Short Papers

Statistical Timing Verification for Transparently Latched Circuits

Ruiming Chen and Hai Zhou

***Abstract*—High-performance integrated-circuit designs need to verify the clock schedules as they usually have level-sensitive latches for their speed. With process variations, the verification needs to compute the probability of correct clocking. Because of complex statistical correlations and accumulated inaccuracy of statistical operations, traditional iterative approaches have difficulties in getting accurate results. A statistical check of the structural conditions for correct clocking is proposed instead, where the central problem is to compute the probability of having a positive cycle in a graph with random edge weights. The authors proposed two algorithms to handle this. The proposed algorithms traverse the graph only several times to reduce the correlations among iterations, and it considers not only data delay variations but also clock-skew variations. Although the first algorithm is a heuristic algorithm that may overestimate timing yields, experimental results show that it has an error of 0.16% on average in comparison with the Monte Carlo (MC) simulation. Based on a cycle-breaking technique, the second heuristic algorithm can conservatively estimate timing yields. Both algorithms are much more efficient than the MC simulation.**

***Index Terms*—Scheduling, timing analysis, timing verification.**

I. Introduction

With shrinking geometries in deep submicrometer technology, process variation becomes a prominent phenomenon in fabrication. These variations introduce random variables into the timing of a fabricated integrated circuit (IC). These delay variations and clock-skew variations present a new challenge on timing verification and yield prediction.

There are many recent researches that deal with the timing analysis under process variations [1]–[6]. These researches are mainly focused on timing analysis of combinational circuits. However, the validity of a circuit really depends on whether the correct signal values are latched into the memory elements, and the results of timing analysis are to be used to check clocking conditions.

Level-triggered transparent latches are usually used in high-performance circuits because of their high performance and low power consumption [7]. The complexity introduced by latches is that a signal can pass transparently through a latch during its enabling period, which makes time borrowing across latch boundaries possible. Therefore, timing analysis can no longer be carried out only on the separated combinational part since the output time of a latch is now dependent on its input time.

In this paper, we formulate the clock-schedule-verification problem under process variations as computing the probability of correct clocking. Considering the application of iterative approaches to compute the arrival time on different corners, some will converge but others will not. As collections of the arrival time, the distributions will not converge. Furthermore, since a delay will have a fixed value after fabrication, the distributions in different iterations are tightly correlated. These issues make the iterative approaches [8]–[10] difficult to be used under process variations. The accumulated inaccuracy of max and min operations on random variables is also an obstacle to iterative methods. Based on this, we propose to use structural clock-validity conditions with process variations. The relationship between valid clocking and the conditions of no positive cycle in the latest constraint graph or negative cycle in the earliest constraint graph is first established. Then, these structural conditions with delays and clock skews being random variables are checked through noniterative graph-traversal techniques. One advantage of these techniques is that each element in the circuit is traversed only several times and thus, the effect of correlations and accumulated inaccuracy of statistical operations is greatly reduced.

There were some recent works on the statistical timing analysis of latch-based pipeline design [11], [12], where the key problem is to compute the probability that the arrival time of the latches violates setup-time and hold-time constraints. Not involving any cycles, this problem is a special case of our statistical clock-schedule-verification problem, and can be efficiently and accurately solved by our algorithms.

The rest of the paper is organized as follows. In Section II, the models of latches, clocking schemes, and the conditions on correct clock schedules are given without consideration of process variations. With the consideration of process variations, Section III formulates the statistical clock-schedule-verification problem and presents the difficulties involved in iterative approaches under process variations. Section IV establishes the structural conditions for valid clock schedules by extending the work of Szymanski and Shenoy [8]. In Section V, the probability that these structural conditions are held under random element delays and clock skews is computed to give the probability of valid clocking, and an important application of our algorithms in pipeline designs is proposed. The experiments on the proposed approaches and their comparison with the Monte Carlo (MC) simulation are reported in Section VI. Finally, the conclusion is given in Section VII.

II. Deterministic Clock-Schedule Verification

A. Models of Clock and Transparent Latches

A clock is a periodical signal used to regulate other signals in the circuit. Multiple phases of a clock may be used in a circuit. A clock scheme for a circuit is a set of periodical signals $\phi_1, \ldots, \phi_n$ with a common period c. Selecting a period of length c as the global time reference, we can denote each phase ϕ_i by its starting and ending time (s_i, e_i) with respect to the reference. Note that it is possible to have $s_i > e_i$ based on the selection of global time reference. For simplicity, we assume that $s_i < e_i$ for all the phases, which can be easily done by shifting the global time reference. We generally order the phases such that $e_i < e_j$ if $i < j$. Also note that w_i is used to represent the width of phase i.

Memory elements are used in a circuit to store its state. Only under a certain condition of the clock input does the memory element respond to the data input. Memory elements can be categorized into two groups, according to how they respond to the clock input: flip-flops store the data when the clock switches and latches let the output have the input value when the clock level is high.

Manuscript received June 11, 2004; revised January 21, 2005 and May 4, 2005. This work was supported by the National Science Foundation (NSF) under Grant CCR-0238484. This paper was recommended by Associate Editor C. J. Alpert.

The authors are with the Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL 60208 USA (e-mail: rui-chen@northwestern.edu).

Digital Object Identifier 10.1109/TCAD.2005.857395

Because of this, clock-schedule verification is easy in a circuit with only flip-flops but is very difficult when latches are used. We only focus on the latter problem in the succeeding discussion.

B. Clock-Validity Conditions

Without the consideration of process variations, the clock-schedule-verification problem can be formulated as follows.

Problem 1 (Deterministic Clock-Schedule Verification): Given a circuit and a clock schedule without process variations, check whether the clock schedule is correct in the fabricated chips.

Given the pin-to-pin delay of each gate, the maximal and minimal delays from the output of one latch to the input of another latch can be computed by traversing the topology of the gate connections. Let Δ_{ij} and δ_{ij} represent the maximal and minimal combinational delays from latch i to latch j, respectively. Also let A_i and a_i represent the latest and the earliest signal arrival time on the input of latch i, and D_i and d_i the latest and the earliest signal departure time on the output of latch i, respectively. The well-known SMO formulation [13] is

$$A_i = \max_{j \to i} \left(D_j + \Delta_{ji} - E_{p_j p_i} \right) \tag{1}$$

$$D_i = \max \left(A_i, c - w_{p_i} \right) \tag{2}$$

$$a_i = \min_{j \to i} \left(d_j + \delta_{ji} - E_{p_j p_i} \right) \tag{3}$$

$$d_i = \max \left(a_i, c - w_{p_i} \right) \tag{4}$$

where p_i is the clock phase controlling latch i and E_{ij} is defined as

$$E_{ij} = \begin{cases} e_j - e_i & \text{if } j > i \\ c + e_j - e_i & \text{otherwise} \end{cases}.$$

We must also note that the local time used here is referring to local periods that end with the phase falling edges.

