Microelectronics Reliability 54 (2014) 464–474

Contents lists available at ScienceDirect

Microelectronics Reliability

journal homepage: www.elsevier.com/locate/microrel

Monte Carlo Static Timing Analysis with statistical sampling

Michael Merrett, Mark Zwolinski *

Electronics and Computer Science, University of Southampton, Southampton SO17 IBJ, UK

ARTICLE INFO

Article history:
Received 2 July 2013
Received in revised form 12 October 2013
Accepted 14 October 2013
Available online 7 November 2013

ABSTRACT

With aggressive scaling of CMOS technologies, MOSFET devices are subject to increasing amounts of independent local statistical variability. The causes of these statistical variations and their effects on device performance have been extensively studied, but their impact on circuit performance is still difficult to predict. This paper proposes a method for modeling the impact of random intra-die statistical variations on digital circuit timing. The method allows the variation modeled by large-scale statistical transistor simulations to be propagated up the design flow to the circuit level, by making use of commercial STA and standard cell characterization tools. By using statistical sampling techniques, we achieve close to the accuracy of full SPICE simulation, but with a computational effort similar to that of Statistical Static Timing Analysis, while removing some of the inaccurate assumptions of Statistical Static Timing Analysis.

1. Introduction

The International Technology Roadmap for Semiconductors (ITRS) has repeatedly highlighted that improvements in design productivity are not consistent with the scaling of manufacturing technologies, specifically in areas such as designing for performance and power variability [1]. The accuracy of timing validation and power estimation methods are being challenged by the scaling of process dimensions down to the nanometer level, where the impact of statistical process variations has become increasingly significant. Traditionally, process variations have been modeled at the circuit level by performing static timing analysis (STA), which makes use of calibrated lookup tables for standard cells at multiple technology-dependent corners.

The shortfalls of corner-based analysis have been known for many years, [2]. STA can be both overly pessimistic and optimistic, and hence Statistical Static Timing Analysis (SSTA) was proposed. Since then much effort has been placed into the development of practical and accurate SSTA tools but modern SSTA algorithms are still unable to address issues that are overcome by commonly-used STA methods, such as interconnect analysis, latch based designs and clock-skew analysis [3]. There remain large obstacles to the widespread use of SSTA in the industry, including a limited amount of statistical foundry data available in a standardized format.

The effects of localized statistical process variations such as random discrete dopants (RDD) [4] and line edge roughness (LER) [5,6] have been investigated and are understood at the device level, but these effects must be modeled at the circuit level in order to improve design for manufacturability methods. The challenge is to provide circuit designers with a transparent method for modeling the impact of MOSFET variability to allow trade off design decisions between power consumption, performance and manufacturing yield. We are not concerned in this work with systematic variations, due to, for example, the position of a device on a wafer.

This paper extends the use of Monte Carlo Static Timing Analysis (MCSTA), [7], a transparent method of modeling the impact of statistical transistor variations. MCSTA makes use of existing industry standard timing analysis and cell characterization tools, providing a low cost addition to existing design flows, and takes advantage of the maturity of established STA methods. MCSTA can be performed without assumptions of underlying transistor model parameter distributions, providing greater accuracy than SSTA, while executing considerably faster than Monte Carlo SPICE simulations. Nevertheless, MCSTA is still considerably slower than SSTA. This paper describes how statistical sampling may be used to limit the Monte Carlo simulations to the (interesting) tails of the distributions.

2. Simulation methodology

Previous research into the statistical variation of individual transistors has found that Random Discrete Dopants (RDD), Poly Silicon Granularity [8,9] and Line Edge Roughness (LER) [10] are the three main sources of fluctuations in threshold voltage (V_T). Simulations of three dimensional atomistic transistor models have provided distributions of I–V curves for transistors which traditionally would have been represented by a single continuous charge model. These I–V curves have been converted to a library of BSIM models [11], allowing the range of effects of statistical variations on a transistor to be modeled using SPICE.

* Corresponding author. Tel.: +44 2380593528.
E-mail address: mz@ecs.soton.ac.uk (M. Zwolinski).

0026-2714/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.microrel.2013.10.016

From the point of view of a circuit designer, the effect of variability on individual transistors is not necessarily of interest. The aim of this work is to predict the variability of cells and hence of a particular implementation of the overall circuit. In the case of digital circuits, the main characteristics of interest are the delays through combinational logic paths (which determine the maximum clock speed) and the power dissipated by the circuit.

Thus, the effects of these independent local statistical variations on the performance of a number of test circuits were measured and compared using three methods: Monte Carlo transistor-level SPICE simulations, SSTA, and our proposed MCSTA. The large scale statistical SPICE simulations provide a reference.

2.1. Monte Carlo SPICE simulations

We used a statistical circuit generation tool [12] to replace each MOSFET model instance within a SPICE netlist with BSIM models selected randomly, with a uniform distribution, from a process specific statistical library. Each individual transistor within a circuit was therefore modeled by a separate atomistic model. The tool was used to generate 10,000 randomized transistor level netlists of each circuit under test, and SPICE simulations were then performed on each circuit instance. The input vectors to the circuits included the stimulation of critical paths that were reported during STA. This allowed for a direct comparison of delays through fixed paths, as well as comparisons of the longest path delay through the circuit. The distributions of static and dynamic power consumption were also recorded, as reported elsewhere [13].

2.2. Statistical Static Timing Analysis

Traditional STA allows each delay element within a circuit to be replaced with a single delay value. This delay value is interpolated from a look-up table within a standard cell library (SCL) in which the delay elements (cells) have been characterized for a fixed set of process corners. The SCL is generated by passing SPICE-level netlists of standard cells through a cell characterization tool, Fig. 1(a). Timing analysis tools model the delay and output transition time for the element to be obtained for a given input transition time and output load capacitance, Fig. 1(b). Using this method every instance of a cell within a circuit references an identical model in the SCL; the three instances of NAND gates in the example of Fig. 1(b) all refer to the same NAND model. Traditional STA is therefore unable to represent statistical process variations.

A commercial statistical cell characterization tool was used to characterize a selection of standard cells. The same transistor model parameters and distributions used within the Monte Carlo SPICE simulations were used during this process. A statistical standard cell library (SSCL) was created using a 'Transistor Mismatch' model, which allows the performance of the cell to be established for multiple σ levels of each transistor parameter. These results were combined using a method specific to the commercial tool, allowing the variation to be modeled by a single synthetic variation parameter. This process is based on the observation that if each variable has an independent Gaussian distribution then the impact on a given timing parameter (delay, slew or constraints) resembles a Gaussian distribution.

The SSCL was then used within a commercially available SSTA tool, providing a distribution of delays for paths within the test circuits in a single run. Statistical power analysis was not possible using this method.

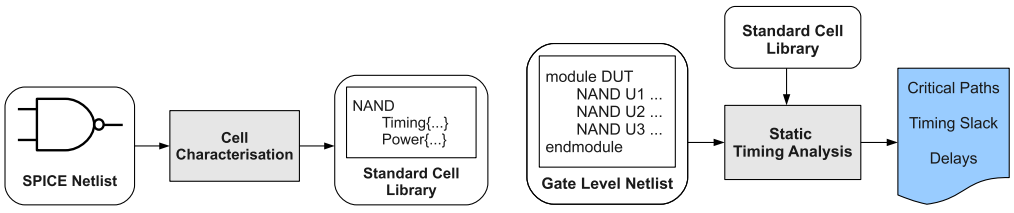
2.3. Monte Carlo Static Timing Analysis

The Monte Carlo Static Timing Analysis method was introduced in a previous paper [7]. It requires no assumptions about the underlying statistics of process parameters, instead using Monte Carlo SPICE simulations of standard cells. Multiple SPICE netlists were generated for selected standard cells, where each transistor referred to different BSIM models selected randomly from the process specific statistical library using RandomSpice. The multiple randomized cell netlists were passed into the same commercial cell characterization tool as with SSTA, generating a Variation Cell Library (VCL) where every standard cell has multiple instances, Fig. 2(a).

