Timing Optimization and Noise Tolerance for Dynamic CMOS Susceptible to Process Variations

Kumar Yelamarthi, Member, IEEE, and Chien-In Henry Chen, Member, IEEE

Abstract—Dynamic CMOS circuits are significantly used in high-performance very large-scale integrated (VLSI) systems. However, they suffer from limitations such as noise tolerance, charge leakage, and power consumption. With the escalating impact of process variations on design performance, aggressive technology scaling, noise in dynamic CMOS circuit has become an imperative design challenge. The design performance of dynamic circuits has to be first improved for reliable operation of VLSI systems. Alongside, this impact of process variation is worse in circuits with multiple timing paths such as those used in microprocessors. In this paper, these problems of process variations, timing, noise tolerance, and power are investigated together for performance optimization. We propose a process variation-aware load-balance of multiple paths transistor sizing algorithm to: 1) improve worst-case delay, delay uncertainty, and sensitivity due to process variations in dynamic CMOS circuits; and 2) optimize dynamic CMOS circuits with MOSFET-based keepers to improve the noise tolerance. Implemented using 90-nm CMOS process, the proposed algorithm has demonstrated an average improvement in worst-case delay by 34%, delay uncertainty by 40.3%, delay sensitivity by 25.1%, and noise margins by 19.4% when compared to their initial performances.

Index Terms—CMOS circuit, delay uncertainty, noise tolerance, process variations, timing optimization.

I. INTRODUCTION

THE PERFORMANCE improvement of microprocessors has been driven traditionally by dynamic CMOS logic and microarchitectural improvements [1], and can be further enhanced through circuit design and topology organization. Of the several methods available for performance improvement, transistor sizing is one of the most effective due to many reasons: 1) several sources of power consumption due to glitches and short-circuit currents can be minimized through efficient transistor sizing; 2) it affects not only the resistance and timing constants but also the propagation delay due to parasitic capacitances; and 3) it can help to maintain sufficient noise margins (NMs).

Of the several logic styles available, dynamic CMOS logic has been predominantly used in microprocessors, and their usage has increased the timing performance significantly over static CMOS circuits [1], [2]. They offer low latency, and unlike static CMOS counterpart, do not require logic implementation in complementary pMOS network. However, timing optimization of dynamic CMOS circuits is challenging due to several issues such as charge sharing, noise-immunity, leakage, environmental, and semiconductor process variations. Technology scaling into the nanometer era has led to a lack of process uniformity in semiconductor manufacturing and has made process variability the primary cause of concern [3]. These variations cause significant unpredictability in the performance characteristics and introduce uncertainties at each step of the process in development, design, manufacturing, and testing. The ratio of these variations to the nominal values has been increasing with the shrinking device size toward 32 nm [4]. Accordingly, these process variations need to be taken into account during the design phase to ascertain the accurate estimation of performance characteristics [5].

One of the many challenges in timing optimization of CMOS logic is delay uncertainty (Δ) from process variations, Δ = Tmax − Tmin, where Tmax and Tmin are the maximum and minimum delays of a timing path after process variations are taken into consideration. In the 180-nm CMOS technology, these process variations have caused about 30% variation in chip frequency, along with 20x variation in current leakage [6]. Accordingly, a large number of chips with significantly high leakage have to be discarded, resulting in a considerable parametric yield loss [3]. Also, this magnitude of intra-die channel length variations has been estimated to increase from 35% of total variations in 130-nm to 60% in 70-nm CMOS process, and variation in wire width, height, and thickness is also expected to increase from 25% to 35% [7]. In 65-nm CMOS process, the parameters that affect timing the most are device length, threshold voltage, device width, mobility, and oxide thickness [8]. For process variation sensitive circuits, such as SRAM arrays and dynamic logic circuits, these process variations may result in functional failure and yield loss [7].

Addressing the challenges of timing optimization and delay uncertainty from process variations, this paper presents a process variation-aware load-balance of multiple path (LBMP) transistor sizing algorithm capable of significantly reducing worst-case delay, delay uncertainty, and delay sensitivity due to process variations in dynamic CMOS circuits that are primarily used in microprocessors and other logic elements, where the effect of random variation is more pronounced [9]. Other advantages of this algorithm include its simplicity,
II. NOISE IN DYNAMIC CMOS CIRCUIT

Dynamic logic is a good choice of design style for high-performance computing applications for its advantages of low area and high speed. The limitation of speed in static CMOS logic can be evaded by using dynamic logic that implement the design using only nMOS transistors. The speed of dynamic circuits is faster compared to its counterpart static circuits due to the lower capacitance and absence of contention during switching. Although using dynamic circuits has advantages, it comes at the additional cost of logic for clocking and high-power consumption. Fig. 1 shows the schematic of a 2-b AND gate using dynamic logic. Dynamic circuit operation is divided into two phases: precharge and evaluation. During the precharge phase as shown in Fig. 2, when the Clk signal is asserted logic low, the dynamic node “d” is precharged to logic high. During the evaluation phase, when the Clk signal is asserted logic high, the logic is evaluated. Based on the primary inputs, the dynamic node “d” will either stay at logic-high or discharge to logic-low.