Ignoring initial hold-condition violations, the SMO formulation is too aggressive on earliest time calculation. A deterministic conservative formulation of earliest time constraints is presented in [14] as

$$a_i = \min_{j \to i} \left(d_j + \delta_{ji} - E_{p_j p_i} \right) \tag{5}$$

$$d_i = c - w_{p_i}. \tag{6}$$

In practice, the aggressive formulation might yield a solution with a shorter period. But it is shown in [15] that there are common situations, such as a latch driven by a qualified clock signal, in which the aggressive formulation is incorrect, and a similar problem arises in circuits that permit the clock to be stopped between adjacent latches to save power. Therefore, in this paper, we choose the conservative formulation.

The solution of the above equations should satisfy the setup and hold-time conditions

$$A_i \leq c - S_i \tag{7}$$

$$a_i \geq H_i \tag{8}$$

where S_i and H_i are the setup time and hold time of latch i, respectively.

III. Problem Formulation

A. Statistical Clock-Schedule Verification

Process variations influence not only the data delays but also the clock network, and there exist correlations among the process variations of all the devices including clock network. There are many recent researches that deal with the timing analysis under process variations [1]–[6]. All these researches only dealt with timing analysis on combinational circuits. However, sequential circuits dominate in reality, and the validity of a circuit really depends on whether the correct signal values can be latched into the memory elements. So all these researches should eventually be used to check the clocking conditions, which is the focus of our work. Neves and Friedman [17] presented a graph-based algorithm to solve the clock-skew optimization problem considering process variations, and it only considered the process variations of clock network while neglecting the dominating data delay variations. On the other hand, our work provides a framework for statistical clock-schedule-verification problems, which considers both data delay variations and clock-network variations. The correlations among data delay variations and clock-network variations are all considered here.

With the consideration of process variations, the clock-schedule-verification problem can be formulated as follows.

Problem 2 (Statistical Clock-Schedule Verification): Given a circuit and a clock schedule under process variations, compute the probability that the clock schedule is correct in the fabricated chips.

Since there exist process variations in the clock network, the starting and ending time (s_p, e_p) are not the same for the latches controlled by the same clock phase p, so we cannot use a single pair of variables (s_p, e_p) to represent this phase. A pair of random variables (s_i, e_i) is used to represent the clock phase controlling latch i. For simplicity, we assume that the influence of process variations on s_i and e_i is the same, so $w_{p_i} = e_i - s_i$ is constant, thus we only need one random variable e_i. Our method can be easily extended to more realistic models. Thus, we need n correlated random variables (e_i) and p constants (w_p) to represent all the clock phases instead of $2p$ constants (e_p, w_p) in the deterministic clock-schedule-verification problem, where n is the number of latches, and p is the number of clock phases. Since the delays are influenced by process variations, Δ_{ij}, δ_{ij}, A_i, a_i, and D_i are random variables. Process variations also influence the setup time and hold time of latches, so S_i and H_i are also random variables. These random variables may be correlated. In the succeeding discussion, we use bold font to differentiate the random variables from the deterministic ones. In summary, the following variables in the conservative formulation are random variables: $\{\mathbf{A_i}, \mathbf{a_i}, \mathbf{D_i}, \mathbf{\Delta_{ji}}, \boldsymbol{\delta}_{\mathbf{ji}}, \mathbf{E_{ji}}, \mathbf{S_i}, \mathbf{H_i}\}$. Changing these variables to random variables, we can translate the deterministic conservative formulation of time constraints to the statistical conservative formulation of time constraints.

B. Difficulties in Traditional Iterative Methods

Considering application of iterative approaches on different corners, some will converge but others will not. As collections of the arrival time, the distributions will not converge.

Also, many max and min operations on correlated random variables are involved in the conservative formulation. Since no accurate analytical formula exists for any of them, the current practice always uses approximation techniques to handle them. For example, some current statistical-timing-analysis algorithms [5], [6] are based on Clark's method [19], which assumes that the maximum of random variables with normal distribution is still with normal distribution. However, from our experience, the actual distribution may be far from normal distribution in some cases. The accumulated inaccuracy of max and min operations on random variables introduces inaccuracy into the ultimate results of iterative methods.

Furthermore, in clock-schedule verification with process variations, the element delay is the same when traversing the same element in different iterations. This means that in an iterative approach, the delays in different iterations are perfectly correlated, which makes some state-of-the-art statistical-timing-analysis techniques inappropriate to

be used in the iterative approaches under process variations. For example, the technique in [6] used a canonical delay model that denotes the delay as the sum of global and local components. The local components are used to improve the accuracy. But since no local components exist in iterative methods, this technique loses correlation information, which results in much inaccuracy in the ultimate results.

One way to simplify the correlations among iterations and reduce the effect of the inaccuracy of statistical operations on ultimate results is to reduce the number of iterations without sacrificing the accuracy too much.

IV. Structural Verification of Clock Schedule

A. Deterministic Situation

Having studied when the iterative approach to the deterministic clock-schedule-verification problem will converge and whether the converged solution is unique, Szymanski and Shenoy [8] came up with some structural characterizations. A circuit C is modeled as a finite edge-biweighted directed graph $G = (V, E, \Delta, \delta)$. For every memory element, primary input or primary output, there is a vertex in V. If there is a path of combinational logic from one memory element, primary input i to another memory element, or primary output j, there is a directed edge e_{ij} from the vertex representing element i to the vertex representing element j. The edge weight $\Delta_{ij}(\delta_{ij})$ is the maximum (minimum) delay of the combinational paths from i to j. G is called latch graph. The following two theorems were given in Shenoy's thesis [9].

Theorem 1: The equation set composed by (1) and (2) has a unique solution if and only if there is no zero Δ-weight cycle in the latch graph.

Theorem 2: If the latch graph has no zero δ-weight cycle, then the equation set composed by (3) and (4) has a unique solution.

Since it is difficult to use an iterative approach when the delays become random variables, a theory of structural conditions for a valid clocking will be established.

First, we consider the deterministic clock-schedule-verification problem. The latest equation set is composed by (1), (2), and (7). The earliest equation set is composed by (5), (6), and (8). We first translate the latest equation set into a system of inequalities

$$\begin{aligned} A_i - D_j &\geq \Delta_{ji} - E_{p_j p_i} \qquad \forall j \to i \\ D_i - A_i &\geq 0 \qquad \forall i \\ D_i &\geq c - w_{p_i} \qquad \forall i \\ -A_i &\geq S_i - c \qquad \forall i. \end{aligned}$$

Based on the correspondence between a system of difference inequalities and the longest path problem on a graph [18], we can construct a latest constraint graph corresponding to this inequality set. For each constraint $x_i - x_j \geq b_{ij}$, vertices v_i and v_j are introduced for variables x_i and x_j, and an edge from v_j to v_i with weight b_{ij} is also introduced. Another vertex O will be introduced as the reference time 0.