Each VCL contained 500 randomized instances of each of the standards cells. Each characterization involves a SPICE simulation and the extraction of cell parameters. The complexity of each simulation depends on the nature of the cell – the number of inputs and outputs and whether there is an internal state. Although we did not characterize every cell in the library for this exercise, only those that were instanced in the circuits, a typical cell library might contain 1000 different cells. This implies up to half a million unique cell characterizations and hence hundreds of hours of CPU time. Fortunately, this exercise only needs to be done once per cell library (and is trivially parallel).

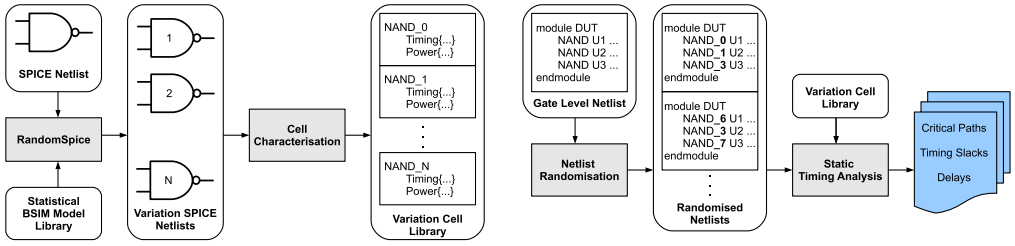
The statistical process variations within a VCL were modeled at the circuit level by making simple modifications to the gate level netlist of a design. Each reference to a standard cell within the original netlist was altered to represent a variation cell reference by the addition of the suffix _x, where *x* refers to the chosen variation model. Each individual cell was therefore modeled by a separate set of atomistic transistor models, rather than the same model as in traditional STA.

Multiple copies of the original netlist were created, where a randomized suffix was added to each SCL reference. This randomization process does not alter the structure or behavior of the circuit, only the statistical process variations of the cells being modeled. Each of these randomized netlists can then be passed through the STA tool, producing individual timing reports for each netlist. These results can then be combined and analyzed to produce a distribution of the timing of a design, measuring any slack within the critical paths and providing the probability of any paths through the design failing to meet timing requirements. The process of generating and performing STA on randomized gate level netlists is referred to as Monte Carlo Static Timing Analysis and is illustrated in Fig. 2(b).



(a) Generation of a Standard Cell Library (SCL) (b) Static Timing Analysis (STA)

Fig. 1. Generating a standard cell library and performing static timing analysis.



(a) Generation of a Variation Cell Library (VCL) (b) Monte Carlo Static Timing Analysis (MCSTA)

Fig. 2. Adding statistical process variations into the timing analysis process.

Table 1
Path delay statistics.

	SPICE	MCSTA		SSTA	
	ns	ns	%Error	ns	%Error
Adder					
Mean	0.0706	0.0707	0.0880	0.0699	1.0175
Std. Dev.	0.0020	0.0020	2.3688	0.0008	60.0179
-3σ (0.3%)	0.0653	0.0656	0.5043	0.0676	3.5034
$+3\sigma$ (99.7%)	0.0764	0.0765	0.1166	0.0721	5.6370
c74283					
Mean	0.0945	0.0942	0.3627	0.0933	1.3022
Std. Dev.	0.0024	0.0023	5.1052	0.0026	7.1089
-3σ (0.3%)	0.0884	0.0882	0.2072	0.0863	2.3595
$+3\sigma$ (99.7%)	0.1015	0.1005	0.9786	0.1009	0.5717
c2688					
Mean	1.1873	1.1816	0.4857	1.1676	1.6636
Std. Dev.	0.0102	0.0088	13.9391	0.0391	282.0789
-3σ (0.3%)	1.1589	1.1581	0.0742	1.0633	8.2529
$+3\sigma$ (99.7%)	1.2165	1.2047	0.9693	1.2857	5.6894

Table 1 shows that MCSTA is far more accurate than conventional SSTA (using the SPICE simulations as a benchmark). The three circuit examples are a small one-bit full adder (13 gates), the ISCAS-85 benchmarks, c74283 (a four-bit fast adder, 36 gates) and c6288 (a 16×16 bit multiplier, 2406 gates). On the other hand, MCSTA is far slower than SSTA, Table 2, and hence not suitable for practical use. The aim of the work presented here is to maintain the accuracy of MCSTA, but to significantly improve the run time.

3. MCSTA with statistical sampling

This section describes a method of statistical sampling that can be used within MCSTA. The method provides a prediction of the extremes of distributions generated by large MCSTA runs, with a fraction of the CPU time.

3.1. Comparing standard cell variation instances

A Variation Cell Library (VCL) [7], contains look up tables (LUTs) of power consumption, output transition times and cell delays for every variation instance of a standard cell. These look up tables can be compared and used to sort the variation instances of a standard cell into any desired order, for any specified measurement. For example, it is possible to compare the LUTs of every variation instance of an Inverter and determine which model contains the longest output delay for a specific input transition and output load. Another example would be to compare the power consumption LUTs and determine which NAND gate uses, on average, the most energy. The comparison can be performed using simple scripting languages, as the LUTs in a .lib file are written in ASCII.

Table 2
Time required for circuit analysis.

	CPU Time (h:m:s)		
	SPICE 10,000 Simulations	MCSTA 10,000 Runs	SSTA 1 Run
Adder	08:25:59	07:36:29	00:00:05
c74283	59:29:39	07:46:48	00:00:02
c2688	315:40:38	14:16:48	00:00:06

Using this information it is possible to find the extreme limits of the performance of a circuit under variation, or at least the limits at which MCSTA will predict the circuit performance. The slowest variation instances could be found for each standard cell, and STA could be performed on a circuit referencing only these slowest instances. Similarly the variation instances of each cell in a circuit could be selected as those with the highest values for leakage energy, for a worst case evaluation of leakage power consumption. The extreme of the distributions generated by MCSTA could be generated by deliberately targeting the worst or best case of each variation instance, saving the need to analyze thousands of randomized samples. However this would simply be an alternate form of corner analysis, where the worst possible variation of every single standard cell would be modeled simultaneously and would produce an overly pessimistic estimate of circuit performance.

The ability to compare the values of timing for each variation instance of a cell in a library allows for the examination of the delay distributions of each standard cell within the library. The distribution of possible values for a given input to output delay of a cell can be seen by the designer before any statistical analysis of the circuit is performed and the position of each variation instance within the overall distribution can be found.

In this paper, we are assuming that the distributions of cell delays and power consumption are approximately Gaussian around the means, but not at the tails. While the assumption of approximately Gaussian behavior is useful from the point of view of simplicity, it is not essential. We discuss the tails further in the next section.

One method of reducing the number of samples required during MCSTA is to find where a random sample of variation instances would fall within a Gaussian model of the circuit, and if these are found to be close to the mean of the distribution then the circuit analysis is not performed, whereas if the instances are found to be at the extremes of the distribution then circuit analysis is performed. If small numbers of samples from the center of the distribution of passed to an STA tool, while large numbers of corner samples are sent to an STA tool, then the overall distribution of circuit performance can be obtained at a fraction of the computation time.

3.2. Method

The initial step is to examine the contents of a chosen VCL, reading the contents of LUTs for the variation instances of each

Table 3
Example of rising cell delay through three variation instances of a standard inverter cell.

INV_1 Rise Delay (ps)	Input slew (ps)			
	2	4	20	60
Output load (fF)				
0.25	3.34	4.36	12.10	21.60
0.50	3.80	4.79	12.49	21.91
2.50	6.10	7.66	16.55	25.81
5.00	9.55	11.72	242.20	36.10
INV_2 Rise Delay (ps)	Input slew (ps)			
Output load (fF)				
0.25	4.72	6.16	17.09	30.50
0.50	5.37	6.76	17.64	30.94
2.50	8.61	10.82	23.37	36.44
5.00	13.48	16.55	341.99	50.97
INV_3 Rise Delay (ps)	Input slew (ps)			
Output load (fF)				
0.25	3.21	4.19	11.62	20.74
0.50	3.65	4.60	11.99	21.03
2.50	5.86	7.35	15.89	24.78
5.00	9.17	11.25	232.51	34.66

Table 4
Gaussian models of the standard inverter cell.

