Invited Paper

From Process Variations to Reliability: A Survey of Timing of Digital Circuits in the Nanometer Era

Bing Li\textsuperscript{1,a)} Masanori Hashimoto\textsuperscript{2,b)} Ulf Schlichtmann\textsuperscript{1,c)}

Received: September 28, 2017

Abstract: In advanced technology nodes, transistors and interconnects with shrinking physical dimensions suffer large process variations during manufacturing and are prone to reliability issues. These underlying changes require an overhaul of the design methodologies for digital circuits. In this paper, we provide an overview of techniques introduced recently to analyze the effect of uncertainty in manufacturing and reliability issues of devices due to the diminishing feature size. These techniques range from variation/aging modeling to circuit-level analysis. In addition, active techniques to counter these effects, such as clock skew tuning and voltage tuning are also covered in this paper.

Keywords: timing analysis, process variations, aging, hierarchical timing, clock skew, noise, circuit tuning

1. Introduction

Digital circuits contain two types of gates. Logic gates, e.g., AND, OR and NOT gates, implement the function of the circuit. The sequential gates, e.g., flip-flops and latches, do not make any contribution to the logic function directly. Instead, they are used to synchronize intermediate steps of the computation. An example of a digital circuit is shown in Fig. 1.

A sequential gate, henceforth with flip-flop as example, is only activated at a given moment to store the data from its input. Except this exact moment, the stored data and the output of the flip-flop do not change, no matter what happens at its input. Consequently, flip-flops sitting on combinational paths in a sequential circuit partition the circuit into separate logic/combinational functional blocks. The data at the outputs of a combinational block are latched into the flip-flops connected to them at a given moment. Thereafter, any changes at the outputs of the combinational block and thus the inputs of the flip-flops do not affect the data stored inside the flip-flops, until the next latching moment comes. Since the outputs of flip-flops are connected to the inputs of combinational blocks, the starting time of logic computation can thus be synchronized.

To coordinate the activities of logic blocks, a clock signal is generated and distributed to all flip-flops. The flip-flops are activated at a clock edge, e.g., the rising clock edge henceforth. Since the clock signal reaches all flip-flops at the same time, the latching activities of flip-flops trigger signal transitions at the inputs of logic blocks simultaneously. Because at each rising clock edge new computations are initiated, the performance of a circuit is thus often represented by its clock frequency, indicating how many input data can be processed in a second by the circuit.

To improve circuit performance, the easiest way is to boost its clock frequency, or, equivalently, shorten the clock period. In 2004 the International Technology Roadmap for Semiconductors (ITRS) has predicted that, driven by advances in manufacturing technology, the clock frequency would increase by 21\% every year, jumping from about 2.9 GHz in 2002 to 33.4 GHz in 2015. This optimistic estimation has been, however, adjusted in 2013, predicting that the clock frequency may reach 5.9 GHz in 2015, leading to only 4\% annual growth rate. This slowdown of clock frequency increase is caused by several factors, including process variations, power consumption, noise as well as reliability and aging issues.

In a digital circuit, the latest time a signal becomes stable is after it travels through the combinational path with the largest delay, called the longest path henceforth. The delay of this path is equal to the sum of all the delays of logic gates and interconnects on the path. These delays are, however, not fixed values in reality due to the imperfection in manufacturing processes. In fact, it is not possible to deliver logic gates or interconnects with the exact design parameters, e.g., physical dimensions, from the manufacturing process. Instead, the parameters of logic gates and interconnects vary in different chips of the same design, a phenomenon called process variations \cite{2}, so that some of them have

\footnotesize
\begin{itemize}
  \item \textsuperscript{1} Technical University of Munich, 80333 Munich, Germany
  \item \textsuperscript{2} Osaka University, Osaka, 565–0871 Japan
  \item \textsuperscript{a)} b.li@tum.de
  \item \textsuperscript{b)} hasimoto@ist.osaka.ac.jp
  \item \textsuperscript{c)} ulf.schlichtmann@tum.de
\end{itemize}

© 2018 Information Processing Society of Japan
large delays but others may have small delays.

Process variations have been existing since the beginning of the semiconductor industry. They did not significantly affect the design flow in the past because compared with the dimensions of the logic devices and interconnects, the variations were relatively small, so that they could be dealt with by allocating performance margins easily. In advanced technology nodes, these variations have become relatively large, e.g., reaching a one-sigma variation of 15.7% in 70 nm manufacturing node [2], so that they cannot easily be compensated anymore by allocating excessive performance margins. In addition, local variations have become more relevant compared to global variations. Some local variations are purely random—therefore the traditional worst-case-based design methodology cannot properly handle them.

Process variations cannot be determined before real manufacturing happens and in different manufactured chips their final effects are different. During design phase, timing analysis is still required because designers need to evaluate how the manufactured chips perform after experiencing process variations. Consequently, the performances of logic gates and interconnects need to be modeled statistically as random variables according to the manufacturing data from foundries, leading to a boom of research on statistical timing analysis (SSTA) [3].

While the feature size is still being scaled down continuously to smaller geometries to enable the development of more complex and powerful integrated circuits (ICs) with a higher transistor density, unfortunately voltage scaling has essentially stopped due to power limitations, resulting in the breakdown of the Dennard Scaling concept [4], which specifies that propagation delays in every new process generation would be reduced by 30%, enabling about 30% higher frequencies. For example, the first microprocessor in 1971, the Intel 4004, ran at 740 kHz. The Intel Pentium 4 introduced in 2004 in 90 nm technology was capable of working at 3.4 GHz, leading to a clock speed increased about 4,600 times, or roughly 29% annually. In advanced nanometer technology nodes, leakage power concerns, however, prevent a further downsizing of threshold voltages $V_{th}$. Performance could be improved by increasing $V_{dd}$, which, unfortunately, would result in increased dynamic power consumption especially in designs with a high clock frequency, and is thus not an option either. This situation becomes even more complicated when noise inside the chip is considered, which also demands a large margin in supply voltages.

Similar to process variations, another effect that has been existing for years but has become practically relevant since about 10 years is aging of transistors as well as the ensuing reliability issues. When currents flow through transistors, stress is accumulated on them. Gradually, chips may become slower compared with the fresh state right after manufacturing. This degradation may also pose a big challenge in some fields such as automotive and medical industries, where safety-critical applications require stable and reliable circuit components. To maintain the designated performance, further timing/voltage margins may need to be allocated, making the design task more challenging.

In this paper, we survey the state of the art of the research on timing of digital circuits considering new design challenges emerging in nanometer manufacturing nodes. These challenges may lead to significant changes in the long-established design flow for integrated circuits.

2. Timing Constraints

Since the logic blocks in a digital circuit are activated by the clock signal periodically, the logic computation inside of a logic block must finish its execution before the data at its outputs are latched into the flip-flops of the next stage. The time difference between two consecutive activations of a logic block is the clock period $T$.

