Static Timing: Back to Our Roots

Ruiming Chen*, Lizheng Zhang*, Vladimir Zolotov, Chandu Visweswariah, Jinjun Xiong
IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA

Abstract— Existing static timing methodologies apply various techniques to address increasingly larger process variations. The techniques include multi-corner timing, on-chip variation (OCV) derating coefficients, and path-based common path pessimism removal (CPPR) procedures. These techniques, however, destroy the benefits of linear run-time and incrementality possessed by classical static timing. The major contribution of this work is an efficient statistical timing methodology with comprehensive modeling of process variations, while at the same time retaining those key benefits. Our methodology is compatible with existing characterization methods and scales well to large chip designs. To achieve this goal, three techniques are developed: (1) building the statistical delay model based on existing multi-corner library characterization; (2) modeling spatial correlation in a scalable manner; and (3) avoiding the time-consuming CPPR procedure by removing common path pessimism in the clock network by an incremental block-based technique. Experimental results on industrial 90 nm ASIC designs show that the proposed timing methodology correctly handles all types of process variation, achieves high correlation with traditional multi-corner timing with more than 4× speedup, and is a vehicle for pessimism reduction.

I. Introduction

Block-based static timing analysis (STA) is popular due to benefits like linear run-time complexity, identification of critical path, and incrementality [1]. These features permit STA to be used in a timing-driven physical synthesis flow to guide optimization. Moreover, both the algorithm and delay models are conservative, thus guaranteeing that STA predicts a bound on actual circuit performance. Hence, STA is also ideal for timing sign-off.

A practical STA implementation in industry is, however, far from ideal. The complications are: (1) gate and interconnect delays are not constant but vary because of process variations, aging and operating conditions; and (2) the data (combinational) network and clock distribution network need to be analyzed simultaneously to ensure correct timing of flip-flop launch and capture. Because a major portion of delay variation comes from chip-to-chip variations, existing industrial STA employs a *multi-corner timing methodology*, in which a separate analysis is conducted at each corner. For N process parameters, a *full-blown* multi-corner timing requires a total of 2^N process corners, which is computationally infeasible. In practice, the number of process corners checked is far smaller than 2^N. In order to guard-band the rest of the timing uncertainty such as within-chip spatial variation, random variation, and varying switching scenarios, existing approaches resort to a delay modeling technique called an *early/late split*, which models the delay of every arc of the timing graph as a minimum delay for early timing purposes, and a maximum delay for late purposes. Choosing the early/late split values properly plays a central role in the tradeoff between aggressive timing and safe design practices. For example, within-chip spatial variation is traditionally accommodated by using an *on-chip variation (OCV) derating coefficient* that derives the two bounded delay values from the library model. Both data and clock networks are timed with their respective early and late arrival times. At the capturing flops, the latest data arrival time is checked against the earliest clock arrival time for a setup test, and vice-versa for a hold test.

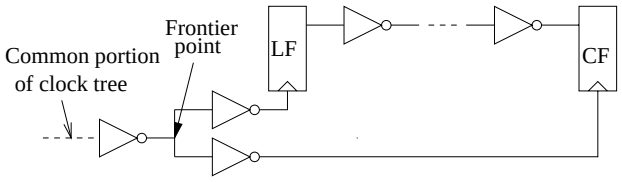


Fig. 1. Illustration of common portion between launching and capturing paths.

Because of these practical considerations, timing can be both risky (due to skipping some of the 2^N corners) and too pessimistic (due to an overly conservative early/late split) at the same time. To reduce pessimism, a procedure called *Common Path Pessimism Reduction* (CPPR) [2] is widely used. Fig. 1 illustrates the need for CPPR. In the Figure, "LF" denotes "Launching Flop" and "CF" denotes "Capturing Flop." The launching and capturing paths contain a *physically common portion* in the clock distribution network upstream of the so-called frontier point, and this portion cannot be both early and late simultaneously. The CPPR algorithm reduces this inherent pessimism in the initial timing. However, because of the path-based nature of CPPR, features such as linear complexity and incrementality are lost.

