Refined Statistical Static Timing Analysis Through Learning Spatial Delay Correlations*

Benjamin N Lee, Li-C. Wang
Dept. of ECE, UC-Santa Barbara
Santa Barbara, CA

Magdy S. Abadir
Freescale Semiconductor, Inc.
Austin,TX

ABSTRACT

Statistical static timing analysis (SSTA) has been a popular research topic in recent years. A fundamental issue with applying SSTA in practice today is the lack of reliable and efficient statistical timing models (STM). Among many types of parameters required to be carefully modeled in an STM, spatial delay correlations are recognized as having significant impact on SSTA results. In this work, we assume that exact modeling of spatial delay correlations is quite difficult, and propose an experimental methodology to resolve this issue. The modeling accuracy requirement is relaxed by allowing SSTA to impose upper bounds and lower bounds on the delay correlations. These bounds can then be refined through learning the actual delay correlations from path delay testing on silicon. We utilize SSTA as the platform for learning and propose a Bayesian approach for learning spatial delay correlations. The effectiveness of the proposed methodology is illustrated through experiments on benchmark circuits.

Categories and Subject Descriptors: B.8.2 [Hardware]: Performance and reliability

General Terms: Algorithms, Design, Performance

Keywords: Statistical timing, Bayesian learning, delay correlations

1. MOTIVATION OF THE WORK

Statistical static timing analysis (SSTA) has attracted much attention in recent years (for example [1–6]). SSTA is attractive because traditional worst-case corner timing analysis has become overly conservative due to the magnitude of process variations at 90nm nodes and below. SSTA aims to recover the timing margin that is lost due to the unrealistic pessimism in traditional worst-case timing tools. Recovering this lost margin can give additional flexibility to design optimization tools and facilitate design timing closure.

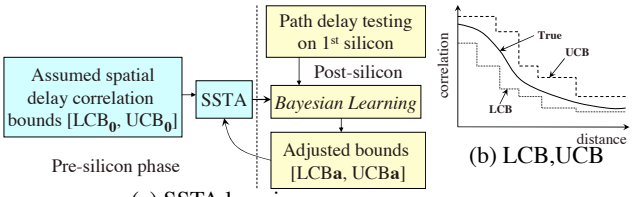
Currently, if we were to take advantage of SSTA technology in practice, one immediate challenge would be obtaining a reliable and efficient statistical timing model (STM). The difficulties are twofold: (1) The statistical data required to develop a reliable STM may not be easily obtainable from a foundry. (2) Characterizing and representing statistical information in a statistical timing model can be much more expensive compared to traditional worst-case corner methods.

*This work was supported in part by National Science Foundation, Grant No. 0312701 and Semiconductor Research Corporation, project 2004-TJ-1173.

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 2006, July 24–28, 2006, San Francisco, California, USA.
Copyright 2006 ACM 1-59593-381-6/06/0007 ...$5.00.

As discussed in [7], there are several issues in process variability modeling and characterization, which may impede modeling for a statistical static timer. Statistical process characterization demands tremendous silicon and test resources and hence, is usually not done frequently. Careful tracking of parameter variations over time can be overly expensive, which leads to gradual decrease in accuracy of a process variability model. Moreover, there are other sources of variability such as environmental sources of variability and systematic layout-dependent variability which may be hard to characterize early in a design cycle. These issues, due to either economic reasons or methodology-related reasons, make worst-case modeling favorable, as modeling just the worst-case bounds reduces the inter-dependency between process characterization and timing modeling.

Among the many types of parameters that need to be carefully considered to develop an STM, spatial correlations are recognized to have significant impact on design timing [8,9], and hence on SSTA analysis results [2, 3]. Characterizing spatial correlations across device and interconnect parameters (such as L_{eff}, V_{th}, ILD) can be complex and resource consuming [10]. Moreover, it is not entirely clear how to effectively model (or aggregate) spatial correlations based on parameters of devices and interconnects [10] into the spatial correlations based on delay elements at the cell level [2] [3] for efficient cell-based SSTA. In addition, correlations may exist between different parameters, which can further complicate the modeling issues [10].



(a) SSTA learning (b) LCB,UCB

Figure 1: Refined SSTA through Bayesian learning of correlations

In this work, we assume that SSTA is cell based. We also assume that obtaining an accurate model on spatial delay correlations at the cell level is difficult. To relax the accuracy requirement on correlation modeling, we assume that SSTA is given an upper-bound model and a lower-bound model of spatial delay correlations. For example, we may assume that the true spatial delay correlation is a smooth distance-based function [9]. Because we do not know the exact shape of the function, we assume an upper-bound step function and a lower-bound step function to bound it (as illustrated in Figure 1). The naive upper bound model is to assume that all delay elements are 100% correlated. Similarly, the extreme lower-bound model is that all delay elements are fully independent. Our objective is to refine these bounds through learning from the results of path delay testing on the silicon.

Figure 1 should be viewed as a *timing margin recovery* methodology, rather than a methodology for design timing closure. How to take advantage of the recovered margin is not suggested, and can be up to the user. In this work, we focus the discussion on developing the

methodology. We note that the learned spatial delay correlations are aggregate results from both design-related and process-related sources. Hence, the learned correlations are based on cell delay elements rather than on individual device or interconnect parameters.

Our learning approach is Bayesian based so that the estimation of correlations can take a *prior* into account. In the proposed methodology, the prior is the initial spatial delay correlation bounds $[LCB_0, UCB_0]$. After learning, we calculate a *posterior* that provides adjusted bounds $[LCB_a, UCB_a]$. It is important to implement a Bayesian based learning so that posterior bounds can be the *prior* for learning in the next run. In contrast, traditional maximum likelihood based statistical inference does not support this prior-posterior iteration [11].

Given that the correlations are constrained within $[LCB_a, UCB_a]$, what correlation values $\vec{\theta} \in [LCB_a, UCB_a]$ cause the worst-case timing in SSTA? This issue is crucial as we do not want SSTA to report optimistic results. Hence, we need to make sure that SSTA with $\vec{\theta} = LCB_a$ and SSTA with $\vec{\theta} = UCB_a$ also represent the lower and upper bounds on the worst-case timing. Otherwise, we need to implement a search method to find such a worst-case $\vec{\theta}$.

