Static Timing: Back to Our Roots

R. Chen[4]*, E. A. Foreman[2], P. A. Habitz[2], J. G. Hemmett[2], K. Kalafala[1], J. S. Piaget[1], P. Qi[1], N. Venkateswaran[1], C. Visweswariah[3], J. Xiong[3] and V. Zolotov[3]†

[1]IBM Microelectronics, East Fishkill, NY
[2]IBM Microelectronics, Essex Junction, VT
[3]IBM Research, Yorktown Heights, NY
[4]Northwestern University, Evanston, IL

ABSTRACT

This methodology paper describes an efficient timing flow with comprehensive modeling and analysis of process, environmental and aging variations. Chip-to-chip, within-chip systematic and independently random variations are handled. Chip-to-chip variations are traditionally accommodated by multi-corner timing, but with the advent of statistical timing, they can be incorporated more efficiently and with reduced timing pessimism. Within-chip variation is traditionally modeled by means of an early/late split, often by so-called "derating OCV (On-Chip Variation) coefficients," which requires a path-based Common Path Pessimism Removal (CPPR) step in the static timer. However, path-based algorithms suffer from superlinear complexity and are not amenable to incremental timing. This paper first describes a comprehensive multi-corner path-based approach, and then presents a novel methodology for handling all three types of variation with linear-time, incremental, statistical timing techniques. Pessimism reduction and speedup results are presented on two 90 nm industrial chip designs.

1. INTRODUCTION

Corner-based static timing has long been the bedrock technique for timing verification of digital integrated circuits. Recently, increasing variability has challenged this well-understood methodology. Variability can be classified as chip-to-chip (or global) and within-chip (or local). Within chip variation is further broken down into spatially correlated variation and independently random variation. In addition, temperature and voltage variation contain global, local and independently random components. A modern timing methodology must deal with all of these sources of variation accurately, efficiently, and with a minimum of pessimism.

Multi-corner timing is a natural extension of corner-based static timing to handle the multitude of significant sources of variation. Within-chip variation has traditionally been handled by an early/late split, i.e., the delay of an arc of a timing graph is larger for late timing purposes than for early mode timing. The early/late split is often introduced by so-called "OCV derating coefficients." Independently random variation leads to statistical cancellation down a path, and is therefore a source of timing credit particularly towards setup tests. This variation is typically lumped into the early/late split in a traditional methodology and is therefore treated pessimistically.

Thus a traditional timing methodology suffers from many drawbacks. First, a large number of corners must be covered to handle chip-to-chip variation. Second, including within-chip variation into an early/late split (irrespective of the physical layout of the circuit) is pessimistic. Third, lumping independently random variation into the early/late split without statistical cancellation is pessimistic. Fourth, any early/late split requires a path-based Common Path Pessimism Removal (CPPR) algorithm for correct handling (more details on this in Section 2), different flavors of which suffer from exponential or quadratic complexity in graph size.

The main contributions of this paper are: (a) A sign-off quality multi-corner methodology that checks all legal corners in a single timing run. This methodology allows for accurate inclusion of spatial correlation, correct handling of independently random variation and statistical treatment of a subset of the sources of variation. (b) A novel statistical timing methodology with the following benefits:

1. Correct handling of all three types of variation.
2. Ability to automatically derive statistical delay models from existing corner-based delay models on the fly.
3. Linear time algorithms that scale well to large chip designs.
4. Reduced timing pessimism.
5. Fully incremental operation that can be used efficiently to guide physical synthesis tools.

It is important to note that ever since the adoption of derating coefficients to handle within-chip variation, sign-off timing has not been either linear-time or incremental; hence the title of this paper.

The organization of the rest of the paper is as follows. The multi-corner path-based approach is described in Section 2, followed by the linear-time modeling and timing methodology in Section 3. Special attention is paid to spatial correlation in Section 4. Section 5 presents numerical results on industrial 90 nm chip designs, followed by conclusions and future work in Section 6.

2. MULTI-CORNER TIMING

This section describes a multi-corner timing methodology that relies on a delay model with the global, spatial and independently random variations broken out [1].