Recently, statistical static timing analysis (SSTA) has been proposed as an alternative way of handling process variations [3, 4]. However, existing literature assumes that parameterized variational delay models are available *a priori*. Unfortunately, this is not the case in reality, as all existing timing characterization is geared towards multi-corner timing. Obtaining a variational timing model based on existing timing characterization is important for a smooth transition from multi-corner timing to statistical timing. A parameterized SSTA algorithm can naturally handle chip-to-chip and independently random variation, but within-chip spatial variation is more troublesome. A grid-based approach in conjunction with Principle Component Analysis (PCA) has been proposed as a way of transforming spatially correlated parameters into a set of uncorrelated random variables

*This work was done while R. Chen and L. Zhang were summer interns at IBM Research; they are now with Cadence and Synopsys, respectively.

978-1-4244-1922-7/08/$25.00 ©2008 IEEE

[4]. However, we will show later that the PCA-based approach is too cumbersome to be used in practice; a more efficient and scalable technique is required. Another drawback of existing SSTA techniques is that they were mainly developed to analyze the data network without considering the clock network. How to address the counterpart of CPPR in SSTA is still not clear.

The purpose of this work is to develop a practical statistical timing methodology that restores all the good features of classical static timing, hence the title of this paper. Towards this goal, three major contributions are made in this paper: (1) the derivation of a variational delay model based on existing timing characterization methods; (2) a novel spatial correlation model that can accurately capture different spatial correlation behaviors with almost negligible overhead; and (3) an effective technique to remove the common path pessimism resulting from the early/late split in the clock network, thus obviating the need for CPPR. The benefits of our proposed timing methodology include linear run-time complexity, reduced timing pessimism with full process space coverage, and fully incremental capability. Experimental results based on industrial 90 nm designs show that the proposed timing methodology correctly handles all types of (chip-to-chip, within-chip, and random) process variations, and achieves high correlation with traditional multi-corner timing. Compared with the state-of-the-art timing sign-off technique, our method achieves more than 4× speedup with full process-space coverage.

The rest of the paper is organized as follows. In Section II, we show how the variational delay model is derived; in Section III, we present our novel spatial correlation modeling technique; we then propose our variational timing methodology in Section IV. Experimental results are presented in Section V with conclusions in Section VI.

II. Variational delay modeling

We use the variational model proposed in [5] for multi-corner timing. Fig. 2 shows the variation of delay due to one parameter. The delay model consists of three components. Intrinsic or global variation captures chip-to-chip variation and defines two corners for the parameter of interest; systematic variation captures "predictable" within-chip variation, including spatial correlation; and independently random variation models uncorrelated variation. The characterization procedure involves creating qualification chips with special test macros repeated at many locations on the chip. Test macros and processing of measurement data are outside the scope of this paper.

For multi-corner timing, a corner is the assignment of each parameter to one of its two extreme values in its intrinsic component. For example, as shown in Fig. 2, the early/late delays at corner 1 are given by

$$Delay^{early} = I_1 - S_1 - R_1, \quad (1)$$

$$Delay^{late} = I_1 + S_1 + R_1. \quad (2)$$

Note that the systematic and independently random parts are defined on a per-corner basis, so no matter how many sources of variation are considered, these effects are lumped into single terms. It is clear that there is inherently an early/late split in the model, which accounts for spatial variation, independently random variation, multiple input switching, initial charge on internal nodes of gates, etc.

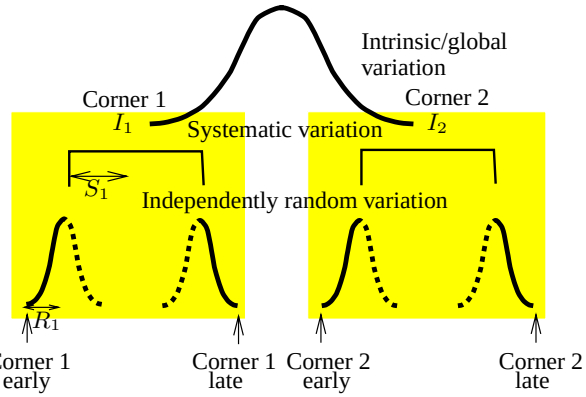


Fig. 2. Variational delay modeling.