Noise in a digital integrated circuit refers to any phenomenon that causes a deviation in the voltage from its nominal value. While this has not been a new problem, there was not much attention in the past as it had little impact on the design performance. Previous research [11]–[13] has shown that the power spent on solving noises issues dominates the overall power consumption in computing systems. With the escalating requirement for performance optimization in NMs and power consumption, efficient noise-tolerant techniques are an impending requirement in dynamic CMOS designs. Dynamic CMOS circuit is one of the most popular logic families adopted in the high-performance designs due to their advantage of timing performance. The switching threshold voltage of a dynamic CMOS logic gate, defined as the input voltage level at which the gate output changes state, is usually the transistor threshold voltage. In comparison, the switching threshold voltage of static CMOS logic gate is typically around half the supply voltage. Therefore, dynamic logic gates have less noise immunity than static CMOS logic gates.

Also, a dynamic circuit is inherently susceptible to noise due to floating nodes that may occur in the evaluation phase. Noise interference in static circuits only leads to glitch while the circuits are still functional. Once the noise interference occurs in dynamic circuits, the leakage charge cannot be recovered and leads to malfunction. In addition, the increase in variability in leakage charge has become a bottleneck in realizing high-performance logic circuits, as the robustness of the dynamic node has to be guaranteed across different process corners without significant loss in the design performance. Overall, dynamic CMOS circuits are the weak link in high-performance designs that utilize deep submicrometer process technology.

III. PREVIOUS WORKS

A. Timing Optimization

There has been research aiming to optimize timing performance through transistor sizing [14]–[18], but many are focused on static CMOS circuits and technologies using multiple threshold voltages. Timed logic synthesizer (TILOS) [14] presented an algorithm of iteratively sizing transistors by a certain factor in the critical path. TILOS is not a deterministic approach, as it does not guarantee convergence in timing optimization. MINFLOTRANSIT [15] is another algorithm proposed for transistor sizing based on iterative relaxation method, but requires iterative generation of directed acyclic graphs in every step of timing optimization. Logical effort computation is one of the other method proposed for timing optimization [16]. However, it has two limitations. First, it requires estimation of input capacitance, of which circuits with complex branches or multiple paths have
difficulty in accurate estimation. Second, it optimizes timing at the cost of increased area \cite{17}. Simple exact algorithm \cite{18} is another approach where series and parallel transistors are grouped for iterative parametric sweep analysis to identify optimal transistor sizes. Although this method works for dynamic CMOS logic, it is not a deterministic approach, as it does not guarantee timing convergence. Moreover, its simulation time increases at a quadratic rate with the number of transistors in the design. Most importantly, none of these algorithms account for process variations.

B. Process Variations

Methods to mitigate the effect of process variations in CMOS circuits were proposed in \cite{6}, \cite{7}, \cite{9}, and \cite{19}–\cite{21}. These methods deal with statistical variations and are not optimal for designs with large number of parameter variations \cite{19}. The adaptive body biasing method was presented in \cite{6} and \cite{20} to compensate for variation tolerance, where each die is provided with a unique bias voltage to reduce the variance of frequency variation. However, this method does not minimize intradie variations, as each block requires a unique bias voltage. Another limitation is the increasing leakage power, caused by the reduction of threshold voltage. Programmable keepers were proposed to compensate for process variations in \cite{21}. This method works for designs with large number of parallel stacks (similar to the NOR gates). However, it requires additional hardware to program the keeper transistors for other designs. Significant disadvantage of this keeper approach is that during transition from precharge to evaluation, considerable noise can be imposed on the dynamic node through coupling capacitances between the dynamic node and other switching transistors in the circuit, making it extremely vulnerable \cite{9}.

C. Noise Tolerance

There has been research performed to improve the noise immunity in dynamic CMOS circuits \cite{13}, \cite{22}–\cite{24}. Recently, the twin-transistor technique was proposed to pull up the source voltage in a noise-dependent manner \cite{13}. As illustrated in Fig. 3, the twin-transistor method requires an extra transistor for every pull-down transistor. The drain nodes of the additional nMOS transistors are connected to the inputs. The charge injected by the noise is drained through the additional nMOS transistors, and the source voltage of pull-down network is raised to enhance the noise tolerance. A current mirror-based keeper technique as shown in Fig. 4 was proposed for a better process tracking \cite{22}. It consists of a replica transistor whose width is a safety factor times the total nMOS pull down logic width. The gate of this transistor is connected to the source and the leakage current is mirrored to the dynamic node through the pMOS current mirror transistors. Although this technique provides excellent tracking of delay, the contention is still high because the keeper is strongly ON during the beginning of the evaluation phase.

The multikeeper technique was proposed in \cite{23} to achieve similar process tracking by adaptively switching the number of keeper transistors. However, this method has disadvantages of high routing complexity, increased dynamic node capacitance, and high contention current. An adaptive keeper design based on rate-sensing technique, as shown in Fig. 5, was proposed in \cite{24}. While this method provides promising results, it has difficulty of finding an accurate reference rate to compute the unique biased voltage for each logic gate. While all these existing techniques can enhance the noise tolerance of dynamic CMOS circuits, they all have different tradeoffs in terms of speed, power, and area in order to achieve the noise tolerance goal. Especially, as the requirement of noise tolerance increases along with the advance of process technology, the design overhead will further increase dramatically. In addition, none of these noise-tolerant techniques include timing optimization in its design constraint.

