Block-based Static Timing Analysis with Uncertainty

Anirudh Devgan
IBM Research
Austin, TX 78758
devgan@us.ibm.com

Chandramouli Kashyap
IBM Microelectronics
Austin, TX 78758
vchandra@us.ibm.com

Abstract

Static timing analysis is a critical step in design of any digital integrated circuit. Technology and design trends have led to significant increase in environmental and process variations which need to be incorporated in static timing analysis. This paper presents a new, efficient and accurate block-based static timing analysis technique considering uncertainty. This new method is more efficient as its models arrival times as cumulative density functions (CDFs) and delays as probability functions (PDFs). Computationally simple expression are presented for basic static timing operations. The techniques are valid for any form of the probability distribution, though the use piecewise linear modeling of CDFs is highlighted in this paper. Reconvergent fanouts are handled using a new technique that avoids path tracing. Variable accuracy timing analysis can be performed by varying the modeling accuracy of the piecewise linear model. Regular and statistical timing on different parts of the circuit can be incorporated into a single timing analysis run. Accuracy and efficiency of the proposed method is demonstrated for various ISCAS benchmark circuits.

1. Introduction

Static timing analysis (STA) is critical to the measurement and optimization of the circuit performance before its manufacture. Full chip static timing analysis is usually performed using efficient block-based techniques. A block-based approach allows incremental, embedded static timing analysis and therefore enables timing-driven flows in logic synthesis and physical design. Hence, block-based static timing analysis has emerged as one of the key technologies in current design methodologies.

The timing or performance of the chip is heavily dependent on the manufacturing process variations (e.g. Vt, Length, etc.) and design environment variations (e.g. VDD & temperature variations, noise impact on timing, etc.). As the feature sizes decrease, the ability to control the manufacturing spread or accuracy of a given feature size is also decreasing. Along with increased process variations, the uncertainty caused by design is also increasing. The increase of uncertainty in design is caused by increase of power supply and temperature variations and interconnect loading uncertainty such as coupling noise impact on timing. Another source of uncertainty is the inherent error in the gate delay models, also called the model-to-hardware correlation error. It is critical that these increased timing uncertainties be handled in the design process in an efficient and accurate manner. Given the pervasive nature of static timing, it is essential that a variation-aware static timing approach be suitable for full chip designs.

Design variations or uncertainty in static timing analysis is typically handled in two broad ways. The first set of techniques handle variations by worst casing the circuit response. In such a scenario, static timing is performed at various design corners (e.g. fast, slow and nominal design corner). For example, the fast corner is computed by placing all the gates (or transistors) at the fast corner and performing a regular deterministic timing analysis. The timing results of the fast, slow and nominal corners can also be combined to minimize the typically large error of worst-case analysis. This approach is computationally attractive but can be inaccurate due to its worst-case nature. The worst-case approach has traditionally been used for industrial designs but it is becoming inapplicable as the timing variations continue to increase. Furthermore, to account for intra-chip or local variations, these techniques scale the data and clock path delays differently using empirical factors.

Another method to handle variations in timing is to perform statistical timing analysis. Statistical static timing analysis has been studied over the years [4-10]. Reconvergent fanouts have caused several statistical timing analysis approaches to have exponential complexity. Efficient approximate techniques for reconvergent fanout are not addressed in these techniques. Block-based approaches to statistical STA have been proposed in [11],[12] and [16]. In [11], the delays of the gates and arrival times are modeled as independent discrete random variables. Reconvergent fanouts are not considered. False path analysis using this basic framework is considered in [12]. A new approach described in [16] proposes a technique which computes both upper and lower bounds to the exact solution, in the presence of reconvergent fanouts. Further, they show that statistical STA performed without accounting for reconvergent fanouts is an upper bound on the actual delay. However, their method of enumerating selected nodes to obtain improved bounds may be cumbersome for large circuits, and has exponential runtime in the worst case. A further drawback of these block-based approaches is that they model both gate delays and arrival times as discrete probability density functions or PDFs. This involves propagating impulse trains across the circuit and taking the statistical maximum of two arrival times, a fundamental operation in STA, becomes inefficient. As we show in this paper, modeling the arrival times as cumulative distribution functions or CDFs is more efficient.

In contrast to the above block-based methods, a path-based approach has been proposed in [13]. Each path delay is modeled as a sum of individual gate delays

ICCAD'03, November 11-13, 2003, San Jose, California, USA.

with each gate delay being a function of random variables. By assuming small deviations from the nominal value, a linear statistical model for gate delay is constructed that can be efficiently summed up to get the path delay. However, to get the circuit delay a statistical maximum is performed assuming independent paths which is not true in general. Another drawback of path-based approaches is that a large number of paths is needed to be examined and can be exponential in the worst-case. Further, path-based approaches are not amenable to incremental STA, a necessary requirement in the synthesis and optimization of designs. Reference [15] describes how to perform circuit optimization in presence of uncertainties by considering the large number of equally critical paths; however, it does not describe any technique for performing statistical STA.