For statistical timing, we extend the first-order canonical statistical delay model [3, 6] for all timing quantities to include spatial correlation as

$$T = a_0 + \sum_i a_i \Delta X_i + a_S \Delta X_S + a_R \Delta X_R, \quad (3)$$

where a_0 is the mean of the timing quantity (such as delay and slew or rise/fall time), a_i are its sensitivities to global sources of variation ΔX_i, a_S is its sensitivity to ΔX_S that represents spatial correlation, and the last term $a_R \Delta X_R$ represents independent randomness. All ΔX represent unit Gaussian distributions. Note that for corner-based timing, the systematic and independently random terms contribute to the early/late split; whereas in this model, they do not contribute to the early/late split because of the explicit modeling of the last two terms in (3).

For the delay and slew of each arc of the timing graph, we derive the coefficients of (3) from the same variational timing models as shown in Fig. 2 by means of finite-differencing, i.e.,

$$a_i = \frac{I_{2,i} - I_{1,i}}{6},\ a_s = \frac{S_1}{3},\ a_r = \frac{R_1}{3}.$$

where, we have assumed the range of variation is ±3 sigma, and sensitivities are normalized to per-sigma values.

III. Spatial correlation

A. Preliminaries

Spatial correlation describes the phenomenon that otherwise identical instances close to each other are more likely to have similar characteristics than instances far apart. It has been shown in [7] that *isotropic* spatial correlation can be described by a spatial correlation function with respect to distance as shown in Fig. 3. The correlation function drops below 1.0 at an infinitesimal distance due to the independently random variation. Then the correlation coefficient gradually reduces as a function of distance to a *correlation distance* beyond which correlation does not change. Beyond this regime, the correlation coefficient represents distance-independent global or intrinsic variation. Given the location of n instances, the covariance matrix of their delays can be obtained from the spatial correlation function. Since correlation distance is much smaller than the size of the chip, such matrices are sparse.

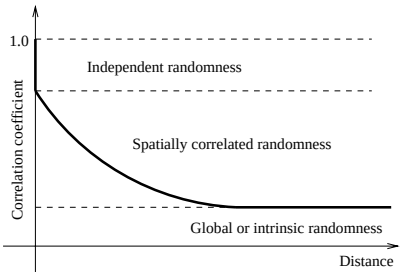


Fig. 3. Spatial correlation function.

In the context of SSTA, two approaches have been proposed in the literature to handle spatial correlation. The first is by using the well-known Principal Component Analysis (PCA) technique to transform spatially-correlated random variables into a new set of independent variables [4]. However, the PCA-based approach suffers from two drawbacks. (1) The computational cost of PCA by Eigenvalue Decomposition (EVD) is high. For example, consider a chip of size $10 \times 10\ mm^2$ with a correlation distance of 300 μm. Three grid cells are barely enough, with reasonable accuracy, to capture the spatial correlation decaying behavior between 0 and 300 μm. So the maximum allowable grid size would be 100 μm, which leads to a 100×100 grid. The covariance matrix that must undergo EVD is thus 10,000×10,000, which is clearly expensive. (2) Even though the covariance matrix is sparse, each physical random variable after PCA will end up depending on almost every new random variable, since the system has 10,000 significant degrees of freedom. Thus the transformed timing quantities will contain about 10,000 terms, which greatly slows down the atomic SSTA operations with increased amount of computation and memory consumption.

The second technique in the literature is to handle spatial correlation directly during SSTA operations [6], i.e., perform statistical operations with basis random variables that are correlated. For the same example as above, to perform the max/min operation on two arguments whose sensitivity vectors are A and B, one first needs to find their correlation coefficient. This requires computation of an inner product of $A^T V B$ with V being the 10,000×10,000 covariance matrix. This repeated computation is unacceptably slow for large circuits.

B. Scalable spatial correlation modeling

In this Section, we seek a sustainable and scalable solution by directly exploiting the spatial sparsity of the correlation.