Mean rise delay (ps)	Input slew (ps)			
	2	4	20	60
Output load (fF)				
0.25	3.75	4.90	13.60	24.28
0.50	4.27	5.38	14.04	24.63
2.50	6.86	8.61	18.60	29.01
5.00	10.73	13.17	272.23	40.58
Std. Dev. rise delay (ps)	Input slew (ps)			
Output load (fF)				
0.25	0.84	1.09	3.03	5.40
0.50	0.95	1.20	3.13	5.48
2.50	1.53	1.92	4.14	6.46
5.00	2.39	2.93	60.60	9.03

standard cell. The mean and standard deviation of each point of each LUT is calculated for each standard cell, so that a separate Gaussian distribution can be used to represent the variation of time through each characterized timing arc through the cell. Examples of LUTs for the rising propagation delay through three variation instances of an inverter are given in Table 3, from which the means and standard deviations are found and stored in a Gaussian model such as that shown in Table 4. Elements from the same position within the LUTs are summed, and the mean value and standard deviations are stored within the equivalent positions of the mean and standard deviation LUTs of the Gaussian model, as highlighted in red and blue respectively within the examples.

The normal probability plot, Fig. 3, shows a comparison between the distribution of rising propagation delays through 500 variation instances of a 35 nm Inverter for a single point within the LUT, against the Gaussian model derived from the mean and standard deviation of the distribution. The distribution matches the model closely between probabilities of 0.1 and 0.9, but is less reliable at the extremes of the distribution. This implies that the Gaussian model could be used to model the average behavior of the inverter, but not the corner cases that circuit designers may be interested in.

We know that the distributions of the physical parameters contributing to the delays and other attributes are not necessarily Gaussian, although we cannot observe these parameters directly. Under fairly weak conditions (Lyapunov), the distributions of the propagation delays will tend towards a Normal distribution, according to the Central Limit Theorem,

$$\frac{1}{s_n}\sum_{i=1}^{N}(X_i - \mu_i) \rightarrow \mathcal{N}(0,1) \tag{1}$$

A significant amount of work has been done on modeling the tails of distributions. Of particular relevance to VLSI design is the *Statistical Blockade* technique [14]. The basic Monte Carlo method will, naturally, select a large number of samples around the mean of a distribution, with relatively few samples in the tails. Therefore a large number of Monte Carlo simulations would be needed to generate an accurate model of the tails. The idea of Statistical Blockade is to model the central region, thus reducing the need for many samples in the predictable region, allowing more sampling in the tails. Fig. 3 tells us that similar assumptions hold in this case: the distribution of the propagation delays is approximately Gaussian around the mean.

The exact fit to a Gaussian model differs between standard cells, between timing paths through a standard cell, and between positions within an LUT, but the overall pattern is that the center of the distribution fits while the extremes are modeled poorly. Therefore we are justified in reducing the sampling around the mean. In this paper, we are not particularly concerned with determining the

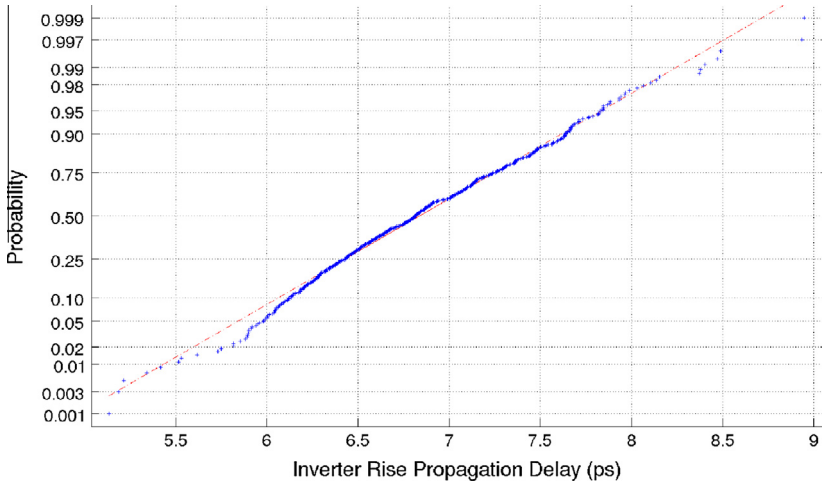


Fig. 3. Plot comparing the distribution of propagation delays for the rising transition of a 35 nm Inverter (blue) with a fitted Gaussian distribution (red). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

shapes of the tails, merely with modeling the extreme behavior. We could, of course, apply a technique like Statistical Blockade to gain a better insight into the exact behavior.

Analysis of the nominal version of a netlist (where no statistical process variations are included) is used to provide the direction of the logic transitions though a critical path, including input transition times and capacitative loads. These transition times and capacitative loads can then be used to determine which locations within each of the standard cell LUTs are used to obtain the correct Gaussian models to predict the variation of delays through each cell. The sum of these individual Gaussian models is then used to predict the distribution of delays through the entire path. The mean of the Gaussian distribution at any depth within the logic path is the sum of the means of each previous logic gate, while the variance of the Gaussian distribution is the sum of the variances of each of the previous logic gates and the standard deviation is the square-root of the variance.

A critical path through the one bit Adder that was analyzed in Section 2 provides a good example of this process, the schematic for this circuit is given in Fig. 4.

Fig. 5 contains plots of the probability density functions for the cumulative Gaussian delay models through each of the seven logic gates within the critical path. The Probability Density Function (PDF) to the furthest right (U12 BUFX2M) represents the distribution of delays at the output of the final gate within the critical path, and is therefore the Gaussian model for this transition through the entire critical path.

This is a very simple and fast method of predicting the distribution of delays through a logic path, and this method resembles some of the earliest forms of SSTA. It should be noted that although this crude method of statistical modeling produces inaccurate estimations of the extremes of circuit behavior, the purpose of this approach is not to accurately model the behavior of the circuit by using statistical models, but is instead to use statistical models to reduce the number of samples required by an accurate MCSTA run.

The goal of reducing the runtime of an accurate MCSTA run can be achieved by combining the above Gaussian models with the MCSTA method described in Section 2. References to standard cells within a netlist are replaced by randomly selected instances from a variation cell library as normal, and then the delays within the randomly selected variation instances are compared with the Gaussian models that have been generated for the circuit. If the combined behaviors of the randomly selected variation instances lie within the tails of the Gaussian model then the circuit is analyzed, but if the predicted behavior is within the center of the distribution, as the vast majority will be, then the effort of performing a full circuit analysis can be avoided as the results are not expected to be of interest to the designer. The predicted behavior of the current randomized netlist is the total delay of the variation instances of interest to the designer; this predicted behavior is compared against the Gaussian model by determining the probability of the predicted behavior being generated by the Gaussian model itself, finding where the predicted behavior lies within the Cumulative Distribution Function (CDF) of the Gaussian model. If the probability of obtaining the behavior using the Gaussian model is below a threshold specified by the designer then it may be discarded, otherwise it may be retained for further, more accurate, analysis. A simplified summary of this process is provided in Fig. 6.

For analyzing critical timing paths the process can be narrowed down to selected timing arcs through specific cells of interest. This may represent a particular set of paths that are known to be problematic, or a set of paths that have been altered by a change in the placement or routing of the design where the designer (or automated design tool) wishes to ensure that the alterations have not caused the paths to become critical paths. The individual cell delays through the path of interest are summed using the appropriate LUTs of the variation instances and the total is compared against the Gaussian model for the same path. If the predicted path delay is within the center of the Gaussian model then the randomized netlist may be discarded and a new one generated, if the predicted path delay is within the extremes of the Gaussian model then the netlist can be passed on for full STA.