This paper presents a new block-based statistical timing analysis technique. The delay and arrival times in the circuit are modeled as random variables. The arrival times are modeled as Cumulative Probability Distribution Functions (CDFs) and the gate delays are modeled as Probability Density Functions (PDFs). This leads to efficient expressions for both max and addition operations, the two key functions in both regular and statistical timing analysis. Although the proposed approach can handle any form of the CDF, in this paper the CDFs are modeled as piecewise linear for computational efficiency. Waveshape of arrival time CDFs are similar to voltage waveforms in deterministic timing analysis. The modeling of arrival time CDFs as piecewise linear is consistent with modeling the voltage waveforms in regular timing as piecewise linear. The accuracy of any CDF can be varied by varying the number of segments in the piecewise linear approximation. Typically, only a few points are sufficient to obtain the desired accuracy. Parts of the circuit (or gates) can be modeled as deterministic. The deterministic part of the circuit is modeled as step CDFs (or impulse PDFs). Piecewise linear CDFs imply piecewise constant PDFs for the gate delays.

Dependency in statistical timing analysis can come from two type of sources. The first source is reconvergent fanout due to the circuit toplogy. The second source of dependency is the manufacturing process parameters. The gates of the circuit which depend on same or similar process parameters cause correlation in delay and arrival times. This paper addresses the dependency caused by reconvergent fanout, which is a necessary first step in a statistical STA framework. Reconvergent fanouts are efficiently handled by a novel common mode removal approach using the idea of a statistical "subtraction" as opposed to expensive path-tracing commonly used in the literature.

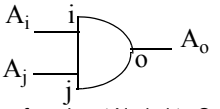
For simplicity, the discussion assumes a late-mode STA, however, the proposed method can be easily applied to early-mode STA as well. The remaining part of the paper is organized as follows. Section 2 describes the basics of the new statistical timing analysis approach. Techniques to account for reconvergent fanout are presented in Section 3. Section 4 presents results for various ISCAS benchmark circuits followed by conclusions and future work in Section 5.

2. Statistical Timing Analysis

The problem in deterministic static timing analysis is to compute arrival times at the output nodes. Using this the slack and hence the critical path of the circuit are determined. Arrival times at the input and delay of the gates are specified as deterministic numbers. In case of statistical timing analysis, the arrival times and delays of the gates are specified as distributions. In general, the distribution of delays of the gates can take any form (i.e. normal, uniform, etc.). The problem in statistical timing analysis is to compute distribution of arrival times at the intermediate nodes and the output nodes. Given the required arrival time and distribution of output arrival times, critical paths and slack distributions can be computed for a given probability or confidence level.

Timing analysis is performed by levelizing the circuit. The arrival time at the input is propagated through the gates at each level till it reaches the output. Propagating the arrival times through a gate is a key function in static timing.

Consider a two input gate shown in Figure 1.



D_{io} : Delay from Input Node i to Output node o

D_{io} : Delay from Input Node j to Output node o

Figure 1 A gate with output o and inputs i and j.

In deterministic static timing analysis, arrival time A_o at output node o is given by:

$$A_o = max(A_i + D_{io}, A_j + D_{jo}) \quad (1)$$

Computation of max and addition is straight forward in regular timing analysis. We now define these operations in statistical timing analysis. In the proposed approach arrival times are modeled as cumulative density functions (CDFs) and the delays are modeled as probability density functions (PDFs).

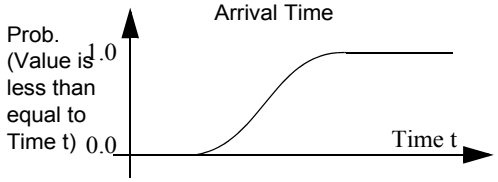


Figure 2 Arrival times are modeled as Cumulative Probability Density Functions (CDFs).

Figure 2 illustrates the modeling of arrival times with CDFs. The first non-zero value of arrival time which has a non-zero cumulative probability is the lower bound on the arrival time. On the other hand, the value of arrival where the cumulative probability reaches 1.0 is the upper bound on the arrival time. Arrival times in regular deterministic static timing can be viewed as a step function (as shown in Figure 3).

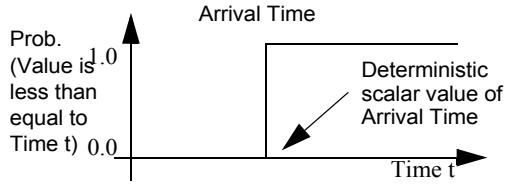


Figure 3 Arrival times in regular deterministic static timing can be viewed as step Cumulative Probability Density Functions (CDFs).