Similarly, the earliest equation set can be translated into the following inequalities, from which an earliest constraint graph can be constructed

$$\begin{aligned} a_i - d_j &\leq \delta_{ji} - E_{p_j p_i} \qquad \forall j \to i \\ d_i &\leq c - w_{p_i} \qquad \forall i \\ -a_i &\leq -H_i \qquad \forall i. \end{aligned}$$

As an example, consider a circuit given in Fig. 1(a). Its latest and earliest constraint graphs are given in Fig. 1(b) and (c), where the setup time and hold time are assumed to be 0 for simplicity.

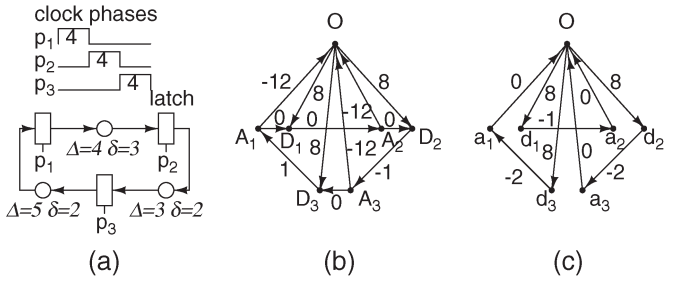

Fig. 1. (a) Clock schedule and (b) its latest constraint graph and (c) earliest constraint graph.

We denote the sum of the weights of all the edges in a path as the weight of this path. If the weight of a cycle is positive (negative), then the cycle is called positive (negative) cycle. Based on the construction of the latest and earliest constraint graphs, the following theorem can be established.

Theorem 3: A given clock schedule is valid if and only if the latest constraint graph has no positive cycle and the earliest constraint graph has no negative cycle.

Proof: For the latest constraint graph, Sakallah *et al.* [14] have proved that the relaxation of the latest equation set is equivalent to the latest equation set. So the constraints in the latest equation set are satisfied if and only if the latest constraint graph has no positive cycle.

For the earliest constraint graph, Shenoy *et al.* [10] proved that for each edge from latch u to latch v, constraint

$$e_{p_v} \leq s_{p_u} + \delta_{uv} + (1 - K_{uv})c - H_u$$

is equivalent to the earliest time constraints in the earliest equation set, where

$$K_{uv} = \begin{cases} 0, & \text{if } e_{p_u} < e_{p_v} \\ 1, & \text{otherwise} \end{cases}.$$

Provided

$$e_{p_u} - s_{p_u} = w_{p_u}$$

and

$$E_{p_u p_v} = e_{p_v} - e_{p_u} + K_{uv}c$$

the constraint

$$\delta_{uv} - E_{p_u p_v} - H_v + c - w_{p_u} \geq 0$$

is equivalent to

$$e_{p_v} \leq s_{p_u} + \delta_{uv} + (1 - K_{uv})c - H_v.$$

This means that the constraints in earliest equation set are satisfied if and only if the earliest constraint graph has no negative cycle.

Therefore, the theorem is true. ■

B. Statistical Situation

In statistical clock-schedule-verification problem, we similarly construct the latest constraint graph and the earliest constraint graph based on constraints (9)–(14), but the weights of edges in these two graphs are random variables. The following corollary can be easily established based on Theorem 3.

Corollary 3.1: The probability that a given clock schedule is valid is equal to the probability that the latest constraint graph has no positive cycle and the earliest constraint graph has no negative cycle.

In the deterministic situation, the negative (positive) cycle detection can be accomplished by the Bellman–Ford algorithm [18]. But to the best of our knowledge, there are no algorithms to deal with the negative (positive) cycle detection problem when the edge weights are random variables. As mentioned before, it is difficult for iterative approaches to achieve accurate and stable results in the statistical situations because of the inaccuracy of the statistical operations. The following example confirms this. The Bellman–Ford algorithm can be extended to handle statistical situations: do the statistical min (max) operations instead of the deterministic comparison and assignment operations in the relaxation steps. Then, the following theorem can be proved.

Theorem 4: The probability of the existence of negative cycles in a graph $G(V, E)$ with random-weight edges is equal to

$$1 - \Pr\left(\max_{(u,v)\in E}(d_v - d_u - w(u,v)) \le 0\right)$$

where d_u is the computed distance label at vertex u after $|V| - 1$ iterations, and $w(u, v)$ is the weight of the edge (u, v).

However, it is obvious that

$$\max_{(u,v)\in E}(d_v - d_u - w(u,v)) \ge 0.$$

Thus,

$$\Pr\left(\max_{(u,v)\in E}(d_v - d_u - w(u,v) \le 0)\right)$$

is actually equal to

$$\Pr\left(\max_{(u,v)\in E}(d_v - d_u - w(u,v) = 0)\right)$$

and the density distribution function of

$$\max_{(u,v)\in E}(d_v - d_u - w(u,v))$$

is truncated at 0. The state-of-the-art statistical-static-timing-analysis (SSTA) algorithms [5], [6] always assume that the maximum of two Gaussian-distributed variables is still Gaussian distributed. Then, these methods must get the mean value of

$$\max_{(u,v)\in E}(d_v - d_u - w(u,v))$$

that is larger than 0, which implies that

$$\Pr\left(\max_{(u,v)\in E}(d_v - d_u - w(u,v)) \le 0\right) < 50\%.$$

Thus, when the actual yields are no less than 50%, the iterative approach cannot get accurate and stable results. In this paper, we use noniterative approaches to solve the statistical clock-schedule-verification problem.

V. Statistical Checking of Structural Conditions

A. Statistical Static Timing Analysis (SSTA)

The SSTA algorithm introduced in [5] is used in our work to calculate the maximal delay between vertices. It uses the principal-component-analysis (PCA) technique [16] to transform a set of correlated parameters into an uncorrelated set and assumes that the delay of a gate or an interconnect is normally distributed. After doing PCA, a delay can be represented as a linear function of principal components (independent random variables with standard normal distributions)

$$\mathbf{d} = d_0 + k_1 \times \mathbf{p_1} + \cdots + k_m \times \mathbf{p_m}$$

where d_0 is the mean value, $\mathbf{p_i}$s are independent principal components, and k_is are coefficients. The sum and max functions of normally distributed random variables were provided, which can maintain the correlation information. Particularly for the max function, Clark's method [19] was used to approximate the result, which assumes that the maximal of two random variables with normal distribution is also normally distributed. Then, it uses a program evaluation and review technique (PERT) like traversal on the circuit graph to calculate the latest arrival time of primary outputs.