2.1 Traditional CPPR

We first review the well-known CPPR algorithm [2] before describing multi-corner extensions. CPPR is an essential analysis counterpart to an early/late split in the delay model. Fig. 1 shows a typical situation with a clock tree and combinational

*This work was done while the author was an intern at IBM Research
†Authors in alphabetical order

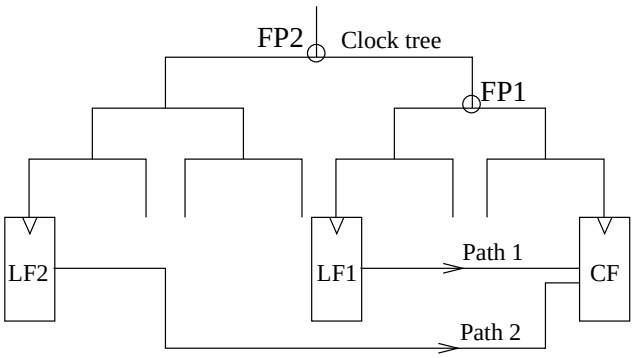


Figure 1: Traditional CPPR.

paths between flip-flops. Each combinational path has a launching clock path and a capturing clock path, which meet in the clock tree at a node called the *frontier point*. Assuming a setup test, the launch path is the late-mode path and the capture path the early-mode path. The physically common part of the clock tree (upstream of the frontier point) cannot be both early and late at the same time, but this is the assumption in the initial timing. So CPPR is used to remove this timing pessimism. In its simplest form, CPPR peels (in order of criticality) all paths leading to a failing test point and times each one with the launching arrival time at the frontier point equalized to the capturing arrival time. The limiting path in this analysis determines the final slack (or timing margin) at the test point, which is never worse than the starting slack. It is important to note that if the first path yields a large CPPR credit (such as Path1 from launching flop LF1 to capturing flop CF in Fig. 1, with frontier point FP1), it is possible that a sub-critical path (such as Path2, with a different frontier point FP2) will now be the limiting path since it obtains a smaller CPPR credit; thus all paths must be checked till the final slack can be correctly justified. A straightforward implementation of the CPPR algorithm is exponential in complexity, although a quadratic algorithm can be achieved with a clever data structure. In practice, the CPU time is kept in check by limiting the number of paths explored per timing test, which is an approximation.

2.2 Generalized CPPR and extensions

The CPPR algorithm above can be extended to a multi-corner situation using an algorithm called generalized CPPR (or gCPPR for short) [3]. In this case, as paths are peeled for CPPR purposes, the physically non-common part of the launching/capturing path pair is timed at all legal corners to find the path-specific worst corner and the corresponding timing slack before moving to the next path. Again, the naïve implementation of this algorithm is exponential (and a quadratic algorithm for gCPPR does not exist), but the number of paths that are analyzed per timing test is typically limited. Thus all corners are covered in a single timing run. Note that slack can get worse during the procedure of checking all corners. Hence we either need a guaranteed pessimistic starting corner for all process variables, or else we have to process all timing tests in a certain positive slack window. Further refinements and efficiency enhancements are described in [3]. In practice, when a chip is close to sign-off, only a small section of the chip is analyzed on a path-and-exhaustive-corner basis, and as soon as a positive slack at a timing test can be justified by all the timing credits, the path peeling loop is exited.

We describe three extensions of gCPPR here. In the inner loop of gCPPR, the independently random portion of the delay of all gates and wires in the physically non-common path can

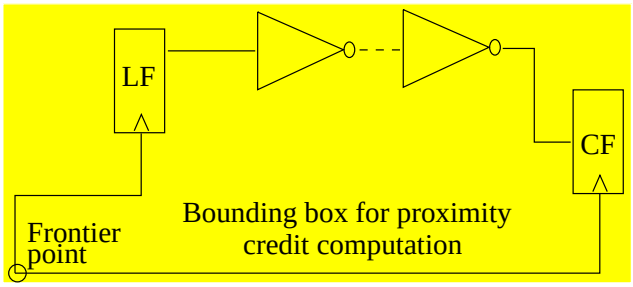


Figure 2: Proximity credit in gCPPR.

