delays of these paths are denoted by vector $\mathbf{d}_a$, while their variation-aware delays are given by

$$\mathbf{d}_{\mathcal{P}} = \boldsymbol{\mu}_{\mathcal{P}} + \mathbf{A}_{\mathcal{P}}^T \mathbf{X}, \tag{2}$$

where $\boldsymbol{\mu}_{\mathcal{P}}$ is the expected delay of the paths in $\mathcal{P}$, and $\mathbf{A}_{\mathcal{P}}$ is the matrix representing the sensitivities of each path with respect to the varying parameters. Using these actual measurements $\mathbf{d}_a$, we identify segments on the failing speedpaths. For a segment s we define

$$\bar{d}_s = d_s | (\mathbf{d}_{\mathcal{P}} = \mathbf{d}_a) \tag{3}$$

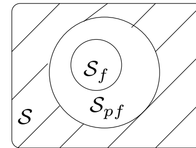

Figure 2: Relationship of different segment types.

In other words, $\bar{d}_s$ is a random variable expressing delay of s given the measurements on the n representative paths.

If we assume the varying parameters in $\mathbf{X}$ have Normal distribution, then we have $\bar{d}_s \sim N(\bar{\mu}_s, \bar{\sigma}_s)$ to also follow a Normal distribution [6]. The mean and standard deviation can be computed using the following equation as in [6]:

$$\begin{aligned} \bar{\mu}_s &= \mu_s + \mathbf{a}_s^T \mathbf{A}_{\mathcal{P}} \boldsymbol{\Sigma}_{\mathcal{P}}^{-1} (\mathbf{d}_a - \boldsymbol{\mu}_{\mathcal{P}}) \\ \bar{\sigma}_s^2 &= \sigma_s^2 - \mathbf{a}_s^T \mathbf{A}_{\mathcal{P}} \boldsymbol{\Sigma}_{\mathcal{P}}^{-1} \mathbf{A}_{\mathcal{P}}^T \mathbf{a}_s \end{aligned} \tag{4}$$

where $\boldsymbol{\Sigma}_{\mathcal{P}} = \mathbf{A}_{\mathcal{P}}^T \mathbf{A}_{\mathcal{P}}$ and σ_s is the standard deviation of d_s at pre-silicon stage. All the parameters in the above equation are known so mean and variance of $\bar{d}_s$ are calculated as soon as the representative path delay measurements are made.

3. PROBLEM DEFINITION

Let us denote an instance of parameter variations for a fabricated chip by $\mathbf{X} = \mathbf{K}$ ($\mathbf{K}$ is a constant vector). The actual delay of a segment s is now given by the expression $\bar{d}_s|_{(\mathbf{X}=\mathbf{K})}$, where the segment delay d_s is evaluated at $\mathbf{X} = \mathbf{K}$.

Now, we define failing and non-failing segments s on a path, which will be used in segment identification.

DEFINITION 1. *Segment s is "failing"* $\iff \bar{d}_s|_{(\mathbf{X}=\mathbf{K})} > \mu_s$.

In other words, the actual delay of s is larger than the expected delay given by pre-silicon model. A segment that is not failing, is referred to as "non-failing".

The above definition of a failing segment is intuitive. As we discuss later, our objective is to accurately identify the failing segments on the failing speedpaths without the knowledge of instance of parameter variations and only relying on post-silicon path-delay measurements.

The following lemma helps us identify some of these non-failing segments of a given path without requiring the knowledge of the instance of parameter variations.

LEMMA 1. *A sufficient condition for a non-failing segment s is $\bar{\mu}_s + 3\bar{\sigma}_s \leq \mu_s$.*

PROOF. Let us denote the left-hand-side of the inequality by $\bar{d}_s^{(wc)}$. It is the worst-case delay of segment s, given the path delay measurements at the post-silicon stage. The right-hand-side is the expected delay of s based on pre-silicon model in Eq. (1). If $\bar{d}_s^{(wc)} \leq \mu_s$ holds, then s is a non-failing segment based on our given definition. □

Similarly, we can conclude s is failing if $\bar{\mu}_s - 3\bar{\sigma}_s > \mu_s$.

The above lemma provides sufficient condition to identify some of the non-failing segments. However, it is not a necessary condition for a segment to be non-failing; i.e., a segment s can be non-failing when $\bar{d}_s^{(wc)} > \mu_s$ holds. In this case, the conditional variance of d_s, $\bar{\sigma}_s^2$, is too large and the actual measurements cannot provide enough information to interpret $\bar{d}_s$. Note that with increase in n, the number of post-silicon measurements, $\bar{\sigma}_s^2$ decreases which allows for higher number of segments to be identified as failing/non-failing.