Since a flip-flop also needs some time to store the data correctly, the data at its input must be stable within a small time window before the rising clock edge, i.e., setup time $t_{su}$, before the rising clock edge. Similarly, the data must stay stable hold time $t_{th}$ after the rising clock edge. As illustrated in Fig. 2, the data must be stable in a window surrounding the rising clock edge to guarantee a correct latching behavior of the flip-flop. Since the latest time a data signal arrives at a flip-flop is determined by the longest combinational path in the logic blocks, and the earliest change is caused by the shortest combinational path in the logic blocks, the timing constraints considering all logic blocks in a circuit can be written as

$$\max_{p \in P} \{d_{pq} + d_p + t_{su}\} \leq T \tag{1}$$

$$\min_{p \in P} \{d_{pq} + d_p\} \geq t_{th} \tag{2}$$

where $p$ is a path with delay $d_p$ from the set $P$ representing all the paths in the combinational blocks. $d_{pq}$ is the delay of the flip-flop. To verify the timing constraints (1) and (2), the delay of combinational paths need to be calculated. However, it is timing-consuming and unnecessary to enumerate all combinational paths. Instead, the delay information can be merged by propagating only the maximum or minimum delay information forward in the circuit. To trace the maximum delay of combinational paths to verify the setup time constraint (1), two atomic operations, $\max$ and $\sum$, are applied recursively in timing analysis. This concept is illustrated in Fig. 3, where the arrival times of signals at nodes are denoted as $a_1$–$a_5$ and the delays of logic gates and interconnects are denoted as $d_1$–$d_4$. When a combina-
tional component is passed, its delay is added to the arrival time. When multiple arrival times converge at a node, the maximum of them is calculated and propagated further. Since logic functions are not considered, the task of timing analysis becomes merely to efficiently traverse a graph representing the circuit structure and delays.

3. Timing with Process Variations

In timing analysis, the delays of the longest and shortest paths should be calculated to verify Eqs. (1) and (2). In the case that all gate delays are fixed values, this is not a challenging task. When process variations are considered, however, the computational complexity increases extraordinarily, because the max and sum operations become statistical. Moreover, the results of these operations should maintain the correlation to the other random variables so that the same formulas of the max and sum operations can be applied recursively. For example, the result $a_5$ of max and sum operations at node 4 in Fig. 3 should be represented in the same form as $a_4$ so that the sum operation can be applied to calculate $a_5$ similarly.

3.1 Process Variation Categorization

Process variations exist inherently in manufacturing. As the dimensions of transistors are being scaled down, these variations, however, do not follow the scaling linearly but lag behind the pace. Consequently, the ratio of process variations to the nominal dimensions of devices becomes larger, to the degree of affecting the delays of logic gates significantly [2]. Since process variations result in transistors with dimensions spreading in much larger relative ranges than previously, the delays of logic gates and interconnects need to be modeled as random variables instead of as fixed values in nanometer manufacturing nodes.

Process variations can be classified into different categories, as shown in Fig. 4. Systematic variations, e.g., the randomness in interconnect metal thickness, are caused in part by design characteristics such as the density of interconnects on a metal layer, so that they can be extracted and modeled according to the design information even before manufacturing. The variations of this type can be directly incorporated into post-OPC (Optical Proximity Correction) timing analysis to improve the accuracy [5].

Non-systematic variations, however, cannot be determined before manufacturing, since they are the results of the inaccuracy in process control during manufacturing. Consequently, they can only be modeled as random variables when the circuit is designed. Examples of these variations are those in doping density and layout independent metal thickness.

Non-systematic variations can be partitioned further into die-to-die variations (interdie variations) and within-die variations (intradie variations) [6]. Die-to-die variations affect all devices and interconnects on a die equally. For example, the chips in the center area of a wafer are normally faster than the chips far away from the center. Within-die variations have different effects on devices or interconnects inside a die, leading to deviation between device parameters inside a chip after manufacturing. If the variations of two parameters exhibit no correlation, within-die variations become a purely random effect, such as the random distortion caused by lens during photolithography and the purely random variations in doping.

3.2 Correlation Modeling

As discussed above, some components of process variations may affect more than one device in a chip. Considered as a whole, performances of devices thus exhibit a correlation. Since the uncertainties during manufacturing process vary continuously, process variations also exhibit a proximity effect, meaning that the smaller the distance is, the larger the correlation becomes [7].

Several models have been proposed to express the correlation between process parameters. The quadtree model in Refs. [8], [9] is illustrated in Fig. 5. In this model, different layers are used to represent the random components shared by devices in areas of different scales. For each grid cell, a random variable is assigned. Therefore, the random variable at the top layer models the component shared by all devices in the chip. As the sizes of grid cells decrease in lower layers, the corresponding variables model finer but more locally restricted correlation. This quadtree model, however, does not model the local correlation uniformly. For example, the distances from (2,4) to (2,1) and to (2,13) are equal. In this model, devices in (2,4) and (2,1) share the same variable at layer 1, but this is not the case for the devices in (2,4) and (2,13).

Another correlation model is proposed in Ref. [10] and is illustrated in Fig. 6. In this model, the die area is partitioned into a uniform grid with $n$ cells, and a random variable is assigned for each grid. The total $n$ random variables are correlated, containing all the components in Fig. 4. To simplify statistical computations in timing analysis, these variables are decomposed into linear combinations of independent random variables, using algorithms such as principal component analysis (PCA) [11]. This correlation model is very flexible, because it can handle any correlation between process parameters rather accurately. This model is revised in Ref. [12] where hexagonal grid cells are used to partition the die area.
The correlation models discussed above are all first-order and only sufficient to model the dependency between Gaussian random variables. To incorporate dependency information of a higher order, methods like independent component analysis [13] have been applied, as in Refs. [14], [15].

### 3.3 Statistical Timing Analysis

With random variables representing process parameters decomposed into linear combinations of independent random variables, the delay of a gate can also be expressed similarly. For example, the canonical delay model in Ref. [16] expresses two gate delays as

\[
A = a_0 + \sum_{i=1}^{n} a_i v_i + a_r v_r
\]

\[
B = b_0 + \sum_{i=1}^{n} b_i v_i + b_r v_r
\]

where \(v_i\) and \(v_r\) are random variables representing the purely independent variations of the gate delays. \(v_i\) are independent random variables from decomposition shared by all gate delays. \(a_0, b_0, a_i, b_i, a_r,\) and \(b_r\) are constant coefficients of the decomposed components, indicating how much a delay is affected by them. The sharing of decomposed components establishes the correlation between two gate delays as

\[
\text{Cov}(A, B) = \sum_{i=1}^{n} a_i b_i \text{Cov}(v_i, v_i) = \sum_{i=1}^{n} a_i b_i \sigma_{v_i}^2.
\]

During statistical timing analysis, the sum of \(A\) and \(B\) is computed easily as

\[
A + B = (a_0 + b_0) + \sum_{i=1}^{n} (a_i + b_i) v_i + (a_r + b_r) v_r
\]

\[
= m_0 + \sum_{i=1}^{n} s_i v_i + s_r v_r
\]

where \(s_i\) is computed by matching the variances of \(s_i v_i\) and \(a_i v_i + b_i v_i\).

The computation of the maximum of \(A\) and \(B\), denoted as \(\text{max}(A, B)\), is, however, more challenging. In Ref. [16], the concept of tightness probability \(T_P\) is introduced to represent the probability a variable is greater than another. When \(A\) and \(B\) are assumed as Gaussian, \(T_P\) can be computed as

\[
T_P = \text{Prob}(A \geq B) = \Phi\left(\frac{m_0 - b_0}{\sqrt{\sigma^2}}\right)
\]

where \(\Phi\) is the cumulative distribution function of the standard Gaussian distribution. \(\theta = \sqrt{\sigma^2 + \sigma^2 - 2\text{Cov}(A, B)}\), with \(\sigma^2\) and \(\sigma^2\) denoting the variances of \(A\) and \(B\) respectively.