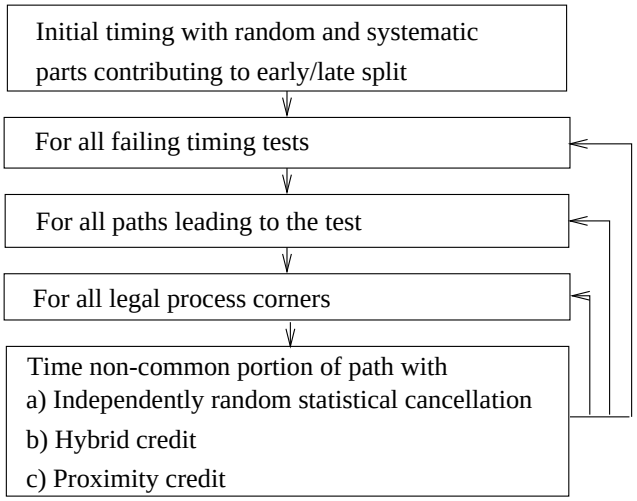


Figure 3: Flow chart for gCPPR.

be statistically summed, leading to an RSS (Root Sum Square) credit *down a path* compared to an initial timing where the independently random part was lumped into an early/late split. The sticky problem is that the independently random part of various path delays are actually correlated since they can traverse some common timing arcs (fortunately, this deviation can easily be bounded by assuming correlation coefficients of 0 and 1). The second extension is to achieve a *hybrid* or mixed corner/statistical timing by treating a subset of the process variables in a statistical fashion in the inner loop of gCPPR, thus achieving RSS credit *across the process space* for this subset of variables. Finally, a *proximity credit* can be achieved by computing the physical bounding box of all pins of the launching and capturing paths and derating the spatial portion of the early/late split of gates and wires in this path by a fraction based on the size of this bounding box (see Fig. 2). Due to the path-based nature of the algorithm, additional enhancements were implemented including a) treatment of known separable parameters in a linear fashion (i.e., setting each such separable parameter to a worst-case corner independently); b) pessimism relief for multiple noise coupling events along a path (based on a user-specified threshold on the maximum number of aggressor nets to consider per path); and c) various path filtering techniques based on, e.g., non-common path latency, to reduce the number of cases where full corner analysis is required. Fig. 3 summarizes the sequence of timing steps to apply gCPPR with all the extensions described in this section.

3. STATISTICAL TIMING METHODOLOGY

This section describes a novel statistical timing methodology

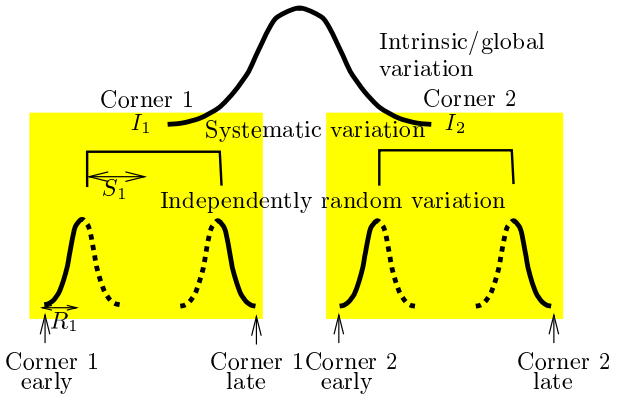


Figure 4: Variational delay modeling.

that is incremental, possesses linear complexity, and reduces pessimism.

3.1 Variational delay modeling

Our methodology uses the variational modeling scheme proposed in [1] for multi-corner timing. Fig. 4 shows the variation of delay due to one source of variation. The delay model consists of three components. An intrinsic or global component captures chip-to-chip variation. Systematic variation captures "predictable" across-chip variation and includes the spatial component of the delay model. Finally, independently random variation has no correlations and can be RSSed down a path. The characterization procedure required to produce such models involves creating qualification chips with special test macros at many locations on the chip. Each macro has repeated instances of circuitry that focus on a particular parameter of interest. The variance of the chip means give us the intrinsic or global statistics, the variance of macro means within the chip gives us the systematic component, and the variance within each macro gives us the independently random component. The delays of Fig. 4 are also characterized as functions of voltage and temperature. The intrinsic component defines the corners for multi-corner timing. A corner is simply an assignment of each source of variation to one of its extreme values.

