Performance Sensitivity Analysis Using Statistical Methods and Its Applications to Delay Testing

Jing-Jia Liou Angela Krstić Kwang-Ting Cheng Deb Aditya Mukherjee Sandip Kundu

ECE Department,
University of California, Santa Barbara
{jjliou, angela, timcheng}@bigbend.ece.ucsb.edu

Intel Corporation,
U.S.A
{dmukher,skundu}@scdt.intel.com

Abstract— The performance of deep submicron designs can be affected by various parametric variations, manufacturing defects, noise or modeling errors that are all statistical in nature. We propose a statistical framework for analyzing the performance sensitivity of designs to various timing related defects/noise/variations. The core engine of our approach is a highly efficient statistical timing analysis tool. We describe the application of our framework for delay fault modeling and analysis of resistive opens and shorts and as well as interconnect crosstalk. We present experimental results demonstrating the accuracy of our statistical framework as compared to SPICE (for a given set of input patterns) and nominal worst-case analysis. Experimental results for analysis of resistive opens and shorts are also included.

I. Introduction

In deep submicron designs, the timing of the circuit can be considerably affected by process variations, various kinds of manufacturing defects as well as by noise effects (such as power supply noise and crosstalk). For example, interconnect resistive opens and shorts can result in a circuit failure due to their effects on the delays even if no logic faults have been produced. Also, simulations show that excessive noise most of the time leads to delay faults [3]. Many parametric variations, noise sources and modeling errors are statistical in nature. Therefore, to incorporate statistical deviations of various parameters for statistical worst-case analysis of circuits with or without defects, the use of statistically based methods seems to be inevitable.

In this paper, we propose a performance sensitivity analysis method based on a statistical framework. Our objective is to apply this method for analyzing the sensitivity of the circuit performance to manufacturing defects, process variations, noise effects. The core engine of our technique is statistical timing analysis. In statistical timing analysis, the propagation delays of interconnects/cells are modeled as correlated random variables with known distribution functions (extracted from layout and process parameters). Several statistical timing analysis methods have been proposed in the literature [12, 7, 6]. However, due to their high computational complexity, these methods are rarely used in practice. As a core engine for our performance sensitivity tool we have developed a highly efficient statistical timing analysis framework that can target larger designs. It can derive the probability density functions of the signal arrival times at internal signals and primary outputs. The 3σ (or 6σ) delay of the critical paths can then be computed. We perform two kinds of statistical timing analysis: *dynamic* and *static timing analysis*. Dynamic statistical analysis involves simulating a set of input patterns. With input vectors, we can evaluate the transition state and sensitization type of logic gates to determine the random variables. Static analysis is vectorless analysis which reports the results based only on the circuit function and structure. Static timing analysis provides and upper bound on the circuit delay (we do not incorporate false path analysis due to high computational complexity), while the dynamic analysis provides a lower bound. This is due to the fact that the arrival times of the signals in the dynamic analysis are derived with respect to a given set of input vectors. Therefore, dynamic analysis is is false-path aware. The main problem with the dynamic timing analysis is deriving the right set of patterns that could excite the longest paths in the circuit.

We apply our statistical timing analysis framework to analyzing the circuit performance sensitivity in the presence of delay faults caused by manufacturing defects and noise effects. Manufacturing defects and excessive noise can cause perturbations of cell/interconnect delays. For a given defect or source of noise, these perturbations can be described as random variables. To derive the performance sensitivity of the circuit to the given defect or noise, we run our statistical timing analysis framework twice: first time using the cell/interconnect delays without considering the perturbations caused by the defect or noise and the second time with the perturbed delays. Defects/noise with high sensitivity are then selected as target faults for testing. This performance sensitivity analysis process is illustrated in Figure 1.

In addition to the performance sensitivity analysis described in this paper and the system performance analysis, our statistical timing analysis framework has several other potential applications: (1) The analysis results can be used for yield estima-

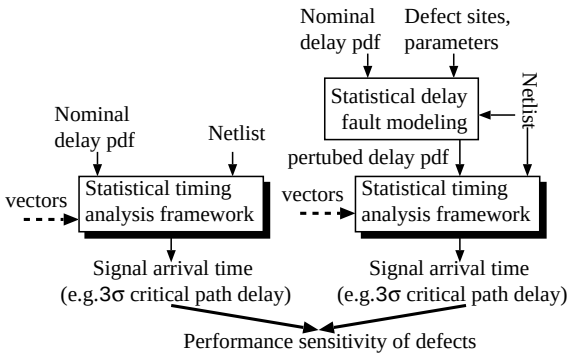


Fig. 1. Estimating the performance sensitivity.

tion. (2) The dynamic timing analysis framework can be used for selecting a subset of design verification or ATPG vectors for speed binning. (3) The dynamic timing analysis can also help in the estimation of the circuit glitching probability and thus power estimation.