The rest of the paper is organized as the following. Section 2 reviews the background of SSTA and illustrates the impact of spatial correlations on SSTA results. Section 3 studies this impact in detail and suggests the use of UCB_a in SSTA for obtaining the worst case. Section 4 presents our Bayesian learning method. Section 5 summarizes the experimental results. Section 6 concludes the paper.

2. BACKGROUND

SSTA can be categorized into *path-based* SSTA [5] and *block-based* SSTA [1,6], where various techniques to perform $+$ and $\max$ of correlated random variables based on Gaussian and non-Gaussian assumptions have been proposed. How to model and handle spatial correlations in SSTA is also an important research topic. Various spatial correlation models have been introduced [2,3].

Our work was inspired by the work in [4] where several interesting observations were made regarding the practical use of statistical timing methods. Suppose we are given with a n-stage path. Suppose that the delay of each stage i can be characterized as a Normal distribution $d_i \sim N(\mu_i, \sigma_i^2)$ for $1 \leq i \leq n$. The path delay P_d is simply the summation of these n Normal distributions: $P_d \sim N(\mu, \sigma^2)$

where $\mu = \Sigma_{i=1}^{n} \mu_i$ and $\sigma = \sqrt{\Sigma_{i=1}^{n} \sigma_i^2 + 2\Sigma_{i=1}^{n} \Sigma_{j=i+1}^{n} (\rho_{ij} \sigma_i \sigma_j)}$

$\rho_{ij} = 1$ for $i = j$ and ρ_{ij} for $i \neq j$ denotes the correlation between the delay d_i and delay d_j. If $\rho_{ij} = 0$ for all $i \neq j$, i.e. delays are mutually independent, then we see that $\sigma = \sigma_{ind} = \sqrt{\Sigma_{i=1}^{n} \sigma_i^2}$. If $\rho_{ij} = 1$ for all i, j, i.e. delays are fully correlated, then we see that

$$\sigma = \sigma_{full} = \sqrt{\Sigma_{i=1}^{n} \sigma_i^2 + 2\Sigma_{i=1}^{n} \Sigma_{j=i+1}^{n} (\sigma_i \sigma_j)} = \Sigma_{i=1}^{n} \sigma_i.$$

If $\rho_{ij} = a_{ij}$ for all $i \neq j$ where $0 < a_{ij} < 1$, we have

$$\sigma = \sigma_a = \sqrt{\Sigma_{i=1}^{n} \sigma_i^2 + 2\Sigma_{i=1}^{n} \Sigma_{j=i+1}^{n} (a_{ij} \sigma_i \sigma_j)}$$

Similarly, we can define σ_b by letting $\rho_{ij} = b_{ij}$.

For simplicity, we assume $0 < a_{ij} < b_{ij} < 1$ for all $i \neq j$. Then we see that $\sigma_{ind} < \sigma_a < \sigma_b < \sigma_{full}$. Because correlations do not impact the resulting mean μ, we can also say that $\mu + 3\sigma_{ind} < \mu + 3\sigma_a < \mu + 3\sigma_b < \mu + 3\sigma_{full}$. In other words, the 3σ worst-case delay of the path increases *monotonically* as the correlations increase. The upper bound is when $\rho_{ij} = 1$ and the lower bound is when $\rho_{ij} = 0$.

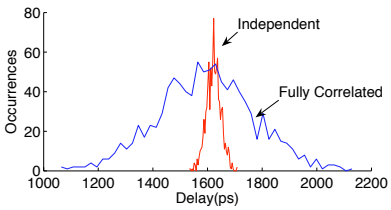


Figure 2: c2670 timing distribution (x-axis is in psec)

The analysis based on this simple path view gives the intuition on why delay correlations are important. Without knowing the actual correlations between delay elements, we can only bound the actual path standard deviation σ to be between σ_{ind} and σ_{full}. Figure 2 presents the Monte Carlo simulation results of the worst-case delay points on 1000 samples (occurrences) on the benchmark c2670 based on a 90nm statistical cell-based timing library characterized in the past [12]. In the fully-correlated case, we assume all delay random variables are 100% correlated. If we assume that delays are neither totally independent nor fully correlated, we would expect the worst-case delay distribution to be somewhere between these two extremes.

3. BOUND ON WORST-CASE TIMING

In this paper, when we refer to worst-case timing, we are referring to a simple notion of the 3-σ value of the circuit delay distribution of a combinational circuit. Thus the only operations we are concerned with are addition and $\max$. It's quite clear that the worst case value of a sum of random variables increases monotonically with a monotonic increase in the correlations between the variables. That is given n delay random variables $d_1 \ldots d_n$, their correlations can be specified with a symmetric positive semi-definite $n \times n$ correlation matrix $\rho = [\rho_{ij}]$. Suppose we have bounds on this matrix U and L. Let $U = [U_{ij}]$ be another symmetric positive semi-definite matrix such that $U_{ij} \geq \rho_{ij}$ for all i, j. Similarly, let $L = [L_{ij}]$ be the matrix such that $L_{ij} \leq \rho_{ij}$ for all i, j. Let $A = \Sigma_{i=1}^{n} d_i$. From the previous analysis, we have seen that $\mu_A + 3\sigma_L \leq \mu_A + 3\sigma_\rho \leq \mu_A + 3\sigma_U$ where σ_L, σ_ρ and σ_U are the standard deviations of A calculated using the respective correlation matrices.

Hence, given $[L, U]$ bounds on the correlations, we see that $[\mu_A + 3\sigma_L, \mu_A + 3\sigma_U]$ are also the bounds on the worst-case timing $\mu_A + 3\sigma_\rho$ This is a result for a single path. However, in a real circuit, the circuit delay distribution can be thought of as the result of applying a sequence of $+$ operations for the delays along each path and finally a single $\max$ operation. Thus, increasing correlation will have an effect on both the $+$ operations and the $\max$ operation.