Referring to Fig. 4, for corner-based purposes the delays at corner 1 are:

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

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

and likewise for corner 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.

Thus we see that there is inherently an early/late split in the modeling to account for within-chip variation. This early/late split accounts for spatial variation, independently random variation, multiple input switching, initial charge on internal nodes of gates, coupling noise effects, etc. From a timing methodology point of view, an early/late split has numerous pernicious effects, including the requirement of path-based non-incremental timing algorithms. One goal of our methodology is to get rid of the early/late split in the clock tree so as to be able to return to our block-based (and hence fully incremental) timing roots.

3.2 The canonical model

We adopt an extended form of a first-order canonical statistical delay model [4, 5]:

$$\text{Timing quantity} = a_0 + \sum_i a_i \Delta X_i + a_S \Delta X_S + a_R \Delta X_R \quad (3)$$

where a_0 is the nominal value, a_i are the sensitivities to global sources of variation ΔX_i, a_S is the sensitivity to a random variable ΔX_S that represents spatial variation, and the last term represents independent randomness. For the delay and slew (transition time) of each arc of the timing graph, the coefficients of the equation above are readily derived from the variational timing models (Fig. 4) by means of finite-differencing. In principle, each gate has its own ΔX_S random variable and physically nearby ΔX_S variables are more correlated than far-away variables. Note that for corner-based timing, the systematic and independently random terms contribute to an early/late split, whereas in this model, due to the last two terms of (3), early and late delays have the same model.

As a practical matter, what we require to eliminate CPPR from our methodology is that early and late delays have the same model only in the clock network. In the data network, the early/late split can be retained, partially eliminated or fully eliminated in favor of spatial random variables, depending on the level of pessimism reduction desired and the confidence in the delay models. Ideally, we need a scheme that can be used both at a pre-physical-design stage when (x, y) coordinates are not known, and in a post-physical-design stage when location information is available.

The independently random part poses a problem in the case of path reconvergence [5] in general, and CPPR in particular. If the arrival time at the frontier point has an independently random part which survives to the clock and data pins of the capturing flop, the independently random part should cancel. But since the "source" of the independently random part is lost during timing traversal, a straightforward approach would treat this situation conservatively. The solution is to introduce new global random variables at potential frontier points of the clock tree[1]. While this may seem expensive, three criteria 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 applied; (b) at each node of the timing graph, random variables with small sensitivities are pruned; (c) canonical forms are stored in a sparse format so as to avoid storing of zero-valued sensitivities. With these measures, introduction of new random variables in the clock tree turns out to pose a relatively small overhead, as shown in the results section.

3.3 Proposed timing methodology

Fig. 5 shows the main steps of the novel timing methodology. The first step is to use a grid-based approach to handle spatial correlation (similar to [6, 7], more details in the next Section). The second step is to convert the variational delay model into a canonical statistical delay model (3) on-the-fly to assign systematic variability partially or fully to grid random variables. The third step is mapping of grid random variables to create a basis set of independent random variables for statistical timing. Once the modeling has been done, the next step is to introduce new random variables to represent independent randomness at all potential frontier points of the clock tree. The next step is to conduct block-based statistical timing in terms of the global sources of variation, the independent grid random variables, and the newly introduced variables at divergence points of the clock tree.

[1] [5] introduced new random variables to handle reconvergent fanout rather than to explicitly target CPPR.