The cumulative density functions are modeled as piecewise linear. Figure 4 shows the approximation of a cumulative density function with a three point piecewise linear function. The accuracy of the CDF modeling can be increased by increasing the number of points (or segment) in the piecewise linear model. Later on in the paper, the results are presented for three point, five point, and seven point piecewise linear model.

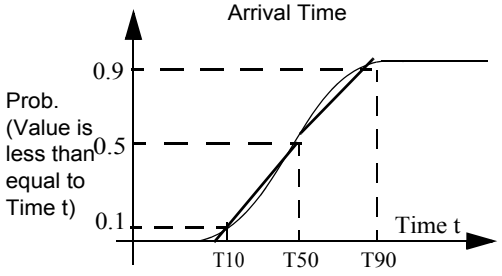


Figure 4 Three point piecewise linear (PWL) modeling of the cumulative density function (CDF).

Piecewise linear modeling of CDFs lead to probability density functions (PDFs) being piecewise constant. The PDFs are not continuous, though their CDFs are continuous. The PDFs being non-continuous is not an issue, as long as CDFs of gate delays are continuous, CDFs of all the arrival times are also continuous. Previous techniques[11][16] have used impulse train modeling for PDFs (or piecewise constant CDFs). Piecewise linear CDFs (or piecewise constant PDFs) used in this paper are much more efficient representation of the distributions as compared to piecewise constant CDFs (or impulse train PDFs) as piecewise linear CDFs require much lower number of segments for the same accuracy.

At any node i in the timing graph, $C_i(t)$ is defined as the notation for its CDF. For delay between node i and node j, $P_{ij}(t)$ is defined as the PDF and $C_{ij}(t)$ is defined as the CDF. From the definition of CDF and PDF, $P_{ij}(t)$ is the derivative of $C_{ij}(t)$. Using this notation the max and addition operations are defined next. For now, these expressions assume that the variables (arrival time, delays) are independent. Techniques to handle interdependence due to reconvergent fanouts will be presented later in Section 3.

Addition Operation:

Let D_{ij} be the delay between node i and node j. Or

$$A_j = D_{ij} + A_i \tag{2}$$

The sum of two random numbers is convolution of their probability functions. The CDF of delay at node j (A_j is given by the convolution of $C_i(t)$ (i.e. CDF of A_i) with $P_{ij}(t)$ (i.e. PDF of D_{ij}). Or

$$C_j = \int_0^t C_i(t-\tau)P_{ij}(\tau)d\tau \tag{3}$$

Max Operation:

Let A_o be the max of arrival time as node a(A_a) and arrival time at node b(A_b). That is,

$$A_o = max(A_a, A_b) \tag{4}$$

The CDF at node o can simply be given by

$$C_o(t) = C_a(t)C_b(t) \tag{5}$$

That is, the CDF of the maximum of two independent random variables is simply the product of the CDF of the two variables. This simple expression is only possible if the variables are modeled as CDFs. If the variables are modeled as PDFs the max is significantly more complicated and which is why we model the arrival times as CDFs. Let

$$A_o = max(A_a, A_b) \tag{6}$$

The CDF of A_o is given by probability of $A_o \le t$. If A_a and A_b are independent, the probability of both A_a and A_b being less than t can be computed by the multiplication of CDF of arrival time at node a ($C_a(t)$) with the CDF of arrival time at node b ($C_b(t)$), giving equation (5).

Once the expression for max and addition have been defined, statistical timing can be performed just like regular timing. The difference is that when addition of arrival time A_i is needed with delay D_{io} a convolution of C_i is performed with P_{io} instead of an algebraic add. And when max of two variables is needed, their CDFs are multiplied to yield the resultant CDF. Hence, the fundamental static timing operation in equation (1) is modified as follows in the proposed statistical timing analysis:

$$A_o = max(A_i + D_{io}, A_j + D_{jo}) \tag{7}$$

$$C_o = (C_i \otimes P_{io})(C_j \otimes P_{jo}) \tag{8}$$

where C_o is the CDF of arrival time at node o (A_o).

$C_i \otimes P_{io}$ is the convolution of CDF of A_i with PDF of D_{io} and $C_j \otimes P_{jo}$ is the convolution of CDF of A_j with PDF of D_{jo} as defined in Equation (3). $(C_i \otimes P_{io})(C_j \otimes P_{jo})$ denotes the multiplication of two resultant CDFs to get the max.