According to Ref. [17], the mean (\(\mu\)) and variance (\(\sigma^2\)) of \(\text{max}(A, B)\) can be computed by

\[
\mu = T_P a_0 + (1 - T_P) b_0 + \Phi\left(\frac{m_0 - b_0}{\sqrt{\sigma^2}}\right)
\]

\[
\sigma^2 = T_P (\sigma_A^2 + \sigma_B^2) + (1 - T_P) (\sigma_A^2 + b_0^2)
\]

\[
+ (a_0 + b_0) \Phi\left(\frac{m_0 - b_0}{\sqrt{\sigma^2}}\right) - \mu^2
\]

where \(\Phi\) is the probability density function of the standard Gaussian distribution.

In order to apply the \text{max} and \text{sum} operations iteratively to propagate arrival times, \(\text{max}(A, B)\) is approximated in the canonical linear model as

\[
\text{max}(A, B) \approx m_0 + \sum_{i=1}^{n} m_i v_i \quad (10)
\]

where \(m_0\) is equal to \(\mu\). \(m_i\) is computed by \(m_i = T_P a_i + (1 - T_P) b_i\). \(m_r\) is computed by matching the variance of the linear form Eq. (10) and \(\sigma^2\) in Eq. (9).

The example above has exposed the difference between statistical timing analysis (SSTA) and traditional static timing analysis (STA). In STA, the maximum of two arrival times is only a simple comparison of two numbers. But in SSTA, it involves a lot computation not only for the mean and the variance of the result but also for the linearized expression Eq. (10), which must guarantee that the approximation form maintains the correlation between \(\text{max}(A, B)\) and any other variable so that the \text{max} and \text{sum} computations can be applied recursively during arrival propagation with the same formulas as in Eqs. (6)–(10) [10], [16]. The \text{sum} operation Eq. (6) is easier but is still computationally much more demanding compared to the sum of two simple numbers in STA. This complexity makes SSTA relatively slow and thus has hampered its application in industrial design flows.

The linear timing analysis methods above assume that gate delays are approximated as Gaussian random variables. This limitation has been relaxed in Refs. [18], [19], [20] by representing timing properties as quadratic functions of independent Gaussian random variables. Furthermore, gate delays can also be expressed as linear combinations of non-Gaussian variables, as in the method in Refs. [14], [15], while the method in Ref. [21] proposes a general framework to incorporate linear/nonlinear combinations of Gaussian and non-Gaussian random variables.

The methods above are block-based because at each component only the maximum or minimum of the variables is propagated. Path-based methods have also been explored for statistical timing analysis, e.g., in Refs. [22], [23]. These methods can only process a given set of combinational paths, which, however, are still not easy to identify accurately from the circuit [24].

### 3.4 Hierarchical Statistical Timing Analysis

To count the immense computational effort required for timing analysis considering process variations, hierarchical statistical timing analysis has been investigated to accelerate system-level timing analysis.

Hierarchical timing analysis splits timing analysis into two
steps. At first, timing properties inside submodules are extracted individually. Afterwards, the timing models of all the submodules in a circuit are combined together to perform timing analysis of the whole design.

A timing model contains interface timing information of a module, which is unusually the delay information of: 1) direct paths from inputs to outputs, type A in Fig. 7; 2) paths from inputs that can reach a flip-flop or a latch inside the module, type B; 3) paths from an internal flip-flop or a latch to an output, type C; 4) paths between flip-flops or latches, type D.

For static timing analysis without consideration of process variations, timing model extraction has been explored in Ref. [25] to extract black-box models, which only contain interface timing information. The methods in Refs. [26], [27], [28], [29] apply graph transformation operations to merge nodes and edges representing the timing information of the original circuit, leading to gray-box timing models because timing information inside the modules is exposed. For sequential circuits, Interface Logic Model (ILM) in Ref. [25] keeps all the combinatorial paths between input ports to the first-level flip-flops and from the last-level flip-flops to the output ports. The Extracted Timing Model (ETM) [25], however, collapses all such paths to reduce the size of timing models, sacrificing the flexibility of keeping the original parasitics information associated with the original logic gates and interconnects. For latch-based circuits, all latches can be retained in the timing model as in Ref. [27], or the depth of transparency from the input ports to internal latches and from the internal latches to the output ports are assumed as given [30], so that the number of latches potentially traveled transparently can be determined.

When process variations are considered, timing model extraction methods for combinational circuits and sequential circuits with flip-flops in Refs. [25], [26], [27], [28], [29] can be applied similarly, but with the maximum and sum computations replaced by the corresponding statistical versions. For sequential circuits with latches, the depth of transparency becomes statistical [31], [32]. Therefore, timing models should be extracted with respect to the probabilities of transparency depths [33], [34], which are affected by the statistical path delays in the original circuit.

When integrating extracted statistical timing models into timing analysis of the whole circuit, a special challenge should be met when process variations are considered. At the top level, the die area occupied by each module is partitioned to model the statistical variations when generating timing models, such as using the uniform grid in Ref. [10]. When the extracted models are placed together, the grids inside them may not be aligned, as illustrated in Fig. 8. Therefore, the relation between these grids should be established. In Refs. [35], [36], [37] by using the coefficient matrices of submodules and the top grid directly.

3.5 Clock Skew and Jitter Aware Setup and Hold Verification

3.5.1 Setup and Hold Verification

Slack computation for setup and hold constraints can be explained using Fig. 9, where slack is the timing margin between required signal arrival time to meet timing constraints and actual arrival time. All arrival times and element delays are supposed to be expressed and computed with the canonical form Eq. (3).

First, the procedure to calculate slack for setup constraint is explained as follows.

(1) Take the source of clock network as an origin, and set the arrival time at the clock source to 0. Then perform SSTA, and obtain the latest arrival time $t_1$ at input D of each FF. Here, the clock signal propagates to the combinational circuit through CLK-to-Q path in FFs, and hence each FF is regarded as a combinational cell, and CLK-to-Q delay is added to the arrival time.

(2) Take the clock source as an origin and set the arrival time at the clock source to clock cycle $T$. Then obtain the arrival time $t_2$ at clock input CLK of each FF.

(3) Slack for setup constraint $S_{su}$ is calculated for every FF.

$$S_{su} = t_2 - t_1 - T_{su}$$

where $T_{su}$ is the setup time of the FF.

Similarly, slack for hold constraint can be computed with the following procedure.
(1) Take the clock source as an origin, and set the arrival time at the clock source to 0. Then perform SSTA, and obtain the earliest arrival time \( t_1 \) at input D of each FF. The arrival time \( t_2 \) at clock input CLK of each FF is also computed.

(2) Slack for hold constraint \( S_{\text{hold}} \) is calculated for every FF.

\[
S_{\text{hold}} = t_1 - t_2 - T_{\text{hold}} \tag{12}
\]

where \( T_{\text{hold}} \) is the hold time of the FF.

### 3.5.2 Common Path Pessimism

When two arrival times that share common paths are mixed or subtracted, this structural correlation should be considered. This situation arises when setup and hold slacks are computed by subtracting two arrival times of launch and capture paths at each FF because both the launch and capture paths share some clock buffers and the common path inherently exists. The problem arises from subtractions \( t_2 - t_1 \) in Eq. (11) and \( t_4 - t_3 \) in Eq. (12). The path corresponding to \( t_1 \) partially overlaps the paths of \( t_1 \) inside the clock distribution network (CDN).