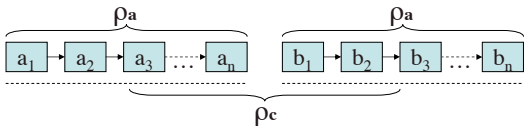
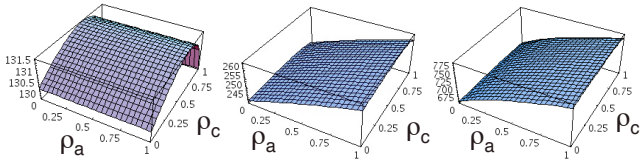


Figure 3: A simple example of mixing $+$ and $\max$

Figure 3 shows a simple example by assuming an intra-path correlation coefficient ρ_a and an inter-path correlation coefficient ρ_c. By assuming that each delay element is Normal $N(100, 10^2)$, we plot the 3-σ worst case delay ($\max$ of the two paths) calculated from Clark's formula by varying ρ_a and ρ_c.



(a) n=1 (b) n=2 (c) n=6

Figure 4: ρ_a dominates the 3σ worst case delay calculation

In Figure 4-(a), the plot is with $n = 1$ (path length $= 1$), and hence ρ_a does not apply. We note that in this plot, we see a non-monotonic effect on the worst case timing from increasing ρ_c. With $n = 2$ as shown in plot (b), we see that non-monotonicity nearly gone as the worst case delay is *dominated* by the value of ρ_a. Plot (c) shows a similar case with $n = 6$. The monotonic dependency of the worst-case delay on the value of ρ_a can clearly be observed.

Note that in this simple example, ρ_a impacts the $+$ operations on individual path delays and ρ_c impacts the $\max$ of the two delays. If we assume $\rho_a \in [l_a, u_a]$ and $\rho_c \in [l_c, u_c]$, we see that the worst-case delay primarily depends on $\rho_a = u_a$ and depends much less on ρ_c.

The dependency on ρ_c diminishes when the path length n increases. We can continue examining this on full circuits.

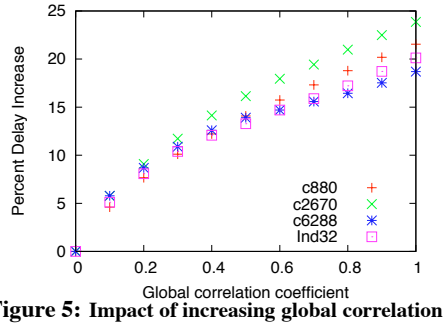


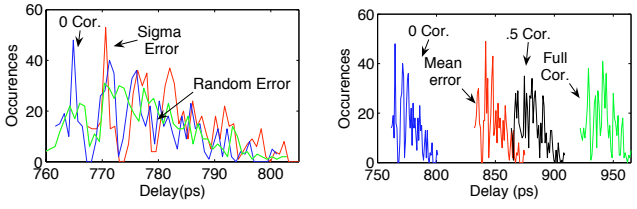
Figure 5: Impact of increasing global correlation ρ

Figure 5 shows Monte Carlo simulation results on three benchmarks and one industrial custom design, Ind32. In this experiment, we assume a global correlation coefficient ρ applicable to all pairs of delay random variables. We plot the percentage of change on the "$\mu + 3\sigma$" worst-case point from the circuit delay distribution given by each ρ value. We clearly see that this delay point increases as ρ increases.

In theory, we know that the "$\mu + 3\sigma$" worst-case delay based on SSTA-computed circuit delay distribution using the upper correlation bound UCB, may not be the true worst case. However, the analysis shown in this section suggests that in practice, applying SSTA with the UCB gives a good approximation on the worst-case circuit delay. This assumption simplifies the methodology in Figure 1. Otherwise, the methodology needs to include a delay maximization method to search for the worst-case timing in $[LCB, UCB]$.

3.1 Correlations vs. other modeling errors

In a cell-based STM, there are other ways for the model to be inaccurate. How does the impact of spatial correlations compare to those from other sources of modeling errors? We experimentally consider other sources of modeling errors: (1) Random errors: A random zero-mean Gaussian noise is added to each pin-to-pin delay in the model with the standard deviation equal to 10% of the delay. (2) Sigma shift: The standard deviation of each pin-to-pin delay is increased by 10%. (3) Mean shift: The mean of each pin-to-pin delay is increased by 10%. (4) Correlation 0.5: A global correlation coefficient 0.5 is applied to every pair of delay variables. (5) Correlation 1: A global correlation 1.0 is applied. In addition, we use the case where all delay variables are independent, i.e. correlation = 0 as the basis for comparison.



(a) (b)

Figure 6: Ind32 critical path delay histograms

We select the 1000 most critical paths based on the base case by running block-based SSTA [6] on Ind32. For each path, we focus on its "$\mu + 3\sigma$" delay point. Figure 6-(a) shows the path delay histogram on the base case ("0 Cor."). In addition, we also show the results of applying the error assumptions (1) and (2) above. We observe that random Gaussian noise and systematic shift of standard deviation do not alter the path delay results much. In contrast, plot (b) shows the results of applying the assumptions of (3), (4), and (5). We can clearly see the impact of delay correlations on these path delays. The simple experimental analysis demonstrates the importance of correct modeling of spatial delay correlations and therefore, motivates us to develop a method to learn the actual correlations from silicon.

4. LEARNING THE CORRELATIONS

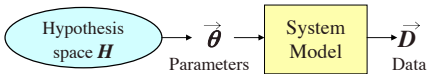


Figure 7: Bayesian inference: $Pr(\vec{\theta}|\vec{D}, \vec{H})$

Bayesian parameter estimation is a framework for estimating unknown system parameters $\vec{\theta}$ given observed data vector $\vec{D}$ and an assumed hypothesis space $\mathcal{H}$. Figure 7 illustrates the concept.

$$Pr(\vec{\theta}|\vec{D}, \mathcal{H}) = \frac{Pr(\vec{D}|\vec{\theta}, \mathcal{H}) Pr(\vec{\theta}|\mathcal{H})}{Pr(\vec{D}|\mathcal{H})}, i.e. \text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}} \quad (1)$$

Equation 1 re-states the Bayes rule [11]. The *evidence* can be expressed as $Pr(\vec{D}|\mathcal{H}) = \int_{\vec{\theta}} Pr(\vec{D}|\vec{\theta}, \mathcal{H})\ Pr(\vec{\theta}|\mathcal{H})\ d\vec{\theta}$, which is the normalizing constant to make the posterior a valid probability density.