B. Latest Time Constraints

Statistical verification of the latest time constraints needs to calculate the probability that all the cycles are nonpositive in the latest constraint graph.

Let $C = \{c_1, c_2, \ldots, c_n\}$ be the set of cycles, and $|\mathbf{c_i}|$ be the delay of cycle c_i. Let $F(|\mathbf{c_1}|, |\mathbf{c_2}|, \ldots, |\mathbf{c_n}|)$ be the joint cumulative distribution function (JCDF) of the delays of cycles, then the probability that all the cycles are nonpositive is $F(0, 0, \ldots, 0)$, which is equal to

$$P_r\left(|\mathbf{c_1}| \le 0, |\mathbf{c_2}| \le 0, \ldots, |\mathbf{c_n}| \le 0\right).$$

However, we know that

$$P_r\left(|\mathbf{c_1}| \le 0, |\mathbf{c_2}| \le 0, \ldots, |\mathbf{c_n}| \le 0\right) = P_r\left(\max_{j=1}^{n}(|\mathbf{c_j}|) \le 0\right).$$

Thus, if we can get the distribution of the maximum delay of all the cycles, we can get the probability of the circuit satisfying the latest time constraints. But the number of cycles is often exponential in the number of vertices of a graph, so the methods based on the enumeration of cycles [20] are prohibitive.

However, if we can classify the edges of a directed graph into two disjoint sets such that each cycle is just formed by two simple paths, one composed of edges from each set, then the enumeration of cycles is not necessary.

Based on this idea, we perform depth-first search on the latest constraint graph, and classify the edges into two sets: one contains all the backward edges and the other contains the rest. Then, if the backward edges are deleted from the graph, the graph becomes a directed acyclic graph (DAG), and we can use a PERT traversal to compute the longest path between any two points. When they are combined with the weights of backward edges, we can get the weight distribution of the longest cycle.

Unfortunately, our study shows that some cycles may be missed in the above approach.

Theorem 5: The edges in a directed graph cannot always be divided into two disjoint sets, such that any simple cycle is formed by two simple paths, one composed of edges from each set.

A simple example to show the correctness of this theorem is shown in Fig. 2. Enumerating all the bipartitions of the cycle $A(a, b, c, d, e, f, a)$ of the graph in this example, we will see that all the partitions do not satisfy the condition that all the simple cycles in this graph can be formed by two simple paths, one composed of edges from each set.

This theorem tells us that if more than one backward edge is involved in a cycle, they possibly do not form a simple path on this cycle, and the traversal in any topological order of vertices ignoring these backward edges does miss this cycle. For example, for the graph in Fig. 3, if the topological order is (a, c, b, d), edges (b, c) and (d, a)

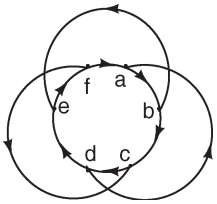

Fig. 2. Case against simple-path bipartitioning.

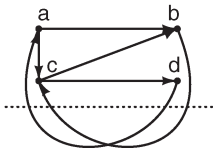

Fig. 3. Cycle containing two backward edges.

are backward edges, and the cycle (a, b, c, d, a) involves these two edges that do not form a simple path, so this order misses this cycle.

Based on these discussions, we design a heuristic method by performing depth-first search many times, and the order of choosing edges in each search is randomly selected. When the search is performed many times, the overall probability of missing cycles can be smaller. This algorithm PCycle is shown in Fig. 4.

We assume that the random variables $\mathbf{\Delta_{ji}}$, $\boldsymbol{\delta}_{\mathbf{ji}}$, $\mathbf{e_i}$, $\mathbf{S_i}$, and $\mathbf{H_i}$, in our statistical conservative formulation, satisfy normal distribution. Any one of these random variables $\mathbf{r}$ can be represented by the linear function of independent variables

$$\mathbf{r} = \mu + \sum_{i=1}^{m} a_i \mathbf{l_i}$$

where μ is the mean value of $\mathbf{r}$, $\mathbf{l_i}$s are independent random variables, a_is are constants, and m is the number of independent random variables. This representation can be always achieved using the models presented in [1] or [5]. Correlated variables can be represented by the linear function of independent random variables with standard normal distribution using PCA. So for simplicity, we assume that each $\mathbf{l_i}$ is a standard normally distributed variable. So in PCycle, we can use statistical-timing-analysis methods [5], [6] to compute the maximal delay of cycles.

An important property of PCycle is that it can get a good estimation of yield very efficiently. In Section VI, we will see that it can get accurate estimation of yield in a much shorter time compared with MC simulations. We also implemented a cycle-enumeration algorithm based on the algorithm proposed by [20], and found that it is only four times faster than the MC simulation in most cases. Thus, PCycle is still a good algorithm to estimate the yield although it may miss cycles.

The complexity of this algorithm is $O[MN(V + E)]$, where M is the number of depth-first-search iterations for each backward edge, N is the number of edges in backward edge set $\mathcal{E}$, V is the number of vertices in G, and E is the number of edges in G.

C. Earliest Time Constraints

Statistical verification of the earliest time constraints needs to calculate the probability that all the cycles are nonnegative in the earliest constraint graph.

Similar to the verification of latest time constraints, let $C = \{c_1, c_2, \ldots, c_n\}$ be the set of cycles, and $|\mathbf{c_i}|$ be the weight of cycle c_i. Let $F(|\mathbf{c_1}|, |\mathbf{c_2}|, \ldots, |\mathbf{c_n}|)$ be the JCDF of the weight of cycles,

Algorithm PCycle
Input: constraint graph $G(V, E)$
Output: probability of no positive cycles existing in G
Notations:
L_e: the weight of edge e
D_e: the maximal weight between the end vertex and start vertex of edge e
M: the maximal number of depth first search iterations.