II. Statistical Timing Analysis

As the core engine for our performance sensitivity analysis we use a statistical timing analysis framework. Even though statistical timing analysis is a well known concept, its implementation involves dealing with many different details. In this section, we describe some key features of our framework.

We model the propagation delays of cells/interconnects as correlated random variables with known probability density functions which can be extracted from layout and process parameters. To achieve balance between accuracy and efficiency, we develop a cell-based statistical timing analysis framework. It requires pre-characterization of cells, i.e., building libraries of cell delays and output transition times (as random variables). The input transition time and output loading of the cells are used as indices for building these libraries. Since the goal of statistical timing analysis is to describe the timing behavior of the circuit regardless of the applied input patterns, we adopt the worst-case analysis in building the cell libraries. The interconnect delay is also modeled as a random variable and is pre-characterized once the RCs are extracted. Next, the random variable of the signal arrival time at any cell/interconnect output can be computed using the information on the arrival and transition times of the cell fanins as well as the information on the cell/interconnect delay [5].

For large designs with various sources of correlations between the propagation delay random variables of different cells, determining closed forms for the probability density functions of the arrival times at the primary outputs is computationally expensive and impractical. Our framework uses Monte Carlo simulation based technique to approximate the probability density function of the signal arrival time at the internal signals and primary outputs. Each run of Monte Carlo simulation consists of two steps: sampling step and analysis step. In the sampling step a single value is chosen from each random variable according to the law of probability. The analysis step utilizes these sampled values to derive the arrival times at all signals for the given circuit instance. The stop (convergence) criteria are decided based on desired accuracy of results (such as the convergence rate of the mean or variance of the arrival times at the primary outputs). Once the mean or variance converges within the desired precision range, the procedure terminates. Figure 2 illustrates a levelized Monte Carlo timing simulation. The flowchart describes one run of Monte Carlo simulation in which one data entry of the arrival time for each signal delay is obtained. Steps 1 through 3 represent cell/interconnect delay estimation. Step 1 determines the effective load capacitance seen by the current cell output. Any of the previously published methods can be used to obtain this capacitance [10, 8]. Step 2 represents computation of the signal arrival time and transition time for all inputs to the given cell and interconnect delay analysis. The interconnect RC delay can be derived using any of the known methods (e.g., [11]). Given the output loading obtained in Step 1 and the input transition time obtained in Step 2, the proper pin-to-pin delay random variables of cells are obtained in Step 3 (using the pre-characterized cell library). The calculation of delays is repeated in each run of Monte Carlo

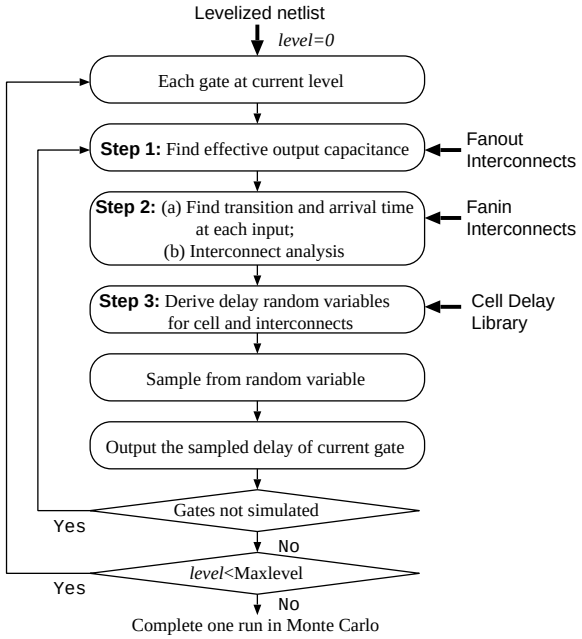


Fig. 2. Flow of one Monte Carlo run.

since cell/interconnect delays depend on transition times and transition times depend on the latest arriving signal which in turn depends on delays.

In general, to achieve the convergence of the results, the number of Monte Carlo runs ranges from tens to hundreds. This means that the timing analysis described in Step 1 through 3 should be repeated that many times. In practice this is not feasible no matter how efficient the timing analysis is. Therefore, we propose an approximate algorithm. In our simulation framework, we move Steps 1 through 3 outside the Monte Carlo simulation loop. Therefore, in these steps, we perform only the nominal delay analysis to obtain the random variables of interconnect and cell delays. We have carefully validated this approximation and found that there are negligible errors in the final arrival times.

A. Handling cross-correlations

In Monte Carlo sampling, we need to deal with the fact that the cell/interconnect delay random variables are strongly correlated. It is common that process variations are shifted in one direction in a specific part of the chip. Devices residing in this part of the chip will almost all be speed up or slowed down. This correlation has been ignored in all previously proposed statistical simulation methods.