In learning spatial delay correlations, we assume that the parameter space consists of k correlation coefficients $\vec{\theta} = [\rho_1, \ldots, \rho_k]$. As an example, one can model the spatial correlations by a distance-based function such as that illustrated in Figure 1 before. Then, each ρ_i can be the value at a step in the distance-based step function, i.e. if the distance between two delay elements is between a given range $[c_i, c_{i+1}]$, then their correlation is estimated as ρ_i. In the extreme case, if we are given n delay variables, we can assume $k = {}_nC_2$, i.e. each pair of variables has a unique correlation between them.

For each ρ_i, we assume that ρ_i falls into a range $[l_i, u_i]$. We let $UCB = [u_1, \ldots, u_k]$ and $LCB = [l_1, \ldots, l_k]$. Hence, the hypothesis space $\mathcal{H}$ is the vector space bounded by UCB and LCB. In between UCB and LCB, because initially we do not have knowledge on $\vec{\theta}$, we can assume a uniform distribution in the space, i.e. for each $\rho_i, Pr(\rho_i \in [x, y]) = |y - x|/|u_i - l_i|$ for $l_i < x < y < u_i$. We note that the learning framework allows any distribution to be specified in the prior, as long as it is efficiently computable.

4.1 Naive method for correlation learning

The naive method for learning correlations via path delay testing is to measure the delay of a path, apply the measured value to a theoretical path delay formula and solve for the correlations. This method is extremely unrobust since there are many other factors that affect the path delay. For example, there may be a systematic error in the characterization of the individual delay elements that is manifest as a shift in the mean or standard deviation. It is therefore very risky to use correlation to explain the absolute delay observed.

4.2 Overview of our approach

Instead of basing our learning on absolute path delays, we utilize path delay correlations as the measured data. This makes the learned results less sensitive to other modeling errors such as systematic shift in the mean or sigma discussed in Section 3.1 before. This point will be illustrated later in Section 4.7.

In our method, we measure the delay correlation between paths and derive the theoretical correlation between paths as a function of the individual delay element correlations. If we were only interested in obtaining point estimates of the individual delay element correlations, we could directly solve for the individual delay element correlations. However, we're interested in using the Bayesian framework to obtain the confidence of the point estimate. Thus, we derive the likelihood function for an observed path correlation and provide a method for determining the mean and standard deviation from the posterior probability density of the individual correlations.

4.3 Measuring path delay correlations

Given two paths P_a, P_b, suppose that we can measure their path delays on t silicon samples to obtain the measurements $M_a = [x_1, \ldots, x_t]$ and $M_b = [y_1, \ldots, y_t]$, respectively. We assume that this can be achieved by varying testing frequency in path delay testing.

There are several test related issues to be resolved in order to ensure accurate measurement of these path delays. These issues include, how to generate a test pattern to sensitize a path but not other paths, how to apply the test pattern in a scan mode properly, etc. We do not go into these issues because they are out of the scope of this paper. However, we note that the selection of P_a, P_b can be flexible in our methodology.

Given M_a, M_b, we calculate the Pearson's correlation coefficient:

$$r_{ab} = \frac{\Sigma_i(x_i y_i) - (\Sigma_i x_i)(\Sigma_i y_i)/t}{\sqrt{\Sigma_i(x_i^2) - (\Sigma_i x_i)^2/t}\sqrt{\Sigma_i(y_i^2) - (\Sigma_i y_i)^2/t}} \quad (2)$$

We call r_{ab} a measured path correlation. For learning, our data space $\vec{D}$ consists of several measured path correlations.

4.4 Bayesian inference of $\vec{\theta}$ based on r_{ab}

Given r_{ab}, we would like to express the likelihood as a function of $\vec{\theta}$. Let P_a consist of n delay elements $\{a_1, \ldots, a_n\}$ and let P_b consist of m delay elements $\{b_1, \ldots, b_m\}$. By assuming Gaussianality on all random variables, we have

$$C_{ab} = Corr(P_a, P_b) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} Cov(a_i, b_j)}{\sigma_{p_a}\sigma_{p_b}} \quad (3)$$

where $Cov(a_i, b_j) = \rho_{[a_i b_j]}\sigma_{a_i}\sigma_{b_i}$, and

$$\sigma_{p_a}^2 = \sum_{i=1}^{n}\sigma_{a_i}^2 + 2\sum_{i=1}^{n}\sum_{j=i+1}^{n}\rho_{[a_i a_j]}\sigma_{a_i}\sigma_{a_j}$$

$$\sigma_{p_b}^2 = \sum_{i=1}^{n}\sigma_{b_i}^2 + 2\sum_{i=1}^{n}\sum_{j=i+1}^{n}\rho_{[b_i b_j]}\sigma_{b_i}\sigma_{b_j}$$

We note that $\rho_{[a_i b_j]}$, $\rho_{[a_i a_j]}$, and $\rho_{[b_i b_j]}$ can be determined from the parameters in $\vec{\theta}$ and the distances between a_i and b_j, between a_i and a_j, and between b_i and b_j, respectively. σ_{a_i} and σ_{b_i} can be determined from the given statistical timing model. Let $\vec{\sigma_a} = [\sigma_{a_1}, \ldots, \sigma_{a_n}]$ and $\vec{\sigma_b} = [\sigma_{b_1}, \ldots, \sigma_{b_m}]$. Then, we see that equation 3 says that C_{ab} is a function of $\vec{\theta}$, $\vec{\sigma_a}$, and $\vec{\sigma_b}$:

$$C_{ab} = Corr(P_a, P_b) = f(\vec{\theta}, \vec{\sigma_a}, \vec{\sigma_b}) \quad (4)$$

Given C_{ab}, we ask the question: what is the probability of observing r_{ab}, i.e what is $Pr(r_{ab}|C_{ab})$? Because $\vec{\sigma_a}$ and $\vec{\sigma_b}$ can be determined from the timing model, the only unknown is $\vec{\theta}$. Hence, we can rephrase the question as: what is the probability $Pr(r_{ab}|\vec{\theta})$?