IV. DELAY UNCERTAINTY DUE TO PROCESS VARIATIONS

Process variations are broadly classified into two types: inter-die variations representing variations from chip to chip for the same circuit and intra-die variations representing the process variations at different locations on the same chip.

Research has shown that intra-die variations primarily impact the mean delay and inter-die variations impact the vari-
paths to obtain the performance uncertainty characteristics and worst-case paths in each case, and take the union of these. This is to pick some sets of technology parameters, find the increasing number of process variations parameters, it is impractical to use enough corners to estimate the performance measurement and analysis using multiple corner method (slow, typical, fast) is impractical as hundreds of thousands, of variables. On the other hand, measurement and analysis involves tens of thousands, if not statistically calculated and multidimensional space. The true (physics-based) circuits, where process variations occur in a highly uncorrelated and multidimensional space. The true (physics-based)Extensive research was performed to understand and reduce the significance of process variations. These algorithms deal with statistical variations and are not optimal for designs with a large number of parameter variations or require the parameters to be correlated, which is not the case in nanoscale CMOS circuits, where process variations occur in a highly uncorrelated and multidimensional space. The true (physics-based) measurement and analysis involves tens of thousands, if not hundreds of thousands, of variables. On the other hand, performance uncertainty measurement and analysis using multiple corner method (slow, typical, fast) is impractical as some problems are almost uncaught in this method. With the increasing number of process variations parameters, it is impractical to use enough corners to estimate the performance uncertainty in nanoscale CMOS circuits. A better solution for this is to pick some sets of technology parameters, find the worst-case paths in each case, and take the union of these paths to obtain the performance uncertainty characteristics such as μ, σ, Δ, and δ of the design under test as in Fig. 7. Some of the parameters variations considered in this method are gate oxide thickness, threshold voltage, mobility variation due to dopant mismatch, variation due to gate orientation, drain overlap capacitance, junction capacitance, and gate length and width, as these are parameters that have a significant effect on performance characteristics. Later, SPICE-level Monte Carlo simulations with 1000 runs are performed on the design netlist with the process and design physical parameters, which are measured previously, changed in the range of ±3σ (σ of process parameters provided by the CMOS technology library). Delay profiles obtained from these simulations are used to compute the average (μ), standard deviation (σ), uncertainty (Δ), and sensitivity (δ) due to process variations.

V. MOSFET-BASED KEEPER

Circuits designed using MOSFET devices that exhibit negative differential resistance (NDR) have been studied extensively in the literature. Several systematic methods have been developed to construct NDR circuits using transistors, constituting a pool of MOSFET-based keepers (Mk). A two-transistor simple NDR circuit illustrated in Fig. 8 was proposed in [26]. It is composed of a cross-coupled depletion-mode nMOS transistor and enhancement-mode pMOS transistor. Since the gate of nMOS transistor is connected to the dynamic node S, the current through the two transistors will be cut off immediately when the voltage at S drops to the turn-off voltage of nMOS transistor. It is noted that in our application the gate of transistor K1 connects to a constant voltage source, the power-supply node.

A typical ac I-V characteristic of the proposed keeper together with its corresponding dc I-V characteristic is shown in Fig. 9. It can be seen that the two curves are very close to each other and they reach their peaks at approximately the same voltage value across the keeper. This is a distinct contrast to the case of the feedback keepers presented earlier. It is mainly because of the fact that the gate of nMOS transistor is directly connected to the dynamic node S, and therefore is able to cut off the current through the keeper instantaneously when the voltage at S drops. The slight difference between the two I-V curves is caused by the small amount of time required to discharge the parasitic capacitance of the internal node residing between the two transistors.
based keeper offers other advantages such as no requirement to compute unique bias voltages for each design, limited area overhead, and delay contention. Accordingly, this MOSFET-based keeper has been used in our process variation aware LBMP transistor sizing algorithm proposed in the next section.

Previous research in [28] has shown that peak current in the Mk is crucial to ascertain optimal performance in terms of noise tolerance. Careful sizing of transistors \(K_1\) and \(K_2\) in the Mk will help to achieve this goal peak current and increase the noise tolerance. The initial step in sizing these transistors is finding all the unique timing paths \(p_i = \{T_0, T_3, \ldots\}\) with respect to each output. The timing paths for output \(Y_1\) are \(p_2 = \{T_0, T_3, T_5\}\) and \(p_3 = \{T_4, T_5\}\). Next, a transistor weight \(w_i\) in the range of 0.05–0.5 is assigned to each transistor relative to its distance from the dynamic output node. For instance, the circuit in Fig. 8 is comprised of four transistor stacks relative to their distance from the output nodes: stack-1, closest to the dynamic output node, includes transistors \(T_3\) and \(T_8\), stack-2 includes transistors \(T_1\) and \(T_4\), stack-3 includes transistor \(T_3\), and stack-4 includes transistor \(T_0\). Accordingly, transistors in stacks 1–4 are assigned weights of 0.05, 0.1, 0.2, and 0.4, respectively. Later, the MOSFET-sizing factor \(M_p\) for each path is found through \(M_p = \max(S_i \times w_i, S_{i+1} \times w_{i+1}, \ldots)\), where \(S_i\) and \(w_i\) are the size and weight of each transistor in a timing path. For instance, the sizing factors for path-2 and path-3 are \(M_2 = \max(S_0 \times w_0, S_3 \times w_3, S_5 \times w_5)\) and \(M_3 = \max(S_4 \times w_4, S_5 \times w_5)\). Upon computing these MOSFET factors, widths of keeper transistors \(K_1\) and \(K_2\) are found through \(K_1 = \sum w_i M_p\) and \(K_2 = f \times K_1\), where \(f\) is a technology-dependent initial sizing factor between 1.1 and 1.5. The widths of keeper transistors in Fig. 8 are \(K_1 = M_2 + M_3\) and \(K_2 = f \times K_1\).  