Procedure:
1. Depth first search graph G to find all backward edges, stored in set $\mathcal{E}$;
2. For each edge e in $\mathcal{E}$ {
   t := 1;
   LOOP: While t<M+1 {
    depth first search graph G starting from the end vertex of e to find backward edges, stored in set $\mathcal{E}'$;
    topological sort graph G starting from the end vertex of e after ignoring all the edges in $\mathcal{E}'$;
    clear $\mathcal{E}'$;
    t := t+1;
    if the resulting order is ever found goto LOOP;
    calculate D_e using statistical timing analysis method;
    $C_e^t = D_e + L_e$;
   }
   $C_e := \max_{t=1}^{M}(C_e^t)$;
  }
3. Compute $\max_{e\in\mathcal{E}} C_e$ using statistical timing analysis method;
4. Probability of no positive cycles is equal to $P_r(max_{e\in\mathcal{E}}(C_e) \leq 0)$

Fig. 4. PCycle algorithm.

then we can calculate the probability that all cycles are nonpositive, that is,

$$P_r\left(|\mathbf{c_1}| \geq 0, |\mathbf{c_2}| \geq 0, \ldots, |\mathbf{c_n}| \geq 0\right).$$

However, we know that

$$P_r\left(|\mathbf{c_1}| \geq 0, |\mathbf{c_2}| \geq 0, \ldots, |\mathbf{c_n}| \geq 0\right) = P_r\left(\min_{j=1}^{n}\left(|\mathbf{c_j}|\right) \geq 0\right).$$

This problem is similar to the latest time verification: we only need to calculate the distribution of the minimal weight of all the cycles in the earliest constraint graph. However, since the earliest constraint graph is much simpler, where every cycle in the earliest constraint graph includes the vertex O, the verification can be done optimally. The earliest time constraints verification algorithm NCycle is shown in Fig. 5. Since all cycles are considered in NCycle, it can check the earliest time constraints accurately. Since

$$\min(\mathbf{A}, \mathbf{B}) = -\max(-\mathbf{A}, -\mathbf{B})$$

Algorithm NCycle

Input: constraint graph $G(V, E)$
Output: probability of no negative cycles existing in G

Procedure:
1. Split vertex O into O_1 and O_2, all the outgoing edges of O in original G are outgoing from O_1, and all the incoming edges of O in original G are incoming edges of O_2;
2. PERT-traversal the new G to calculate the shortest distance $\mathbf{D}$ from O_1 to O_2;
3. Probability of no negative cycles is equal to $P_r(\mathbf{D} \geq 0)$

Fig. 5. NCycle algorithm.

the $\min$ operations involved here can be transformed to $\max$ operations, thus we can still use Clark's method here.

D. Combined Verification

It is obvious that the cycles in the latest constraint graph share some elements with the cycles in the earliest constraint graph. So there exist correlations between the probabilities that the clock schedule does not violate the latest or earliest time constraints. In the two sections above, we have shown how to calculate the probability that the clock schedule does not violate the latest or earliest time constraints, respectively. However, since the earliest and the latest time are correlated, multiplying the probabilities of correct conditions in them does not give the right answer. Therefore, we need a combined verification.

In PCycle, let

$$\mathbf{A} = \max_{e \in \mathcal{E}} \mathbf{C_e}$$

and in NCycle, let

$$\mathbf{B} = \min_{e \in \mathcal{E}} \mathbf{C_e}.$$

Then, the probability that the latest constraint graph has no positive cycles and the earliest constraint graph has no negative cycles is equal to

$$P_r\left(\max(\mathbf{A}, -\mathbf{B}) \leq 0\right).$$

When the correlation coefficient of $\mathbf{A}$ and $-\mathbf{B}$ is more negative, the Clark's method is more inaccurate. For example, $\mathbf{A} = (-1, 4)$, $-\mathbf{B} = (-1, 4)$, and their correlation coefficient is -1. We compute $P_r(\max(\mathbf{A}, -\mathbf{B}) \leq 0)$, then the result of Clark's method has a 7% error compared with the result of MC simulation. Here, $\mathbf{A}$ and $-\mathbf{B}$ often fall into this situation. Furthermore, we do not need to keep the linear-function format for the results in this step. So in order to improve the accuracy, we use MC simulation to accurately calculate the probability that the maximum of $\mathbf{A}$ and $-\mathbf{B}$ is not positive.

E. Latch-Based Pipeline Designs

Recently, some researches are focusing on the SSTA for latch-based pipeline designs [11], [12]. Besides the computation of the latest and earliest arrival time at the primary outputs, we need to compute the probability that the arrival time on latches violates setup-time and hold-time constraints, which is the data transmission error probability. Not involving any cycles, this problem is similar with the earliest constraint verification problem, and can be efficiently and accurately solved by our algorithm.

For the pipeline design with level-sensitive latches, since the pipeline design does not involve cycles, we can use PERTlike traversal to compute the arrival time at the primary outputs. The remaining task is to compute the data transmission error probability. We first construct the latest constraint graph G_p and the earliest constraint graph G_n. Similar with NCycle algorithm, vertex O in $G_p(G_n)$ is split into two vertices O_1 and O_2: all the outgoing edges of O in original graph $G_p(G_n)$ are outgoing from O_1, and all the incoming edges of O in original $G_p(G_n)$ are incoming edges of O_2. Then, $G_p(G_n)$ becomes a DAG, so we can traverse the new $G_p(G_n)$ from O_1 to calculate the maximal (minimal) weight from O_1 to O_2, which is also the maximal (minimal) weight of all cycles in the original $G_p(G_n)$. Then, we can use the combined-verification technique to compute the data transmission error probability.

F. Cycle-Breaking Strategy

PCycle may miss some cycles, so it may overestimate the yield. Another important aspect of the statistical clock-schedule problem is to conservatively estimate the yield.

Inspired by the cycle-breaking ideas proposed in [21], we break the cycles at the positions of latches instead of at the backward edges. For each cycle in the latch graph, we select one latch to be treated as a flip-flop. These flip-flops are called virtual flip-flops. In order to get a tight lower bound of the yield, we need to select an appropriate set of latches to be treated as virtual flip-flops, and set appropriate departure time for each virtual flip-flop.

Let X_i be the weight of the edge from O to D_i, and Y_i be the weight of the edge from A_i to O. If we select latch i to be treated as a virtual flip-flop, the physical meanings of X_i and $-Y_i$ are the departure time and the arrival time of the flip-flop, respectively.

Definition 1 [Cycle Break Operation (CBO)]: In the latest constraint graph for the clock-schedule verification, for latch i, delete the edge from A_i to D_i, and set X_i and Y_i such that X_i and Y_i satisfy the following three conditions

$$X_i + Y_i = 0$$
$$Y_i \geq S_i - c$$

and

$$X_i \geq c - w_{p_i}.$$

This operation on the graph is called CBO.

An important property of CBO is as follows.

Theorem 6: CBO is conservative, that is, if there are no positive cycles in the graph after CBO, there are no positive cycles in the original graph.

Proof: Part of a latest constraint graph and its corresponding graph after the CBO on latch i are shown in Fig. 6(a) and (b), where s, t, p, q, m, n, p' and q' are the weights of the corresponding paths. According to the definition of CBO, p' and q' satisfy

$$p' + q' = 0$$
$$p' \geq S_i - c$$

and

$$q' \geq c - w_{p_i}.$$

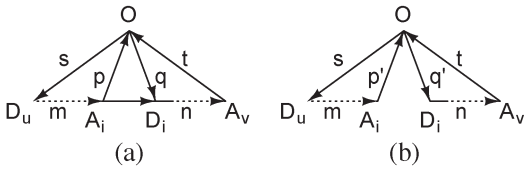

Fig. 6. Cycle break operation. (a) Original latest constraint graph. (b) Constraint graph after CBO.