Suppose we can express $Pr(r_{ab}|\vec{\theta}) = g(r_{ab}, \vec{\theta})$. We see that given a set of values on the parameters in $\vec{\theta}$, we can calculate the probability of r_{ab}. It is more interesting to see that if we are given a value of r_{ab}, g becomes a function $h_{r_{ab}}(\vec{\theta})$ that entirely depends on $\vec{\theta}$ only. Multiplying $h_{r_{ab}}(\vec{\theta})$ with the prior $Pr(\vec{\theta}|\mathcal{H})$ and normalizing the result, we can obtain the posterior probability distribution $Pr(\vec{\theta}|r_{ab}, \mathcal{H})$.

Recall that equation 2 is an estimation of the true path correlation C_{ab} based on the observed samples. Hence, r_{ab} in essence is also a random variable characterizing the *sampling distribution* of the correlation coefficient. This sampling distribution is actually a skewed, non-normal form. If we express $Pr(r_{ab}|\vec{\theta})$ directly with r_{ab}, $g(r_{ab}, \vec{\theta})$ becomes complex and hard to manage. Thus, it is convenient to take the *Fisher Transformation* of r_{ab} [13] and obtain a new statistic z_{ab} so that the result is mathematically more manageable. After the transformation, z_{ab} can be expressed as a Normal distribution whose mean is the transformation of the theoretical correlation C_{ab} [13]. Hence, we have (the resulting equation 5 is a Normal distribution):

Let $z_{ab} = tanh^{-1} r_{ab} = \frac{1}{2}\ln(\frac{1+r_{ab}}{1-r_{ab}})$, (Fisher Transform [13])

Let $\mu_{z_{ab}} = tanh^{-1} C_{ab}$ and let $\sigma_{z_{ab}} = \frac{1}{\sqrt{t-3}}$

$$Pr(r_{ab}|\vec{\theta}) = g(r_{ab}, \vec{\theta}) \sim N(\mu_{z_{ab}}, \sigma_{z_{ab}}^2) = \frac{e^{-\frac{(z_{ab}-\mu_{z_{ab}})^2}{2(t-3)}}}{\sqrt{2\pi}\sqrt{t-3}} \quad (5)$$

In the actual application of equation 5, r_{ab} is a value calculated using equation 2 based on the measured path delays on the t sample chips. $C_{ab} = f(\vec{\theta}, \vec{\sigma_a}, \vec{\sigma_b})$ is calculated based on a simple path-based SSTA on the two paths P_a, P_b. We note that in this path-based SSTA, $\vec{\sigma_a}, \vec{\sigma_b}$ are calculated as vectors of values based on the timing model. However, $\vec{\theta}$ is kept *symbolically* because it is the unknown vector of parameters to be estimated. Once the value r_{ab} (and hence z_{ab}) is determined, equation 5 expresses $Pr(r_{ab}|\vec{\theta})$ as a function of $\vec{\theta}$ only, i.e. denoted as $h_{r_{ab}}(\vec{\theta})$ as mentioned above. Hence, we have

$$Pr(\vec{\theta}|r_{ab}, \mathcal{H}) \propto Pr(r_{ab}|\vec{\theta}) * Pr(\vec{\theta}|\mathcal{H}) = h_{r_{ab}}(\vec{\theta}) * prior \quad (6)$$

4.5 From r_{ab} to $\vec{R} = \{r_1, \ldots, r_s\}$

In the above analysis, we establish a likelihood function given a single sampled delay correlation between a pair of paths. To extend this likelihood function for s pairs of paths, ideally, we would replace the scalar correlation r_{ab} with a vector of correlations $\vec{R} = \{r_1, \ldots, r_s\}$, and directly obtain a joint likelihood function over the s observations. However, to our best knowledge, there is no closed form for such a function with multiple Pearson correlations. Our solution is to use a *Naive Bayesian* approach [11]. This approach assumes that the different observations $r_i$are conditionally independent. Although this may be flawed, it is an assumption that is often used to simplify Bayesian approaches.

$$Pr(\vec{R}|\vec{\theta}) = Pr(r_1|\vec{\theta}) * Pr(r_2|\vec{\theta}) ... * Pr(r_s|\vec{\theta}) = h_{\vec{R}}(\vec{\theta}) \quad (7)$$

In other words, the multivariate likelihood function is the product of the individual likelihoods. Multiplying the multivariate likelihood with the prior, we have

$$Pr(\vec{\theta}|\vec{R}, \mathcal{H}) \propto Pr(\vec{R}|\vec{\theta}) * Pr(\vec{\theta}|\mathcal{H}) \quad (8)$$

To normalize $Pr(\vec{R}|\vec{\theta}) * Pr(\vec{\theta}|\mathcal{H})$ for obtaining the probability density function for $Pr(\vec{\theta}|\vec{R}, \mathcal{H})$, we need to calculate $\int_{\vec{\theta}} (Pr(\vec{R}|\vec{\theta}) * Pr(\vec{\theta}|\mathcal{H}))\, d\vec{\theta}$ which can be quite complicated. Because our objective is to learn the statistics such as the mean and standard deviation of the estimated $\vec{\theta}$ in order to get a confidence interval on $\vec{\theta}$, it is not necessary to explicitly express the probability density $Pr(\vec{\theta}|\vec{R}, \mathcal{H})$.

4.6 Rejection sampling

To get the mean and standard deviation of the posterior $Pr(\vec{\theta}|\vec{R}, \mathcal{H})$, we utilize *rejection sampling* method [11]. Although there are more involved methods such as Metropolis-Hastings sampling, rejection sampling is sufficient for our purposes. In rejection sampling, we first draw a random sample $\vec{\rho}$ from the parameter space $\vec{\theta}$ according to the prior $Pr(\vec{\theta}|\mathcal{H})$. Recall that we can assume a prior where in $\mathcal{H}$, $\vec{\theta}$ has a multivariate uniform distribution in $[LCB, UCB]$ to begin with. Then, we need to decide if the random sample $\vec{\rho}$ should be accepted or rejected according to the likelihood $h_{\vec{R}}(\vec{\theta})$ in equation 7 above.