VI. LBMPs for Timing Optimization and Noise Tolerance

The delay of dynamic circuit is highly dependent on the number and size (width) of transistors in the critical path. Increasing width of transistors in a path will increase the discharging current and reduce the output pull-down path delay. However, increasing width of transistors to reduce one path delay may increase the capacitive load of channel-connected transistors on other paths and, substantially increase their delays. This complexity increases along with the number of paths present in the circuit. A 2-b weighted binary-to-thermometric converter (WBTC) used in high-performance binary adders is shown in Fig. 10 and its 34 timing paths are listed in Table I. This design is used as an example to explain the path delay optimization complexity and the process variation-aware LBMPs transistor sizing algorithm presented in Fig. 11.

Consider the two timing neighboring paths: path-18 (\(T_{28} - T_7 - T_8 - T_{12} - T_{18}\)) and path-29 (\(T_{28} - T_0 - T_4 - T_{11} - T_{15} - T_{16}\)) in Fig. 10. A test was performed to optimize path-18 by gradually increasing widths of \(T_7, T_8, T_{12},\) and \(T_{18}\). It was observed that the delay of path-18 reduced by 4%, but delay of path-29 increased by 9.3%. This is a result of transistors on path-29 being channel-connected to transistors on path-18. For instance, \(T_0\) and \(T_{12}\) are channel-connected to \(T_7\) and \(T_8\), and \(T_{15}\) and \(T_{16}\) are channel-connected to \(T_{11}\) and \(T_{13}\). Increasing widths of \(T_7, T_8, T_{12},\) and \(T_{18}\) in path-18 increased the capacitive load of \(T_0, T_{11}, T_{15},\) and \(T_{16}\) and further increased delay of path-29. This illustrates that increasing widths of
transistors on one path increases capacitive load and delay of other path. Our transistor sizing algorithm computes the number of paths a transistor is present in and increases the sizes of transistors that appear in the most number of paths to reduce delays of most paths.

In a stack of transistors connected in series, the discharge time of a transistor increases with its relative distance from the output. Accordingly, the transistor sizes are made progressively larger, starting from a minimum-size transistor, which is closest to the output. The width of the next-to-last transistor is scaled up by a factor. In the proposed LBMP sizing algorithm, a transistor weight \( w \) in the range of 0.05–0.5 is assigned to each transistor relative to its distance from the dynamic output node. For instance, the 2-b WBTC in Fig. 10 is comprised of seven transistor stacks relative to their distance from the dynamic output node. Stack-1, closest to the dynamic output node includes transistors \( T_0, T_9, T_15, T_20, \) and \( T_24 \), Stack-2 includes transistors \( T_0, T_1, T_12, T_23, \) and \( T_29 \). Stack-3 includes transistors \( T_2, T_6, T_{13}, T_{22}, \) and \( T_{27} \). Stack-4 includes transistors \( T_0, T_2, T_7, \) and \( T_{18} \). Stack-5 includes transistors \( T_0, T_1, T_2, \) and \( T_{16} \). Stack-6 includes transistors \( T_0, T_1, T_2, \) and \( T_{15} \). Stack-7, farthest from the dynamic output node, includes transistors \( T_0, T_1, T_2, \) and \( T_3 \). Accordingly, transistors in stacks 1–7 are assigned weights of 0.05, 0.1, 0.15, 0.2, 0.3, 0.4, and 0.5, respectively. For designs with different number of stacks, weights are evenly distributed in the range 0.05–0.5 relative to its distance from the dynamic output node with a weight of 0.05 assigned to stack closest to the dynamic output node, and a weight of 0.5 assigned to stack farthest from the dynamic output node.

Since the magnitude of leakage charge and input noise result in significant performance degradation of high-performance dynamic CMOS circuits, answering this challenge is the initial step in LBMP algorithm. A MOSFET-based keeper presented in the previous section is designed and inserted at each of the dynamic CMOS output node. As increasing width of transistor that appears in the most number of paths reduces overall delay, the number of paths a transistor is present in is computed and denoted as \( N \). The next step in LBMP sizing algorithm is to compute the repeat and weight profiles of all transistors \( \{T_0, T_1, \ldots, T_n\} \). Upon computing these profiles, all transistor sizes are initialized with \( S_i = S_{\text{init}} \times f \) for faster optimization convergence, where \( f \) is a technology-dependent initial sizing factor between 1.1 and 1.5, and \( u \) is the relative position of transistor from dynamic output node. With this initial transistor sizing, process variations are induced to obtain delay distribution of all paths. The transistors on the top 20% critical paths based on delay \( (\mu_i + \sigma_i) \) are grouped to \( \text{set-}x \), and their widths are increased by an optimization formula,