In the absence of knowledge of instance of parameter variations, we categorize a segment into one of the two cases:

- Segment s is "definitely non-failing" $\iff \bar{\mu}_s + 3\bar{\sigma}_s \leq \mu_s$
- Segment s is "potentially failing" $\iff \bar{\mu}_s + 3\bar{\sigma}_s > \mu_s$

For an instance of parameter variations, we identify a set of failing speedpaths which we denote by $\mathcal{P}_f$. Let us assume these paths are covered by a set $\mathcal{S}$ of non-overlapping segments. We divide $\mathcal{S}$ into two subsets consisting of definitely non-failing and potentially failing segments and denote them by $\mathcal{S}_{dnf}$ and $\mathcal{S}_{pf}$, respectively. For the corresponding instance of variations, we denote the segments that are actually failing by $\mathcal{S}_f$. Fig. 2 shows the relationship between these sets, where the dashed area corresponds to $\mathcal{S}_{df}$, the definitely-failing segments. We further denote L as the number of edges covered by $\mathcal{P}_f$. The number of edges covered by $\mathcal{S}_{dnf}$, $\mathcal{S}_{pf}$, and $\mathcal{S}_f$ are denoted by L_{dnf}, L_{pf}, and L_f, respectively. Therefore $L_{pf} + L_{dnf} = L$.

For a given set of non-overlapping segments covering the failing paths $\mathcal{P}_f$, we define *diagnostic resolution*, DR, as

$$DR \triangleq \frac{L_f}{L_{pf}} \quad (5)$$

A high DR indicates that the set of potentially failing edges in $\mathcal{S}_{pf}$ is close to the actually failing ones in $\mathcal{S}_f$. It implies that the formation of segments $\mathcal{S}_{pf}$ for the failing speedpaths is a good one, providing high DR. Note, this definition is in terms of number of edges, and not number of segments.

Problem Definition: Given post-silicon delay measurements of n representative paths denoted by $\mathcal{P}$, and m failing paths $\mathcal{P}_f$ which have been isolated, our objective is to divide $\mathcal{P}_f$ into a set of non-overlapping segments and form $\mathcal{S}_{pf}$ such that diagnostic resolution given in Eq. (5) is maximized.

Note that our focus is on the segment identification problem, assuming the failing speedpaths are provided. *We assume isolation of failing speedpaths is done using existing works such as [1], [3], [5], [7], while post-silicon path delay measurement is done using existing techniques such as [9].*

4. SEGMENT IDENTIFICATION: ILP

In this Section, we first introduce a mathematical formulation of the segment identification problem, and then discuss how it is linearized into an ILP formulation.

4.1 Mathematical Formulation

Here we explain our formulation for the simple case when we solve the segment identification problem on one failing speedpath, given post-silicon delay measurements on n representative paths. We can then iteratively apply this formulation on each of the m failing speedpaths in $\mathcal{P}_f$.

The objective of our segment identification problem is to maximize DR, given in Eq. (5). Since L_f, the numerator of DR, is unknown for an instance of parameter variations, we alternatively minimize the denominator L_{pf} which is the number of edges in the potentially-failing segments. On the other hand, since $L_{pf} + L_{dnf} = L$ holds, minimizing L_{pf} is equivalent to maximizing L_{dnf} which is the number of edges covered by the definitely non-failing segments.

Therefore, in our approach to form $\mathcal{S}$, we focus on dividing $\mathcal{P}_f$ into a set $\mathcal{S}_{dnf}$ of definitely non-failing segments. This leaves the remaining edges to be potentially-failing segments (of length 1). For example, Fig. 3(a) shows a valid solution for a path which is composed of two definitely non-failing segments, while the remaining edges e_3 and e_8 are potentially-failing segments of length 1.

To achieve a correct mathematical formulation of the problem, we need to describe a definitely non-failing segment as a continuous connection of edges, and ensure that distinct definitely non-failing segments are apart by at least one edge.

To mathematically describe the above problem, we start by describing our notations with an example.

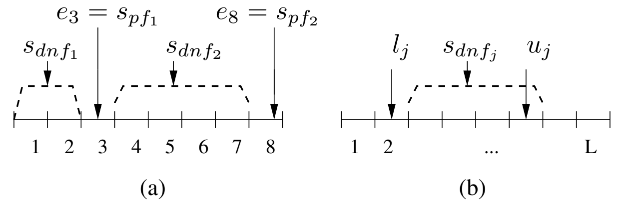

Figure 3: Illustration of our variable definitions.