In a similar vein as [8, 4], we employ a grid to model spatial correlation. Unlike previous attempts, however, we use a hexagonal grid as shown in Fig. 4. The reasons that the hexagonal grid is preferable are: (1) the hexagonal grid is the most isotropic among regular grids which can uniformly tile 2D space, which helps to achieve high accuracy and efficiency in approximating the spatial correlation function; (2) a hexagonal cell has only 6 immediate neighbors (all sharing 1 edge and 2 vertices) while a rectangular cell has 8 neighbors (some sharing an edge, and some sharing only a vertex), hence we can get by with fewer terms in a linear model. Nonetheless, it should be noted that our technique can be applied to other grid schemes as well.

Each grid cell has a *physical* random variable ΔX_S associated with it. Delays and slews are modeled as a function of the physical random variable of the grid cell where the gate or wire occurs. Physical random variables are correlated in order to model spatial correlation. We also associate an independent unit Gaussian *virtual* random variable (i.e., without physical meaning) with each grid cell. We express the physical random variable ΔX_S of each grid cell as a linear combination of the virtual random variables of its own cell and its neighbors. For simplicity of presentation, we consider only immediate neighbors. Mathematically, the model is

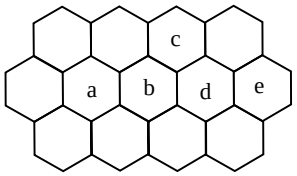


Fig. 4. Hexagonal grid.

$$\Delta X_S = a_{sd} \Delta Y_c + a_{ld} \sum_{k=1}^{6} \Delta Y_{N,k} \tag{4}$$

where ΔY_c and $\Delta Y_{N,k}$ are the virtual random variables of the grid cell of interest and its six immediate neighbors; a_{sd} represents short-distance correlation, and a_{ld} represents long-distance correlation; these coefficients are chosen to best represent the decay of correlation with distance. To preserve isotropic behavior, the coefficients a_{ld} are the same for all immediate neighbors. Because global variation ΔX_i and uncorrelated variation ΔX_R are explicitly modeled in (3), the spatial correlation coefficient of ΔX_S varies from 1 to 0 as a function of distance.

Fig. 5 shows the distance characteristics of a hexagonal grid, which are summarized in Table I. Consider a first instance at the center of grid cell a. Now consider a second instance at varying distances d from the first one as shown by the concentric circles in the Figure. Assuming R to be the radius of the circumscribing circle of any hexagon, the second instance must be at a distance between 0 and R to be in the same cell a (i.e., lie between concentric circles 1 and 3), and thereby share 1 cell virtual variable and 6 neighbor variables. This is shown in the first row of the table as 1c+6n and the corresponding correlation coefficient is shown in the last column. If the second instance is in cell b, the distance range is $\sqrt{3}R/2$ to $\sqrt{7}R$ between concentric circles 2 and 6, and four neighbor virtual variables are shared. Similar considerations allow us to fill out the rest of the table. Note that the distances form overlapping regions.

TABLE I
CORRELATION COEFFICIENT AS A FUNCTION OF DISTANCE.

Range of distance d	**Circles**	**2nd inst.**	**Shared vars.**	ρ
$[0, R]$	$1 \rightarrow 3$	a	1c+6n	$a_{sd}^2 + 6a_{ld}^2$
$[\frac{\sqrt{3}}{2}R, \sqrt{7}R]$	$2 \rightarrow 6$	b	0c+4n	$2a_{sd}a_{ld} + 2a_{ld}^2$
$[2R, 4R]$	$4 \rightarrow 8$	c	0c+2n	$2a_{ld}^2$
$[3\frac{\sqrt{3}}{2}R, \sqrt{19}R]$	$5 \rightarrow 9$	d	0c+1n	a_{ld}^2
$[\sqrt{13}R, \infty]$	$7 \rightarrow 10$	e	0c+0n	0

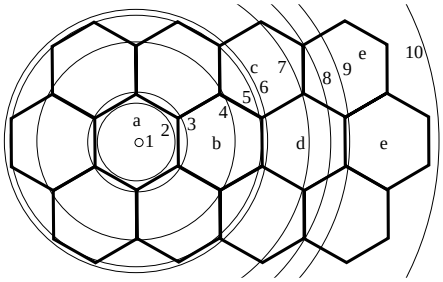


Fig. 5. Distance characteristics of hexagonal grid.