The Gaussian model that was generated for the 1bit Adder in Fig. 5 provides a good example of how this form of statistical sampling can save computation time for MCSTA without large reductions in accuracy. The mean and standard deviation of each delay through each cell is obtained from the distributions of delays within a VCL, the appropriate Gaussian models are then combined to estimate the mean and standard deviation of an entire critical path. An example of this is given in Table 5, where a Gaussian model is selected for each cell within the path based upon the pins used, the direction of the input transition, and the capacitative loading. The

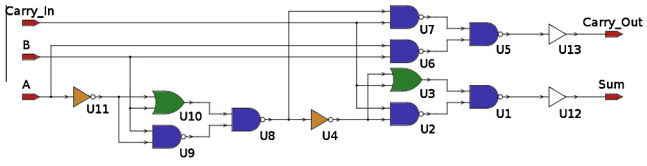


Fig. 4. Gate level schematic of a one bit full adder.

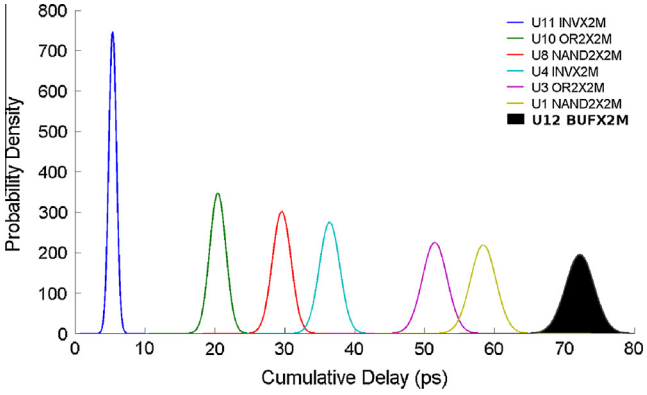


Fig. 5. Cumulative PDFs of delays through the critical path of a 35 nm 1bit Adder circuit, using Gaussian models for each individual cell delay.

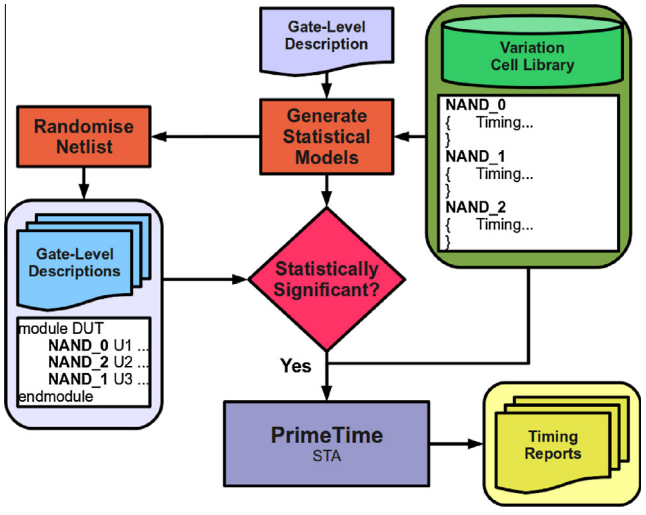


Fig. 6. Performing Monte Carlo Static Timing Analysis with statistical sampling.

Table 5
The mean and standard deviation of the delay through a critical path calculated from the multiple variation instances of each cell within a variation cell library.

Component	Cell	Mean delay (ps)	Standard deviation (ps)	Cumulative mean (ps)	Cum. Std. Dev. (ps)
U11	INVX2M	5.36	0.53	5.36	0.53
U10	OR2X2M	15.05	1.02	20.41	1.15
U8	NAND2X2M	9.15	0.66	29.56	1.32
U4	INVX2M	6.85	0.59	36.41	1.45
U3	OR2X2M	15.05	1.02	51.45	1.77
U1	NAND2X2M	6.93	0.43	58.39	1.82
U12	BUFX2M	13.83	0.93	72.22	2.05
Path				72.22	2.05

Table 6
Estimates of delays through the critical paths of two randomized instances of a 35 nm 1-bit Adder circuit.

Component	Variation cell instance	Cell delay (ps)	Total delay (ps)
Component			
Estimated delays for randomised netlist 1			
U11	INVX2M_331	5.33	5.33
U10	OR2X2M_53	15.61	20.94
U8	NAND2X2M_200	9.03	29.97
U4	INVX2M_133	7.04	37.00
U3	OR2X2M_213	13.12	50.12
U1	NAND2X2M_471	7.01	57.13
U12	BUFX2M_130	15.43	72.56
Path delay			72.56
Estimated delays for randomised netlist 2			
U11	INVX2M_435	5.32	5.32
U10	OR2X2M_268	15.28	20.59
U8	NAND2X2M_134	9.71	30.30
U4	INVX2M_329	7.19	37.49
U3	OR2X2M_474	16.12	53.62
U1	NAND2X2M_310	7.32	60.94
U12	BUFX2M_304	15.45	76.39
Path delay			76.39

Table 7
Probability of obtaining a faster critical path delay than the estimates of randomized netlists.

	Path delay (ps)	Probability
Randomised netlist 1	72.56	0.57
Randomised netlist 2	76.39	0.98

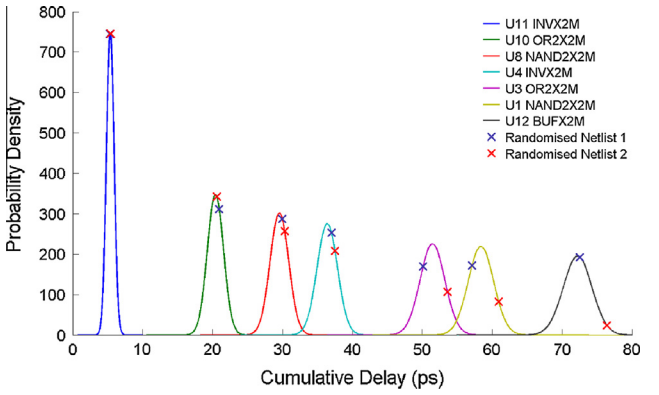


Fig. 7. The cumulation of the Gaussian models of delays through the critical path of a 35 nm 1bit Adder circuit.

Gaussian models of the standard cells were derived from a 35 nm VCL containing 500 variation cell instances of each standard cell.

The process of MCSTA is then modified to allow estimations of critical path delays to be made for each randomized netlist that is generated. Table 6 contains examples from two randomized netlists where the LUTs of each randomly selected variation cell have been read for the same output load and input transition as was used to generate the Gaussian model of the critical path delay. The individual cell delays are combined to form an estimate of the critical path delay for each randomized netlist; in this example the second netlist is estimated to be nearly 4 ps slower than the first netlist. This is an extremely simplistic way of obtaining a delay for the path, as it does not take the output transition times of the variation cells into account, nor does it use constraints, interconnect parasitics or any other detailed analysis that may be performed by an STA tool. The point of this analysis is not to provide the designer with an accurate calculation of the path delay, but instead to see where within the Gaussian model the currently selected variation instances are expected to perform. This is achieved by comparing the critical path delay for each randomized netlist against the Gaussian model of the critical path delay. In this case the Gaussian model has a mean of 72.22 ps and a standard deviation of 2.05 ps, and the cumulative distribution function for these values returns a probability of 0.5661 and 0.9792 for obtaining the predicted path delays of the first and second randomized netlists respectively, Table 7. This means that there is 57% chance of producing a faster netlist than netlist 1 by repeating the randomization process, and a 98% chance of producing a faster netlist than netlist 2. Conversely there is only a 2% chance of producing a slower randomized netlist than netlist 2, and so performing a complete STA analysis on netlist 2 will provide a much more accurate delay calculation and this may represent the slowest 2% of circuit performance under statistical process variations. These percentages are represented by the area under the curve for component U12 in Fig. 7, where the positions of the critical path delay estimates within the Gaussian models are labeled for the two randomized netlists for each cell within the path. The area to the left of an estimate represents the probability of obtaining a faster circuit, while the area to the right of the estimations represents the probability of obtaining a slower circuit. If this model holds true and the designer is only interested in the behavior of the slowest 2% of circuits then the results of STA on randomized netlist 2 may be used instead of performing full MCSTA with hundreds of samples.