Notations: Consider one speedpath with L edges $e_1, \ldots, e_L$, and all edges are labeled in increasing order from the path input to its output. For the definitely non-failing segment set $\mathcal{S}_{dnf}$, we denote its j-th segment by s_{dnf_j}, and also assume that these segments are indexed in increasing order from the path input towards its output.

Since we require that the definitely non-failing segments be apart from each other by at least one edge, an upper bound on the number of such segments is $N_m = \lfloor \frac{L}{2} \rfloor$. The remaining edges between these segments belong to $\mathcal{S}_{pf}$. We further define binary variables x_{ij} for $i = 1, 2, \ldots L$ and $j = 1, 2 \ldots N_m$. If edge e_i belongs to the j-th definitely non-failing segment s_{dnf_j}, we set x_{ij} to 1. Otherwise, x_{ij} is equal to 0, and e_i belongs to $\mathcal{S}_{pf}$ of potentially-failing segments.

Since the number of definitely non-failing segments is unknown and we only have its upper bound, we define binary variables Δ_j to denote whether s_{dnf_j} exists for $j = 1, 2 \ldots, N_m$. If Δ_j is set to 1, it indicates that s_{dnf_j} has been formed. Otherwise, it means that there is no need to form the j-th definitely non-failing segment.

We further define integer variables l_j and u_j for j=1, …, N_m. If s_{dnf_j} is not formed ($\Delta_j = 0$), we force $l_j = u_j = L$. Else, we have formed s_{dnf_j}, and u_j returns the edge index of the last edge covered by s_{dnf_j} while l_j gives the index of the edge immediately before the first edge covered by s_{dnf_j}. Fig. 3(b) illustrates the definitions of l_j and u_j.

Example: Consider Fig. 3(a). Here, this speedpath includes $L = 8$ edges. The maximum number of definitely non-failing segments N_m is equal to $L/2 = 4$. For the particular solution shown in the figure, we have two definitely non-failing segments s_{dnf_1} and s_{dnf_2}. The remaining edges, i.e., e_3 and e_8, belong to $\mathcal{S}_{pf}$. Therefore, we have $\Delta_1 = \Delta_2 = 1$ and $\Delta_3 = \Delta_4 = 0$. For s_{dnf_2}, we have $l_2 = 3$ and $u_2 = 7$. We also have $u_3 = l_3 = u_4 = l_4 = L = 8$.

The mathematical formulation is given by:

$$\max \quad \sum_{i=1}^{L} \sum_{j=1}^{N_m} x_{ij} \quad (6)$$

$$\text{s.t.} \quad \sum_{j=1}^{N_m} x_{ij} \le 1, \quad \forall i = 1, \ldots, L \quad (7)$$

$$\bar{\mu}_{s_j} + 3\bar{\sigma}_{s_j} \le \mu_{s_j}, \quad \forall j = 1, \ldots, N_m \quad (8)$$

$$\sum_{i=1}^{L} x_{ij} = u_j - l_j, \quad \forall j = 1, \ldots, N_m \quad (9)$$

$$l_j = L - \max_{i=1,\ldots,L} (L+1-i) x_{ij}, \forall j = 1, \ldots, N_m \quad (10)$$

$$u_j \ge \max_{i=1,\ldots,L} (i x_{ij}), \quad \forall j = 1, \ldots, N_m \quad (11)$$

$$u_j + \Delta_{j+1} \le l_{j+1}, \quad \forall j = 1, \ldots, N_m - 1 \quad (12)$$

$$\frac{l_{j+1} - (L-1)}{L} \le 1 - \Delta_{j+1} \le \frac{l_{j+1}}{L}, j = 2, ., N_m - 1 \quad (13)$$

$$x_{ij} \in \{0, 1\}, \quad \forall i = 1, \ldots, L; j = 1, \ldots, N_m \quad (14)$$

$$\Delta_j \in \{0, 1\}, \quad \forall j = 1, \ldots, N_m \quad (15)$$

Considering the objective, we aim to maximize DR, which alternatively turns into maximizing L_{dnf}. The L_{dnf} is equal to the total number of edges covered by the set $\mathcal{S}_{dnf}$, and can be expressed using Eq. (6); based on our definition, the edges that are covered by $\mathcal{S}_{dnf}$ will have $x_{ij} = 1$ and otherwise 0.