Nearby physical variables share more common dependencies than distant variables, and hence are more correlated. We approximate the specified correlation function with a stepwise constant function. For a given technology, we perform a one-time fitting procedure to determine the proper grid size and the coefficients a_{sd} and a_{ld}. This is formulated as an optimization problem that minimizes either the least square error or maximum absolute error between the given correlation function and the stepwise constant approximation. Solving such an unconstrained minimization problem in 3 variables is straightforward, hence we omit the details.

Additional accuracy can be obtained by including dependence on second-order neighbors, increasing the number of terms in (4) from 7 to 19. Our experiments showed, however, that considering immediate neighbors provides sufficient accuracy. Fig. 6 illustrates the approximation of two spatial correlation functions: Gaussian and exponential, respectively.

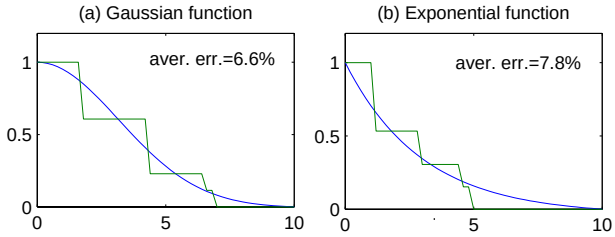


Fig. 6. Approximation of correlation functions.

C. Fast indexing

An efficient indexing scheme was developed to locate the grid cell containing any point on the chip and its neighbors. Denote n_r as the number of cells in a row, so that cell indices at each row k start from $k \times n_r$ for $k \geq 0$ (see Fig. 7, $n_r = 5$). We overlay an auxiliary rectangular grid as shown in Fig. 7 on top of the hexagonal grid, and denote the horizontal and vertical indices for each rectangular cell as (i_{hr}, i_{vr}), all starting at zero. The index of the left hexagonal cell covered by each rectangular cell is

$$i \quad = \quad i_{vr} \cdot n_r + 2 \cdot i_{hr}. \tag{5}$$

Given any point (x, y) on the chip, it is straightforward to find (i_{hr}, i_{vr}) based on the rectangular grid. Then (5) gives us the index of the left hexagon within the rectangle and depending on (x, y)'s location within the rectangle, the final index is computed by referring to the right side of Fig. 7 which depicts the offsets of the four hexagonal cells touched by a rectangular cell.

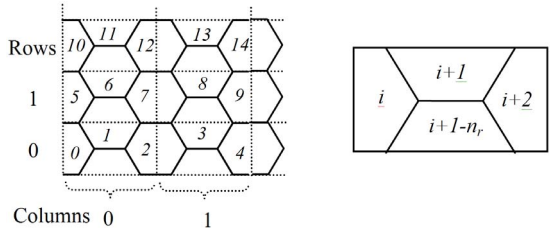


Fig. 7. Hexagonal index scheme.

D. Timing with spatial correlation

The delay of each arc of the timing graph is first modeled in terms of physical variables, and then transformed to virtual space. All statistical calculations are conducted on virtual variables. During statistical timing, when a clock and data arrival time are compared at a flip-flop to evaluate a timing test, these arrival times automatically contain a geometrical "trace" or "history" of the grid cells traversed by the most critical path or paths leading up to the flop. Thus common dependencies on grid cells and nearby neighbors cancel out, thus obtaining spatial timing credit automatically.

Compared with the PCA-based approach, the main benefits of the proposed method are: (1) no EVD, matrix multiplication or other matrix operations; (2) spatial sparsity of the correlation matrix is directly exploited; (3) the extended canonical forms remain reasonably short and sparse, and the overhead of accommodating spatial correlation is small; (4) the method scales to very large chip designs; and (5) it is easy to trace the areas of the chip that contribute to spatial variation based on the canonical form of the timing quantity of interest.

IV. Statistical timing methodology

The early/late split has numerous pernicious effects, including the requirement of a path-based non-incremental algorithm CPPR. What we require to eliminate CPPR is that the clock network should have the same early and late delays. In the data network, the early/late split can be either retained, partially eliminated or fully eliminated, depending on the level of pessimism reduction desired and the confidence in the delay models.