Statistical timing analysis can be added to a regular static timing engine by replacing the fundamental max and add operation in Equation (1) by the one in Equation (8). Circuit parsing and setup, timing graph construction, graph traversal and incremental capabilities of regular timing can be used as is in statistical timing analysis. Techniques described in this section can be used for any model or form (i.e. normal, uniform, measured, etc.) of the variations (or the CDF). Since we use piecewise linear models for CDF, in the next section we show how the add and multiplication is performed in this framework.

2.1 Max and Addition with PWL modeling

As mentioned previously, convolution is performed to add the input arrival time - which is modeled as a piecewise linear CDF - and the gate delay - which is modeled as a piecewise constant PDF. We break the CDF into a sum of ramps and the PDF into a sum of step signals. For instance, a 3-piece CDF and a 4-piece PDF are decomposed as shown in Figure 5.

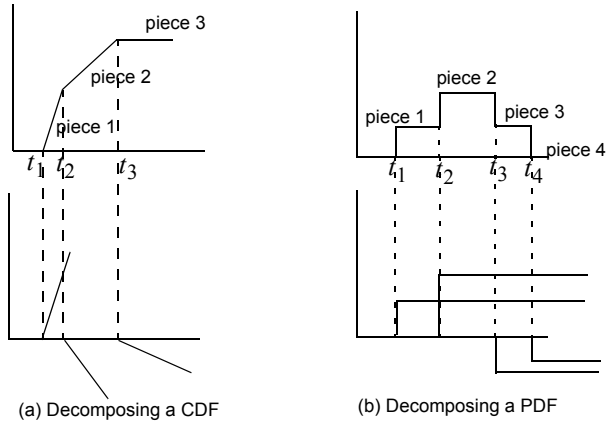


Figure 5 Decomposing a CDF and a PDF into ramps and steps

Each ramp and each step waveform are convolved individually as shown in Figure 6. Here s is the slope of the ramp. The resultant waveform is a quadratic. For an n-piece CDF and an n-piece PDF, a total of n^2 convolutions are performed. The n^2 quadratics are then summed together to obtain the resultant CDF. Finally, for forward propagation the quadratic CDF is converted back to a piecewise linear CDF by sampling at the preset probability values. No nonlinear iterations are required for the sampling since closed-form formulas for the crossing times can be obtained since the result of convolution is a quadratic.

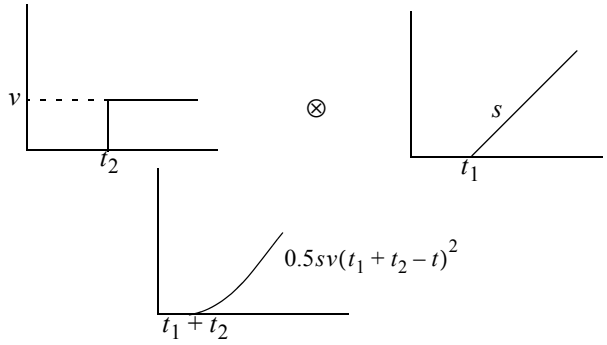


Figure 6 Convolution of a step and a ramp

The multiplication of two piecewise linear CDFs proceeds analogously. We decompose the piecewise linear CDF as a sum of ramps as shown in Figure 5(a). Each ramp of the first CDF is multiplied by each ramp of the second CDF as shown in Figure 7. The resultant waveform is again a quadratic. For n pieces in each of the two CDFs, n^2 quadratic pieces are produced due to multiplication. These are then summed up to get the resultant CDF. As before, the CDF is sampled at the preset probability values to get the piecewise linear CDF for forward propagation.

2.2 Time Complexity of the Proposed Method

It is well known that block-based deterministic timing analysis can be performed in $O(E+V)$ where E and V are the number of edges and vertices in the timing graph respectively. In the statistical case, convolution is performed for each edge in the timing graph and the multiplication is performed at each vertex for all the incident edges in the timing graph. Since each of these operations takes $O(n^2)$ time for an n-piece CDF/PDF model, the overall complexity of our approach is $O(n^2E + V)$.

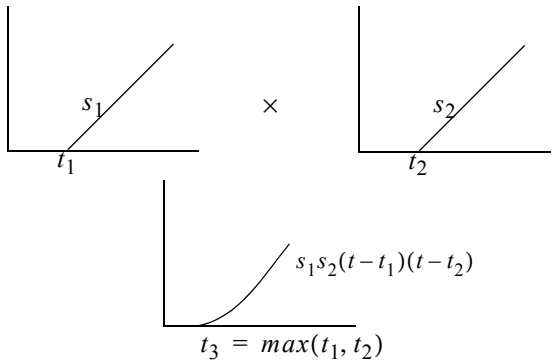


Figure 7 Multiplication of two ramps

3. Handling Reconvergent Fanouts

The complexity of statistical timing analysis usually increases due to reconvergent fanounts. In this section, a new technique is presented to capture reconvergent fanouts in the proposed statistical timing

framework. We illustrate the basic principle behind our approach through the circuit in Figure 8. In this example, two paths originating from node r reconverge as inputs to the same gate at nodes i and j. This causes both the arrival time A_i and A_j to depend on arrival time A_r. This dependency potentially complicates the computation of arrival time A_o since equation (5) can no longer be used.