Eq. (7) indicates that each edge on the path should be covered by at most one definitely non-failing segment s_{dnf}. For edge e_i, if all x_{ij}s are 0 for $j = 1, 2, \ldots, N_m$, then e_i belongs to the set $\mathcal{S}_{pf}$ of potentially-failing edges.

Eq. (8) ensures that the j-th segment formed by our formulation is actually s_{dnf_j}. This is consistent with the definition of definitely non-failing segment explained in Section 3.

Eq. (9) ensures each segment is one continuous ordering of consecutive edges. This can be expressed based on our definition of variables l_j and u_j given for segment s_{dnf_j}. The inequality basically states that the summation of the number of edges covered by s_{dnf_j} should be equal to $u_j - l_j$.

Eq. (10) and (11) describe our definitions of variables l_j and u_j in terms of x_{ij}. The highest edge index on segment s_{dnf_j}, u_j, satisfies Eq. (11). Similarly, Eq. (10) gives the expression for l_j. In the example of Fig. 3, we verify these expressions as $l_2 = 8 - \max(2, 3, 4, 5) = 3$, and $u_2 = \max(4, 5, 6, 7) = 7$. Note, even though the expression for u_j is an inequality, Eq. (9) forces this to become an equality for formed segments s_{dnf_j}.

Eq. (12) uses Δ_{j+1} to ensure that the distinct definitely non-failing segments are at least apart by one edge. Recall that if the $(j+1)$-th segment $s_{dnf_{j+1}}$ is formed, then we have $\Delta_{j+1} = 1$. When $\Delta_{j+1} = 1$, we get $u_j + 1 \leq l_{j+1}$, indicating that the formed segments s_{dnf_j} and $s_{dnf_{j+1}}$ are apart by at least one potentially-failing edge.

In our formulation, we define N_m number of Δ_j variables, where N_m is the maximum number of $\mathcal{S}_{dnf}$ segments we can possibly form. However, when we cannot form N_m number of $\mathcal{S}_{dnf}$ segments, the Δ_js with higher indexes of j should be set to 0 for satisfying Eq. (12). Therefore, we have $u_j \leq l_{j+1}$. Combining this case with Eq. (9), we conclude that when $\Delta_{j+1} = 0$, we have $u_k = l_k = L$ for $k \geq j+1$. We use this special relationship to enforce the definition of Δ_{j+1} variable which we discuss next.

The binary variable Δ_{j+1} is defined using Eq. (13). When $\Delta_{j+1} = 0$, we have $u_j \leq l_{j+1} = L$. The inequality in Eq. (13) turns into $\frac{L-L+1}{L} \leq 1 \leq \frac{L}{L}$ which is always true. When $\Delta_{j+1} = 1$, we have $\frac{l_j - L}{L} \leq 0 \leq \frac{l_{j+1}}{L}$ which is again a valid inequality. Therefore, we enforce our definition of Δ_{j+1} using the alternative conditions of $u_j \leq l_{j+1} = L$ (when $\Delta_{j+1} = 0$) and $u_j + 1 \leq l_{j+1}$ (when $\Delta_{j+1} = 1$).

4.2 Linearization

In the formulation presented in Section 4.1, Eqs. (8), (10), (11) are in non-linear form. Here, we describe how to linearize these constraints **without approximation**.

The Eqs. (10) and (11) are nonlinear due to max operation. In general, we can define an auxiliary variable y_{max} to replace the max expressions, which are in the form of $\max_{i=1,\ldots a}(y_i)$. Then, we enforce y_{max} to satisfy: $y_1 \leq y_{max}$, ..., $y_a \leq y_{max}$.

Now, we are left to explain how Eq. (8) can be linearized. First, we equivalently rewrite Eq. (8) as

$$\mu_{s_j} - \bar{\mu}_{s_j} \geq 0, \quad \forall j = 1, 2, \ldots N_m \tag{16}$$

$$\left(\mu_{s_j} - \bar{\mu}_{s_j}\right)^2 \geq 9\bar{\sigma}_{s_j}^2 \quad \forall j = 1, 2, \ldots, N_m \tag{17}$$

where Eq. (16) can be transformed into

$$\mu_{s_j} - \bar{\mu}_{s_j} \geq 0 \Leftrightarrow \sum_{i=1}^{L} x_{ij} \left(\mu_{e_i} - \bar{\mu}_{e_i}\right) \geq 0. \tag{18}$$

The μ_{e_i} and $\bar{\mu}_{e_i}$ in the above equation are defined using Eqs. (1) and (4), when the segment is just a single edge. It also makes use of the linearity property in expectation on conditional random variables in writing μ_{s_j} and $\bar{\mu}_{s_j}$ in terms of μ_{e_i} and $\bar{\mu}_{e_i}$, respectively. Note that we can pre-compute μ_{e_i} and $\bar{\mu}_{e_i}$ for all edges using Eqs. (1) and (4). Consequently, Eq. (16) becomes in linear form.