Also, according to the weight definition of the latest constraint graph, $p = S_i - c$ and $q = c - w_{p_i}$. So $p' \geq p$ and $q' \geq q$.

For any cycle L in the original constraint graph perform the following.

1) If L does not pass through the edge from A_i to D_i, since the edge weights do not decrease in the CBO if L is not a positive cycle in the new graph, L is not a positive cycle in the original graph either, which means that the CBO is conservative.
2) If L passes through the edge from A_i to D_i, there are two situations.
   a) If L does not pass vertex O, we can replace the edge from A_i to D_i in L by one edge from A_i to O and another edge from O to D_i, and it is obvious that the weight of the new cycle is the same with the weight of L.
   b) If L passes vertex O, we do the above replacement, and we get two new cycles L_1 and L_2. As shown in Fig. 6, the weight of L is equal to

$$s + t + m + n$$

and the weights of L_1 and L_2 are equal to

$$s + m + p'$$

and

$$t + n + q'$$

respectively. If

$$s + m + p' \leq 0$$

and

$$t + n + q' \leq 0$$

we get

$$(s + m + p') + (t + n + q') \leq 0$$

while $p' + q' = 0$, so $s + m + t + n \leq 0$, which means that if L_1 and L_2 are not positive cycles, L is not a positive cycle either.

So the CBO is always conservative. ■

Let $\mu_{\mathbf{a}}$ and $\sigma_{\mathbf{a}}$ denote the mean value and the deviation of random variable $\mathbf{a}$, respectively. Let $\mathrm{PreSet}(i)$ be the set of latches that can reach latch i through a combinational path. Let Best_i and Worst_i represent the earliest and the latest arrival time of latch i, respectively, when the combinational delays between any two latches j and i are $\boldsymbol{\Delta}_{\mathbf{ji}}$, that is

$$\mathrm{Best}_i = \min_{j \in \mathrm{PreSet}(i)} (c - w_{p_j}) + \mu_{\boldsymbol{\Lambda}_{\mathbf{ji}}} - 3\sigma_{\boldsymbol{\Lambda}_{\mathbf{ji}}}$$

and

$$\mathrm{Worst}_i = \max_{j \in \mathrm{PreSet}(i)} c + \mu_{\boldsymbol{\Lambda}_{\mathbf{ji}}} + 3\sigma_{\boldsymbol{\Lambda}_{\mathbf{ji}}}$$

where $\boldsymbol{\Lambda}_{\mathbf{ji}} = \boldsymbol{\Delta}_{\mathbf{ji}} - \mathbf{E}_{\mathbf{ji}}$.

We define the weight of a latch as

$$\mathrm{LW}_i = \min(\mathrm{Worst}_i, c) - \max(\mathrm{Best}_i, c - w_{p_i}).$$

Now, we propose our latch-selection algorithm. We need to select as few latches as possible in breaking the cycles. We split the vertex O into two vertices O_1 and O_2 to break the cycles involving the vertex O: all the outgoing edges of O in the original graph are outgoing from O_1, and all the incoming edges of O in the original graph are incoming edges of O_2. Thus, the cycles involving the vertex O are broken at O. In the following, we focus on how to select latches to break the cycles not involving vertex O. For each latch i, the corresponding vertices A_i and D_i are collapsed to be a single vertex i: all the incoming edges of A_i in the original graph are incoming edges of i, and all the outgoing edges of D_i in the original graph are outgoing edges of i. We set the weight of vertex i to be LW_i. Then, our latch-selection problem becomes the well-known minimum weighted feedback vertex set problem, which has been proved to be NP-complete [22]. We use the heuristic algorithm proposed by Demetrescu and Finocchi [23] to compute the solutions.

After the selection of latches, we need to set the departure time for each virtual flip-flop, or determine X_is and Y_is. The problem to determine X_is and Y_is can be formulated as

$$\begin{aligned}
\text{Maximize} \quad & \Pr(L \leq 0) \\
\text{s.t.} \quad & L = \max_{i \mapsto j}(X_i + Y_j + L_{ij}) \\
& c - w_{p_i} \leq X_i \leq c \qquad \forall i \in \mathrm{Sel} \\
& Y_i = -X_i \qquad \forall i \in \mathrm{Sel} \\
& X_i = c - w_{p_i} \qquad \forall i \notin \mathrm{Sel} \\
& Y_i = S_i - c \qquad \forall i \notin \mathrm{Sel}
\end{aligned}$$

where L_{ij} is the maximal weight of the paths from i to j, $i \mapsto j$ means that latch i can reach latch j, and Sel is the set of virtual flip-flops (the selected latches).

From the proof of Theorem 6, we can see that the yield pessimism comes from the increase of the probabilities that the cycles not through the edge (A_i, D_i) in CBO are positive. Thus, if we can reduce this probability increase, the yield pessimism is expected to be reduced. Furthermore, since the correlations between path weights are typically positive, we may increase the $\Pr(L \leq 0)$ by separately increasing

$$\Pr(X_i + Y_j + L_{ij} \leq 0)$$

on each path $i \mapsto j$.

Based on these, we propose a novel approach to determine X_is. Starting from each vertex a connected with O_1, we traverse the graph to compute the maximal weights of the paths from a to the vertices connected with O_2. Since all the edge weights involved in this computation are known, for each pair of vertices, we can compute the probability that the maximal weight of the paths between them is positive, and insert it into a max-heap. Then, we use a greedy algorithm to set the unknown X_is. We assume that the setup time of latches is 0. The algorithm selects the paths one by one in the nonincreasing order of the probabilities that the path weights are positive. For each selected path from D_i to A_j in the graph after CBOs, there exists a cycle $\{O \to D_i \to A_j \to O\}$ with weight

$$L_{ij} + (c - w_{p_i}) - c = L_{ij} - w_{p_i}$$

TABLE I
COMPARISON RESULTS OF PCYCLE AND MC-SIMULATION METHOD

circuit				PCycle		Monte Carlo		error%
name	inputs#	latches#	gates#	yield%	time(s)	yield%	time(s)	
s27	4	3	10	94.41	0.01	94.39	96	0.02
s298	3	14	119	95.35	0.36	95.41	170	-0.06
s349	9	15	161	97.72	0.93	97.72	498	0.00
s526	3	21	193	95.35	0.22	95.25	287	0.10
s820	18	5	289	92.79	0.24	93.07	3,381	-0.28
s1238	14	18	508	96.71	0.21	96.67	3,312	0.04
s1488	19	6	653	87.49	4.43	87.86	3,651	-0.37
s1494	8	6	647	88.69	6.47	88.32	3,050	0.37
s5378	35	179	2,779	83.15	10.98	82.99	31,965	0.16
s9234	19	228	5,597	92.92	175.55	93.24	66,351	-0.32
s13207	31	669	7,951	93.94	1,024.79	93.48	118,046	0.46
s15850	14	597	9,772	88.49	1,154.83	87.70	164,130	0.79
s35932	35	1,728	16,065	96.25	938.28	96.12	392,806	0.13

in the original constraint graph, so the greedy algorithm always sets X_i and Y_j such that

$$\Pr(X_i + Y_j + L_{ij} \leq 0)$$

is as close to

$$\Pr(L_{ij} - w_{p_i} \leq 0)$$

as possible, which means that the increase of the probabilities that the cycles not through (A_i, D_i) are positive is minimized.