Let us examine a simple example of Fig. 10. For simplicity, this example assumes all variations are static and uncorrelated. In this case, \( \mu(t_1), \sigma(t_1), \mu(t_2) \) and \( \sigma(t_2) \) are estimated to be 110 ps, 6.6 ps, 120 ps and 5 ps respectively, and \( T_{\text{ww}} \) is 0 ps. When Eq. (11) is computed without considering the common path, \( \mu(S_{\text{ww}}) \) and \( \sigma(S_{\text{ww}}) \) are 10 ps (= 120 - 110) and 8.3 ps (= \( \sqrt{6.6^2 + 6.6^2} \)). In contrast, the correct \( \mu(S_{\text{ww}}) \) and \( \sigma(S_{\text{ww}}) \) considering the common path are 10 ps and 4.2 ps. In this case, the correct values are estimated by virtually eliminating the leftmost inverter, namely, \( \mu(t_1), \sigma(t_1), \mu(t_2) \) and \( \sigma(t_2) \) becomes 60 ps, 4.2 ps, 70 ps and 0 ps respectively. Ignoring the structural correlation over-estimates the standard deviation by 98%.

Common path pessimism reduction has been studied to tackle this common path problem. References [38], [39] proposes a method to solve this problem from the viewpoint of path-based analysis. First, the authors determine a target path for timing verification, and they obtain the pair of FFs which are the source and sink of the path. Next, the clock distribution network is traced from the FFs to the clock source, and the branching point of the clock paths of the FFs is found. Finally, by regarding the meeting point as the source of clock signal, the common path is eliminated and setup and hold verification is performed. However, these path-based methods may suffer from excessive CPU time when the number of the candidates for critical path is large.

On the other hand, the block-based analysis, which does not suffer from the number of paths, can cope with common paths by increasing the number of random variables [40]. Let us analyze Fig. 11 and calculate arrival times \( t_1 \) and \( t_2 \) as an example. In this situation, both \( t_1 \) and \( t_2 \) contain the delays of \( c_{ba} \) and \( c_{bh} \) in common, which causes the structural correlation problem. To cope with this problem involved in the slack computation, individual random variables are assigned to each clock driver. As an example, assume a case that each clock buffer \( c_{ba} \), \( c_{bh} \), and \( c_{bh} \) has its random variable \( v_{cb}, \ v_{cb}, \) and \( v_{cb} \). By extending Eq. (3) with \( v_{cb}, \ v_{cb}, \) and \( v_{cb}, \ t_1 \) and \( t_2 \) are expressed in Eqs. (13) and (16).

\[
t_1 = a_{1,0} + \sum_{j=1}^{m} a_{1,j}v_{fj} + a_{1,v}v_{v1} + c_{1,a}v_{cba} + c_{1,b}v_{cb} + c_{1,c}v_{cb} \tag{13}
\]

\[
t_2 = a_{2,0} + \sum_{j=1}^{m} a_{2,j}v_{fj} + a_{2,v}v_{v2} + c_{2,a}v_{cba} + c_{2,b}v_{cb} + c_{2,c}v_{cb} \tag{16}
\]

where \( v_{v1} \) and \( v_{v2} \) are the random variables to represent random components of the gates except the clock buffers, and \( c_{1,[a-c]} \) and \( c_{2,[a-c]} \) are the coefficients which represent the magnitudes of \( v_{cb}, v_{cb} \) in \( t_1 \) and \( t_2 \) respectively. In this example, \( c_{2,c} \) is 0. Here, \( t_2 \) can be obtained by summing the delays of \( c_{ba} \) and \( c_{bh} \). On the other hand, all paths to the D terminal of the topmost FF include \( c_{ba} \), and hence \( t_1 \) includes the delay of \( c_{ba} \). In this case, \( c_{1,a} \) and \( c_{2,a} \) are identical, which means that the terms of \( v_{cba} \) in \( t_1 \) and \( t_2 \) are canceled out when \( t_2 - t_1 \) is computed. By this way, the structural correlation is considered automatically thanks to the assignment of individual random variables to clock drivers.

When the analyzed circuit is large, the number of random variables could be a problem in terms of SSTA run time and memory. However, this problem can be mitigated exploiting the tree structure of the CDN. In the CDN, the clock drivers in the upstream network are often included in the common path. On the other hand, the number of upstream drivers is much smaller than that of downstream drivers. Therefore, assigning random variables to clock drivers from the clock source with breadth first search, as long as the computational cost permits, efficiently mitigates the problem of the structural correlation. If the available computational resources are large enough, each clock driver has its own random variable respectively, which can achieve a full consideration of the structure correlation on CDN. On the other hand, structural correlation is also largely considered even if random variables are assigned only to the upstream drivers because they are shared by downstream components.

Figure 12 shows the accuracy of setup slack estimation and SSTA run time. The cases of 1 and 2019 random variable(s) correspond to (M1) and (M2), respectively. There is a tradeoff between accuracy and run time. As the number of random variables increases, the worst setup slack defined as \( \mu + 3\sigma \) converges and
Supply noise. To consider the correlation and compute even though large clock jitter arises due to resonant power points out that this correlation contributes to mitigate timing vio-
clock cycles need to be taken into consideration. Reference [42] Eq. (11), the correlation between noise waveforms of successive
circuit workload, and hence delay variation due to supply noise is
3.5.3 Impact of Supply Noise on Timing
Power supply noise varies in every clock cycle depending on circuit workload, and hence delay variation due to supply noise is different temporally. In addition, non-uniform circuit switchings in space and topology of power distribution network make supply noise different in space. Therefore, clock arrival time varies in time and space, which causes clock jitter and clock skew.

Such impact of supply noise on timing can be considered consistently by adding random variables modeling noise into Eq. (3) [40], [41]. Let us explain the statistical noise model using Fig. 13. To express dynamic waveforms within a cycle, a clock cycle is partitioned into several time spans, and a representative voltage, e.g., average is computed. Afterwards, a random variable is assigned to power supply or ground voltage at each time span and at each spatial grid. The voltage value at every clock cycle is treated as a different sample. Figure 13 shows an example when the voltage at position \((x, y)\) is divided into five time spans and its random variables are denoted as \(V_{x,y,0}\) to \(V_{x,y,4}\). To compute \(S_{iu}\) in Eq. (11), the correlation between noise waveforms of successive clock cycles need to be taken into consideration. Reference [42] points out that this correlation contributes to mitigate timing viola-
tion even though large clock jitter arises due to resonant power supply noise. To consider the correlation and compute \(S_{iu}\) appropriately, the origin of temporal division is set to the clock launch timings at the source of the CDN. The sample, which is divided into several time spans, is extended in time so that the clock and signal propagations both in the launch path \(t_1\) and the capture path \(t_2\) are included in a single sample. Therefore, a correlation between the variables of successive clock cycles such as between \(V_{x,y,0}\) and \(V_{x,y,3}\) in Fig. 13 is naturally considered. With this modeling, the correlation between noise waveforms of successive clock cycles, which is demanded to accurately compute \(S_{iu}\) in Eq. (11), can be appropriately modeled.

3.6 Timing with Setup-Hold Time Interdependency
Process variations may affect path delays in a sequential circuit statistically. If the delay of a path in a manufactured chip exceeds the clock period minus the setup time of a flip-flop, a timing violation is assumed to happen. In reality, however, the data may still be latched into the flip-flop correctly even in view of a violation of the setup time constraint (1), but the clock-to-q de-
lay may increase significantly, leading to a delay increase of the combina-
tional paths of the next stage. If those paths are short, the increased delay can be absorbed automatically. As shown in Fig. 14 (a), a flip-flop can work with different setup-hold time combinations with different clock-to-q delays. In the traditional definition, the setup time and hold time are simply approximated as the point A in Fig. 14 (b), losing the flexibility of the flip-flop completely.