To implement the rejection sampling, we need to find the maximum point $max = \max\{h_{\vec{R}}(\vec{\rho})|\forall\vec{\rho} \in [LCB, UCB]\}$. This can be done through numerical maximization using a standard mathematical software package. Then, we let $h'_{\vec{R}} = h_{\vec{R}}/max$ so we know that $\forall\vec{\rho}$, $h'_{\vec{R}}(\vec{\rho}) \leq 1$. The rejection sampling is shown below.

> Randomly draw a sample $\vec{\rho}$ according to the prior.
> Randomly draw a sample u from distribution $U(0, 1)$.
> If $u < h'_{\vec{R}}(\vec{\rho})$, accept $\vec{\rho}$ as a sample. Otherwise, reject $\vec{\rho}$.
> Repeat process until there are sufficient samples or convergence criteria is met.

After we obtain sufficient samples, we can then calculate the sample mean μ_{ρ_i} and sample standard deviation σ_{ρ_i} for for each of the parameters, ρ_i. This allows us to estimate a confidence interval such as $[\mu_{\rho_i} - \sigma_{\rho_i}, \mu_{\rho_i} + \sigma_{\rho_i}]$ for each correlation parameter ρ_i.

It is interesting to note that we can store the un-normalized posterior (Equation 8) and reuse it as the prior for the next run of learning, for example, based on a new set of sample chips. Hence, for the new run, we could replace $\mathcal{H}$ with Equation 8 when drawing the sample $\vec{\rho}$.

4.7 Systematic Mean and Sigma shifts

Our Bayesian estimation of the correlations $\vec{\theta}$ does not utilize cell delay means. Hence, if there is a systematic shift on the cell delay means, this shift would not impact the estimation. It can also be shown that systematic shift in the delay standard deviations also do not impact the estimate.

Based on the formulation in equation 3, let us assume that there is a scale factor α being applied to all delay standard deviations, i.e. $\sigma'_{a_i} = \alpha\sigma_{a_i}$ and $\sigma'_{b_j} = \alpha\sigma_{b_j}$. We see that $Cov'(a_i, b_j) = \alpha^2 Cov(a_i, b_j)$, i.e. the covariance will be scaled by the factor α^2. In addition, we have $(\sigma'_{p_a})^2 = \Sigma_i \alpha^2 \sigma_{a_i}^2 + 2\Sigma_i \Sigma_j \alpha^2 \rho_{a_i a_j} \sigma_{a_i} \sigma_{a_j}$. Hence, $(\sigma'_{p_a})^2 = (\alpha\sigma_{p_a})^2$. Similarly, $(\sigma'_{p_b})^2 = (\alpha\sigma_{p_b})^2$. As a result, we have

$$C'_{ab} = \frac{\alpha^2[\sum_i \sum_j Cov(a_i, b_j)]}{\alpha^2(\sigma_{p_a}\sigma_{p_b})} = C_{ab} \quad (9)$$

Therefore, a systematic shift in the delay standard deviations by a factor α would not change the theoretical correlation between the two paths. As a result, the inference is not sensitive to this systematic shift either. This immunity to systematic shifts makes this method of estimating correlation much more robust than the naive method.

5. EXPERIMENTAL RESULTS

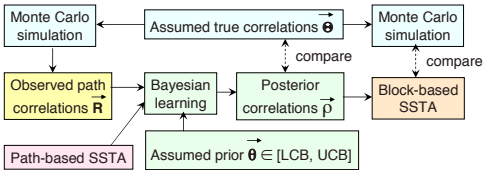


Figure 8: Experimental flow

The experimental flow is summarized in Figure 8. The path-based SSTA was discussed in Section 4.4 before. For the block-based SSTA, we implemented the method in [6]. We used the placement tool from [14] to obtain the x-y location of each gate. In the experiments, we used $[LCB, UCB] = [\vec{0}, \vec{1}]$ as the prior. A "true" correlation model $\vec{\Theta}$ was assumed in each case. Parameter values in $\vec{\Theta}$ were used only in the Monte Carlo simulator. $\vec{\rho}$ is the Bayesian estimation of $\vec{\Theta}$.

To show the results, we can (1) compare the parameters values in between $\vec{\Theta}$ and $\vec{\rho}$, and (2) compare the 3σ circuit timing bound from the block-based SSTA based on $\vec{\rho}$, to the "true" 3σ timing bound from the Monte Carlo simulation based on $\vec{\Theta}$. We note that before the learning, we also have a 3σ timing bound (the "prior" bound) calculated by the block-based SSTA based on the upper correlation bound UCB.

In the results, when we say "timing bound," we always refer to the "$\mu + 3\sigma$" point from a delay distribution. For a parameter $\rho \in \vec{\theta}$, our learning method estimates the mean of ρ (μ_ρ) and the standard deviation of ρ (σ_ρ). Hence, in the block-based SSTA, we have the choice of using the correlation value μ_ρ (mean), the correlation value $\mu_\rho + \sigma_\rho$ (1-sigma), or the correlation value $\mu_\rho + 3\sigma_\rho$ (3-sigma). Hence, in the results shown, when we show "mean," "1-sigma," or "3-sigma" in a figure, we always refer to these correlation value(s) in use. Moreover, when we use the "number of samples," we refer to the number of samples in the Monte Carlo simulation producing the path delay results.

5.1 Global view Vs. Local view

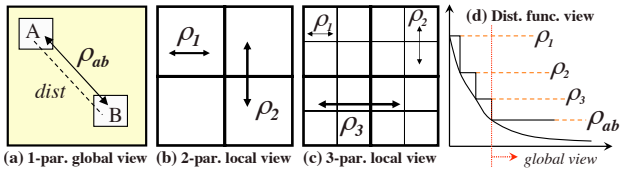


Figure 9: Global view Vs. local views

We conducted two types of experiments: (1) learning a global correlation and (2) learning local correlations. Figure 9 illustrates the difference. In the simple case as illustrated in Figure 9-(a), we consider two blocks A and B separated by a long distance $dist$. Given two paths P_a (in A) and P_b (in B), the assumption is that the distance spanned by each path is much shorter than $dist$. Essentially we can treat $dist$ as the distance between any pair of delay elements a_i, b_j, for $a_i \in P_a, b_j \in P_b$. Then, because delay correlations are distance dependent, there is only one parameter ρ_{ab} to be learned between all pairs of delay elements in between P_a and P_b. Local correlations inside A, B still affects individual path delays but they do not affect the correlation between the two paths. Note from Section 3 (Figure 4) that, local intra-path correlations dominate the worst-case timing of $\max(P_a, P_b)$ (by assuming that they are much larger than ρ_{ab}).