Application of the covariance matrix (V) is a typical way of dealing with multivariate normal random variables. The sampling procedure follows equation $X = \mu + CZ$ [2], where $X = (X_1, X_2, \ldots, X_n)$ is a vector of dependent normal variables, $\mu = (\mu_1, \mu_2, \ldots, \mu_n)$ is the mean vector of X, C is a lower triangular matrix such that covariance matrix V can be expressed as $V = CC^T$ (Cholesky factorization) and Z is the vector of n independent standard normal components. While this is a versatile method to represent arbitrary correlation between pairs of random variables, in practice it is not feasible. This is because: (1) The size of V can be too large to process. For example, for a system with n = 1000, the size of V is already in the order of million, and the Cholesky factorization process with the complexity of N^3 is a tremendous task, too. Note that the cal-

culation of C is needed every time we have a different set of random variables. We cannot afford to do this in our statistical timing analysis framework. (2) A prohibitively large number of samples is needed to show that these samples are produced from a particular V, since the dimension of the sampling space is equal to the number of variables. For example, if we want to obtain the same resolution as in the case of a single variable with 10 samples, in the $n = 1000$ system we need 10^{1000} samples to obtain a similar level of accuracy.

Our solution to the cross-correlation problem is reversing the process of Principal Component Analysis (PCA) [9]. The PCA procedure is used to extract the independent random variable components from the dependent random variables. Thus, the dependent variables can be expressed as the composition of the extracted independent components. Our reverse method consists of adding an independent random variable into the group of variables which share the same correlation factor r (set by the user). Let $X = A + t_1 C$ and $Y = B + t_2 C$, where A, B, C (C is standard normal distribution with $\overline{C} = 0$, $\overline{C^2} = 1$) are independent random variables and t_1, t_2 are constant coefficients. We will show that the correlation factor of X and Y is determined by t_1 and t_2. Therefore, choosing appropriate values for these two parameters can lead to a predefined correlation.

The correlation r_{XY} can be obtained as:

$$\begin{aligned} r_{XY} &= \frac{\overline{(X-\overline{X})(Y-\overline{Y})}}{\sqrt{\overline{(X-\overline{X})^2}}\sqrt{\overline{(Y-\overline{Y})^2}}} = \frac{\overline{(A+t_1C)(B+t_2C)} - \overline{AB}}{\sqrt{\overline{A^2}+{t_1}^2}\sqrt{\overline{B^2}+{t_2}^2}} \\ &= \frac{t_1 t_2}{\sqrt{\overline{A^2}+{t_1}^2}\sqrt{\overline{B^2}+{t_2}^2}} = r_{XC} r_{YC} \end{aligned}$$

where $r_{XC} = \frac{t_1}{\sqrt{\overline{A^2}+{t_1}^2}}$ and $r_{YC} = \frac{t_2}{\sqrt{\overline{B^2}+{t_2}^2}}$. To find t_1 and t_2 when $r_{XY} = r$, we need another constraint. We assume $r_{XC} = r_{YC} = \sqrt{r}$. Therefore, $t_1 = \sqrt{r}\sqrt{\overline{X^2}}$ and $t_2 = \sqrt{r}\sqrt{\overline{Y^2}}$. After we derive the random variables in Step 3, X and Y are known and we can calculate t_1, t_2 using the above described procedure. Next, we can determine A, B from t_1 and t_2. In the sampling process, we can sample values from A, B and C and using $X = A + t_1 C$ and $Y = B + t_2 C$, we can get $r_{XY} = r$. The above process can be easily extended to multivariate random variables. The advantage of our procedure is that it avoids the problem of Cholesky decomposition and the large size of covariance matrix, while supporting a large number of elements with the same correlation factor.

B. Interconnection Modeling

To demonstrate the modeling of interconnect delays in a statistical form, we limit our discussion to the first order RC interconnect model shown in Figure 3. Even though our framework can accept any other more sophisticated interconnect model we use this model for simplicity reasons.