The setup-hold interdependency has been investigated in Refs. [43], [44] to exploit the compensation between setup time and hold time combinations with respect to a given clock-to-q delay. In addition, a method based on Euler-Newton tracing is introduced in Refs. [45], [46] to characterize the delay curves of flip-flops. These methods, however, do not consider the relation between clock-to-q and setup/hold times, leading to a limited performance improvement. To remove this limitation the method in Ref. [47] uses a quadratic programming model to calculate the optimal clock period directly, but it is incapable of solving the high-order programming problem for large circuits. To simplify the three dimensional model, the method in Ref. [48] applies an analytic function and calculates the minimum clock period by it-
erations. In addition, the method in Ref. [49] approximates the three dimensional delay surface using linear planes. In calcul-
lating the minimum clock period of a circuit, this method, how-
ever, splits the problem into two-dimensional problems. Further-
more, the method in Ref. [50] proposes an efficient algorithm to capture timing violations in a circuit very efficiently, but only the relation between clock-to-q delay and setup slack is con-
sidered in this method. Most recently, the method in Ref. [51]
models the delay surface using piecewise polygons and performs the timing analysis using integer linear programming together with reduction techniques. In addition, process variations are considered together with setup-hold interdependency for flip-flop models in Ref. [52], but how to incorporate the generated statistical models in statistical timing analysis is still open.

4. Aging

With the diminishing dimensions of devices, a new challenge that has been around for a long time, aging of ICs, has also started to attract increasing attention since about 10 years. The term “aging” covers a number of effects impacting devices in nanometer manufacturing nodes, most prominently Hot Carrier Injection (HCI), Negative Bias Temperature Instability (NBTI), Electromigration (EM) and Time-Dependent Dielectric Breakdown (TDDB). Moreover, Positive BTI (PBTI) is also increasingly becoming a consideration. For example, NBTI and HCI result in an increase of $V_{th}$ or decrease of $I_{on}$, respectively, causing a loss of performance, up to 20% over time.

With aging the performance of devices deteriorates over time. The analysis of aging effects is challenging, since the amount of aging depends on a number of factors, among which are $V_{dd}$, Temperature, frequency and also the amount of switching or the specific voltage level a transistor experiences. Accordingly, both the specific structure of a circuit and its real usage contribute to the results of aging. Aging analysis is further complicated by the fact that NBTI degradation partly recedes once a stress condition is removed. This phenomenon is known as “recovery effect.”

Aging analysis traditionally has focused on potential effect of aging on individual transistors. As a result of such transistor analysis, an overall safety margin was added into the timing signoff to guarantee the correct functionality of the chips. This coarse-grained approach is increasingly less feasible because, on the one hand, aging has become more pronounced, and on the other hand, the timing budget is becoming increasingly tight, thus not being able to tolerate a large timing margin anymore. Consequently, it has become desirable to analyze the aging effect of a given circuit specifically to provide more fine-grained information.

To calibrate aging effects, transistor-level simulation is required. Though traditional transistor aging models are accurate, they are too slow for analyzing large ICs. To solve this problem, research has been undertaken to analyze HCI and NBTI and the factors influencing them, and to develop timing models and algorithms for aging analysis on gate level, leading to a speedup of aging analysis by orders of magnitude [59], [60], [61]. The AgeGate model [53], [62], [63] is probably still the state of the art in gate-level aging analysis today. This aging model can handle both HCI and NBTI. It is independent of the current use profile, defined by $V_{dd}$ and temperature $T$ over the lifetime of the IC. It also incorporates the aging effect of each transistor in a gate individually and the aging of output slope, both of which are required for accurate timing analysis. The approach of AgeGate builds on the canonical delay model [16] so that it can be integrated into standard industrial STA-based timing signoff flows smoothly [64].

The AgeGate model was extended later to take module-level aging analysis into account, speeding up aging analysis further, while maintaining a good accuracy [65]. This approach extracts a timing graph of a module from the original gate-level netlist. Thereafter, the graph is pruned significantly. For example, any path that can never become critical under aging and process variations is not considered in the analysis. The concept of this pruning is illustrated in Fig. 15. Empirical results have confirmed that the number of timing paths can be reduced by up to four orders of magnitude or even more for aging analysis. The reduced timing graphs of modules can not only be used to accelerate aging analysis, but also to identify critical paths to be monitored online, hence enabling a systematic design approach by combining periodic monitoring of a circuit during its operation and online tuning.

Similar to statistical timing analysis, the characterization of aging still aims to capture more detailed delay information of logic gates for timing analysis in the current design flow. During the design phase, aging effect still have to be considered statistically since both process variations and aging affect manufactured chips individually and produce different critical paths in different chips after manufacturing. To counter the aging effects actively, post-silicon tuning techniques can also be applied to adjust the timing properties of individual chips. Techniques in this category include body bias tuning [66], [67], voltage control [68], [69], [70] as well as clock tuning [71], [72], [73].

5. Circuit Tuning

While process variations must be modeled as random variables during the design phase, their effects become fixed in individual chips after manufacturing. These chips have different performance because process variations affect them differently. Similarly, aging is also a temporal effect of individual chips. To counter these effects actively, manufactured chips maybe tuned accordingly.
5.1 Post-silicon Clock Tuning

A representative post-silicon tuning technique is clock skew tuning. In a sequential circuit, the clock signal reaches all flip-flops at the same moment. With this strict assumption, the performance of the circuit is determined by the longest path in the circuit, no matter how fast the other paths are. To balance the performance between different paths, clock edges can be tuned toward fast paths so that slow paths receive more timing budget to finish their signal propagation to improve the overall yield [74], [75].

There are various structures of post-silicon clock tuning [71], [76], [77], [78]. For example, the delay buffer in Ref. [71] is illustrated in Fig. 16, where the delay between the clock input CLK_IN and the output CLK_OUT is controlled by the values of three registers configured through the test access port (TAP).

To perform post-silicon clock tuning, delay buffers should be inserted into the circuit during the design phase. Since these buffers occupy die area, a tradeoff should be made to balance the yield improvement and the enlarged area. When deciding how many buffers to insert into a circuit, the method in Ref. [79] presents a technique based on clock scheduling to balance the skew resulting from process variations. In Ref. [80] this problem is solved using a graph-based algorithm. In Ref. [72] several algorithms are proposed to insert buffers into the clock tree to guarantee a given yield and minimize the total area taken by these tunable buffers. Furthermore, the methods in Refs. [81], [82] solves this problem with a sampling-based method to recognize a limited number of locations to insert tunable buffers for yield improvement, while the computational complexity of this method is reduced using machine learning in Ref. [83]. In Ref. [84], this problem is solved together with gate sizing. In Ref. [85], the placement of tunable buffers is explored and a considerable yield improvement has been observed.

After manufacturing, the tunable buffers should be configured to adjust the clock skews in the manufactured chips. This configuration requires that delay information should be captured for these chips. In Ref. [73], this post-silicon configuration is performed by searching a configuration tree together with graph pruning and buffer grouping. In Refs. [86], [87] path delays are measured individually in manufactured chips and delay buffers are tuned accordingly. Since this individual measurement is not efficient, statistical prediction and aligned delay test have been introduced in Ref. [88]. Furthermore, in Refs. [89], [90], post-silicon tuning is applied for online adjustment to counter aging effect. Moreover, in Ref. [91] this method is applied to compensate dynamic delay uncertainty induced by temperature variations.

5.2 Body Bias Tuning

Post-silicon performance can also be tuned by supply voltage scaling or body biasing. Various supply voltages can be provided by an external voltage regulator or an on-chip DC-to-DC converter. Here, low impedance distribution of multiple supply voltages is a burden to physical design. On the other hand, generation and distribution of body voltages are relatively easier compared to those of supply voltage, because the flowing current is very small.