The independently random part poses a problem in the case of path re-convergence in general, and CPPR in particular. If the arrival time at the frontier point as shown in Fig. 1 has an independently random part which survives to the clock and data pins of the capturing flop (CF), the independently random parts should cancel each other out. But since the "source" of the independently random part is lost during timing traversal, a straightforward approach would treat this situation conservatively, which results in additional pessimism. Our solution is to introduce new global random variables at potential frontier points of the clock tree[1]. While this may seem expensive, three techniques are applied to improve efficiency: (a) the independently random part of a frontier point arrival time has to exceed a threshold for a new random variable to be introduced; (b) small sensitivities are accumulated into an independently random term as timing proceeds, and at each node of the timing graph, random variables with small sensitivities are pruned; (c) canonical forms are

[1] [6] introduced new random variables to handle re-convergent fanout rather than to explicitly target CPPR.

stored in a sparse format so as to avoid storing of zero-valued sensitivities. With these techniques, introduction of new random variables turns out to pose a relatively small overhead. Because of the explicit modeling of independently random variation, statistical cancellation credit is automatically obtained without any undue pessimism.

We propose a novel statistical timing methodology shown in Fig. 8. The first step is to use a grid-based approach to handle spatial correlation. A hexagonal grid with proper size is determined and overlaid on top of the chip. The second step is to convert the variational delay model into the extended canonical delay model (3), which assigns systematic variability partially or fully to a spatial random variable ΔX_S. The third step is the mapping of spatial random variables to a set of independent virtual random variables for statistical timing. Once the modeling has been done, the next step is to introduce new random variables to represent independent variation at all potential frontier points of the clock tree. Finally, we conduct a block-based statistical timing in terms of the global sources of variation, independent virtual random variables, and the newly introduced random variables at divergence points of the clock tree.

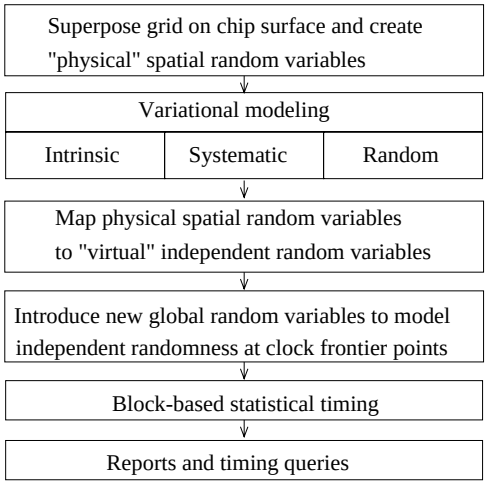


Fig. 8. The proposed timing methodology.

The benefits of our proposed flow are: (a) fully incremental because of the block-based nature of our timing; (b) controlled amount of pessimism reduction; and (c) correct treatment of all three types of variation. The efficiency, incrementality and pessimism reduction features lend themselves well both to timing sign-off and timing optimization.

V. Experimental results

A. Spatial correlation results

Three 90 nm industrial ASIC designs, whose sizes range from 10K to 100K gates are used to show the effectiveness of our spatial correlation modeling technique. Among the six process parameters in our experiments, we only allow one to have spatial correlation with a linear decay function. The 3-sigma variation of this spatially correlated process parameter is 15% of its nominal value, consisting of 20% for chip-to-chip variation, 60% for spatial correlation, and 20% for independently random variation. The 3-sigma variation for the rest of the process parameters is set to be 5% in total. This setting accentuates the effect of spatial correlation. For comparison purposes, we implemented the PCA-based approach with a sparse EVD algorithm, which we use as a baseline for comparisons.

Table II compares the accuracy between our approach and the existing PCA approach. The correlation distance $\overline{d}$ is shown in column 2, and the grid cell size R is in column 3. Compared to the grid size R, the correlation distance $\overline{d}$ in this experiment is purposely chosen to be relatively large, which favors the PCA-based approach. We compare the mean and standard deviation of the circuit delay from both approaches. We observe that our approach achieves the same mean delay value as the PCA-based approach, and sigma is almost the same as the one from the PCA-based approach.