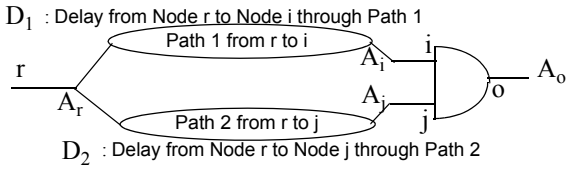


Figure 8 Example of reconvergent fanout. Two fanouts from node r reconverge downstream as inputs i and j of the same gate.

It should be noted that the interdependence of arrival time A_i and A_j has a very specific linear form. That is

$$A_i = A_r + D_1 \tag{9}$$

$$A_j = A_r + D_2 \tag{10}$$

The variable of interest, arrival time at node o, A_o, is given by

$$A_o = max(A_r + D_1 + D_{io}, A_r + D_2 + D_{jo}) \tag{11}$$

The computation in Equation (11) can be exactly rewritten as

$$A_o = A_r + max(D_1 + D_{io}, D_2 + D_{jo}) \tag{12}$$

The expression is simplified by taking out the common mode A_r. Since D_1, D_2, D_{io} and D_{jo} are independent, simple expression derived in Section 2 can be used. In other words, the CDF at node o can simply be rewritten as

$$C_o = C_r \otimes [(C_1 \otimes D_{io})(C_2 \otimes D_{jo})]' \tag{13}$$

where C_r is the CDF of arrival time at node r, C_1 is the CDF of delay D_1, C_2 is the CDF of delay D_2 and $\otimes$ is the convolution operator defined in Equation (3). Note that the expression within the square bracket is a CDF. Therefore, the dash at the end denotes the derivative (which yields the PDF) of the expression within the square brackets. This is required so that convolution with C_r produces the right CDF.

To compute A_o, the computation of D_1 and D_2 is required. One way is to perform path tracing to get these values. However, it is simpler and more desirable to compute D_1 and D_2 through statistical subtraction. i.e.

$$D_1 = A_i - A_r \tag{14}$$

$$D_2 = A_j - A_r \tag{15}$$

The statistical subtraction is equivalent to inverse of convolution, and one way to do this is by moment matching. Consider the case

$$A_z = A_x - A_y \tag{16}$$

where A_x and A_y are known and A_z is the unknown. This can be rewritten as

$$C_x = C_z \otimes P_y \tag{17}$$

$$P_x = P_z \otimes P_y \tag{18}$$

Since z and y are independent, their mean and variances will add in a convolution. That is,

$$\mu_x = \mu_z + \mu_y \tag{19}$$

$$\sigma_x^2 = \sigma_z^2 + \sigma_y^2 \tag{20}$$

Since P_x, C_x, P_y and C_y are known, their mean and variances can be computed from their PDFs (or their piecewise linear CDF). Hence, the required mean and variance of C_z can be computed from algebraic subtraction, or

$$\mu_z = \mu_x - \mu_y \tag{21}$$

$$\sigma_z^2 = \sigma_x^2 - \sigma_y^2 \tag{22}$$

Once the mean μ_z and variance σ_z^2 is computed, the CDF, C_z, can be determined by fitting the mean and variance to a probability distribution. Two moments (mean and variance) are matched to determine the distribution. This method can also be extended by matching higher order moments and performing Pade approximation to determine the CDF.

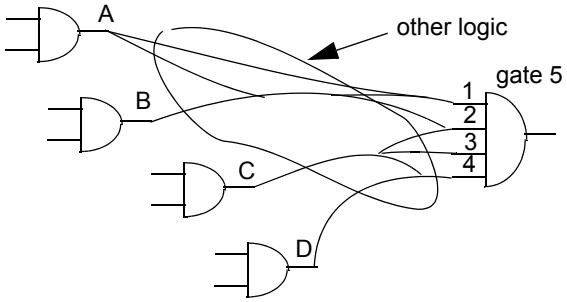


Figure 9 A general case of reconvergent fanout

In general, an input of a gate may depend on more than one previous node. For example, in Figure 9, the inputs of the gate 5 depends on A, B, C and D. Some of these vertices may also share subpaths when reaching the inputs of gate 5. Therefore, when computing the arrival time at the output of gate 5, this dependency must be accounted for. To accomplish this, we maintain a Dependency List (DL) with each vertex in the timing graph which lists the vertices on which the arrival time of the current vertex depends. The vertices are sorted by the level in a descending order i.e. the most recent vertex (i.e. the one with the highest level) appears first and so on. The DL is propagated as we compute the statistical arrival times using the *DLPropagate* algorithm shown in Figure 10. In the algorithm DL_i denotes the DL of the ith input and DL_o denotes the DL of the output node. If an