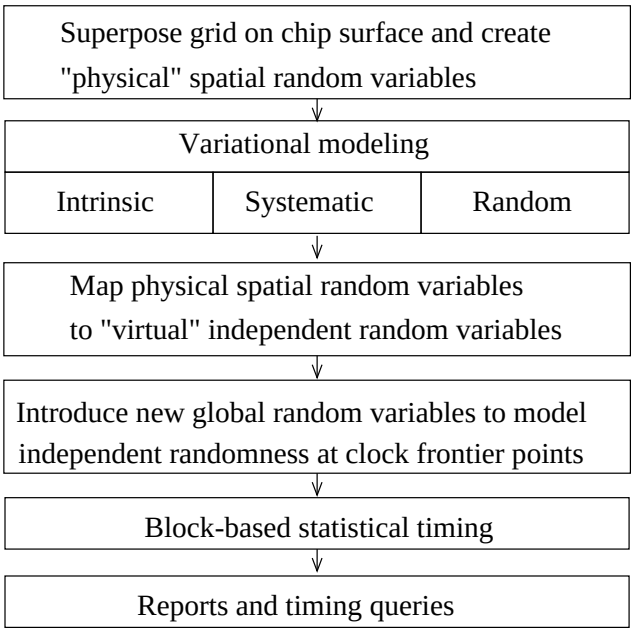


Figure 5: The proposed timing methodology.

It is important to note that the independent grid random variables and the random variables introduced at clock frontier points are *internal* to the timer. For any external purposes such as querying delays or writing timing reports, these sensitivities are RSSed and rolled up into a single term within each category.

To re-iterate, the benefits of this flow are: (a) fully incremental [8], block-based, one-pass timing; (b) controlled amount of pessimism reduction; (c) correct treatment of all three types of variation. The efficiency, incrementality and pessimism reduction features of this timing methodology lend themselves well both for timing sign-off and to a timing mode for guiding physical synthesis and optimization.

4. SPATIAL CORRELATION

A key contribution of this paper is the elimination of the early/late split in delay modeling of the clock tree. While Section 2 explained how proximity credit is obtained in a multi-corner timing context, this section presents details of spatial correlation during statistical timing.

Spatial correlation means that otherwise identical instances close to each other are more likely to have similar characteristics than instances far apart. By means of extensive hardware measurements, a correlation function can be established, as shown in Fig. 6. The correlation function drops below 1.0 at an infinitesimal distance due to the independently random component of delay. 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 global or intrinsic variation. Methods for deriving a physically valid correlation function from measurement data are described in [9].

As suggested in the literature [6, 7], we employ a grid to model spatial correlation. Unlike previous attempts, however, we use a hexagonal grid as shown in Fig. 7 since hexagons can be used to uniformly tile two-dimensional space with minimum anisotropy, and any pair of neighbors shares precisely one side and two vertices. Associated with each grid cell j is a physical zero-mean random variable ΔX_{Sj} that represents spatial variation. Delays

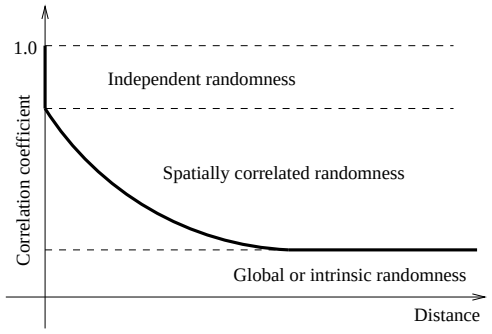


Figure 6: Spatial correlation function.

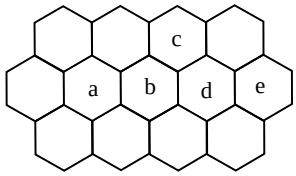


Figure 7: Hexagonal grid.

and slews are modeled in terms of these physical random variables as shown in (3).

The physical random variables of any two grid cells are correlated with each other as a function of the distance between the cells. There are two ways that these correlations are handled in the literature. The first is by using the well-known Principal Component Analysis (PCA) technique (e.g., [7]) to transform correlated random variables into a new set of independent variables. This technique suffers from two drawbacks. Consider a chip of size 10 mm $\times$ 10 mm, with a correlation distance of 300 μm. To represent the correlation function decay between 0 and 300 μm with any reasonable accuracy, assume that we need at least 3 grid cells, so the maximum allowable grid size is 100 μm, leading to a 100 $\times$ 100 grid. The covariance matrix that must undergo Eigen Value Decomposition (EVD) during PCA is thus 10,000 $\times$ 10,000. Thus the first drawback is the requirement of decomposing such a large matrix. Even though the covariance matrix is sparse, each physical random variable will end up depending on almost every grid random variable, and *vice versa* (since the system has 10,000 significant degrees of freedom), which is the second drawback. Thus the canonical forms that are computed and manipulated during statistical timing will contain 10,000 terms, slowing down the computations. The long canonical forms complicate timer implementation since timing results can no longer be stored in memory and must be written to disk during timing propagation.