We assume an exponential rising input voltage $V_i(t) = 1 - e^{-\omega t}$(output of the driving gate) whose Laplace domain equivalent is $V_i(s) = \frac{1}{s} - \frac{1}{s+\omega}$. The output voltage at the capacitor node can be obtained as $V_o(s) = (\frac{1}{s} - \frac{1}{s+\omega})(\frac{p}{s+p})$, where $p = 1/RC$ (p is a random variable). Therefore, we have

$$V_o(t) = \begin{cases} 1 - \frac{1}{\omega - p}(\omega e^{-pt} - p e^{-\omega t}) & \omega \neq p \\ 1 - (1+pt)e^{-pt} & \omega = p \end{cases}$$

The random variable corresponding to the interconnect delay can be obtained as follows. Consider function $d = f(x_1, x_2, \ldots, x_n)$, where x_i ($i = 1, 2, \ldots, n$) is a random variable with mean $\overline{x_i}$ and standard variation σ_i and x_i's are all independent. Then, we have $\overline{d} = f(\overline{x_1}, \overline{x_2}, \ldots, \overline{x_n})$ while the standard variation (σ_d) can be found from ${\sigma_d}^2 = \sum_i (\frac{\partial f}{\partial x_i})^2 {\sigma_i}^2$. Our goal is to find time t such that $V_o(t) = 0.5$ (50% of the output voltage). We denote this time $t_{0.5}$. Time $t_{0.5}$ is a function of p and ω. If we assume $\omega = p$, we get $1 - pe^{-pt_{0.5}} - t_{0.5}e^{-pt_{0.5}} = 0.5$. Substituting the mean value of p will yield the mean of $t_{0.5}$ as suggested by the $\overline{d} = f(\overline{x_1}, \overline{x_2}, \ldots, \overline{x_n})$ equation. The standard deviation of $t_{0.5}$ can be obtained by using the formula for ${\sigma_d}^2$. The partial derivative $\frac{\partial V_o}{\partial p}$ for $\omega = p$ and $V_o = 0.5$ yields $(t^2 + pt - 1)e^{-pt}\frac{\partial t}{\partial p} = 0$. By plugging in the mean of p and t in this equation, we can obtain $\frac{\partial t}{\partial p}$ and hence σ of $t_{0.5}$ from $(\frac{\partial t}{\partial p})^2 \sigma_p^2$.

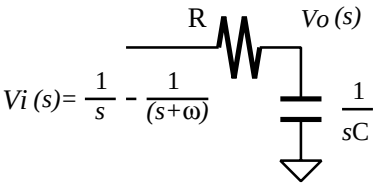


Fig. 3. First order RC interconnect model (All R's and C's are random variables with known distributions.)

TABLE I
COMPARISON WITH SPICE FOR THE PATTERN PRODUCING THE MAXIMAL MEAN(MEAN IN PS, AND σ IN %.)

	Monte Carlo SPICE	Dynamic(r=0)	Dynamic(r=1)
Ckt	Mean/σ	Mean/σ	Mean/σ
s208	103.8/3.4	118.7/2.3	118.9/3.5
s344	140.2/2.2	154.0/0.8	154.1/3.3
s349	139.4/2.2	161.3/0.8	161.6/3.1
s420	189.8/1.6	220.5/0.7	220.1/2.9
s838	364.2/0.8	423.1/0.5	423.2/2.7

C. Experimental results

We have implemented our statistical timing analysis framework in C++ and Python scripts. In this section, we present experimental results obtained for fully scanned ISCAS89 benchmark circuits. We use ELDO/SPICE [1] to extract the cell statistical delay information for a 0.25μm CMOS technology. The results for the circuit performance obtained by simulating these cell delay data are compared to the results obtained by simulating the whole circuits by ELDO/SPICE. To be able to compare the results of our framework to the results obtained by ELDO/SPICE, we also use Python and Esim [1] scripting languages to program ELDO/SPICE to extract the required arrival times for comparison.

We perform experiments for both static and dynamic statistical timing analysis framework. For dynamic analysis, we use

TABLE II
COMPARISON WITH SPICE FOR A SET OF PATTERNS(MEAN IN PS, σ IN %, AND TIME IN S.)

	Monte Carlo SPICE		Dynamic (r=1)		Error Percentage	
Ckt	mean/σ	time	mean/σ	time	mean	σ
s208	106.9/2.1	524	124.1/2.0	0.73	16.1	-4.7
s344	144.0/2.0	933	159.2/2.1	1.03	10.6	5.0
s349	143.0/2.0	1161	162.8/2.3	1.10	13.8	15
s420	191.6/2.4	878	222.5/2.3	1.08	16.1	-4.2
s838	369.5/2.2	2849	429.8/2.3	1.79	16.3	4.5

TABLE III
STATISTICAL WORST-CASE VS. TRADITIONAL WORST-CASE(MEAN IN PS, σ IN %, AND TIME IN S.)