TABLE II
Comparison of timing accuracy.

Chip	$\overline{d}$ (μm)	R (μm)	PCA (ns)		Ours (ns)	
			Mean	Sigma	Mean	Sigma
D1	400	80	3.715	0.178	3.715	0.176
D2	400	80	4.210	0.624	4.210	0.652
D3	4000	800	11.45	0.918	11.45	0.987

We also compare runtime performance between our approach and the PCA-based approach in Table III for design D1. Various grid cell sizes R are tested. As shown in Table II, when the grid cell size is small, the PCA-based approach slows down due to the large number of grid cells. The resulting large correlation matrix makes EVD the computational bottleneck. In contrast, our proposed method is almost insensitive to the number of grid cells. Compared to the PCA-based approach, our method achieves more than 150$\times$ speedup.

TABLE III
Run time comparison for D1.

R (μm)	160	80	40	20	10
Cell #	63	143	378	1167	4599
PCA (sec.)	127	124	128	390	17703
Ours (sec.)	80	80	83	87	98

We also compare the runtime and memory usage of our approach with basic SSTA without modeling spatial correlation. The runtime and memory overheads are both less than 50%.

B. Timing methodology comparison

We compare the proposed statistical timing methodology to an existing production multi-corner timing methodology based on case studies of two large 90 nm industrial designs, D4 and D5. D4 is a 69.2 mm^2 chip containing 227,286 placeable objects, while D5 has 387,488 objects[2] with an area of 22.2 mm^2.

In the case of multi-corner timing, timing credits due to spatial correlation and statistical cancellation of independently random variation are obtained in relation to an initial timing run, in which the spatial and random components of delays contribute to the early/late split. To compare multi-corner and statistical results, we project the statistical slacks to their worst corner values as follows: depending on the sign of each sensitivity, the corresponding process parameter is moved to its +3σ or -3σ corner to convert the canonical slack into a single number. In other words,

[2] Some of the placeable objects are large macros or memory blocks, so in this case the larger chip has fewer objects.

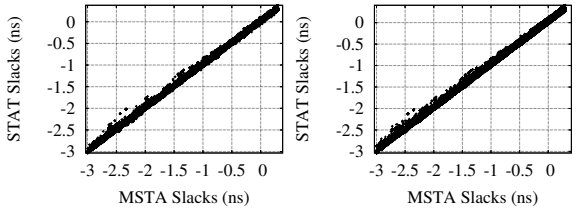
the statistical timing was configured to behave like multi-corner timing for the purpose of comparison.

Two scenarios are tested. The first scenario is without spatial correlation, i.e., we lump the spatial variation into the chip-to-chip variation bucket. The second scenario is with spatial correlation and the correlation distance is set as $\overline{d} = 200\mu m$. Table IV shows the slack and CPU time comparisons between the multi-corner timing (MSTA) and the proposed statistical timing (SSTA). We report the difference between all timing test slacks, and the biggest ("Maxdiff"), smallest ("Mindiff") and average absolute ("Avgdiff") differences are shown. We also report the CPU time on an IBM 1.9 GHz Power 4 Risc/System 6000 machine. On average, the SSTA flow produces timing results with less pessimism than the multi-corner flow even with its results projected to the worst corner, and the relative speedup of SSTA over MSTA ranges from $4\times$ to $8\times$.

TABLE IV
COMPARISON OF TIMING METHODOLOGIES.

Scenario	Chip	SSTA-MSTA slack (ps)			SSTA	MSTA
		Maxdiff	Mindiff	Avgdiff	(sec.)	(sec.)
No Spat.	D4	311	-84	8	357	2682
	D5	667	-73	40	1523	6203
With Spat.	D4	327	-64	20	395	2658
	D5	631	-93	101	1634	6207

We also show the slack differences between multi-corner timing and statistical timing on design D1 graphically. The slack comparison without and with consideration of spatial correlation is shown in Fig. 9. The x-axis shows multi-corner timing slacks; while the y-axis shows the worst-case projected statistical timing slacks. Both plots have about 15,000 data points. The results indicate a good correlation between these two approaches. Differences are seen in the range of [-50ps, 50ps], which arise due to both modeling differences and the different approaches taken to handle spatial correlation. In most cases, SSTA achieves reduced timing pessimism compared to MSTA, even though the results from SSTA have been projected to the unlikely region of a worst-case corner.