input does not contribute to the output of the gate (for example, it may be 1 well before any other input arrives), its DL is not propagated to th output. In addition, we use two other pruning heuristics to limit the size of the DL. We allow the user to specify the size of the list. In addition, we only carry-forward the n most recent (i.e. level) vertices where n is again set by the user. Thus, we know all the prior vertices that impact the arrival time of a given vertex *without any path tracing.*

Algorithm DL Propagate
DL_o = NULL
if gate output has fanout > 1
  add output node to DL_o
for each input i of gate contributing to output
  for each node v in DL_i and not in DL_o
    add v in DL_o using insertion sort
    in descending order by level
return DL_o

Figure 10 Algorithm to propagate dependency list

Now, suppose we wish to compute the arrival time at the output of a multi-input gate, with each input having possibly a non-empty dependency list. An approximate algorithm is given in Figure 11. The key idea is to reduce the dependency of each input to a single vertex so that (13) may be applied. In the algorithm, A_o is the output arrival time, A_i is the arrival time at the i th input and A_v is the arrival time at vertex v.

Algorithm depMax
$A_o = -\infty$
L=NULL
for each input i
  for each vertex v in DL_i
    if (v covers more than one input)
      &&(v does not appear in L)
        insert v according to level in L
        mark inputs that v covers
  if (L is empty)
    proceed as in independent case
  else
    for each v in L
    {
      $A_{ov} = -\infty$
      for each input i that it covers
        $A_{ov} = max(A_{ov}, A_i - A_v + D_{io})$
      $A_o = max(A_o, A_{ov})$
}
  return A_o

Figure 11 Algrithm to compute output arrival time in presence of reconvergent fanout

Once a dependent max is computed at the output of a gate, th dependency lists of the inputs is not propagated forward.

For example, referring to Figure 9, the output of gate 5 depends on A, B, C and D. However, *depMax* will identify that inputs 1 and 2 depend on B (since B is at a higher level than A) and that inputs 3 and 4 depend on C. Thus the arrival times due to 1 and 2 (as well as 3 and 4) will be computed using (13) and the two resultant arrival times will be treated as independent and combined using (5) to get the output arrival time. The dependency lists of the inputs to gate 5 will not be carried forward.

4. Results

The proposed block-based static timing analysis approach in presence of uncertainty has been implemented and its results are presented for various ISCAS benchmark circuits. The ISCAS circuit have been mapped using a commercial logic synthesis system to a recent library consisting of gates with maximum of four fanins. The cell library consists of the following gates: inverter, 2-input nand, 2-input nor, 2-input and, 2-input or and 4-input nand, 4-input nor, 4-input and 4-input or cell. The circuits and their number of gates, input and outputs are shown in . The variations of the individual gate delays can be modeled as any distribution. The results in this section use normal distribution for the modeling of the variations in the proposed method and the Monte Carlo method.

Circuit	Gates	Inputs	Outputs
C432	125	45	8
C499	544	75	33
C880	365	88	27
C1908	495	60	26
C2670	657	299	65
C3540	1120	74	23
C6288	2727	66	33
C7552	2608	316	108

Table 1 compares the accuracy of the proposed statistical timing method. The comparisons are made against golden Monte Carlo method (each with 10,000 timing runs). The results of worst case method are also shown. The worst case method uses a regular deterministic timing analysis with the delay of each gate set to 3σ from the mean value. The 99% confidence point of the CDF is shown for the Monte Carlo and the proposed method. As seen from the table, the worst case method can be 21-24% off from the desired monte carlo method. The proposed method produces very accurate results as compared to Monte Carlo method with error of only 0.3% to 0.79%. The results shown are with a 7-point piecewise linear CDF modeling with no reconvergent fanout correction. The Monte Carlo method takes into account reconvergent fanout. In reference [16], it is shown that statical timing analysis ignoring reconvergent fanouts produces an upper bound on the true delay. However, the table also indicates that a statistical timing analysis without reconvergence

fanout correction is very close to the real answer. Therefore, reconvergence does not cause a significant difference in overall answer for these circuits. Later, we also present results for our technique for reconvergence fanout correction.Number of gates, inputs and outputs for ISCAS benchmark circuit mapped to a current library.