To linearize Eq. (17), we first rewrite its left-hand-side in the following linear form:

$$\left(\mu_{s_j} - \bar{\mu}_{s_j}\right)^2 = \sum_{k=1}^{L} \sum_{i=1}^{L} x_{ij} x_{kj} \left(\mu_{e_i} - \bar{\mu}_{e_i}\right)\left(\mu_{e_k} - \bar{\mu}_{e_k}\right). \tag{19}$$

Here, we introduce binary auxiliary variables $\min(x_{ij}, x_{kj})$ to replace $x_{ij}x_{kj}$ since both x_{ij} and x_{kj} are binary variables. In addition, we need to introduce additional linear constraints for these auxiliary variables expressing the min operation, which is skipped due to lack of space.

We are left to transform the right-hand-side of Eq. (17) into a linear form. Using Eq. (4), we express $9\bar{\sigma}_{s_j}^2$ as

$$9\bar{\sigma}_{s_j}^2 = 9\left(\sigma_{s_j}^2 - \mathbf{a}_{s_j}^T \mathbf{A}_{\mathcal{P}} \mathbf{\Sigma}_{\mathcal{P}}^{-1} \mathbf{A}_{\mathcal{P}}^T \mathbf{a}_{s_j}\right) = 9\mathbf{a}_{s_j}^T \mathbf{T} \mathbf{a}_{s_j} \tag{20}$$

where $\mathbf{T} = \mathbf{I} - \mathbf{A}_{\mathcal{P}} \mathbf{\Sigma}_{\mathcal{P}}^{-1} \mathbf{A}_{\mathcal{P}}^T$ is a constant matrix and can be pre-computed.

Let us denote the (i, j)-th entry of $\mathbf{T}$ and the k-th entry of $\mathbf{a}_{s_j}$ by t_{ij} and $a_{s_{jk}}$, respectively. We rewrite $\bar{\sigma}_{s_j}^2$ as

$$\bar{\sigma}_{s_j}^2 = \sum_{k=1}^{m} a_{s_{jk}}^2 t_{kk} + \sum_{i \neq k} \sum_{k=1}^{m} a_{s_{ji}} a_{s_{jk}} (t_{ik} + t_{ki}), \tag{21}$$

Here, we have two non-linear terms $a_{s_{jk}}^2$ and $a_{s_{ji}} a_{s_{jk}}$. Due to the lack of space, we only discuss how to linearize the first term. We can follow the same procedures to linearize the other one. For $a_{s_{jk}}^2$, we have

$$a_{s_{jk}}^2 = \left(\sum_{i=1}^{L} a_{e_{ik}} x_{ij}\right)^2 = \sum_{i=1}^{L} \sum_{l=1}^{L} a_{e_{ik}} a_{e_{lk}} x_{ij} x_{lj} \tag{22}$$

Note, we have already introduced auxiliary variables to linearize $x_{ij}x_{lj}$, as previously discussed. Thus, we do not need additional variables or constraints to linearize $a_{s_{jk}}^2$.

Solving ILP for a path is extremely fast (fraction of a second in our simulations). For more paths, we iteratively apply the ILP and show this still gives us high accuracy in our simulations. Therefore, the use of ILP is justified for practical considerations.

5. SIMULATION RESULTS

Our framework was implemented using C++, and we used CPLEX 9.0 to solve the generated ILP formulation. In our experimental flow, we start by synthesizing ISCAS'89 benchmarks using 90nm TSMC library and Synopsys Design Compiler for minimum area under a stringent timing constraint to ensure having many critical paths. We assume parameter variations in effective channel length L_{eff} and zero-bias threshold voltage V_{th} to be Gaussian distributed with standard deviations of 5% and 10% of their mean, respectively.

Table 1: Results of our segment identification framework for two sets of failing speedpaths.