It is not important for the delay estimates of the randomized netlists to match with the results of full STA analysis for this method to function effectively. It is important that there is a correlation between the position of the estimated delay within the Gaussian model, and the position of the delay calculated by STA within the distribution of delays generated by a full MCSTA run. If such a correlation exists then it would allow the designer or automated analysis tool to discard the vast majority of samples that are of little or no interest, if there is no such correlation then samples that are viewed as trivial may be discarded when in fact they represent the extremes in circuit behavior.

Two test circuits were selected for an investigation into whether such a correlation exists between the proposed Gaussian estimations and the actual distribution of delays; a one bit full adder and an ISCAS-85 benchmark four bit fast adder (c74283). These are two of the test circuits that were used within Section 2, and

were synthesized using a custom scaled 35 nm technology. The same VCL was used as within the previous experiments, containing 500 variation instances of an inverter, a buffer and two input NOR, NAND, OR and AND gates. These variation instances were again based upon multiple width 35 nm transistor models libraries within RandomSpice, where Vth0, U0, rdsw and dsub were varied with Gaussian distributions, and the means and standard deviations of the transistor model parameters were dependent upon the transistor widths.

The Monte Carlo Static Timing Analysis method described in Section 2 was adjusted to include the generation of Gaussian models as described within this section. STA was performed on nominal versions of the two test circuits, using SCLs that contained no statistical process variation data, and the critical path delays were reported. The input transitions and output loads at each stage of the critical paths were recorded from these nominal analyses, to determine which LUT elements were read within the SCL by the timing analysis tool. The distributions of delays within the VCL were then read at each of these LUT elements, so that the mean and standard deviation of each relevant cell delay could be calculated. These means and standard deviations were used to generate Gaussian models for the delays through the critical paths of the two test circuits. 50,000 randomized netlists were generated for both of the test circuits, and the variation instances within these randomized netlists were used to calculate a predicted delay and predicted probability for the critical paths. All of the randomized netlists were then passed to a commercial STA tool, allowing the predicted path delays to be compared with the actual path delays, and more importantly allowing the predicted probabilities to be compared with the shape of the measured delay distributions.

4. Results

This section provides a comparison of the critical path distributions generated by MCSTA with statistical sampling and the estimations of delays and significance made by using Gaussian models.

4.1. 35 nm 1bit Adder

The estimated delays and probabilities of the 50,000 randomized 1bit Adder netlists were collected and compared against the Gaussian model of the critical path. Fig. 8 shows the PDF of the Gaussian model, combined with the positions of the 50,000 samples within the distribution, which represents the statistical coverage of the samples used within the MCSTA run. When the CDF of the distribution is viewed, Fig. 9, the highest estimated probability is 0.99999999206 while the lowest is 0.00005544515, showing that the samples both represent the extremes of the distribution and that the center of the distribution is effectively covered.

A comparison of the distribution of measured delays against the predicted delays reveals that the measured distribution is 8% slower than the estimated distribution, showing as expected that the model used to produce the Gaussian model does not accurately replicate the STA process, Table 8. The differences in delays occur due to the absence of interpolation between LUT elements used by the generation of the Gaussian model, combined with the lack of constraints, parasitics and no propagation of transition times. Histograms of the measured and estimated critical path delay distributions are shown for comparison in Fig. 10. A scatter plot of the estimated versus the measured delays of the randomized netlists shows that although there is an average 8% error between the values, there is a strong correlation between the estimated and measured figures, Fig. 11. This means that a relatively slow estimated delay is likely to correspond to a slow measured delay, which supports the goal of using the Gaussian model for predicting which samples will produce an extreme of circuit performance.

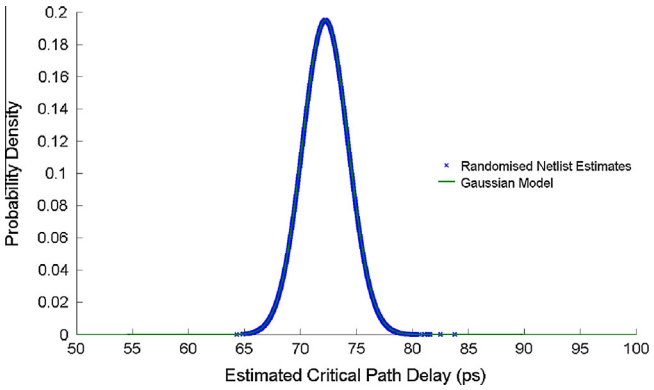


Fig. 8. The Gaussian model of the distribution of delays through the critical path of the 1bit Adder circuit.

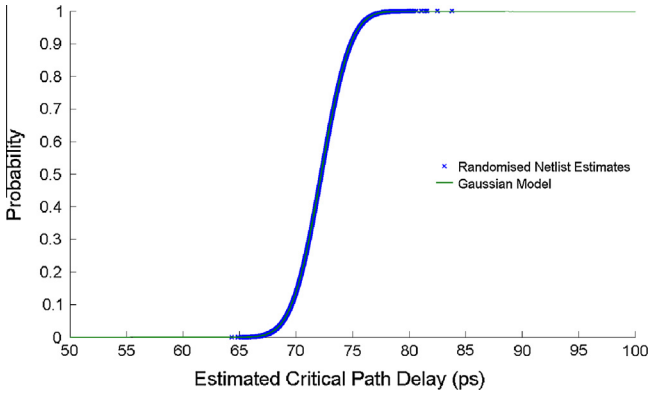


Fig. 9. CDF of the Gaussian model of delays through the critical path of the 1bit Adder circuit. 648 of the samples are estimated to be within the slowest 1% of circuit performance.

Table 8
Comparison of the estimated critical path delays with the measured path delays.

	Mean (ps)	Std Dev (ps)	Maximum (ps)	Minimum ps)
Estimated delays	72.21	2.06	83.78	64.31
Measured delays	78.23	2.12	89.80	69.92
Error (ps)	6.02	0.07	6.02	5.61
% Error	8.34	3.34	7.18	8.73

For the estimated probabilities to be useful it must be possible to select certain areas of the full MCSTA distribution that may be of interest for a designer, which is the case if the designer sets a threshold for full STA against which the estimated probabilities can be compared. Fig. 12 contains four histograms where the threshold for estimated probabilities has been adjusted, a full MCSTA run (including estimated probabilities between 0 and 1), and estimated probabilities of 0.05 (from 0 to 0.1), 0.50 (from 0.45 to 0.55) and 0.95 (from 0.90 to 1). The distributions obtained from the smaller sample sets within the estimated probability thresholds lie within the appropriate areas of the full MCSTA distribution, i.e. the lowest 10%, middle 10% and highest 10%. The shapes of these sampled distributions do not match perfectly, spilling into the regions outside of the desired thresholds, with the shortest delay from the highest 10% samples overlapping with the longest delay from the middle 10% samples. This is due to the way in which

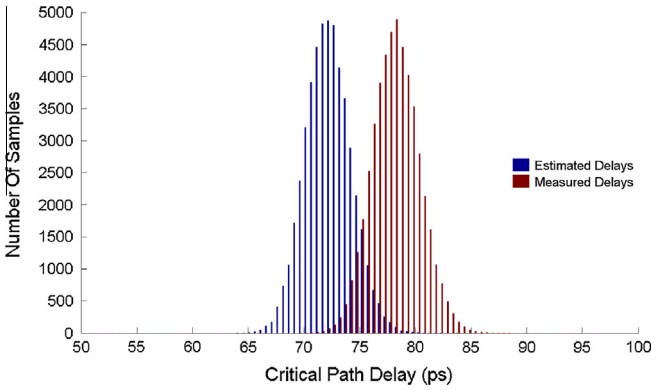


Fig. 10. Histograms of estimated and measured critical path delays for 50,000 randomized samples of the 35 nm 1bit Adder circuit. The measured delays are on average 8% slower than the predicted delays.