Circuit	Monte Carlo Method 99%	Worst Case Method		Proposed Method 99%	
	Delay (ps)	Delay (ps)	Error (%)	Delay (ps)	Error (%)
C432	3588	4426	23.3%	3610	0.61%
C499	3505	4283	22.1%	3525	0.57%
C880	3344	4088	22.2%	3359	0.44%
C1908	3574	4355	21.8%	3587	0.27%
C2670	2549	3094	21.3%	2557	0.31%
C3540	5251	6500	23.7%	5280	0.55%
C6288	20704	26351	27.2%	20868	0.79%
C7552	5262	6565	24.7%	5299	0.69%

Table 1 Accuracy comparison of the timing results by the proposed method as compared to worst case method and Monte Carlo method.

The accuracy of the proposed method is further illustrated in Table 2. This table shows the results obtained by Monte Carlo and the proposed method for the 1% point in the CDF. The error compared to exact Monte Carlo is small and varies from 0.09% to 2.42%.

The proposed method can be run with different modeling complexity of the piecewise linear (PWL) CDF. All the CDFs in each circuit are modeled as a 3-point PWL model, a 5-point PWL model and a 7-point PWL model. Accuracy comparisons of these three modeling levels is shown in Table 3. As expected, the accuracy of the proposed technique decreases with reduction in number of segment in the PWL approximation. However, even the 3-pt. PWL model gives good accuracy. The computational cost of different CDF PWL models is illustrated in Table 4. Performance is shown as the ratio to the 3-point PWL statistical timing run. For example, in circuit C880, the 5-point PWL model takes 2.5 times longer to run as compared to the 3-point PWL model and the 7-point PWL model takes 4.5 times longer to run as compared to the 3-point PWL model.

Accuracy of statistical timing with and without reconvergence fanout handling in shown in Figure 12 and Figure 13. The figures show the CDF of the arrival time of the most critical output as computed by Monte Carlo, proposed method without reconvergence and proposed method with reconvergence. The proposed method with reconvergence handling compares very well to Monte Carlo validating our dominant common mode algorithm of Section 3. Performance impact of reconvergence handling in the proposed method is shown in Table 5. This table shows the percentage runtime overhead with reconvergence handling as compared to run time without reconvergence. The efficient nature of reconvergent handling procedure yields only about 10-30% penalty over timing without reconvergence handling.

Circuit	Monte Carlo Method 1% pt.	Proposed Method 1%CDF point	
	Delay (ps)	Delay (ps)	Error (%)
C432	3285	3327	0.12%
C499	3250	3278	0.86%
C880	3057	3108	1.66%
C1908	3323	3403	2.4%
C2670	2287	2289	0.09%
C3540	4899	4967	1.38%
C6288	20008	20326	1.58%
C7552	4876	4997	2.42%

Table 2 Accuracy comparison of the timing results by the proposed method as compared to Monte Carlo method.

Circuit	7-pt. PWL CDF	5-pt. PWL CDF	3-pt. PWL CDF
	Error (%)	Error (%)	Error (%)
c432	0.61%	1.8%	-2.2%
c499	0.57%	1.76%	-2.4%
C880	0.44%	1.7%	-2.54%
C1908	0.27%	1.65%	-2.63%
C2670	0.31%	1.6%	-2.74%
C3540	0.55%	1.81%	-2.15%
C6288	0.79%	1.69%	-1.38%
C7552	0.69%	1.84%	-1.98%

Table 3 Accuracy comparison for variable accuracy modeling of the CDFs for the 99% point in timing distribution.

Circuit	Perf. Impact of Variable Accuracy	
	5-pt. PWL CDF	7-pt. PWL CDF
C432	2.0	4.0
C499	2.7	4.7
C880	2.5	4.5
C1908	3.3	5.3
C2670	2.5	4.2
C3540	2.1	3.7
C6288	2.5	4.6
C7552	2.7	4.7

Table 4 Performance impact of variable accuracy in CDF modeling. The runtimes are shown as ratio to the runtime of 3-pt. PWL

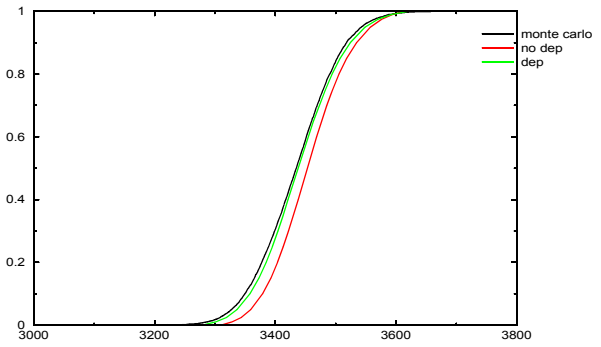


Figure 12 Accuracy of the timing distribution CDF for circuit C432 with and without reconvergence handling as compared to Monte Carlo.