BENCH	$\lvert\mathcal{P}\rvert$	Small Violation Case								Large Violation Case			
		N_{FS}	N_{FP}	DR (%)	$\frac{L_{pf}}{L}$ (%) Before	After	L_{FP}	ΔN_e	DR_p (%)	N_{FS}	N_{FP}	DR (%)	DR_p (%)
S1423	119	1581	23.56	86.72	68.76	61.14	18.22	18	92.20	939	12.65	90.84	94.81
S1488	24	1289	2.09	85.46	74.74	59.91	10.16	9	86.92	660	1.79	89.61	91.07
S1494	11	1016	1.32	72.18	83.64	80.81	11.48	6	73.64	612	1.21	81.75	80.96
S5378	61	1547	19.36	87.27	69.12	62.80	9.99	7	90.89	1041	16.03	91.44	94.11
S9234	53	1228	141.80	71.23	60.32	55.38	11.27	9	76.35	835	122.10	72.97	80.22
S13207	37	1935	4.58	75.57	74.37	70.32	11.21	9	81.70	1137	2.90	82.79	86.33
S15850	34	1294	13.30	62.84	59.64	54.83	8.81	9	66.55	756	12.87	63.75	69.89
S35932	295	624	267.16	71.12	53.63	46.12	7.37	4	83.84	468	267.97	71.96	83.78
S38417	173	1950	56.09	82.06	53.77	40.59	11.76	14	86.95	1122	40.24	86.19	91.61
S38584	101	1286	72.56	73.45	44.69	31.07	8.44	6	79.89	826	69.59	75.25	82.63
Ave				76.79	64.27	56.30			81.89			80.65	85.54

To capture spatial correlation between the varying parameters, we use the multi-level hierarchical model of [2], which defines rectangular regions on the chip. The gates/interconnects in the same region or in physically close-by regions will share all or some of their parameter variations and be correlated to each other using this hierarchical model.

For smaller benchmarks (S1423 to S9234), we use a 3-level hierarchical model, resulting in $\sum_{i=0}^{3} 2^i = 21$ regions. In the remaining benchmarks, we use a 5-level model with $\sum_{i=0}^{5} 2^i = 341$ regions for each benchmark. This assumption is consistent with [6]. Consequently, for each region, we had two distinct random variables of L_{eff} and V_{th}.

To obtain a "golden model" capturing post-silicon behavior in our simulations, we assume the instance of parameter variation is known. We then experiment by analyzing many such instances generated using Monte Carlo (MC) simulation reflecting various post-silicon cases. Furthermore, for each variation instance, we assume an additional Gaussian-distributed error of ϵ_i is introduced for the delay of each gate i representing a mismatch between pre- and post- silicon delay models. We randomly generate this error for each gate and assume it to be at most 6% of the nominal gate delay in our simulations.

Our framework requires as input, post-silicon delay measurements on representative paths (which can include failing ones). However, the problem of representative path selection is outside the scope of this paper and is studied separately, such as in [3]. While our framework can take as input any method for selecting and measuring representative paths, the important factor in our simulations is to ensure that the measured paths are indeed representative so they can more effectively reflect the silicon impact. We briefly summarize our procedure to get measured representative paths:

1. We first identify a large set of critical paths $\mathcal{P}_c$, using pre-silicon timing models for nominal case of variability. These paths are selected if their nominal delays fall within a 20% window of the timing constraint.
2. We use the procedure of [10] to identify a subset of these paths denoted by $\mathcal{P}$, as representative paths ($\mathcal{P} \subseteq \mathcal{P}_c$). These paths are highly correlated with the target critical paths in presence of variability. We indeed verified that our representative paths could predict the delays of critical paths with an average error of less than 3% over a large set of parameter variation instances.
3. We assume the measured delays of these representative paths are obtained using the explained "golden model". So we assume an instance of parameter variations $\mathbf{X}$ and an instance of random noise for each gate, and incorporate these in Eq. 2 to find the delay of each path for each simulation.

5.1 Comparisons on Diagnostic Resolution

We generate MC samples for parameter variations, which follow Gaussian distributions. Due to the very large dimension of the parameter variations, we generate several MC sample sets and all of them include 2,500 samples. We chose the sample set, which yields the largest number of failing samples, for performance validation. We declare an instance to be a "failing sample" if at least one path in P_C fails the timing. Recall, we assume failing paths are identified and isolated using existing techniques, such as [7]. Here, we use our golden model to localize failing paths for a variation instance, while the representative paths are determined once and remain the same over all variation instances. The number of failing samples N_{FS} is given in column 3 of Table 1. Note this number is high compared to the total number of samples for some benchmarks. It is because 2,500 samples are selected to cover a high number of failing samples, representing various failure possibilities. In addition, this number also differs among benchmarks due to their topologies.