	Statistical Static			Worst-case	
Ckt	mean	3σ delay	average time	$C=0$	$C=3$
s5378	238.4	244.8	6.62	238.2	268.3
s9234	420.0	425.6	14.08	420.1	460.5
s13207	480.8	488.1	26.91	481.1	529.4
s15850	656.4	664.8	38.8	656.5	722.5
s35932	204.5	206.6	26.42	204.6	223.6
s38417	362.1	368.4	69.71	361.8	396.9
s38584	566.9	579.8	124.42	566.8	620.0

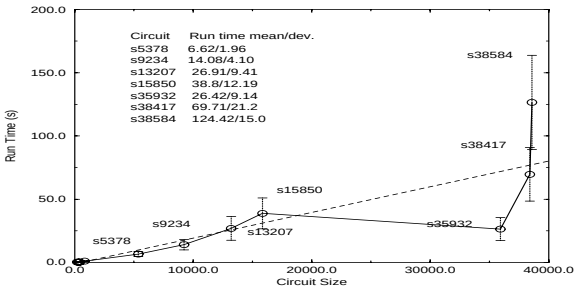


Fig. 4. Runtime growth.

the path delay fault tests generated for 10 long critical paths for each tested circuit [4].

As a stopping criterion, in the Monte Carlo simulation, we set a threshold for the change of the mean/standard deviation for the circuit's longest delay. One run of Monte Carlo simulation produces one data entry of arrival times at each primary output. Next, we find the maximum among these data, and record it as one instance of the circuit performance. After the simulation converges, we can calculate the sample mean and variance. In the case of static timing analysis, these results represent probability distribution of the circuit performance. In the case of dynamic timing analysis, there is one probability distribution for each test pattern. Table I shows the comparison of the results for sample mean and standard deviation (σ) for the dynamic framework. The results are given for the input pattern that produces the maximal mean among all applied patterns. We show results for two dynamic simulation runs along with SPICE: one assumes independent random variables ($r=0$), while the other assumes a correlation factor of one ($r=1$). As it can be seen the mean values are close, while the σ values are quite different. This is because the circuit with the correlation in the samples tends to present more extreme cases than if the samples were independent. The results also show that our dynamic simulator produces delays that are very close to the SPICE results for this single vector pattern.

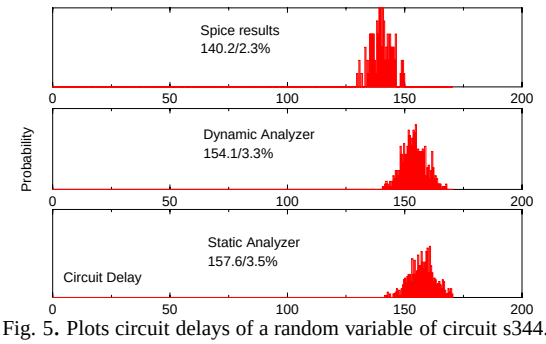


Fig. 5. Plots circuit delays of a random variable of circuit s344.

To estimate the performance of the circuit with respect to the applied set of input vectors, we find the statistical maximal distribution. Since the probability distribution functions for each of the test patterns are independent of each other, the maximal distribution can be found by sequentially taking the maximum between two random distributions. For example, assume that $r_1, r_2, \cdots, r_n$ are the distributions for n input patterns. First, we find $r_t = max(r_1, r_2)$, and then continue to find $r_t = max(r_t, r_3)$, etc. This way, the circuit performance with respect to the given patterns is represented by one worst-case random variable. Table II shows the statistical maximal results obtained as explained above. As it can be seen the error percentage is small. Our simulator can produce reasonable results within only a small fraction of run time consumed by SPICE.

The difference between statistical worst-case analysis and traditional worst-case analysis is illustrated in Table III. Column marked *Worst-case* shows the results of static timing analysis in which the cell delays are fixed to their mean value plus a multiple (C) of σ. Even though the sample mean is the same for both experiments, the traditional worst-case simulation is clearly too pessimistic (from comparing columns 3 and 6). To demonstrate the efficiency of our approach for larger circuits, in Figure 4 we plot the runtime vs. the circuit size. As it can be seen, the run time grows roughly linearly with the circuit size. Figure 5 illustrates the histogram of the sampled distributions of SPICE/ELDO, dynamic and static timing analysis for circuit s344.

III. PERFORMANCE SENSITIVITY ANALYSIS

In this section, we describe application of our statistical timing analysis framework for analyzing the circuit performance sensitivity in the presence of delay faults caused by manufacturing defects and noise effects. We concentrate on delay faults caused by interconnect resistive open and short faults and on crosstalk induced delay faults.

A. Resistive interconnect defects

Interconnect resistive opens and shorts can cause logic or delay faults in the circuit. For example, a resistive short between two interconnects results in an increased load for the driving cell and a lower driving voltage for the driven cell. The fault effect of a resistive short depends on the resistance value. For certain resistance values (R ≈ 0) this defect will cause logic faults while for some other values it will result in delay faults.