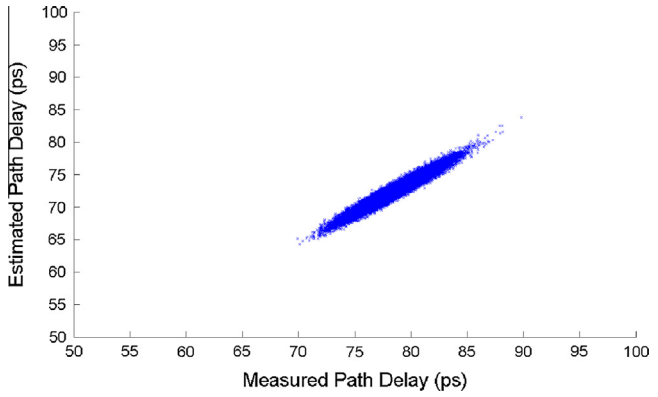


Fig. 11. A scatter plot of the estimated delay against measured delay for each randomized Adder netlists. The correlation between the estimated and measured delay shows that the Gaussian model can be used to predict extremes of behavior before STA.

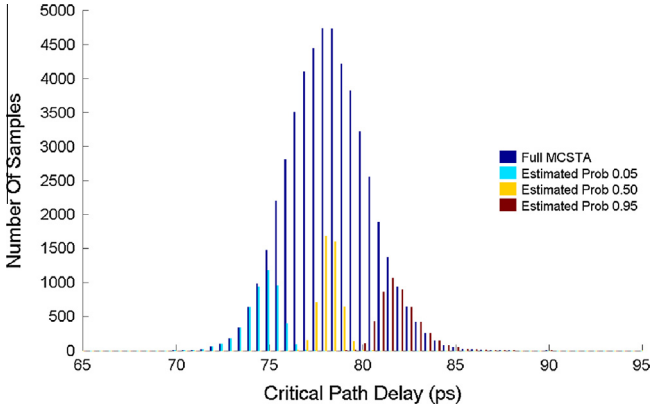


Fig. 12. Histograms of delays for randomized netlists, selected for their estimated probabilities.

the estimates are calculated, quick and inaccurate, meaning that the average value of the retained samples is of more value than either of the extremes.

A measure of the effectiveness of the probability estimations is to attempt to predict the critical path delays of the adder to the 3σ level that is often the goal of statistical analysis, where the designer may wish to be able to predict the behavior of 99.73% of the distribution of delays, derived from the fact that the mean of a Gaussian distribution plus/minus 3σ represents a 99.73% confidence interval. In the case of predicting the worst case delays of a circuit this requires predicting the behavior of 99.865% of the circuits, which is the probability of obtaining a value at plus 3σ of a Gaussian distribution. Fig. 13 contains a plot of the histogram of critical path delays from select samples, where only those randomized adder netlists whose estimated probability was greater than or equal to 0.99865 were retained; the tail of the distribution of delays generated by the 50,000 MCSTA samples are also provided as a reference. The delay with a probability of 0.99865 in the CDF for the full MCSTA distribution was found to be 85.13 ps, while the mean delay of the selected samples was found to be 85.28 ps, an relative error of 0.18%. The slowest tail of the 50,000 sample MCSTA distribution is accurately recreated by the selected samples, of which there were 126, requiring 0.25% of the computation time of the full distribution. These results are shown in Table 9, where the mean delay of the samples obtained from a range of probability thresholds is compared with the delay obtained for the same probability threshold from the CDF of the 50,000 sample MCSTA run. A small but significant point to observe from this table is that there is a 0.05% error between the full 50,000 sample MCSTA distribution and the 50,000 samples retained when the probability threshold is set from 0 to 1. This is due to the fact that the median of the distribution is not equal to the mean of the distribution, which is what is being compared in the first entry within the results table. The accuracy of sampled method remains within 1%

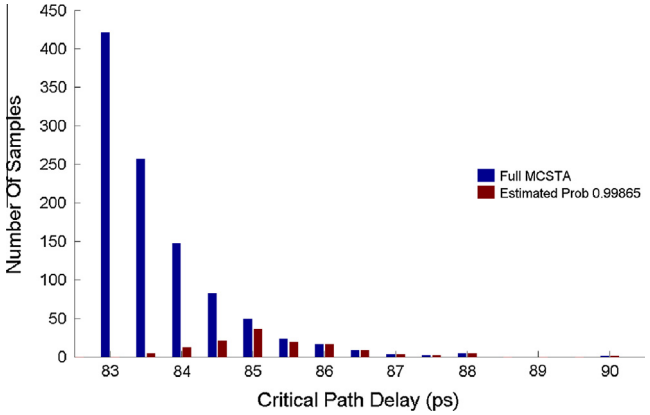


Fig. 13. Histogram of delays for randomized netlists, with an estimated probability of over 3-sigma, 0.99865. The tail of the distribution of delays from 50,000 MCSTA samples is also included.

Table 9
Comparison of critical path delays from samples selected by their estimated probability thresholds and critical path delays from the CDF of the 50,000 samples 1bit Adder MCSTA run.

Probability threshold	Full MCSTA delay (ps)	Mean sampled delay (ps)	%Error	No. samples	%CPU time
0.00000–1.00000	78.19	78.23	0.05	50000	100.00
0.00000–0.00135	72.13	71.64	0.69	42	0.08
0.00000–0.10000	74.82	74.70	0.15	4917	9.83
0.45000–0.55000	78.19	78.23	0.06	4968	9.94
0.90000–1.00000	81.80	81.95	0.18	5025	10.05
0.98000–1.00000	83.40	83.36	0.04	1191	2.38
0.99865–1.00000	85.13	85.28	0.18	126	0.25
0.99900–1.00000	85.39	85.53	0.16	97	0.19
0.99990–1.00000	87.54	86.75	0.91	23	0.05

Table 10
The number of randomized netlists generated to find 50 netlists with a probability estimate within the target threshold, and the time taken to perform STA on the 50 samples.

Probability threshold	Samples generated	Generation time (s)	50 Sample STA time (s)	Generated STA time (s)
0.000000–1.00000	50	5	62	62
0.000000–0.00135	62,364	8	67	49,392
0.000000–0.10000	579	5	65	464
0.450000–0.55000	553	4	62	443
0.900000–1.00000	366	4	61	293
0.980000–1.00000	2402	5	69	1925
0.998650–1.00000	21,270	6	63	16,846
0.999000–1.00000	24,134	6	63	19,114
0.999900–1.00000	137,862	11	68	109,187

of the whole distribution, even when the retained sample size is only 23 for a probability threshold of between 0.9999 and 1 (1 in 10,000 circuits), which requires 0.05% of the computation time required for the 50,000 samples.

The results so far have been based upon a comparison of a 50,000 sample size MCSTA run and the estimated probabilities of each of the 50,000 randomised netlists within the run. For the proposed method to be of use it must be possible for a designer to specify which probability threshold is required for examination, and to specify a limit on the number of samples that must be obtained within the threshold. For this reason the experiment was repeated with a limit on the number of samples that were retained for each probability threshold. In this case randomized netlists and probability estimations were repeatedly generated until 50 of the estimations were found to be within the target threshold, these 50 netlists were retained and STA was performed upon each one. Table 10 shows the number of randomized netlists that had to be generated to find 50 probability estimates within the same probability thresholds that were used previously. STA of the 50 retained samples required an average of 65 s, including initialization, and this run time is independent of the probability threshold that was chosen.