For a failing sample, we then measure representative path delays ($\mathcal{P}$) according to our golden model. Our framework then generates and solves the ILP formulation using these measurements to identify segments on the failing paths.

Here, we consider two sets of failing speedpaths $\mathcal{P}_{f,1}$ and $\mathcal{P}_{f,2}$. For an instance of parameter variations, $\mathcal{P}_{f,1}$ includes all failing speedpaths in $\mathcal{P}_f$ with timing violation less than 3% of the timing constraint. It represents small violation case. Similarly, $\mathcal{P}_{f,2}$ includes all failing speedpaths with timing violation between 3% and 6% of the target circuit delay. It represents large violation case. Note that the paths in $\mathcal{P}_{f,1}$ and $\mathcal{P}_{f,2}$ are different for each variation case and are defined based on the post-silicon delays.

In table 1, Column 2 provides the number of paths ($\lvert\mathcal{P}\rvert$) for direct measurement. Columns 3-10 show the performance of our framework for small violation case where the segment identification is performed over the set of $\mathcal{P}_{f,1}$. Column 4 gives the average number of failing paths (N_{FP}) in our MC simulation. Column 5 gives the average diagnostic resolution (DR) after ILP formulation. As can be seen, the average DR over all circuits is very high (76.79%). *Note that in our simulations the actual failing set $\mathcal{S}_f$ was indeed contained in the potentially failing set $\mathcal{S}_{pf}$.*

Columns 6-7 give the ratios of the average number of potential failing edges (L_{pf}) with respect to the total number of edges covered by the failing speedpaths (L). We compute this percentage before ILP (where each edge is a segment of length 1 of either definitely non-failing or potentially failing types). We also compute this percentage after ILP (where some edges are merged into definitely non-failing segments). As can be seen, after ILP, we can reduce this percentage by on-average 8% in small and 13% in large benchmarks.

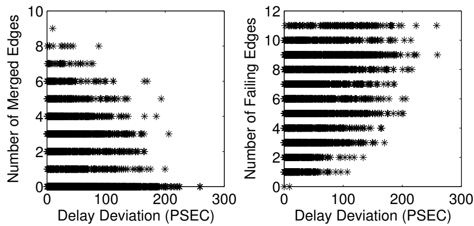

Figure 4: Scatter plots of number of edges merged into segments, and number of actual failing edges drawn with respect to the path delay deviation.

We also report the average length of the failing speedpaths (L_{FP}) in Column 8. Column 9 gives the *maximum* number of edges which can be merged (ΔN_e) in our MC Simulation. Compared with Column 8, we sometimes merge a large proportion of edges into non-failing segments.

Column 10 gives the average diagnostic resolution of a path (DR_p) over MC samples. The DR_p is defined with respect to each path; the ILP is solved for each path and the diagnostic resolution measured for each path separately, and then averaged over the number of paths and the MC samples. As can be seen, DR_p is very high, indicating that the set of potentially-failing edges per path is very close to the actually-failing ones, if paths are individually considered.

Columns 11-14 give simulation results for large violation case (the set of failing speedpaths is $\mathcal{P}_{f,2}$). In this case, we observed that the values of DR and DR_p are higher than those in small violation case. This could be due to the fact that in the large violation case, there are likely more number of failing edges on the paths (L_f is larger).

To further show the effectiveness of our ILP formulation, we performed another experiment considering all failing speedpaths (for all the post-silicon delay violation cases). For each path, we recorded the number of edges merged into definitely non-failing segments after applying our ILP formulation. We also recorded the number of actual failing edges (obtained using golden model) over all the violation cases. Fig. 4 shows the scatter plots of these two quantities for S1488. As shown, for the cases of small delay violations, we can merge more edges into non-failing segments. For large violations, since more edges are prone to failure, there is less benefit in solving the ILP.

Note that solving ILP for each path is extremely fast (fraction of a second), which makes our algorithm practical.

5.2 Comparisons using Other Metrics

Consider small violation case (i.e., the set of failing speedpaths is $\mathcal{P}_{f,1}$). We define two ranked lists of 1) potentially failing segments, and 2) actually failing ones. This allows us to evaluate *a match ratio* as we discuss later. The two lists are constructed as follows. After applying our framework, we obtain a set of potentially failing segments $\mathcal{S}_{pf}$, and for each one compute the expected segment delay given the path measurements (i.e., $\bar{\mu}_s$) using Eq. (4). We rank these segments in non-increasing order in terms of the segment delay deviation ($\bar{\mu}_s - \mu_s$), and derive a rank list $\mathcal{L}_1$. We also rank all the segments in $\mathcal{S}_f$ in terms of their actual post-silicon delay deviations from μ_s in non-increasing order to get the list $\mathcal{L}_2$. Note the length of all segments in $\mathcal{L}_1$ and $\mathcal{L}_2$ is 1.