(a) No spatial (b) With spatial

Fig. 9. Comparison between multi-corner and statistical timing.

To verify the statistical results, Monte Carlo analysis was conducted on the smaller design D3. We find that the relative error of the mean of the slack distribution is 0.6%, while the worst-case projected slack is within 2.6% of the Monte Carlo results.

Spatial correlation is handled by creating a grid and introducing a new "virtual" random variable for each grid cell. This increases the memory consumption of the proposed statistical timing methodology. The introduction of global random variables to represent independent randomness at potential frontier points in the clock network increases memory further. For example, for chip D4, introducing these random variables blows up the memory consumption from 448 MB to 1.03 GB. To tame this problem, a sparse filtering technique was implemented whereby small sensitivities are pruned off (and combined into the independently random term). Since zero-valued sensitivities are not stored, this leads to big memory savings. With a sensitivity filtering threshold of 1 ps/σ, the memory consumption is reduced to 550 MB, with only slightly changed timing slack (5 ps or less) among all timing tests.

The results above demonstrate the pessimism reduction capability of statistical timing when spatial correlation is treated statistically. The memory overhead is controllable by sparse filtering. We should also not lose sight of the fact that the statistical methodology covers the entire process space (like the full-blown multi-corner approach), but is incremental. This makes it a good candidate for guiding physical synthesis, optimization and fix-up.

VI. CONCLUSIONS AND FUTURE WORK

This paper presented a novel methodology to accurately handle chip-to-chip, within-chip spatial and within-chip independently random variations. It is a statistical timing methodology, which possesses many benefits, chief among them being linear run time and fully incremental operation, reminiscent of simpler days prior to the advent of the early/late split. The proposed methodology was implemented in an industrial timer in a manner fully compatible with traditional multi-corner timing. Experiments on large designs proved that timing results of the new methodology correlate very well with both Monte-Carlo and traditional multi-corner timing.

ACKNOWLEDGEMENTS

The authors sincerely thank all members of the extended IBM EinsStat and IBM ASIC timing methodology teams, especially N. Venkateswaran, K. Kalafala and P. Qi for extensive consultations.

REFERENCES

[1] S. Sapatnekar. *Timing*. Kluwer Academic Publishers, 2004.
[2] D. J. Hathaway, J. P. Alvarez, and K. P. Belkhale. Network timing analysis method which eliminates timing variations between signals traversing a common circuit path. *U. S. Patent 5,636,372*, June 1997.
[3] C. Visweswariah, K. Ravindran, K. Kalafala, S. G. Walker, and S. Narayan. First-order incremental block-based statistical timing analysis. *Proc. 2004 Design Automation Conference*, pages 331–336, June 2004. San Diego, CA.
[4] H. Chang and S. S. Sapatnekar. Statistical timing analysis considering spatial correlations using a single PERT-like traversal. *IEEE International Conference on Computer-Aided Design*, pages 621–625, November 2003. San Jose, CA.
[5] P. S. Zuchowski, P. A. Habitz, J. D. Hayes, and J. H. Oppold. Process and environmental variation impacts on ASIC timing. *IEEE International Conference on Computer-Aided Design*, pages 336–342, November 2004. San Jose, CA.
[6] L. Zhang, Y. Hu, and C. C. Chen. Block based statistical timing analysis with extended canonical timing model. *Proc. Asia South Pacific Design Automation Conference (ASPDAC)*, pages 250–253, January 2005. Shanghai, China.
[7] J. Xiong, V. Zolotov, and L. He. Robust extraction of spatial correlation. *Proc. International Symposium on Physical Design*, pages 2–9, April 2006. San Jose, CA.
[8] A. Agarwal, D. Blaauw, and V. Zolotov. Statistical timing analysis for intra-die process variations with spatial correlations. *IEEE International Conference on Computer-Aided Design*, pages 900–907, November 2003. San Jose, CA.