Smaller probability thresholds require larger numbers of randomized netlists, especially when the thresholds are set to the extremes of the distribution. The generation and probability estimations for 137,862 netlists was achieved within 11 s, saving approximately 109,000 s (30 h) that would be required to perform STA on all of the discarded netlists, rather than simply the 50 that were retained for the probability threshold of between 0.9999 and 1. This probability threshold was only represented by 23 of 50,000 randomized netlists in the previous experiment, but 50 samples were generated and analyzed within 79 s. Table 11 shows the delays obtained from each of the samples of 50 randomized netlists, compared with the delays obtained from the CDF of the 50,000 sample MCSTA run. The percentage error between the large MCSTA run and the small sample sizes remains very small, with the error only just rising above 1% for the most extreme of the probability thresholds, where twice as many netlists were generated using the probability threshold than within the 50,000 netlist MCSTA run. In this case the distribution of 50,000 delays may not accurately reflect the distribution that would have been obtained by the 137862 netlists generated to find 50 samples within the probability threshold, in which case the 50,000 samples no longer serves as a golden reference. The probability threshold that was chosen to reflect an accuracy of 3-sigma (0.99865–1) provides a mean delay that is within 0.2% of delay obtained from the 50,000 sample MCSTA run, saving over 11 h of computation time for a very acceptable loss of accuracy.

Table 11
Comparison of critical path delays from the first 50 samples to be generated within a range of estimated probability thresholds and the CDF of the 50,000 Sample MCSTA run.

Probability threshold	Full MCSTA delay (ps)	Mean sampled delay (ps)	%Error
0.000000–1.00000	78.19	78.68	0.63
0.000000–0.00135	72.13	71.65	0.67
0.000000–0.10000	74.82	74.61	0.28
0.450000–0.55000	78.19	78.19	0.00
0.900000–1.00000	81.80	81.87	0.10
0.980000–1.00000	83.40	83.37	0.03
0.998650–1.00000	85.13	85.31	0.20
0.999000–1.00000	85.39	85.49	0.11
0.999900–1.00000	87.54	86.6	1.09

4.2. 35 nm 4bit Fast Adder, c74283

Gaussian models were generated for the critical path through the 4bit Fast Adder circuit, with an estimated mean of 75.66 ps and estimated standard deviation of 2.06 ps. STA was performed on 50,000 randomized netlists of the 4bit Fast Adder using a commercial STA tool. The sample size was found to be large enough to generate estimated delays that covered the entire range of the Gaussian distribution, Fig. 14, but the mean and spread of the measured delays were found to be considerably larger than the estimates. There was a 26% increase in the mean delay and a 10% increase in standard deviation when compared to the estimates, providing no overlap between the range of the Gaussian model and the measured distribution of delays, Table 12. This can be more clearly seen from a plot of histograms taken of the distributions of estimated and measured delays Fig. 15.

A scatter plot of the estimated probabilities against the measured probabilities indicates that there is a weaker correlation between the probabilities than was found in the results obtained from the 1bit Adder, Fig. 16. The increase in error between the estimates and the measured results after STA is an indication that the Gaussian approach to statistically modeling critical path delays becomes weaker with longer logic paths, but that it is still possible to

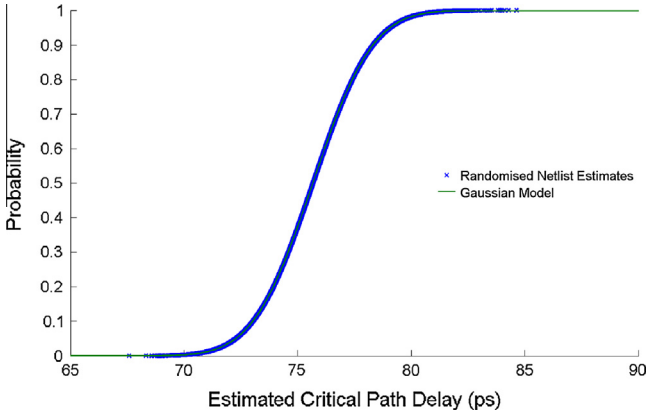


Fig. 14. CDF of the Gaussian model of delays through the critical path of the Fast Adder. 595 of the samples are estimated to be within the slowest 1% of circuit performance.

Table 12
Comparison of the Gaussian estimations of critical path delays with the 50,000 sampled and measured path delays, for the 35 nm Fast Adder.

	Mean (ps)	Std Dev (ps)	Maximum (ps)	Minimum (ps)
Estimated delays	75.66	2.06	84.64	67.57
Measured delays	95.56	2.28	106.20	86.29
Error (ps)	19.90	0.21	21.56	18.71
% Error	26.3	10.28	25.47	27.69

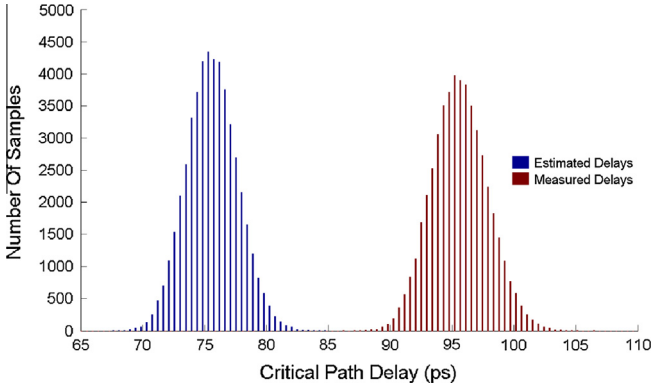


Fig. 15. Histograms of estimated and measured critical path delays for 50,000 randomised samples of the 35 nm Fast Adder circuit. The measured delays are on average 26% slower than the predicted delays.

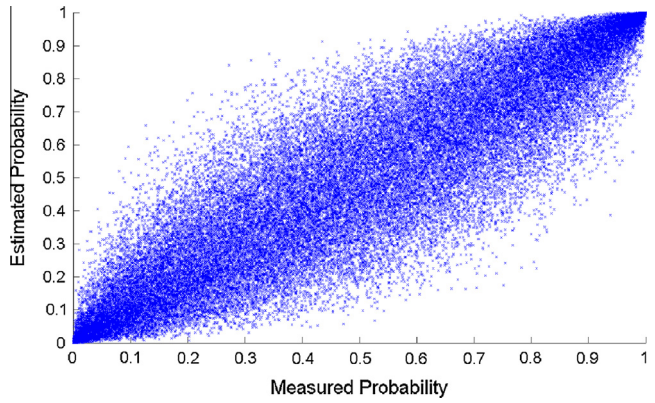


Fig. 16. Scatter plot of estimated probabilities against measured probabilities for the Fast Adder shows weak correlation in the center of the distribution, but stronger correlations at the extremes.

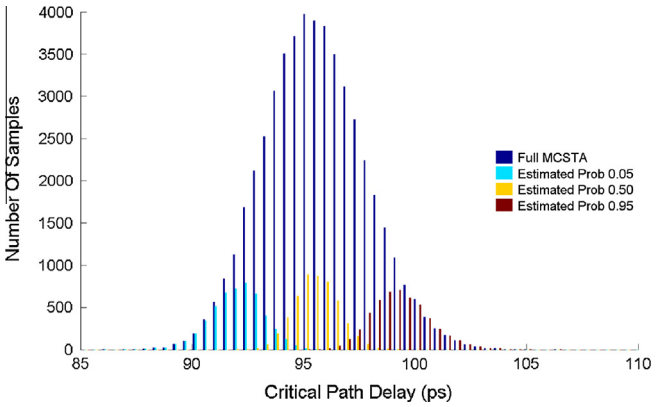


Fig. 17. Histograms of delays for randomized netlists of the Fast Adder that were selected for their estimated probabilities.

Table 13
Comparison between critical path delays from samples selected by their estimated probability thresholds and critical path delays from the CDF of the 50,000 sample Fast Adder MCSTA run.

Probability threshold	Full MCSTA delay (ps)	Mean sampled delay (ps)	%Error	No. samples
0.000000–1.00000	95.5	95.56	0.06	50000
0.000000–0.00135	89.11	88.92	0.22	47
0.000000–0.10000	91.91	92.03	0.13	4959
0.450000–0.55000	95.5	95.56	0.06	4996
0.900000–1.00000	99.39	99.32	0.07	5009
0.980000–1.00000	101.06	100.71	0.35	1138
0.998650–1.00000	102.75	102.46	0.29	109
0.999000–1.00000	102.98	102.77	0.21	73
0.999900–1.00000	104.19	104.01	0.17	11

Table 14
Comparison of critical path delays from the first 50 samples to be generated within a range of estimated probability thresholds and the CDF of the 50,000 sample Fast Adder MCSTA run.