\[
S_i = S_{\text{old}} \times (1 + ((1 + \alpha_r)W_r)/(1 + \alpha_r)),
\]

where \( r_j \) and \( w_j \) are the repeat and weight profiles of transistor \( j \). As the delay of critical path is dependent on the capacitive load of channel-connected transistors, reducing this capacitive load from neighboring paths reduces the overall delay. The first-order connection transistors in \( \text{set-}x \) are identified and grouped to \( \text{set-y} \). Then transistors in \( \text{set-y} \) are excluded from \( \text{set-y} \), if any, and sizes of \( \text{set-y} \) transistors are decreased by an optimization formula,

\[
S_i = S_{\text{old}} \times (1 - ((1 + \alpha_r)W_r)/(1 + \alpha_r))).
\]

Next, the circuit is updated with these new sizes in the pull-down network, and sizes of transistors in the MOSFET-based keeper are updated with \( K_f = K_{\text{init}} \times f \) and \( K_f = f \times K_f \). While computing transistor sizes in every iteration, the LBMP transistor sizing algorithm considers delay distribution and transistor sizes from the previous iterations to identify paths that: 1) act as a load through transistor channel connection to the critical path, and 2) can sacrifice its path-delay without increasing the worst-case delay of the circuit in the new iteration. Once these paths are identified, sizes of transistors in these paths are decreased to reduce the channel capacitive load on the critical path. This will reinforce that the worst-case delay in the new iteration is lower or at least the same compared to the previous iteration in the LBMP algorithm.

![Fig. 10. 2-b WBTC.](image-url)
Next, in order to find the footer transistor size, the first step is to identify all the first-order connection transistors (Tl, Tl+1, ... Tn) to the footer transistor Tf. Later, the footer transistor size Sf is found through Sf = \sum_{l=1}^{n} S_l \times W_l, where S_l is the width of transistor Tl. Once sizes of all the transistors are updated, process variations are induced to obtain new delay profiles for further timing optimization. This algorithm is repeated until the delay converges to an acceptable value.

VII. PERFORMANCE MEASUREMENT OF TIMING, NOISE TOLERANCE, DELAY UNCERTAINTY AND SENSIVITY

A 2-b WBTC used in high-performance binary adders is shown in Fig. 10. This circuit is used as an example to illustrate the complexity of transistor-sizing optimization with multiple timing paths. With less than 50 transistors, the 2-b WBTC has 34 timing paths and of which path delays change significantly with different transistor sizes. Prior to optimization, the worst-case delay of 2-b WBTC was 328 ps from path-1 (T28, T0, T4, T11, T22, and T26). The top 20% critical paths are path-1, 2, 5, 8, 26, and 29. Widths of all transistors in these critical paths are initially increased by a ratio S_i to their initial values. For example, the sizes of transistors (T28, T0, T4, and T11) in path-1 are increased to 120 × 1.11 = 132 nm, 120 × 1.12 = 145 nm, 120 × 1.13 = 160 nm, and 120 × 1.14 = 176 nm, respectively. After initial transistor sizing, process variations are induced and delay distribution of each path are obtained. Accordingly, transistor sizes are updated using the respective formulas and simulations are performed to obtain a new critical path order. After a few iterations of the LBMP sizing algorithm, the worst-case delay of 2-b WBTC converged to an optimal delay of 194 ps, which accounts for a 40.8% improvement.

The LBMP transistor sizing algorithm has been implemented by considering both mean delay as well as a linear combination of mean and the standard deviation (μ + σ) of the delay distribution for the path-ranking criteria. Results obtained in Fig. 12 shows that the linear function of (μ + σ) results in a better performance with lower mean delay and standard deviation from process variations. Fig. 13 shows the significance of LBMP sizing algorithm through comparison of delay distribution in preoptimized and postoptimized 2-b WBTC design, where the delay reduced from 328 ps to 194 ps and delay uncertainty reduced from 133 ps to 36 ps, an improvement of 34.9% and 73%, respectively. Efficiency of the LBMP sizing algorithm is further illustrated in Fig. 14 through a deterministic improvement of worst-case delay (μ + σ), delay uncertainty (Δ1), and delay sensitivity (δ) in the 2-b WBTC. In order to demonstrate the effectiveness of MOSFET-based keeper and updating footer transistor sizes, analysis was performed in two cases. In case-1, LBMP optimization was performed without updating the footer transistor size and without using the MOSFET-based keeper. In case-2, the MOSFET-based keeper was inserted at each dynamic output node, and the sizes of the keeper and footer transistors were updated as necessary.

In addition to optimizing delay, the LBMP transistor sizing algorithm also optimizes power consumption as shown in Fig. 15. The distribution of average power consumption due to process variations in the preoptimized 2-b WBTC is shown in Fig. 15 (red color), with an average of 0.13 μW. When the 2-b WBTC was optimized as in case-1, delay reduced from 328 ps to 194 ps, and the average power increased by 553% to 0.85 μW in Fig. 15 (blue color) due to charge leakage and contention. Performing optimization as in case-2 reduced the average power to 0.22 μW in Fig. 15 (brown color) with almost the same delay, 196 ps, further highlighting the significance of LBMP algorithm.