The second technique in the literature [5] handles the correlations directly during statistical operations such as min and max. Consider two arguments to the statistical min or max function with sensitivity vectors A and B. An essential step in the statistical calculator is to find the correlation coefficient between A and B, which requires computation of an inner product A^TVB where V is the 10,000 $\times$ 10,000 covariance matrix in the example of the previous paragraph. These repeated matrix operations and the long resulting canonical forms are drawbacks of this technique.

In our approach, we seek a sustainable and scalable solution by directly exploiting the spatial sparsity of the situation. We first introduce a "virtual" random variable corresponding to each grid cell. These virtual random variables are independent, and will be the basis for all our statistical calculations that concern spatial correlation. Then we express the physical random variable of

each grid cell as a linear combination of the virtual random variables of its own cell *as well as its nearby neighbors*. For example, assume dependence on immediate neighbors only. Referring to Fig. 7, physical variables a and b share a common dependency on 4 virtual grid variables, while physical variables a and c share 2, physical variables a and d share one, and physical variables a and e share none. Thus by adjusting the coefficients of the dependencies of physical variables on virtual variables, the required correlation function (Fig. 6) can be approximated.

In practice, the following steps are used. For a given technology, there is a one-time determination of the grid size and a fitting procedure to determine the coefficients of the dependence of physical variables on virtual variables. During the timing run, the chip size is determined and a hexagonal grid overlaid. Given an (x, y) location of a gate, we need to quickly find the hexagon to which it belongs and its immediate neighbors; this is achieved by a straightforward indexing scheme. When a gate is timed, its delay and slew are first expressed as a function of the physical random variable of the grid cell (see (3)) by using the variational model of Fig. 4. Then this model is transformed to one in terms of the independent virtual grid random variables, and timing propagation continues as usual. Note that such location-based modeling can also be used to handle voltage and temperature variability. Small sensitivities are pruned and accumulated into an independently random term as the timing proceeds. By using sparse storage in the canonical form, memory is saved by this pruning procedure.

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 during the slack margin computation. On the other hand, if the launching and capturing paths are physically far-removed, a timing penalty automatically results.

The main benefits of this method of handling spatial correlation are: (a) no EVD, matrix multiplication or other matrix operations required; (b) spatial sparsity is directly exploited; (c) canonical forms remain reasonably short and the overhead of accommodating spatial correlation is small; and (d) the method scales to very large chip designs.

5. RESULTS

Two comprehensive timing methodologies were described in this paper: a multi-corner methodology using the gCPPR algorithm, and a statistical methodology using block-based statistical timing. Both have been implemented in the framework of EinsTimer, an industrial static timing tool. The methodologies themselves are codified as Tcl scripts and embedded in a graphical user interface.

This section presents case studies of two 90 nm chip designs, D1 and D2. D1 is a 69.2 mm^2 chip containing 227,286 placeable objects, while D2 has 387,488 objects[2] in an area of 22.2 mm^2. The sources of variation activated were Product Reliability (combination of NBTI and hot-electron effects), threshold voltage of each of 3 Vt families, and mistracking between NFET and PFET strengths. The tool set also permits metal layer, temperature and voltage variations. Corner-based models as described in Section 3 were employed. Various spatial correlation scenarios were benchmarked on both designs using the two methodologies. In addition, Monte Carlo results were obtained on a much smaller design, D3.

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

In order to compare multi-corner and statistical results, the statistical slacks were *projected* to a worst corner. In other words, depending on the sign of each sensitivity, the corresponding process parameter was moved to a $+3\sigma$ or -3σ corner to convert the canonical slack into a single number. This way, the statistical flow was configured to behave like multi-corner timing for comparison purposes.