After the determination of the X_is, we can traverse the constraint graph from O_1 to calculate the distribution of the maximal weight of the paths from O_1 to O_2. Then, we can compute the lower bound of the timing yield using the combined-verification algorithm. ■

VI. EXPERIMENTAL RESULTS

PCycle, NCycle, and the combined-verification algorithm NPCycle have been implemented and tested on the ISCAS89 benchmark circuits where the flip-flops are replaced by level-sensitive latches. For simplicity, we have only considered the single-phase clock, and since PCA can transform the correlated variables to uncorrelated variables, any random variable $\mathbf{r}$ in statistical conservative formulation can be represented by a linear function of independent random variables with standard normal distribution

$$\mathbf{r} = r_0 + k_1 \times \mathbf{p_1} + \ldots + k_m \times \mathbf{p_m}$$

where r_0 is the mean value, $\mathbf{p_i}$s are independent random variables with standard normal distribution, and k_is are coefficients. Then, a random-number generator is used to generate r_0 and k_i for $i = 1, \ldots, m$. All the random variables are normally distributed with a 10–20% deviation. In our test cases, the number of random variables (m) is between 10 and 100, and $|k_i| \leq 5$. This test-case generator introduces the spatial correlations of process variations of gates, interconnects, and the clock network. For example, if for some $1 \leq i \leq m$, $k_i \neq 0$ for one gate, and $k_i \neq 0$ for another gate, there exists spatial correlation between the delay of these two gates. All experiments are run on a Linux PC with a 2.4-GHz CPU and 2.0-GB memory.

To verify the results of the PCycle, we used an MC simulation as a comparison. In each iteration, an MC simulation assigns values to $\mathbf{p_i}$ for $i = 1, \ldots, m$, then calculates the gate delays and the clock schedule, then performs a Bellman–Ford shortest-path algorithm to detect positive cycles. Here, the assigned values of $\mathbf{p_i}$ for $1 \leq i \leq m$ satisfy standard normal distribution. We run 10 000 iterations for each case in the MC simulations. A comparison of results on the latest time constraints is shown in Table I. The number of depth-first search (M) in PCycle is always less than ten. We can see that the results of PCycle are very close to the MC results with an average error of 0.24%, which confirms that the influence of missed cycles in PCycle is insignificant. PCycle is also much faster than the MC simulations.

TABLE II
COMPARISON RESULTS OF NPCYCLE AND MC-SIMULATION METHOD

circuit	NPCycle		Monte Carlo		error
	yield%	time(s)	yield%	time(s)	%
s27	92.15	0.05	92.13	180	0.02
s298	93.60	0.48	93.47	273	0.13
s349	92.91	2.00	92.71	970	0.20
s526	93.59	1.86	93.71	580	-0.12
s820	90.68	1.82	90.66	4,310	0.02
s1238	89.30	1.81	89.47	4,416	-0.17
s1488	83.88	6.82	83.89	4,261	-0.01
s1494	84.91	8.23	84.59	4,710	0.32
s5378	78.08	18.72	77.65	41,813	0.43
s9234	88.13	207.22	88.00	104,935	0.13
s13207	0.00	140.12	0.00	113,321	0.00
s15850	0.00	160.31	0.00	153,625	0.00
s35932	94.08	1,342.00	-	>10 d	-

A comparison of the results by the NPCycle with those by the MC simulations is shown in Table II. For each case, an MC simulation performs the Bellman–Ford algorithm to detect positive cycles in the latest time constraint graph, if no positive cycles are found, then it performs Bellman–Ford algorithm to detect negative cycles in the earliest time constraint graph. Since there exists a pair of latches between which the minimal combinational delays are 0 in s13207 and s15850, there always exist earliest constraint violations in them, so the corresponding timing yields are 0%. We can see that the results from NPCycle are very close to the MC-simulation results with an average error of 0.16%. The circuit with the largest run time s35932 is verified in 1342 seconds, while the MC simulation cannot finish within ten days.

We also implemented the conservative timing-yield estimation algorithm CBVerify. CBVerify computes the yields considering only the latest time constraints. Many cycles in ISCAS89 benchmark circuits contain only one flip-flop, so in order to test the performance of our latch-selection strategy, we transformed the circuits by replacing edge-triggered devices with pairs of level-sensitive latches controlled by a clock with dual phases, and then the circuits are retimed to minimize cycle time using the retiming algorithm proposed by Maheshwari and Sapatnekar [24]. A comparison of the results by the CBVerify with those by the MC simulations is shown in Table III, where "Strat. A" is the latch-selection strategy to select latches with the least LWs, "Strat. B" is the latch-selection strategy to select latches with the

TABLE III
COMPARISON RESULTS OF CBVERIFY AND MC-SIMULATION

circuit		MC	Yield of CBVerify (%)	
name	#latch	Yield(%)	Strat. A	Strat. B
s5378	844	98.87	98.86	74.86
s9234	990	91.25	88.30	83.13
s13207	1,295	89.50	89.25	89.07
s15850	1,719	100.00	100.00	22.06
s35932	5,207	98.87	98.81	98.30

largest LWs. Since it is obvious that our algorithm is much more efficient than MC simulations, we do not report the running time here. We can see that both strategies get the conservative yields. Particularly, Strategy A gets much more accurate yield estimation than Strategy B, which confirms that the latch-selection strategy greatly influences the accuracy of the yield estimation.

VII. CONCLUSION

In statistical clock-verification problems, because of complex statistical correlations and the accumulated inaccuracy of statistical operations, it is difficult for traditional iterative approaches to get accurate results. A statistical checking of the structural conditions for correct clocking is proposed instead, where the central problem is to compute the probability of having a positive cycle in a graph with random edge weights. We proposed two algorithms to handle this problem. The proposed methods traverse the graph several times to reduce the correlations among iterations. Although the first algorithm is a heuristic algorithm that may miss cycles, experimental results showed that it has an error of 0.16% on average, in comparison with the MC simulations. As an important application, this algorithm can accurately solve the SSTA problem in latch-based pipeline design. The second algorithm is based on a cycle-breaking technique, and can conservatively estimate the timing yield. Both algorithms are much more efficient than the MC simulation.