Fig. 16 shows the robustness of the LBMP algorithm to gate input noise level. The 2-b WBTC is used as the test case in this paper. The load capacitance of each output in this design is 25F and the clock frequency used in simulation is 500 MHz. The normalized gate delay versus the maximum input noise voltage through SPICE simulations is shown in Fig. 16. It is clear that designs optimized with LBMP
TABLE II
Delay and Power Optimization Results from LBMP Transistor Sizing Algorithm

<table>
<thead>
<tr>
<th>Design</th>
<th>Pre-Optimized Delay (ps)</th>
<th>Post-Optimized Delay (ps)</th>
<th>Delay Improvement (%)</th>
<th>Pre-Optimized Power (μW)</th>
<th>Post-Optimized Power (μW)</th>
<th>Power Increase (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pre-Optimized</td>
<td>Post-Optimized</td>
<td></td>
<td>Pre-Optimized</td>
<td>Post-Optimized</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Case-1</td>
<td>Case-2</td>
<td></td>
<td>Case-1</td>
<td>Case-2</td>
<td></td>
</tr>
<tr>
<td>2-b WBTC</td>
<td>328.4</td>
<td>194.0</td>
<td>42.0</td>
<td>0.13</td>
<td>0.85</td>
<td>0.72</td>
</tr>
<tr>
<td>74181-ALU</td>
<td>231.5</td>
<td>108.9</td>
<td>49.4</td>
<td>0.07</td>
<td>0.13</td>
<td>0.11</td>
</tr>
<tr>
<td>74181-ALU</td>
<td>193.5</td>
<td>99.7</td>
<td>57.2</td>
<td>0.09</td>
<td>0.34</td>
<td>0.25</td>
</tr>
<tr>
<td>2184-CC2</td>
<td>80.19</td>
<td>35.5</td>
<td>57.6</td>
<td>0.06</td>
<td>0.11</td>
<td>0.05</td>
</tr>
<tr>
<td>2184-CC2</td>
<td>132.0</td>
<td>114.1</td>
<td>23.1</td>
<td>0.14</td>
<td>0.2</td>
<td>0.06</td>
</tr>
<tr>
<td>73515-CC2</td>
<td>105.1</td>
<td>81.6</td>
<td>23.8</td>
<td>0.08</td>
<td>0.14</td>
<td>0.06</td>
</tr>
<tr>
<td>73515-CC4</td>
<td>111.7</td>
<td>89.9</td>
<td>27.3</td>
<td>0.10</td>
<td>0.25</td>
<td>0.15</td>
</tr>
<tr>
<td>73515-CC6</td>
<td>87.95</td>
<td>76.9</td>
<td>27.3</td>
<td>0.09</td>
<td>0.2</td>
<td>0.06</td>
</tr>
<tr>
<td>7352-CC17</td>
<td>58.05</td>
<td>37.4</td>
<td>35.4</td>
<td>0.03</td>
<td>0.06</td>
<td>0.03</td>
</tr>
<tr>
<td>7352-CC34-4</td>
<td>218.4</td>
<td>105.5</td>
<td>57.7</td>
<td>0.03</td>
<td>0.08</td>
<td>0.05</td>
</tr>
<tr>
<td>7352-CC34-4</td>
<td>111.1</td>
<td>82.3</td>
<td>27.3</td>
<td>0.1</td>
<td>0.25</td>
<td>0.15</td>
</tr>
<tr>
<td>7352-CC40</td>
<td>60.19</td>
<td>35.5</td>
<td>35.6</td>
<td>0.06</td>
<td>0.11</td>
<td>0.05</td>
</tr>
<tr>
<td>Average (%)</td>
<td>34.8</td>
<td>34.0</td>
<td>169.7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Delay and Power Optimization Results from LBMP Transistor Sizing Algorithm

algorithm have significantly reduced performance overhead in comparison with the initial design. The performance benefit of using the MOSFET-based keepers as in case-2 is significantly increased when the noise-tolerance requirement is increased. Comparisons of power consumption for 2-b WBTC at different noise robustness levels and delay are shown in Figs. 17 and 18, respectively. Our simulations have shown that LBMP-optimized design with MOSFET-based smart keeper as in case-2 consumes less power than their counterparts in case-1 due to its faster switching of the internal dynamic node that results in a shorter period of contention. Previous research presented in [5] shows that a decrease in supply voltage degrades cell timing at a quadratic rate; a 5% drop in total rail-to-rail voltage may result in a 15% timing degradation. After the implementation of LBMP transistor sizing algorithm on this design, it was observed that a 20% decrease in supply voltage resulted in only 4% variation in timing, further demonstrating the algorithm’s robustness to variation in supply voltage. Furthermore, the LBMP sizing algorithm was also implemented on several ISCAS benchmark circuits listed in Table II. When the optimization was performed per case-1, the
LBMP algorithm has shown the average delay improvement by 34.8%, while the power has increased significantly by 169.7% due to leakage and contention. Incorporating the MOSFET-based keepers and updating transistor sizes per case-2 into the LBMP optimization allowed for a faster switching of dynamic nodes, and resolved the contention to reduce the average power increase to an acceptable value of 69.5%.