Traditionally, chip-level and block-level tuning have been studied and adopted in some commercial chips. On the other hand, to improve timing while keeping power dissipation low, fine-grained tuning only for regions with timing violation is effective, since most other regions in fabricated chips and blocks do not violate timing specification. However, fine-grained tuning involves an implementation overhead, because of the need of proper separation and distribution of multiple body and supply voltages, as well as level shifters. All of them need extra area and/or power dissipation. The finest granularity is the logic gate, but the overhead is impractically huge and therefore not acceptable. Another problem of fine-grained tuning is the test cost required to obtain an optimal tuning result after fabrication. If there are \( M \) tuning variables and each variable can take \( N \) values, the number of combinations is \( N^M \), and finding an optimal combination for every chip after fabrication becomes less practical, as \( M \) and \( N \) increases.

To minimize power dissipation after post-silicon tuning with acceptable implementation overhead, gate clustering has been proposed [92]. The goal of this approach is to attain a tuning quality that is close to gate-level tuning while reducing the implementation overhead and the number of tuning variables \( M \) to \( M' (\ll M) \). Although the number of combinations is reduced to \( N^{M'} \), the test cost is still expensive for most products, which prevents post-silicon tuning.

To further reduce the test cost, Ref. [93] proposes to preset the testing order of combinations during design time. This work limits \( N = 2 \), and determines an ordered cluster set that allows only \( M + 1 \) combinations for test and guarantees monotonic decrease/increase in delay/power irrelevant of manufacturing variability (Fig. 17), where this monotonicity results from the fact that changing a region with more FBB while the other regions unchanged must reduce delay and increase power. Thanks to this monotonic leveling and limitation of the number of levels, the test cost of post-silicon tuning is significantly reduced.
Reference [94] extends this pre-order body biasing to multiple bias voltages. During the body bias clustering, this method explicitly estimates and minimizes the average leakage after the post-silicon tuning. Experimental results in Ref. [94] demonstrate that this method reduces the average leakage by 25.3 to 51.9% compared to the non-clustering case. It is also revealed that two bias voltages are sufficient when only a small number of compensation levels are allowed for test cost reduction.

5.3 Monitoring and Dynamic (Online) Tuning
5.3.1 Opportunities and Challenges

To minimize design and operating margins, adaptive circuit design is promising where each chip self-adjusts its operating condition, such as supply voltage and body bias. Run-time adaptation can cope with dynamic environmental fluctuation and aging in addition to static process variations. The most popular adaptation is adaptive voltage scaling (AVS). There are two AVS strategies in literature; error detection and recovery based control (e.g., Razor [95]), and error prediction and prevention based control (e.g., canary FF [96], [97], slack monitor [98], timing error predictive FF (TEP-FF) [99]). In both strategies, sensors are embedded to detect/predict timing errors, and the supply voltage is controlled according to the sensor outputs.

Figure 18 shows a circuit that adaptively controls the speed and power dissipation using a warning signal generated by a TEP-FF [99]. The TEP-FF consists of a normal flip-flop, a delay buffer and a comparator (XOR gate). When the timing margin is gradually decreasing, a timing error occurs at the TEP-FF before the main FF captures a wrong value due to the delay buffer, which enables us to predict that the timing margin of the main FF is not large enough. A warning signal is generated to predict the timing errors.

Let us focus on timing margin degradation due to aging. Figure 19 illustrates how the operational margin in the chip lifetime varies with and without adaptation. Without adaptation, the operational margin at the beginning of the chip lifetime is large, and the margin decreases as the chip ages. If the delay increase due to aging exceeds the timing margin, timing errors occur in the chip. This time to the first error corresponds to time to failure (TTF), and TTF varies depending on chip operating environment, workload and process variations. Mean TTF (MTTF) is often used as a metric of device lifetime and reliability. With adaptation, ideally a constant operational margin can be set for the entire chip lifetime. Especially, the margin can be reduced at the beginning of the chip lifetime. If the large operational margin can be translated into supply voltage reduction, the aging process, i.e., the performance degradation, can be slowed down, and the chip lifetime is extended. For pursuing these advantages, performance adaptation has widely been studied. For example, Ref. [97] reports that the power dissipation with performance adaptation is smaller than that with conventional worst-case design by 46% in 65 nm subthreshold design.

On the other hand, the negative impact of performance adaptation, which is illustrated in Fig. 20, is less studied. Performance adaptation degrades noise immunity, especially at the beginning, and hence the possibility that an unexpected timing error due to, for example, unexpected large supply noise occurs becomes higher. In addition, the margin checking performed in the chip is not perfect due to the limited area available for test logic or the limited time allowed for test. Therefore, there is a fundamental problem that the possibility of timing error occurrence cannot be completely reduced to zero, since, for example, a sudden delay increase larger than expectation can induce a timing error without error detection or before error prediction. Similarly, offline delay testing may miss the error because delay testing is carried out with a certain time interval. It should be noted that the timing errors due to such a sudden delay increase arise even in the chips without adaptive speed control, especially at the end of the chip lifetime because the operational margin is small.

5.3.2 MTTF Estimation for Design Optimization

To obtain a good adaptive circuit design mitigating the above negative impact, the adaptive circuit needs to be optimized. Each adaptation scheme has some design parameters to optimize and the built-in test can be tuned. However, it is difficult to evaluate how much improvement in MTTF and power is achieved by the optimization and tuning, since the device lifetime is
5.3.3 MTTF Extension by Critical Path Isolation

The above MTTF estimation enables MTTF-aware design optimization. Here, critical path isolation (CPI) in design time is introduced [101]. CPI enforces larger slacks on the FFs that have frequent input transitions immediately before the clock edge since those FFs tend to cause setup timing errors. Three CPI circuits have been designed with the proposed methodology, where the numbers of FFs with larger slack $N_{\text{cpi}}$ are set to 10/20/30, respectively. For each CPI circuit, the MTTF is calculated. Nine supply voltages are used, ranging from 1.2 V to 0.85 V with 0.05 V interval, and fixed cycle time to 2.1 ns. Figure 22 shows the MTTF of CPI circuits generated with this methodology. From Fig. 22, it can be seen that the CPI circuit with $N_{\text{cpi}} = 10$ achieves the same MTTF at 0.9 V as that of the initial circuit at 1.2 V, i.e., $MTTF_{\text{min}}$. In other words, the supply voltage can be reduced from 1.2 V to 0.9 V by 25.0% without MTTF degradation. This 25% supply voltage reduction corresponds to 44% dynamic power reduction, demonstrating the significant impact of critical path isolation on the power reduction.

MTTFs of the pre-isolated and CPI circuits can also be compared at 0.9 V. With the proposed critical path isolation of $N_{\text{cpi}} = 10$, the MTTF is improved from $1.38 \times 10^2$ cycles to $1.00 \times 10^7$ cycles, with an MTTF improvement ratio $7.24 \times 10^4$.

Thus, the power reduction and MTTF improvement thanks to critical path isolation are remarkable while the area overhead is a few percent. The longer MTTF means fewer timing errors in the field, which is also desirable for run-time adaptation designs. With critical path isolation, the power dissipation of such adaptation circuits could be reduced further and/or the reliability can be improved.

6. Conclusion

The concept of digital circuits has been serving the IC industry for a half century. Even with increasing process variations and aging in the nanometer era, researchers are still able to find ways to overcome new challenges with techniques such as statistical analysis and circuit tuning. When the manufacturing technology advances further reaching 5 nm node or below, variations in manufacturing and reliability issues of devices may challenge the research community again. Looking beyond the realm of nanometer manufacturing, incremental improvements in the existing design flow may not be sufficient any more. Consequently, techniques such as design for tuning, variation-tolerant design, building reliable systems upon unreliable devices, need to be examined together to redefine a new timing paradigm.