Using a control parameter $\eta \in (0, 1]$, we define a window of size m to compare the segments of the two ranked lists, where $m \triangleq \max(\eta|\mathcal{L}_2|, \min(10, |\mathcal{L}_2|))$. This ensures that for fair comparison, the window size is at least 10 segments, or the entire list (if the list size is less than 10 segments).

Table 2: Results for rank list evaluation.

BENCH	$\lvert G\rvert$	MRRL (%) 20%	40%	100%	Sharing Info n_{fe}	n_e
S1423	1045	93.98	94.23	97.35	5.44	4.13
S1488	702	95.71	95.70	96.93	1.44	1.30
S1494	711	89.97	89.97	90.83	1.15	1.12
S5378	1911	92.52	92.38	96.22	4.91	4.11
S9234	1688	84.51	85.88	92.71	22.26	16.45
S13207	2185	87.77	87.86	91.70	1.95	1.70
S15850	1007	82.16	82.77	85.53	2.59	1.87
S35932	16202	89.42	91.26	93.02	1.90	1.23
S38417	14772	90.65	90.97	94.50	6.92	4.31
S38584	11944	91.68	92.46	93.21	4.25	2.10
Ave		89.84	90.35	93.20		

We then consider the first m segments of the two ranked lists $\mathcal{L}_1$ and $\mathcal{L}_2$ and define the metric "match ratio of the rank list" as $\text{MMRL}(\eta) = \frac{n}{m}$, where n is the number of segments in the window of $\mathcal{L}_2$ which were contained in the window of $\mathcal{L}_1$. A high value of MMRL indicates a high accuracy of our framework. Table 2 gives MMRL for $\eta = 20\%$, 40%, and 100%. Column 2 gives the total number of gates in the entire circuit. Columns 3-5 show that for nearly all cases, MMRL is larger than 85% indicating a large matching. It indicates that *our framework can determine the top* η *(in %) failing segments very accurately, even when* η *is small.*

In addition, Columns 6-7 provide the sharing information of the failing speedpaths. For the failing speedpaths, first we consider the *failing edges* and compute the average number of failing speedpaths going through each failing edge (over all the paths and MC samples). We denote this by n_{fe}. Next, we consider all the *failing and non-failing edges* of the failing speedpaths. We then compute the average number of failing speedpaths going through each edge. We denote this by n_e. As shown in Table 2, n_{fe} is higher than n_e, indicating that the edges which are covered by a higher number of failing paths are more probable to fail. We plan to incorporate this observation to improve our framework in future.

6. CONCLUSIONS

We presented a framework for fast post-silicon segment diagnosis on failing speedpaths, which was based on our proposed ILP formulation. We showed through simulation that we can achieve a very high diagnostic resolution of failing segments in the speedpaths. A failing segment means that its silicon delay may be deviating (substantially) than its pre-silicon delay. We also show a higher accuracy for ranking the failing segments based on the degree of failure.

7. REFERENCES

[1] P. Bastani, et al. Speedpath prediction based on learning from a small set of examples. *DAC*, 2008.

[2] D. Blaauw, et al. Statistical timing analysis: From basic principles to state of the art. *IEEE TCAD*, 27(4), 2008.

[3] N. Callegari, et al. Path selection for monitoring unexpected systematic timing effects. *ASPDAC*, 2009.

[4] K.-H. Chang, et al. Reap what you sow: spare cells for post-silicon metal fix. *ISPD*, 2008.

[5] K. Killpack, et al. Case study on speed failure causes in a microprocessor. *IEEE DTC*, 25(3):224–230, 2008.

[6] Q. Liu and S. S. Sapatnekar. Synthesizing a representative critical path for post-silicon delay prediction. *ISPD*, 2009.

[7] S. Onaissi, et al. PSTA-based branch and bound approach to the silicon speedpath isolation problem. *ICCAD*, 2009.

[8] S.-B. Park, et al. IFRA: instruction footprint recording and analysis for post-silicon bug localization in processors. *DAC*, 2008.

[9] X. Wang, et al. Path-RO: a novel on-chip critical path delay measurement under process variations. *ICCAD*, 2008.

[10] L. Xie, A. Davoodi. Representative path selection for post-silicon timing prediction. *Technical Report, University-Wisconsin ECE 09-04, 2009.*