In addition to optimizing worst-case delay as in Fig. 19, accounting for process variations, the LBMP algorithm also optimizes delay uncertainty ($\Delta = T_{\text{max}} - T_{\text{min}}$) and delay sensitivity ($\delta = \sigma/\mu$), where $T_{\text{max}}$ and $T_{\text{min}}$ are the maximum and minimum delays, and $\sigma$ and $\mu$ are the standard deviation and average in the path-delay distributions due to process variations. Results presented in Figs. 20 and 21 show an average improvement in delay uncertainty by 40.3% and delay sensitivity by 25.1%. NMs determine the allowable maximum noise voltage on the input of a gate so that the output does not corrupt [17]. Comparison of both the low and high NMs (NM,
TABLE III
NM Comparison in Preoptimized and Postoptimized Designs

<table>
<thead>
<tr>
<th>Design</th>
<th>Pre-Optimized NM&lt;sub&gt;L&lt;/sub&gt;/NM&lt;sub&gt;H&lt;/sub&gt; (V)</th>
<th>Post-Optimized NM&lt;sub&gt;L&lt;/sub&gt;/NM&lt;sub&gt;H&lt;/sub&gt; (V)</th>
<th>Improvement (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-b WBTC</td>
<td>0.45/0.45</td>
<td>0.50/0.50</td>
<td>11.1</td>
</tr>
<tr>
<td>74181-ALU</td>
<td>0.45/0.45</td>
<td>0.50/0.50</td>
<td>11.1</td>
</tr>
<tr>
<td>c2670-CLA</td>
<td>0.35/0.35</td>
<td>0.40/0.35</td>
<td>7.1</td>
</tr>
<tr>
<td>c3540-CC5</td>
<td>0.40/0.45</td>
<td>0.55/0.55</td>
<td>29.8</td>
</tr>
<tr>
<td>c3540-CC9</td>
<td>0.30/0.30</td>
<td>0.35/0.50</td>
<td>41.6</td>
</tr>
<tr>
<td>c5315-CalP2</td>
<td>0.35/0.35</td>
<td>0.50/0.40</td>
<td>28.5</td>
</tr>
<tr>
<td>c5315-CB4</td>
<td>0.35/0.40</td>
<td>0.40/0.50</td>
<td>19.6</td>
</tr>
<tr>
<td>c5315-GLC4</td>
<td>0.40/0.40</td>
<td>0.50/0.50</td>
<td>25.0</td>
</tr>
<tr>
<td>c7552-CGC17</td>
<td>0.45/0.45</td>
<td>0.50/0.55</td>
<td>16.6</td>
</tr>
<tr>
<td>c7552-GLC34</td>
<td>0.35/0.35</td>
<td>0.25/0.45</td>
<td>0.0</td>
</tr>
<tr>
<td>c7552-GLC5</td>
<td>0.40/0.40</td>
<td>0.40/0.50</td>
<td>12.5</td>
</tr>
<tr>
<td>c7552-CGC30</td>
<td>0.40/0.45</td>
<td>0.50/0.55</td>
<td>20.8</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td>19.4</td>
</tr>
</tbody>
</table>

VIII. Conclusion

Effective performance optimization techniques are vital to the success of very large-scale integrated circuits as timing optimization and noise tolerance become ever-increasing problems with the relentless scaling of CMOS process technology. A desirable performance optimization technique should be able to improve the circuit robustness against all these challenges, be suitable for all logic functions, and have very low overhead in silicon area, delay, and power consumption. In this paper, such a performance optimization technique was proposed. First, we identified the complexity in timing optimization of dynamic CMOS logic. And last, we demonstrated the robustness of algorithm in optimization of delay, delay uncertainty, sensitivity, noise tolerance, and power consumption.

It has been shown that the proposed technique improved performance of dynamic CMOS logic with little overhead in area and power. Simulation results on several ISCAS and other benchmark circuits have shown an average improvement in worst-case delay by 34%, delay uncertainty by 40.3%, delay sensitivity by 25.1%, and NMs by 19.4%. Furthermore, in contrast to existing techniques, the optimization method does not modify or change the pull-down network in dynamic CMOS circuits. It is fairly easy to be adopted in circuit design practice.

REFERENCES

YELAMARTHI AND CHEN: TIMING OPTIMIZATION AND NOISE TOLERANCE


Kumar Yelamarthi

Chien-In Henry Chen

Dr. Yelamarthi was a Technical Reviewer of several IEEE/ASME/ASEE International Conferences and Journals. He is a member of Tau Beta Pi, Engineering Honor Society, and Omicron Delta Kappa National Leadership Honor Society.
Timing Optimization and Noise Tolerance for Dynamic CMOS Susceptible to Process Variations
Kumar Yelamarthi, Member, IEEE, and Chien-In Henry Chen, Member, IEEE

Abstract—Dynamic CMOS circuits are significantly used in high-performance very large-scale integrated (VLSI) systems. However, they suffer from limitations such as noise tolerance, charge leakage, and power consumption. With the escalating impact of process variations on design performance, aggressive technology scaling, noise in dynamic CMOS circuit has become an imperative design challenge. The design performance of dynamic circuits has to be first improved for reliable operation of VLSI systems. Alongside, this impact of process variation is worse in circuits with multiple timing paths such as those used in microprocessors. In this paper, these problems of process variations, timing, noise tolerance, and power are investigated together for performance optimization. We propose a process variation-aware load-balance of multiple paths transistor sizing algorithm to: 1) improve worst-case delay, delay uncertainty, and sensitivity due to process variations in dynamic CMOS circuits, and 2) optimize dynamic CMOS circuits with MOSFET-based keepers to improve the noise tolerance. Implemented using 90-nm CMOS process, the proposed algorithm has demonstrated an average improvement in worst-case delay by 34%, delay uncertainty by 40.3%, delay sensitivity by 25.1%, and noise margins by 19.4% when compared to their initial performances.