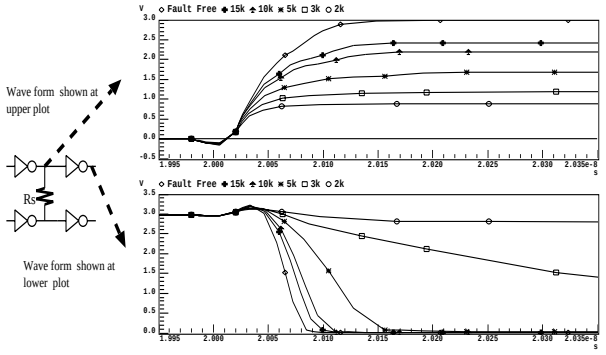
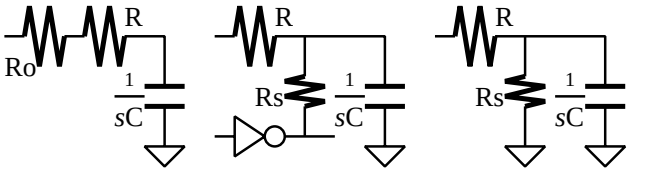


Fig. 6. Plots of signal waveforms for faulty circuits with various short resistances.



(a)resistive open Ro (b)resistive short Rs (c)resistive short Rs

Fig. 7. Resistive defects interconnect model (All R's and C's are random variables with known distributions.)

The fault effects of one resistive short circuit are demonstrated in Figure 6. The lower the resistance, the larger the fault effect is. At 2kΩ, the fault behaves as a stuck-at fault. In this paper, we assume that the defects causing logic faults will be covered by stuck-at tests and we model only the defects with a range of resistance causing delay faults.

Very often, for resistive shorts and opens, we do not know the exact resistance value and in the analysis, we would like to model the defects for a wide range of resistance values. In our framework, we treat the resistance value as a random variable which can have any arbitrary but given distribution. To demonstrate modeling delays caused by interconnect resistive open and short defects in a statistical form, we use the first order RC interconnect models shown in Figures 7(a) and (b), respectively. These models are extended from the model in Figure 3. In general, the delay calculation steps are the same as the ones described in section II-B. The effect of the resistive open on the delay is easy to obtain by substituting $p = 1/(R+R_o)C$ in the expression for $V_o(t)$ where R_o is the resistance of the open fault. For the resistive short defects, the transfer function becomes $\frac{R_s}{R_s+R+sRR_sC}$ where R_s is the resistance of the short fault, so we have a slightly different form for $V_o(t)$. The resistive short defect is assumed to be bridging two interconnects together as in Figure 7(b). However, using an effective resistance to ground as shown in Figure 7(c) would produce the worst-case delay effect. This can be achieved by requiring that the test generator produces a stable 0 (1) value at the output of the inverter in Figure 3 for a rising (falling) transition at the fault site. In addition, the short fault needs to be also tested by applying test patterns that excite it through the other interconnect bridged by the fault. Note that the case shown in Figure 7(c) can also happen in practice. The main problem in modeling resistive short faults is that $V_o(t)$ will never reach V_{dd} (only the case of rising transition at the fault site is described). Even though this is similar to lower voltage supply in low power design, the situation here is different: The voltage supply is maintained, but the input voltages of gates at the fanout of the resistive short are lowered. The effect is that the rising transition at this defect site is slower. To find the delays under the reduced voltage, we translate the lower voltage level to the corresponding output rise time with the same delay. This can be done with the assistance of predefined table from SPICE simulations. This way, we can use the original cell delay library with input transition time as the table parameter to propagate the fault effect forward without the extra characterization effort for constructing a new library for lower input voltage.

B. Crosstalk

It has been shown that crosstalk between interconnects can significantly impact the performance of deep submicron designs. Figure 8 shows a simple first order interconnect model of two cross-coupled lines. In the case of two transitions that simultaneously or within a small time interval switch in the opposite directions, the interconnect delay of both lines increases while if the transitions are in the same direction, the delay of both lines decreases. The increase and decrease are measured with respect to the case when the aggressor line has a transition and the victim line is quiet.

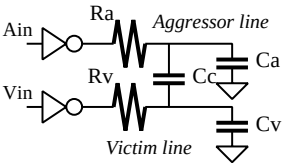


Fig. 8. First order RC model for cross-coupled interconnects.

To statistically analyze the circuit performance in the presence of crosstalk, the crosstalk induced delay increase/decrease needs to be modeled as a random variable. Since we are interested in the worst-case statistical analysis, we only consider the case when coupled lines have opposite transitions. We assume that the set of lines with a significant cross-coupling capacitance between them is given. This can be obtained using RC extraction tools given the circuit layout.