In the case of multi-corner timing, both proximity and independently random credits were obtained (see Fig. 3) in relation to an initial timing run in which the systematic and random components of delay contributed to an early/late split. In both multi-corner timing and statistical timing, a linear decay of correlation coefficient with distance was assumed (see Fig. 6).

Four scenarios shown in Table 1 were tested. The first two scenarios ignored (x, y) location information, even though both designs are fully placed and routed. In this case, assuming a correlation distance of infinity is equivalent to lumping the systematic variation into the chip-to-chip bucket (and setting the distance to zero is tantamount to treating systematic variation as independently random). The next two scenarios use actual (x, y) information and assume a correlation distance of 200 μm. Each of these was run without and with an independently random part of delay variation; statistical cancellation down a path accrues when this is turned on.

Table 2 shows the slack and CPU time comparisons between multi-corner timing (gCPPR) and statistical timing (STAT). The first two columns denote the scenario and design. The next three columns show the difference between all test-point slacks, using a worst-case projection in the case of STAT. The biggest ("Maxdiff"), smallest ("Mindiff") and average difference ("Avgdiff") are shown. Finally, the CPU time on a 1.9 GHz IBM Power 4 Risc/System 6000 43P-681 machine is shown. On average, the STAT flow produces timing results with less pessimism than the multi-corner flow even with the projection to a worst corner, and achieves a speedup of about 4x. While proximity credit provides timing relief in multi-corner timing, it is still a pessimistic treatment of spatial correlation.

Table 1: Experimental scenarios.

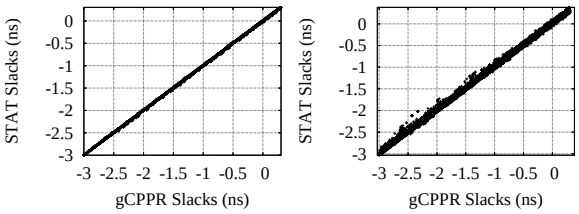
Scenarios	I.1	I.2	II.1	II.2
Correlation distance	Infinity	Infinity	200 μm	200 μm
Independently random?	No	Yes	No	Yes

Table 2: Comparison of gCPPR and STAT.

Scenario	Chip	STAT-gCPPR slack (ps)			STAT	gCPPR
		Maxdiff	Mindiff	Avgdiff	(sec.)	(sec.)
I.1	D1	48	-40	3	321	1216
I.1	D2	186	-50	5	1445	4076
I.2	D1	311	-84	8	357	2682
I.2	D2	667	-73	40	1523	6203
II.1	D1	101	-22	14	359	1158
II.1	D2	267	-100	55	1688	4075
II.2	D1	327	-64	20	395	2658
II.2	D2	631	-93	101	1634	6207

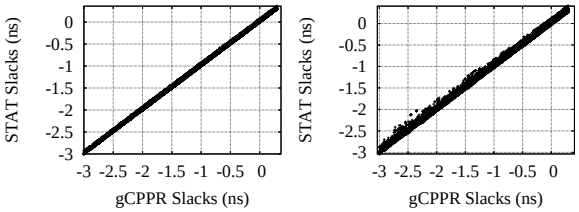
5.1 Perfectly correlated variations

Here we show graphically the slack differences between multi-corner timing and statistical timing on design D1 in scenarios I.1 and I.2. In these scenarios, the spatial variation is lumped with chip-to-chip variability, in other words this variability is considered to be perfectly correlated across the chip. Fig. 8(a) shows the slack comparison with multi-corner timing slacks on the X axis and worst-case projected statistical timing slacks on the Y axis. The plots in Figs. 8 and 9 have about 15,000 data points apiece. The results indicate a good match. Differences



(a) Scenario I.1 (b) Scenario I.2

Figure 8: Comparison between multi-corner and statistical timing with an infinite correlation distance (a) without independent randomness; (b) with independent randomness.