Acknowledgments

This work is partially supported by STARC and Socionext Inc.

References

[1] Bruglez, F., Bryan, D. and Kozminski, K. : Combinational Profiles of Sequential Benchmark Circuits, Proc. Int. Symp. Circuits and Syst. (ISCAS), pp.1929–1934 (1989).
[2] Nassif, S.R. : Modeling and Analysis of Manufacturing Variations, Proc. Custom Integ. Circuits Conf. (CICC), pp.223–228 (2001).
[3] Blauw, D., Chopra, K., Srivastava, A. and Scheffer, L. : Statistical Timing Analysis : From Basic Principles to State of the Art, IEEE Trans. Comput.-Aided Design Integ. Circuits Syst., Vol.27, No.4, pp.589–607 (2008).
[4] Dennard, R.H., Gaensslen, F.H., Yu, H.-N., Rideout, V.L., Bassous, E. and LeBlanc, A.R. : Design of ion-implanted MOSFETs with very small physical dimensions, IEEE J. Solid-State Circuits, Vol.9, No.5, pp.256–268 (1974).
[5] Yang, J., Capodieci, L. and Sylvester, D. : Advanced timing analysis based on post-OPC extraction of critical dimensions, Proc. Design Automation Conf. (DAC), pp.359–364 (2005).
[6] Stine, B.E., Boning, D.S. and Chung, J.E. : Analysis and Decomposition of Spatial Variation in Integrated Circuit Processes and Devices, IEEE Trans. on Semiconductor Manufacturing, Vol.10, No.1, pp.24–41 (1997).
[7] Cline, B., Chopra, K., Blauw, D. and Cao, Y. : Analysis and modeling of CD variation for statistical static timing, Proc. Int. Conf. Comput.-Aided Des. (ICCAD), pp.60–66 (2006).
[8] Agarwal, A., Blauw, D., Zolotov, V., Sundareshwar, S., Zhao, M., Gala, K. and Panda, R. : Statistical Delay Computation Considering Spatial Correlation, Proc. Asia and South Pacific Des. Autom. Conf. (ASP-DAC), pp.271–276 (2003).
[9] Agarwal, A., Blauw, D. and Zolotov, V. : Statistical Timing Analysis for Intra-Die Process Variations with Spatial Correlations, Proc. Int. Conf. Comput.-Aided Des. (ICCAD), pp.900–907 (2003).
[10] Chang, H. and Sapatnekar, S.S. : Statistical Timing Analysis
Improved Reliability of Nanopatterning Circuits, Proc. Design, Autom., and Test Europe Conf. (DATE) (2006).

Kleeberger, V.B., Barke, M., Werner, C., Schmitt-Landsiedel, D. and Schlichtmann, U.: A compact model for NBTI degradation and recovery under use-profile variations and its application to aging analysis of digital integrated circuits, Microelectronics Reliability, Vol.54, No.6-7, pp.1083–1089 (2014).

Amrouch, H., Khaleghi, B., Gerstlauer, A. and Henkel, J.: Reliability-aware design to suppress aging, Proc. Design Autom. Conf. (DAC) (2016).

Koppeatzky, N., Metzdorf, M., Eilers, R., Helms, D. and Nebel, W.: RT level timing modeling for aging prediction, Proc. Design, Autom. and Test Europe Conf. (DATE) (2016).

Lorenz, D., Georgakos, G. and Schlichtmann, U.: Aging analysis of circuit timing considering NBTI and HCI, IEEE Int. On-Line Testing Symp. (IOLTS) (2009).

Lorenz, D., Barke, M. and Schlichtmann, U.: Efficiently analyzing the impact of aging effects on large integrated circuits, Microelectronics Reliability, Vol.52, No.8, pp.1546–1552 (2012).

Karapetyan, S. and Schlichtmann, U.: Integrating aging aware timing analysis into a commercial STA tool, Int. Symp. on VLSI Des., Aut. and Test (VLSI-DATE) (2015).

Lorenz, D., Barke, M. and Schlichtmann, U.: Monitoring of aging in integrated circuits by identifying possible critical paths, Microelectronics Reliability, Vol.54, No.6-7, pp.1075–1082 (2014).

Kulkarni, S.H., Sylvester, D. and Blaauw, D.: A statistical framework for post-silicon timing through body bias clustering, Proc. Int. Conf. Comput.-Aided Des. (ICCAD) (2006).

Geng, H., Liu, J., Luo, P.-W., Cheng, L.-C., Grant, S.L. and Shi, Y.: Selective Body Biasing for Post-Silicon Tuning of Sub-Threshold Designs: An Adaptive Filtering Approach, IEEE Trans. Comput.-Aided Design Integr. Circuits Syst. Vol.34, No.5, pp.713–725 (2015).

Ando, Y.: Integrated circuits having post-silicon adjustment control, US Patent 6,957,163 (2005).

Kuo, J., Chao, C., Lin, C., Katta, N., Yang, K. and Wang, C.: Post-silicon tuning in voltage control of semiconductor integrated circuits, US Patent 9,564,896 (2017).

Kumar, R., Li, B., Shen, Y., Schlichtmann, U. and Hu, J.: Timing verification for adaptive integrated circuits, Proc. Design, Autom., and Test Europe Conf. (DATE) (2015).

Nadlzi, S., Stackhouse, B., Grutkowski, T., Josephson, D., Desai, J., Alon, E. and Horowitz, M.: The implementation of a 2-core, multi-threaded Itanium family processor, IEEE J. Solid-State Circuits, Vol.41, No.1, pp.197–209 (2006).

Tsai, J., Zhang, L. and Chen, C.C.P.: Statistical timing analysis driven post-silicon tunable clock-tree synthesis, Proc. Int. Conf. Comput.-Aided Des. (ICCAD) pp.575–581 (2005).

Lak, Z. and Nicolici, N.: A Novel Algorithmic Approach to Aid Post-Silicon Delay Measurement and Clock Tuning, IEEE Trans. Comput. Vol.63, No.5, pp.1074–1084 (2014).

Li, B., Chen, N. and Schlichtmann, U.: Fast statistical timing analysis for circuits with Post-Silicon Tunable clock buffers, Proc. Int. Conf. Comput.-Aided Des. (ICCAD) (2011): pp.11–17 (2011).

Li, B. and Schlichtmann, U.: Statistical Timing Analysis and Criticality Computation for Circuits With Post-Silicon Clock Elements, IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., Vol.34, No.11, pp.1784–1797 (2015).

Tam, S., Rusi, N., Nagajishi, U., Kim, R., Zhang, J. and Young, I.: Clock generation and distribution for the first IA-64 microprocessor, IEEE J. Solid-State Circuits, Vol.35, No.11, pp.1545–1552 (2000).

Takahashi, E., Kasa, Y., Murakawa, M. and Higuchi, T.: Post-fabrication clock-timing adjustment using genetic algorithms, IEEE J. Solid-State Circuits, Vol.39, No.4, pp.634–650 (2004).

Mahoney, P., Fetzer, E., Doyle, B. and Naflie, S.: Clock distribution on a dual-core, multi-threaded Itanium® family processor, Proc. Int. Solid-State Circuits Conf. (ISSCC), pp.292–293 (2005).