Also, we assume that any number of cross-coupled lines can occur in the circuit. Therefore, since the coupling on the two lines can change the timing information of all gates/interconnects in their fanout cones, we process the gates in the topological order from PIs to POs. For each interconnect pair, we need to first check if opposite transitions can occur temporally close to each other. To perform this check, in addition to the worst-case statistical analysis, the best-case statistical analysis also needs to be completed. The framework for the best-case statistical analysis is analogous to the worst-case analysis. After obtaining the random variables of the earliest and latest arrival time for each interconnect, we can determine if opposite transitions can simultaneously or near simultaneously occur for a given interconnect pair. For those interconnect pairs for which the intervals between earliest and latest arrival time have a non-empty intersection, we need to find the perturbation of the random delay variable caused by cross-coupling capacitance. The value of this delay perturbation depends on the time skew z between the two switching signals. The interconnect delay in the presence of crosstalk can be obtained by replacing the coupling capacitance C_c with a capacitance to the ground multiplied by a factor k, i.e., the total capacitance for the aggressor line can be written as $C_a + kC_c = C_a + C'$. The value of C' depends on the time skew between the aggressor and victim line, i.e., $C' = f(z)$. In order to be able to use our cell delay libraries to perform analysis of the crosstalk effects on the performance, for each pair of cross-coupled interconnects we can derive the curve $C' = f(z)$ and use it to access the cell/interconnect delay libraries during Monte Carlo simulation.

C. Experimental Results for Resistive Defects

To demonstrate the application of our statistical timing analysis framework for deriving the circuit performance sensitivity with respect to manufacturing defects/noise, we present preliminary results for resistive open and short faults. Table IV shows the results for a chosen resistive open fault. The analysis is done by inserting a resistor whose resistance value is a random variable with mean of 10kΩ and a standard deviation of 5% of mean (10kΩ/5%) at the fault site and by simulating a pattern that activates the defect. We compare the results for

TABLE IV
RESISTIVE OPEN ANALYSIS AND COMPARISON WITH SPICE(MEAN IN PS, σ IN %, AND TIME IN S.)

	With Defect			Fault Free		
	SPICE	Dynamic	Static	SPICE	Dynamic	Static
Ckt	Mean/σ	Mean/σ	Mean/σ	Mean/σ	Mean/σ	Mean/σ
s208	155.3/3.3	163.4/3.7	164.5/3.8	102.7/3.6	117.3/3.3	118.9/3.5
s344	202.1/2.9	213.6/3.6	214.9/3.8	140.2/3.1	153.6/3.2	157.2/3.1
s349	200.7/2.8	214.3/3.8	214.6/3.8	139.4/3.1	153.6/3.2	157.6/3.1
s420	216.1/2.7	238.6/3.8	239.7/3.7	159.4/3.0	183.7/3.1	221.6/3.2
s838	390.1/2.8	441.1/3.3	443.4/3.4	333.8/3.0	386.7/3.1	424.4/3.1

TABLE V
RESISTIVE SHORT ANALYSIS AND COMPARISON WITH SPICE(MEAN IN PS, σ IN %, AND TIME IN S.)

	With Defect			Fault Free		
	SPICE	Dynamic	Static	SPICE	Dynamic	Static
Ckt	Mean/σ	Mean/σ	Mean/σ	Mean/σ	Mean/σ	Mean/σ
s208	117.6/4.3	136.4/4.7	138.5/4.7	102.7/3.6	117.3/3.3	118.9/3.5
s344	141.2/3.2	163.0/3.1	164.5/3.1	140.2/3.1	154.0/3.1	157.2/3.1
s349	140.3/3.1	163.2/3.0	164.1/3.0	139.4/3.1	153.7/3.3	157.6/3.1
s420	163.8/3.2	188.8/3.2	221.1/3.1	159.4/3.0	183.7/3.1	221.2/3.0
s838	338.0/3.1	392.4/3.2	425.2/3.1	333.8/3.0	386.1/3.0	424.7/3.0

SPICE, static and dynamic timing analysis. This particular defect shows up in both the dynamic and static simulation.

Table V shows results for simulating resistive short faults. The value of the inserted resistance is 50kΩ/5% random variable for the first three circuits, and 10kΩ/5% random variable for the last two circuits. We choose different resistance values according to the fault site resistance in order to excite the delay fault effect in each of the circuits. The resistance outside the range of random variables show either a stuck-at fault effect or no fault at all. Though SPICE/ELDO simulations for 50kΩ/5% resistive short show a clear delay fault effect for s208, the fault is harder to detect for s344 and s349. On the other hand, our dynamic and static simulators clearly show that there is a delay fault on these two circuits. For s420 a static simulation does not reveal the defect. Our analysis shows that this is not a missed fault because the delays of the paths involving this fault are smaller than the delay of some other fault free critical paths. These critical paths mask the fault effects. If we set the clock speed according to the results of static analysis, the results of the dynamic simulation are still within this bound. In Figures 9 and 10, we plot the increase of critical path delay with a uniformly distributed resistance of $[25k, 75k]$ and $[100k, 300k]$ for resistive open defect, respectively, inserted at all nodes for circuit s5378. We show the scaled plot at the right of both figures. From these figures it is clear that not all fault sites are equally sensitive to defects. Performing this analysis for each node in the circuit may cause efficiency problems for large circuits. The efficiency issue can be alleviated by performing a defect fault collapsing prior to the analysis.