(a) Scenario II.1 (b) Scenario II.2

Figure 9: Comparison between multi-corner and statistical timing with a correlation distance of 200 μm (a) without independent randomness; (b) with independent randomness.

are seen in the range [-50ps, 50ps], and arise due to modeling differences and the different approaches taken to handle spatial correlation. Likewise, Fig. 8(b) shows the comparison when independent randomness is turned on. The pessimism reduction is noticeable in this case.

5.2 Monte Carlo result

To verify the statistical results, a Monte Carlo analysis was conducted on a smaller design D3 and timing results were compared on a timing test that showed the largest difference between multi-corner timing and statistical timing. The relative error of the mean of the slack distribution was 0.6% and the worst-case projected slack was within 2.6% of the Monte Carlo results.

5.3 Uncorrelated variations

Fig. 9(a) shows the comparison on design D1 between multi-corner slacks on the X axis and statistical slacks on the Y axis, assuming a correlation distance of 200 μm. Likewise, Fig. 9(b) shows the situation with independent randomness turned on. When spatial correlation is handled statistically, larger pessimism reduction is obtained.

5.4 A comment on memory consumption

In scenarios II.1 and II.2, spatial correlation was handled by creating a grid and introducing a new "virtual" random variable for each grid cell. This increases the memory requirement. The introduction of variables to represent independent randomness at potential frontier points in the clock distribution tree also increases memory. In the case of D1, introducing these random variables blows up the memory requirement for the timing step from 448 MB to 1.03 GB. However, a sparse filtering technique was implemented whereby small sensitivities are pruned off (and combined into the independently random term) as the timing proceeds. Since canonical forms are stored in a sparse format, zero-valued sensitivities lead to memory savings. With a sensitivity filtering threshold of 1 ps/σ, memory consumption reduced to 550 MB, with a slack change of 5 ps or less at all timing tests. Without filtering, the worst-case time and space complexity would be $O(n \log n)$ in the graph size.

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 multi-corner approach), but is incremental. This makes it a good candidate for guiding physical synthesis, optimization and fix-up. Diagnostics such as criticality [10] and yield gradients [11] can also be helpful during optimization. Pessimism reduction and timing risk can be balanced by: (a) choosing a conservative spatial correlation function; (b) allocating less of the systematic variation to spatial random variables in the data network; (c) RSSing only those process variables that are known to be independent and projecting the rest to their individual worst-case corners.

6. CONCLUSIONS AND FUTURE WORK

This paper presented two methodologies for accurately handling chip-to-chip, within-chip spatial and within-chip independently random variations. The first is a multi-corner timing methodology, with extensions of the CPPR algorithm to handle spatial correlation, independent randomness and statistical treatment of a subset of process variables. The second 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 derating OCV coefficients. Future work items include handling of non-separable sources of variation and thorough treatment of all sources of an early/late split including noise impacts on timing, IR drop effects, and so on.

7. ACKNOWLEDGMENTS

The authors acknowledge the contributions of L. Zhang (now with Cadence) towards implementation of spatial correlation during his summer internship at IBM Research. The authors thank all members of the extended IBM EinsStat and IBM ASIC timing methodology teams spread across East Fishkill, Yorktown Heights, Burlington, Poughkeepsie and Waltham for useful discussions, collaboration and software support.

8. REFERENCES

[1] 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.

[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] D. J. Hathaway, K. Kalafala, A. J. Suess, P. Qi, and C. Visweswariah. System and method for correlated process pessimism removal for static timing analysis. *U. S. Patent 7,117,466*, October 2006.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] C. Visweswariah. System and method for incremental statistical timing analysis of digital circuits. *U. S. Patent 7,111,260*, September 2006.

[9] 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.

[10] J. Xiong, V. Zolotov, C. Visweswariah, and N. Venkateswaran. Criticality computation in parameterized statistical timing. *Proc. 2006 Design Automation Conference*, pages 63–68, July 2006. San Francisco, CA.

[11] V. Zolotov, J. Xiong, and C. Visweswariah. Computation of yield gradients from statistical timing analysis. *Proc. 2006 TAU (ACM/IEEE workshop on timing issues in the specification and synthesis of digital systems)*, pages 125–130, February 2006. San Jose, CA.