REFERENCES

[1] A. Agarwal, D. Blaauw, S. Sundareswaran, V. Zolotov, M. Zhao, K. Gala, and R. Panda, "Path-based statistical timing analysis considering inter- and intra-die correlations," in *Proc. ACM Int. Workshop Timing Issues Specification and Synthesis Digital Systems*, Monterey, CA, 2002, pp. 16–21.

[2] A. Devgan and C. Kashyap, "Block-based static timing analysis with uncertainty," in *Proc. Int. Conf. Computer-Aided Design*, San Jose, CA, 2003, pp. 607–614.

[3] S. Bhardwaj, S. B. Vrudhula, and D. Blaauw, "Tau: Timing analysis under uncertainty," in *Proc. Int. Conf. Computer-Aided Design*, San Jose, CA, 2003, pp. 615–620.

[4] A. Agarwal, D. Blaauw, and V. Zolotov, "Statistical timing analysis for intra-die process variations with spatial correlations," in *Proc. Int. Conf. Computer-Aided Design*, San Jose, CA, 2003, pp. 900–907.

[5] H. Chang and S. S. Sapatnekar, "Statistical timing analysis considering spatial correlations using a single pert-like traversal," in *Proc. Int. Conf. Computer-Aided Design*, San Jose, CA, 2003, pp. 621–625.

[6] C. Visweswariah, K. Ravindran, K. Kalafala, S. G. Walker, and S. Narayan, "First-order incremental block-based statistical timing analysis," in *Proc. Design Automation Conf.*, San Diego, CA, 2004, pp. 331–336.

[7] C. Ebeling and B. Lockyear, "On the performance of level-clocked circuits," in *Proc. Advanced Research VLSI*, Chapel Hill, NC, 1995, pp. 342–356.

[8] T. G. Szymanski and N. Shenoy, "Verifying clock schedules," in *Proc. Int. Conf. Computer-Aided Design*, Santa Clara, CA, 1992, pp. 124–131.

[9] N. Shenoy, "Timing issues in sequential circuits," Ph.D. dissertation, Dept. Elect. Eng. Comput. Sci., Univ. California, Berkeley, 1993.

[10] N. Shenoy, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "Graph algorithms for clock schedule optimization," in *Proc. Int. Conf. Computer-Aided Design*, Santa Clara, CA, 1992, pp. 132–136.

[11] L. Zhang, Y. Hu, and C. C. Chen, "Statistical timing analysis in sequential circuit for on-chip global interconnect pipelining," in *Proc. Design Automation Conf.*, San Diego, CA, 2004, pp. 904–907.

[12] M. C. T. Chao, L. C. Wang, K. T. Cheng, and S. Kundu, "Static statistical timing analysis for latch-based pipeline designs," in *Proc. Int. Conf. Computer-Aided Design*, San Jose, CA, 2004, pp. 468–472.

[13] K. A. Sakallah, T. N. Mudge, and O. A. Olukotun, "$checkT_c$ and $mint_c$: Timing verification and optimal clocking of synchronous digital circuits," in *Proc. Int. Conf. Computer-Aided Design*, Santa Clara, CA, 1990, pp. 552–555.

[14] ——, "Analysis and design of latch-controlled synchronous digital circuits," in *Proc. Design Automation Conf.*, Orlando, FL, 1990, pp. 111–117.

[15] T. G. Szymanski, "Computing optimal clock schedules," in *Proc. Design Automation Conf.*, Anaheim, CA, 1992, pp. 399–404.

[16] W. J. Krzanowski, *Principles of Multivariate Analysis*. New York: Oxford Univ. Press, 2000.

[17] J. L. Neves and E. G. Friedman, "Optimal clock skew scheduling tolerant to process variations," in *Proc. Design Automation Conf.*, Las Vegas, NV, 1996, pp. 623–628.

[18] T. H. Cormen, C. E. Leiserson, and R. H. Rivest, *Introduction to Algorithms*. Cambridge, MA: MIT Press, 1989.

[19] C. E. Clark, "The greatest of a finite set of random variables," *Oper. Res.*, vol. 9, no. 2, pp. 145–162, 1961.

[20] R. Tarjan, "Enumeration of the elementary circuits of a directed graph," *J. Comput.*, vol. 2, no. 3, pp. 211–216, Sep. 1973.

[21] T. M. Burks and K. A. Sakallah, "Optimization of critical paths in circuits with level-sensitive latches," in *Proc. Int. Conf. Computer-Aided Design*, San Jose, CA, 1994, pp. 468–473.

[22] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to Theory of NP-Completeness*. New York: Freeman, 1979.

[23] C. Demetrescu and I. Finocchi, "Combinatorial algorithms for feedback problems in directed graphs," *Inf. Process. Lett.*, vol. 86, no. 3, pp. 129–136, 2003.

[24] N. Maheshwari and S. S. Sapatnekar, "Optimizing large multi-phase level-clocked circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 18, no. 9, pp. 1249–1264, Sep. 1999.

Reducing Memory Energy Consumption of Embedded Applications That Process Dynamically Allocated Data

V. De La Luz, M. Kandemir, and I. Kolcu

***Abstract*—The authors present a set of strategies for reducing the energy consumption in a multibank memory architecture using an energy conscious dynamic memory allocation/deallocation. Applications that make dynamic memory allocations are used very frequently in embedded computing and mobile networking areas. The authors' strategies focus on such applications exclusively and cluster dynamically allocated data with a temporal affinity in the physical address space such that the data occupy a small number of memory banks. The remaining banks can be shut off, saving energy. One of the authors' strategies also employs dynamic data migration to further increase energy savings. The experimental results show that all the authors' strategies save a significant amount of energy.**

***Index Terms*—Banked memories, compilers, dynamic data allocation, energy minimization, pointer-intensive codes, virtual memory management.**

Manuscript received October 22, 2004; revised March 7, 2005 and May 26, 2005. This work was supported in part by the National Science Foundation Faculty Early Career Development (NSF CAREER) under Award 0093082 and a Grant from the Gigascale Silicon Research Center (GSRC). This paper was recommended by Associate Editor M. F. Jacome.

V. De La Luz is with the Banco de Mexico, Mexico C.P. 06059, Mexico.

M. Kandemir is with Pennsylvania State University, University Park, PA 16802 USA.

I. Kolcu is with the University of Manchester, Manchester MB 9PL, U.K.

Digital Object Identifier 10.1109/TCAD.2005.859521