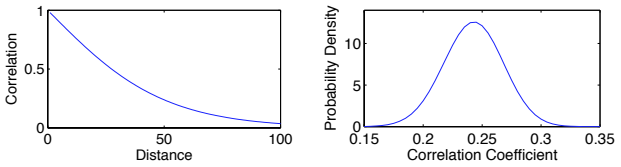
If we focus on P_a itself, we cannot make the same assumption that delay correlation between a pair a_{i_1}, a_{i_2} is the same as that between another pair a_{i_1}, a_{i_3}. We need a higher-resolution model. In Figure 9-(b) and (c), we provide a grid view by discretizing a correlation distance function. With this grid view, we assume that if two delay element is separated by a distance across w grids, then their correlation is given by ρ_w. We also assume that $\rho_1 > \rho_2 > \cdots > \rho_w$. Note that inside a block, a path may pass through multiple grids.

We adopted the grid model in the experiments in order to maintain the efficiency in the Monte Carlo simulation of circuit delay and the block-based SSTA. As far as the learning method concerns, it could be based on an arbitrary distance based model. This is because the learning utilizes only the simple path-based SSTA where handling complex correlations is more efficient [5] than the block-based SSTA. For the block-based SSTA [6], we could not afford to model correlations for every pair of delay elements. A grid model could dramatically reduce the number of correlation parameters to be considered.

One thing to note is that for block-based SSTA, a hierarchical grid model [3] is preferred. Our grid model is a flat model. Hence, for running the block-based SSTA, we need to convert our model into a hierarchical model, which can be done by applying Cholesky Decomposition or other more efficient methods (such as reverse PCA [15]).

5.2 Single parameter experiment - Figure 9-(a)

For the experiment, we assume an arbitrary correlation distance function shown in Figure 10-(a). The number 50 (in abstractly defined units) is also arbitrary. We selected a path P of 20 delay elements from circuit c2670. For simplicity we made two copies P_a, P_b of this path and placed them separated by a distance of 50 which gave a "true" correlation around 0.235 by setting $dist$=50 in Figure 10-(a). In the Monte Carlo simulation, the correlation function was applied to all pairs of delays in between the two paths as well as within each path. However, the path-based SSTA is aware of only one unknown correlation parameter ρ_{ab} to be learned (in between the two paths).

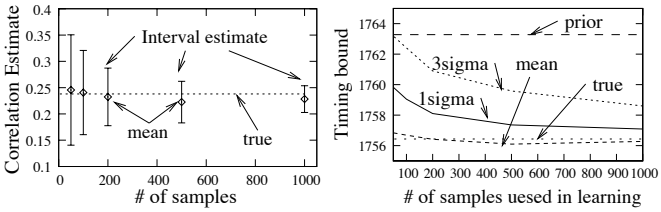


(a) $1 + Tanh(-dist/50)$ (b) Learned ρ_{ab}

Figure 10: Assumed distance function and learned ρ_{ab} distribution

Again, we assume $\rho_{ab} \leq UCB = 1$ in the prior. Based on delays from 1000 simulated samples, the result of estimating ρ_{ab} is shown in Figure 10-(b). Notice that the estimated result is a distribution which in this case, is a Normal like distribution. We obtain $\mu_{\rho_{ab}} = 0.228$ and $\sigma_{\rho_{ab}} = 0.025$. Figure 11-(a) shows the $[\mu_{\rho_{ab}} - \sigma_{\rho_{ab}}, \mu_{\rho_{ab}} + \sigma_{\rho_{ab}}]$ intervals at different sample numbers. The "true" line is the correlation value (0.235). We observe that the estimated interval for ρ_{ab} becomes smaller and closer to the truth as more samples are used.

Figure 11-(b) shows the timing bound comparison (in psec) based on the 3σ delay of $\max(P_a, P_b)$. As mentioned before, the "1sigma"



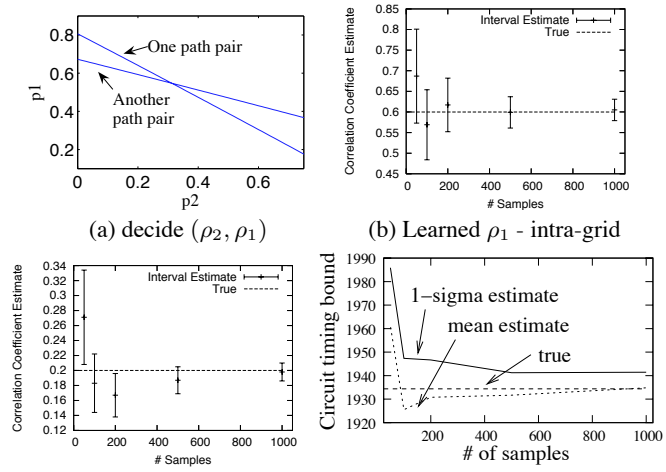
(a) Interval estimate (b) Timing bounds

Figure 11: Single correlation estimate results

curve is based on using the correlation value at $\mu_{\rho_{ab}} + \sigma_{\rho_{ab}}$. The "1sigma" curve represents a good upper bound to the true delay bound. The difference between the "prior" and the "1sigma" (or "3sigma") can be viewed as the timing margin recovered by learning the correlation. This margin is small because local correlations dominate the timing.

5.3 Experiments - Figures 9-(b) and (c)

In the 2nd experiment, we applied the model Figure 9-(b) on the circuit c2670. We selected two pairs of paths for learning. All the paths span more than one grid. This gives two path correlations. We need at least two path correlations because there are two unknown correlations ρ_1 and ρ_2 to be learned. One path correlation can only give us enough information to constrain a relationship between ρ_1 and ρ_2. For example, on a 2-dimensional ρ_1-vs-ρ_2 plot, learning from one path correlation can only give us a curve (looks linear) as shown in Figure 12-(a). We need at least two curves to find an intersection point.