Probability threshold	Full MCSTA delay (ps)	Mean sampled delay (ps)	%Error	No. samples
0.000000–1.00000	95.5	95.54	0.04	50
0.000000–0.00135	89.11	88.96	0.17	50
0.000000–0.10000	91.91	92.07	0.17	50
0.450000–0.55000	95.5	95.81	0.32	50
0.900000–1.00000	99.39	99.4	0.01	50
0.980000–1.00000	101.06	100.89	0.17	50
0.998650–1.00000	102.75	102.54	0.20	50
0.999000–1.00000	102.98	102.66	0.31	50
0.999900–1.00000	104.19	104.17	0.01	50

perform basic statistical sampling. The probability estimations can still be reliably used to predict circuit performance within ranges of desired probability thresholds, such as the slowest, fastest and nominal 10%, Fig. 17. A range of desired probability thresholds were chosen and were used to asses the validity of the Gaussian modelling process, and it was found that the mean delays of the circuits within the threshold bands matched the delay at the appropriate point within the 50,000 sample CDF to within 0.35%, and less than 0.3% for a prediction of 3-sigma circuit performance, Table 13.

The ability to predict the performance of a circuit without prior knowledge is much more important than being able to reproduce a distribution of delays by using sub-samples of the distribution. It is therefore vital to be able to achieve similar results using independently generated random samples. The experiment was altered so that any randomized netlist with an estimated probability outside of the desired threshold was discarded, and only the first 50 randomized netlists that were found within the threshold were retained and analyzed in STA. The results of using these sets of 50 samples are given in Table 14, where the full 50,000 sample MCSTA run of the 35 nm Fast Adder are retained as a reference against which smaller independent samples are be compared. These results indicate that a statistically sampled MCSTA run of just 50 samples can be used to replicate the targeted area of a much larger full MCSTA run to within one third of a percent. This method avoids the pessimism of corner analysis while dramatically reducing the computation time required to produce accurate delay information at the extremes of circuit behavior.

Table 15
Percentage error in the average delay from a range of sample sizes for a series of probability thresholds. The percentage errors are with respect to the delay from 50,000 samples.

Number of samples	Probability threshold				
	0.00000–0.10000 (%)	0.45000–0.55000 (%)	0.90000–1.00000 (%)	0.99865–1.00000 (%)	0.99990–1.00000 (%)
5	0.85	0.26	0.16	0.04	0.37
10	0.11	0.47	0.30	0.27	0.41
15	0.20	0.38	0.25	0.17	0.43
20	0.25	0.35	0.26	0.31	0.33
25	0.30	0.29	0.18	0.28	0.27
50	0.17	0.32	0.01	0.20	0.01

4.3. Sample size reduction

The ability to accurately predict the behavior of a specific percentage of circuits will depend upon the probability thresholds and sample sizes specified by the designer. It was therefore important to assess the impact of the sample size and probability threshold selection on the accuracy of the statistically samples MCSTA run. The MCSTA experiment with 50,000 randomized samples of the Fast Adder was repeated, selecting the first N randomized netlists that were predicted to fall within a chosen probability threshold, where N is the statistical sample size. The average delay obtained from the N retained randomized netlists was then compared against the delay obtained from the distribution of 50,000 netlists at the relevant position within the CDF. The results of this experiment are shown within Table 15, where the percentage error was never found to be over 1% of the delay obtained from a large MCSTA run of 50,000 samples. This suggests that a sample size of as little as 5–10 can be used to predict the behavior of a much larger sample set, by using a simple Gaussian statistical model as a predictor of which samples to select.

5. Conclusions

MCSTA produces predictions of circuit delay with a greater accuracy than SSTA or traditional corner based analysis, when compared to transistor level SPICE simulations. Compared with SSTA, however, MCSTA is slow and not suited to practical use. In this paper, we have described a statistical sampling method that significantly reduces the number of samples needed by a Monte Carlo simulation to evaluate the behavior in the tails of distributions. Hence we have demonstrated a method that has the accuracy close to that of full Monte Carlo SPICE simulations, but with the speed advantage of SSTA.

The method is best suited to path-based STA, in which the path delay has an approximately Gaussian distribution. We will consider block-based STA, where the arrival time delay is less well modeled by a Gaussian distribution, in future work.

References

[1] International Technology Roadmap for Semiconductors, ITRS 2009. edition, 2009. <http://www.itrs.net/Links/2009ITRS/Home2009.htm>.
[2] Berkelaar M. Statistical delay calculation, a linear time method. Proc TAU 1997:15–24.
[3] Blaauw D, Chopra K, Srivastava A, Scheffer L. Statistical timing analysis: from basic principles to state of the art. IEEE Trans Computer-Aided Des Integrated Circuits Syst 2008;27(4):589–607. http://dx.doi.org/10.1109/TCAD.2007.907047. URL 10.1109/TCAD.2007.907047.
[4] Tang X, De V, Meindl J. Intrinsic MOSFET parameter fluctuations due to random dopant placement. IEEE Trans Very Large Scale Integration (VLSI) Syst 1997;5(4):369–76. http://dx.doi.org/10.1109/92.645063. URL 10.1109/92.645063.
[5] Asenov A, Kaya S, Brown A. Intrinsic parameter fluctuations in decananometer MOSFETs introduced by gate line edge roughness. IEEE Trans Electron Dev 2003;50(5):1254–60. http://dx.doi.org/10.1109/TED.2003.813457.
[6] Thiault J, Foucher J, Tortai JH, Joubert O, Landis S, Pauliac S. Line edge roughness characterization with a three-dimensional atomic force microscope: Transfer during gate patterning processes. J Vacuum Sci Technol B: Microelectron Nanometer Struct. http://dx.doi.org/10.1116/1.2101789.
[7] Merrett M, Asenov P, Wang Y, Zwolinski M, Reid D, Millar C. Modelling circuit performance variations due to statistical variability: Monte carlo static timing analysis. In: Design, automation & test in Europe conference & exhibition (DATE); 2011. p. 14.
[8] Roy G, Adamu-Lema F, Brown A, Roy S, Asenov A. Simulation of combined sources of intrinsic parameter fluctuations in a 'real' 35 nm MOSFET. In: Solid-state device research conference, 2005. ESSDERC 2005. Proceedings of 35th European; 2005. p. 337–40. http://dx.doi.org/10.1109/ESSDER.2005.1546654.
[9] Wang X, Cheng B, Roy S, Asenov A. Simulation of strain enhanced variability in nMOSFETs. In: 9th international conference on ultimate integration of silicon, 2008. ULIS 2008; 2008. p. 89–92. http://dx.doi.org/10.1109/ULIS.2008.4527147.
[10] Xiong S, Bokor J. A simulation study of gate line edge roughness effects on doping profiles of short-channel MOSFET devices. IEEE Trans Electron Dev 2004;51(2):228–32. http://dx.doi.org/10.1109/TED.2003.821563.
[11] Sheu B, Scharfetter D, Ko P, Jeng M. BSIM: berkeley short-channel IGFET model for MOS transistors. IEEE J Solid-State Circuits 1987;22(4):558–66.
[12] Gold Standard Simulations Ltd., RandomSpice. <http://www.goldstandardsimulations.com/services/circuit-simulation/random-spice/>.
[13] Wang Y, Merrett M, Zwolinski M. Statistical power analysis for nanoscale CMOS. In: 2010 International conference on signals and electronic systems (ICSES). IEEE; 2010. p. 201–4.
[14] Singhee A, Rutenbar R. Statistical blockade: very fast statistical simulation and modeling of rare circuit events and its application to memory design. IEEE Trans Computer-Aided Des Integrated Circ Syst 2009;28(8):1176–89. http://dx.doi.org/10.1109/TCAD.2009.2020721.