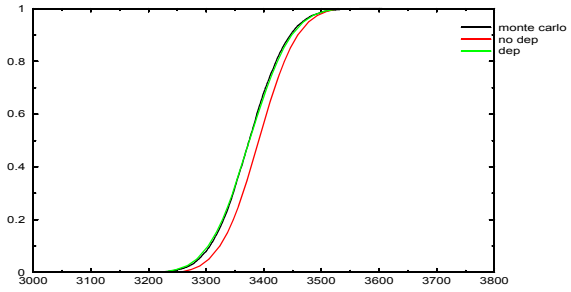


Figure 13 Accuracy of the timing distribution CDF for circuit C499 with and without reconvergence handling as compared to Monte Carlo.

Circuit	Perf. Impact of Handling Reconvergence
C432	10%
C499	21%
C880	33%
C1908	19%
C2670	18%
C3540	30%
C6288	16%
C7552	24%

Table 5 Performance impact of handling reconvergence fanout as measured by ratio of runtime with and without reconvergence handling.

5. Conclusions and Future Work

A new block-based static timing technique with uncertainty has been presented in this paper. This technique models the arrival times as cumulative distribution functions (CDFs) and gate delays as probability density functions (PDFs) for efficient timing analysis. Simple expressions are presented for the key operations: add and max. While the approach works for any type of distribution, for efficiency the CDFs are modeled as piecewise linear distributions. An efficient technique based on statistical subtraction has been presented for handling reconvergent fanouts. Accurate results have been demonstrated for various ISCAS benchmark circuits. The accuracy of the analysis can be varied by varying the accuracy of the piecewise linear CDF model. Regular (or deterministic) timing analysis can be easily incorporated into this statistical framework by modeling the CDF as a step. All the expressions for statistical timing analysis are valid in the limiting case for deterministic timing analysis. Future work include handling correlation in arrival times and gate delays due to process parameters.

6. References

[1] R. B. Hitchcock, “Timing verification and the timing analysis problem”, *ACM Design Automation Conference*, pp. 594-604, 1982.

[2] N. Jouppi, “Timing Analysis for nMOS VLSI”, *ACM Design Automation Conference*, 1983, pp. 411-418.

[3] F. Najm, R. Burch, P. Yang, I. Hajj, “Probabilistic simulation for reliability analysis of CMOS VLSI Circuits,” IEEE Trans. on CAD, April 1990.

[4] S. Devadas, H. Jyu, K. Keutzer, S. Malik, “Statistical timing analysis of combinational circuits, *IEEE ICCD conference, 1992.*

[5] H. Jyu, S. Malik, “Statistical timing optimization of combinational circuits, *IEEE ICCD conference, 1993.*

[6] R. B. Brawhear, et. al, “Predicting circuit performance using circuit-level statistical timing analysis”, *European Design and Test Conference*, 1994.

[7] M. Berkelaar, “Statistical Delay Calculation: a Linear Time Method,” *Proceedings of TAU*, 1997.

[8] E. Jacobs and M. Berkelaar, “Gate sizing using a statistical delay model”, *Europen Design and Test Conference*, 2000.

[9] R. Lin and M. Wu, “A new statistical approach to timing analysis of VLSI cricuits”, *Proc. International Conference of VLSI Design,* 1998.

[10] S. Tongsima, et al, “Optimizing circuits with confidence probability using probabilistic retiming”, *Proceedings of ISCAS,* 1998.

[11] J.J. Liou,et al, “Fast Statistical Timing Analysis by Probabilistic Event Propagation”, *ACM Design Automation Conference*, 2001.

[12] J.J. Liou,et al, “False path aware statistical timing analysis and efficient path selection for delay testing and timing validation”, *ACM Design Automation Conference*, 2002.

[13] M. Orshansky and K. Keutzer, “A general probabilistic framework for worst-case timing analysis, ACM Design Automation Conference, 2002.

[14] A. Gattiker, S. Nassif, R. Dinaker, C. Long, “Timing yield estimation from static timing analysis,” Proceedings ISQED, 2001.

[15] X. Bai, C. Visweswariah, P. Strenski, D. Hathaway, “Uncertainty-aware circuit optimization”, ACM Design Automation Conference, 2002.

[16] A. Agarwal, V, Zolotov, D. Blaauw, S. Vrudhula, “Statistical Timing Analysis using Bounds and Selective Enumeration,” Proceedings of TAU, 2002.

[17] J. A. Jess, et al, “Statistical Timing for Parametric Yield Prediction of Digital Integrated Circuits,” Proceedings of ACM Design Automation Conference, 2003.