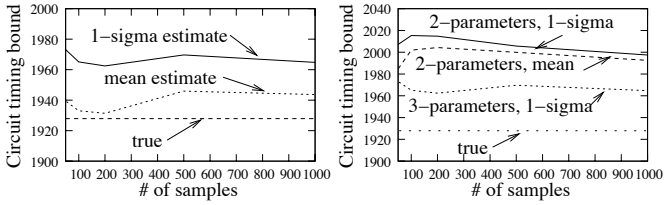


(a) decide (ρ_2, ρ_1) (b) Learned ρ_1 - intra-grid

(c) Learned ρ_2 - inter-grid (d) Timing bound results

Figure 12: c2670 2-parameter estimation results

Figures 12-(b) and (c) show interval estimation results on ρ_1 and ρ_2, respectively. (d) shows the timing bound comparison. We note that using the prior would give the timing bound at 2104.68 psec.



(a) 3-param. case (b) complicated case

Figure 13: c2670 3-parameter estimation results

Figure 13-(a) show results based on the 3-parameter model in Figure 9-(c). In this case, we selected three pairs of paths, each spanning across at least three grids. We observe again that the "1sigma" is an improved timing bound from the prior bound at 2104.68 psec.

Tables 1 and 2 summarize the results on four different circuits. We observe that in all cases, the "Prior" which assumes correlation 1.0 on all parameters are overly conservative. After learning, the "Posterior" results show tighter timing bounds and recovered timing margins.

Circuit	Posterior (1-sigma)	Prior	True	Margin recovered
c880	1310	1431.4	1293.8	9.38%
c1355	1340.5	1394.5	1338.03	4.04%
c2670	1941.4	2104.7	1934.4	8.44%
Ind32	917	982.6	907.8	7.22%

Table 1: 2-parameter results (in psec), Margin=(Prior-Posterior)/True

Circuit	Posterior (1-sigma)	Prior	True	Margin recovered
c880	1324.6	1431.4	1314.4	8.12%
c1355	1341.4	1394.5	1329.08	3.99%
c2670	1954.1	2104.7	1927.9	7.81%
Ind32	919.0	982.6	909.8	6.99%

Table 2: 3-parameter results (in psec), Margin=(Prior-Posterior)/True

In Figure 13-(b), we applied the 3-parameter model in the Monte Carlo simulation but used a 2-parameter model in learning and block-based SSTA. For comparison, we also include the "3-parameters, 1-sigma" curve which was the result of still using the 3-parameter model in learning and SSTA. We observe that, even if the actual correlation model is a 3-parameter model and a 2-parameter model is used in learning, we can still recover some margin. However, the amount of the recovered margin would be less than that by using a higher-resolution 3-parameter model. This result illustrates the trade-off between the correlation model complexity and the amount of recovered margin. In any case, using a lower-resolution model in learning ensures a conservative bound on the worst-case circuit timing.

6. CONCLUSION AND FUTURE RESEARCH

In this work, we demonstrate the importance of accurate modeling of spatial delay correlations in SSTA. We present a Bayesian learning framework for learning these correlations from silicon. Our framework utilizes path-based SSTA as the platform for learning. Learned correlations can then be used in block-based SSTA to recover timing margin. Our learning framework supports iterative learning where the learned result (posterior) from a set of sample chips can be used as the prior assumption (prior) for learning another set of chips. For future work, we plan to investigate in detail this iterative learning as well as the test related issues associated with the proposed methodology.

7. REFERENCES

[1] H. Chang and S. S. Sapatnekar. Statistical timing analysis considering spatial correlations using a single PERT-like traversal. In *Proc of ICCAD*, 2003.

[2] Vishal Khandelwal and Ankur Srivastava. A general framework for accurate statistical timing analysis considering correlations. In *DAC*, 2005.

[3] Assem Agarwal, David Blauuw, and Vladmir Zolotov. Statistical timing analysis for intra-die process variations with spatial correlations. In *ICCAD*, 2003.

[4] C. Amin, N. Menezes, K. Kilpatrick, F. Dartu, U. Choudhury, N. Hakim, and Y. Ismail. Statistical static timing analysis: How simple can we get. In *DAC*, 2005.

[5] M. Orshansky and A. Bandyopadhyay. Fast statistical timing analysis handling arbitrary delay correlations. In *Proc of DAC*, 2004.

[6] C. Visweswariah,et al. First-order incremental block-based statistical timing analysis. In *Proc. of DAC*, 2004.

[7] Nagib Hakim Sani R. Nassif, Duane Boning. The care and feeding of your statistical static timer. In *ICCAD*, 2005.

[8] M. Orshansky, L. Milor. C. Hu. Characterization of spatial intrafield gate cd variability, its impact on circuit performance, and spatial mask-level correction. In *IEEE Tran. on Semiconductor Manufacturing*, Vol 17, No 1, 2 - 11, 2004.

[9] P. Friedberg, et. al. Modeling Within-Die Spatial Correlation Effects for Process-Design Co-Optimization. *Proc. of ISQED*, 2005

[10] Duane S. Boning and Sani Nassif. Models of process variations in device and interconnect. In *Chap 6, Design of High Performance Microprocessor Circuits, edited by W. Bowhill A. Chandrakasan and F. Fox, IEEE Press*, 2000.

[11] David J.C. Mackay. Information Theory, Inference, and Learning Algorithms. Cambridge University Press, 2003.

[12] Benjamin N Lee, Li-C. Wang, and Magdy S. Abadir. Reducing pattern delay variations for screening frequency dependent defects. In *Proceedings of the 23rd IEEE VLSI Test Symposium*, pages 153–160, 2005. IEEE Computer Society.

[13] E. F. Schisterma, et al. Estimation of the correlation coefficient using the Bayesian Approach and its applications for epidemiologic research *BMC Medical Research Methodology*, Vol 3, 2003.

[14] X. Yang, B-K. Choi, and M. Sarrafzadeh. A Standard-Cell Placement Tool for Designs with High Row Utilization *ICCD*, Sep, 2002.

[15] J.-J. Liou, et.al Enhancing Test Efficiency for Delay Fault Testing Using Multiple-Clocked Schemes *DAC*, pp. 371-374, Jun. 10-14, 2002