Tsai, J., Baik, D., Chen, C.C.P. and Saluja, K.K.: A yield improvement methodology using pre- and post-silicon statistical clock scheduling, Proc. Int. Conf. Comput.-Aided Des. (ICCAD), pp.614–618 (2004).

Kim, J. and Kim, T.: Adjustable Delay Buffer Allocation under Useful Clock Slew Scheduling, IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., Vol.36, No.4, pp.614–654 (2017).

Zhang, G.L., Li, B. and Schlichtmann, U.: Sampling-based buffer insertion for post-silicon yield improvement under process variability, Proc. Design, Autom., and Test Europe Conf. (DATE), pp.1457–1460 (2016).

Zhang, G.L., Li, B., Liu, J., Shi, Y. and Schlichtmann, U.: Design-Phase Buffer Allocation for Post-Silicon Clock Tuning by Iterative Learning, IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., Vol.37, No.2, pp.392–405 (online), DOI: 10.1109/TCAD.2017.2702637 (2018).

Yigit, B., Zhang, G.L., Li, B. and Schlichtmann, U.: Application of Machine Learning Methods in Post-Silicon Yield Improvement, IEEE Int. System-on-Chip Conf., pp.243–248 (2017).

Khandelwal, V. and Srivastava, A.: Variability-driven formulation for simultaneous gate sizing and post-silicon tuning allocation, Proc. Int. Symp. Phys. Des. (ISPD), pp.11–18 (2007).

Nagaraj, K. and Kundu, S.: A study on placement of post silicon clock tuning buffers for mitigating impact of process variation, Proc. Design, Autom., and Test Europe Conf. (DATE), pp.292–295 (2009).

Nagaraj, K. and Kundu, S.: An Automatic Post Silicon Clock Tuning System for Improving System Performance based on Tester Measurements, Proc. Int. Test Conf. (ITC), pp.1–8 (2008).

Tadesse, D., Groslstein, J. and Bahar, R.I.: AutoRes: An automated post-silicon clock tuning tool, Proc. Int. Test Conf. (ITC), pp.1–10 (2009).

Zhang, G.L., Li, B. and Schlichtmann, U.: EffiTest: Efficient Delay Test and Statistical Prediction for Configuring Post-silicon Tunable Buffers, Proc. Design Autom. Conf. (DAC), pp.60–60:6 (2016).

Ye, R., Yuan, F. and Xu, Q.: Online Clock Slew Tuning for Timing Speculation, Proc. Int. Conf. Comput.-Aided Des. (ICCAD), pp.442–447 (2011).

Lak, Z. and Nicolici, N.: On Using On-Chip Clock Tuning Elements to Address Delay Degradation Due to Circuit Aging, IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., Vol.31, No.12, pp.1845–1856 (2012).

Chakraborty, A., Duraisami, K., Sathanur, A.V., Sathiambaran, P., Benini, L., Maci, A., Maci, E. and Poncino, M.: Dynamic Thermal Clock Slew Compensation Using Tunable Delay Buffers, IEEE Trans. VLSI Syst., Vol.16, No.6, pp.639–649 (2008).

Kulkarni, S.H., Sylvester, D.M. and Blaauw, D.T.: Design-time optimization of post-silicon tuned circuits using adaptive body bias, IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., Vol.27, No.3, pp.481–494 (2008).

Hamamoto, K., Hashimoto, M. and Onoye, T.: Tuning-friendly body bias clustering for compensating random variability in subthreshold circuits, Proc. Symp. Int. Symp. Power Electron. and Des. (ISPSD), pp.51–56 (2009).

Kimura, S., Hashimoto, M. and Onoye, T.: A Body Bias Clustering Method for Low Test-Cost Post-Silicon Tuning, IEICE Trans. Fundamentals, Vol.E95-A, No.12, pp.2292–2300 (2012).

Das, S., Roberts, D., Seokwoo, L., Pant, S., Blaauw, D., Austin, T., Flautner, K. and Mudge, T.: A Self-Tuning DVS Processor Using Delay-Error Detection and Correction, Vol.41, pp.792–804 (2006).

Sato, T. and Kuntakute, Y. A Simple Flip-Flop Circuit for Typical-Case Designs for DFM, Proc. Int. Symp. Quality Electron. Des. (ISQED), pp.539–544 (2007).

Fuketa, H., Hashimoto, M., Mitsuayma, Y. and Onoye, T.: Adaptive Performance Compensation with In-Situ Timing Error Predictive Sensors for Subthreshold Circuits, IEEE Trans. VLSI Syst., Vol.20, No.2, pp.333–343 (2012).

Benhasain, A., Cacho, F., Huard, V., Saliva, M., Angbel, L., Parthasarathy, C., Jain, A. and Giner, F.: Timing in-situ monitors: Implementation strategy and applications results, Proc. Custom Integ. Circuits Conf. (CICC) (2015).

Iizuka, S., Mizuno, M., Kuroda, D., Hashimoto, M. and Onoye, T.: Stochastic Error Rate Estimation for Adaptive Speed Control with Field Delay Testing, Proc. Int. Conf. Comput.-Aided Des. (ICCAD) (2013).

Iizuka, S., Masuda, Y., Hashimoto, M. and Onoye, T.: Stochastic Timing Error Rate Estimation under Process and Temporal Variations, Proc. Int. Test Conf. (ITC) (2015).

Masuda, Y., Hashimoto, M. and Onoye, T.: Critical Path Isolation for Time-to-Failure Extension and Lower Voltage Operation, Proc. Int. Conf. Comput.-Aided Des. (ICCAD) (2016).
Bing Li received his bachelor’s and master’s degrees in communication and information engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2000 and 2003, respectively, and Dr.-Ing. degree in electrical engineering from Technical University of Munich (TUM), Germany, in 2010. He is currently a researcher with the Institute for Electronic Design Automation, TUM. His research interests include high-performance and lower-power design of computing systems and emerging systems.

Masanori Hashimoto received his B.E., M.E., and Ph.D. degrees in communications and computer engineering from Kyoto University, Kyoto, Japan, in 1997, 1999, and 2001, respectively. Now, he is a Professor in the Department of Information Systems Engineering, Osaka University, Osaka, Japan. His current research interests include design for manufacturability and reliability, timing and power integrity analysis, reconfigurable computing, soft error characterization and low-power circuit design. Dr. Hashimoto was a recipient of the Best Paper Award at ASP-DAC 2004. He was on the technical program committees of international conferences including DAC, ICCAD, ITC, Symposium on VLSI Circuits, ASP-DAC and DATE. He serves/served as an associate editor for IEEE TVLSI, TCAS-I, and ACM TODAES.

Ulf Schlichtmann received his Dipl.-Ing. and Dr.-Ing. degrees in electrical engineering and information technology from Technical University of Munich (TUM), Munich, Germany, in 1990 and 1995, respectively. He was with Siemens AG, Munich, and Infineon Technologies AG, Munich, from 1994 to 2003, where he held various technical and management positions in design automation, design libraries, IP reuse, and product development. He has been a Professor and the Head of the Chair of Electronic Design Automation, TUM, since 2003, where he also served as the Dean of the Department of Electrical and Computer Engineering, from 2008 to 2011, and as the Associate Dean of Studies of International Studies since 2013. Since 2016, he is an elected member of TUM’s Academic Senate and Board of Trustees. He also serves on a number of advisory boards. Ulf’s current research interests include computer-aided design of electronic circuits and systems, with an emphasis on designing reliable and robust systems. In recent years, he has increasingly worked on emerging technologies, such as microfluidic biochips and optical interconnect.

(Invited by Editor-in-Chief: Nozomu Togawa)