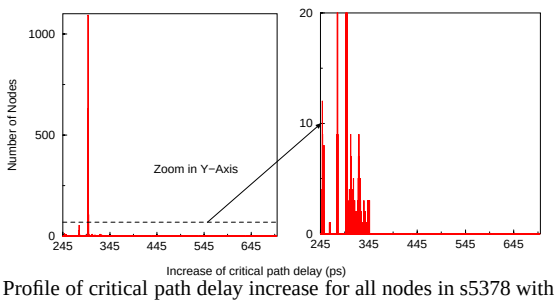


Fig. 9. Profile of critical path delay increase for all nodes in s5378 with defect in $[25k, 75k]$.

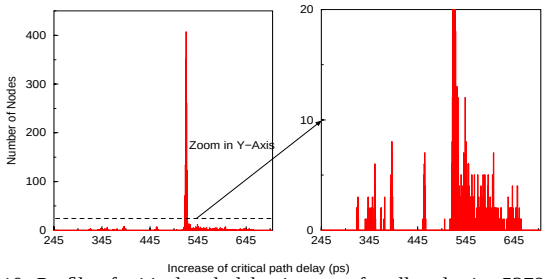


Fig. 10. Profile of critical path delay increase for all nodes in s5378 with defect in $[100k, 300k]$.

IV. CONCLUSIONS

We propose a performance sensitivity analysis method based on statistical framework. Our method can be used for analyzing the circuit performance in the presence of delay faults caused by manufacturing defects, excessive noise, process variations, modeling errors. We model the propagation delays of cells/interconnects as correlated random variables with known probability density functions and use a statistical timing analysis to derive the worst-case arrival times at the primary outputs and internal signals with and without defects. We apply our technique for analysis of interconnect resistive opens and shorts. Our experimental results demonstrate the efficiency and usefulness of statistical vs. nominal worst-case performance analysis.

REFERENCES

[1] Anacad. *Eldo v4.4.x User's Manual*, 1996.

[2] P. Bratley, B. L. Fox, and L. Schrage. *A Guide to Simulation, Second Edition*. Springer-Verlag, New York, 1987.

[3] M. A. Breuer, C. Gleason, and S. Gupta. New Validation and Test Problems for High Performance Deep Sub-Micron VLSI Circuits. *VTS, Tutorial Notes*, Apr. 1997.

[4] K.-T. Cheng, A. Krstic, and H.-C. Chen. Generation of High Quality Tests for Robustly Untestable Path Delay Faults. *IEEE Trans. on Computers*, 45(3):1379–1392, Dec. 1996.

[5] H. Edamatsu, K. Homma, M. Kakimoto, Y. Koike, and K. Tabuchi. Pre-Layout Delay Calculation Specification for CMOS ASIC Libraries. *Proc. of ASP-DAC*, pp. 241–248, Feb. 1998.

[6] H.-F. Jyu and S. Malik. Statistical Delay Modeling in Logic Design and Synthesis. *Proc. of 31th DAC*, pp. 126–130, June 1994.

[7] H.-F. Jyu, S. Malik, S. Devadas, and K. Keutzer. Statistical Timing Analysis of Combinational Logic Circuits. *IEEE Trans. on VLSI Systems*, 1(2):126–137, June 1993.

[8] A. B. Kahng and S. Muddu. New Efficient Algorithms for Computing Effective Capacitance. *Proc. of ISPD-98*, pp. 147–151, Apr. 1998.

[9] J. Power, B. Donnellan, A. Mathewson, and W. Lane. Relating Statistical MOSFET Model Parameter Variabilities to IC Manufacturing Process Fluctuations Enabling Realistic Worst Case Design. *IEEE Trans. on Semiconductor Manufacturing*, 7(3):306–318, Aug. 1994.

[10] J. Qian, S. Pullela, and L. Pillage. Modeling The "Effective Capacitance" for The RC Interconnect of CMOS Gates. *IEEE Trans. on CAD*, 13(12):1526–1535, Dec. 1994.

[11] C. L. Ratzlaff and L. T. Pillage. RICE: Rapid Interconnect Circuit Evaluation Using AWE. *IEEE Trans. on CAD*, 13(6):763–776, June 1994.

[12] D. R. Tryon, F. M. Armstrong, and M. R. Reiter. Statistical Failure Analysis of System Timing. *IBM Journal of Research and Development*, 28(4):340–355, July 1984.