Index Terms—CMOS circuit, delay uncertainty, noise tolerance, process variations, timing optimization.

I. INTRODUCTION

THE PERFORMANCE improvement of microprocessors has been driven traditionally by dynamic CMOS logic and microarchitectural improvements [1], and can be further enhanced through circuit design and topology organization. Of the several methods available for performance improvement, transistor sizing is one of the most effective due to many reasons: 1) several sources of power consumption due to glitches and short-circuit currents can be minimized through efficient transistor sizing; 2) it affects not only the resistance and timing constant but also the propagation delay due to parasitic capacitances; and 3) it can help to maintain sufficient noise margins (NMs).

Of the several logic styles available, dynamic CMOS logic has been predominantly used in microprocessors, and their usage has increased the timing performance significantly over static CMOS circuits [1], [2]. They offer low latency, and unlike static CMOS counterpart, do not require logic implementation in complementary pMOS network. However, timing optimization of dynamic CMOS circuits is challenging due to several issues such as charge sharing, noise-immunity, leakage, environmental, and semiconductor process variations. Technology scaling into the nanometer era has led to a lack of process uniformity in semiconductor manufacturing and has made process variability the primary cause of concern [3]. These variations cause significant unpredictability in the performance characteristics and introduce uncertainties at each step of the process in development, design, manufacturing, and testing. The ratio of these variations to the nominal values has been increasing with the shrinking device size toward 32 nm [4]. Accordingly, these process variations need to be taken into account during the design phase to ascertain the accurate estimation of performance characteristics [5].

One of the many challenges in timing optimization of CMOS logic is delay uncertainty ($\Delta$) from process variations, $\Delta = T_{\text{max}} - T_{\text{min}}$, where $T_{\text{max}}$ and $T_{\text{min}}$ are the maximum and minimum delays of a timing path after process variations are taken into consideration. In the 180-nm CMOS technology, these process variations have caused about 30% variation in chip frequency, along with 20x variation in current leakage [6]. Accordingly, a large number of chips with significantly high leakage have to be discarded, resulting in a considerable parametric yield loss [3]. Also, this magnitude of intra-die channel length variations has been estimated to increase from 35% of total variations in 130-nm to 60% in 70-nm CMOS process, and variation in wire width, height, and thickness is also expected to increase from 25% to 35% [7]. In 65-nm CMOS process, the parameters that affect timing the most are device length, threshold voltage, device width, mobility, and oxide thickness [8]. For process variation sensitive circuits, such as SRAM arrays and dynamic logic circuits, these process variations may result in functional failure and yield loss [7].

Addressing the challenges of timing optimization and delay uncertainty from process variations, this paper presents a process variation-aware load-balance of multiple path (LBMP) transistor sizing algorithm capable of significantly reducing worst-case delay, delay uncertainty, and delay sensitivity due to process variations in dynamic CMOS circuits that are primarily used in microprocessors and other logic elements, where the effect of random variation is more pronounced [9]. Other advantages of this algorithm include its simplicity,
II. NOISE IN DYNAMIC CMOS CIRCUIT

Dynamic logic is a good choice of design style for high-performance computing applications for its advantages of low area and high speed. The limitation of speed in static CMOS logic can be evaded by using dynamic logic that implement the design using only nMOS transistors. The speed of dynamic circuits is faster compared to its counterpart static circuits due to the lower capacitance and absence of contention during switching. Although using dynamic circuits has advantages, it comes at the additional cost of logic for clocking and high-power consumption. Fig. 1 shows the schematic of a 2-b AND gate using dynamic logic. Dynamic circuit operation is divided into two phases: precharge and evaluation. During the precharge phase as shown in Fig. 2, when the Clk signal is asserted logic low, the dynamic node “d” is precharged to logic high. During the evaluation phase, when the Clk signal is asserted logic high, the logic is evaluated. Based on the primary inputs, the dynamic node “d” will either stay at logic-high or discharge to logic-low.

III. PREVIOUS WORKS

A. Timing Optimization

There has been research aiming to optimize timing performance through transistor sizing [14]–[18], but many are focused on static CMOS circuits and technologies using multiple threshold voltages. Timed logic synthesizer (TILOS) [14] presented an algorithm of iteratively sizing transistors by a certain factor in the critical path. TILOS is not a deterministic approach, as it does not guarantee convergence in timing optimization. MINFLOTRANSIT [15] is another algorithm proposed for transistor sizing based on iterative relaxation method, but requires iterative generation of directed acyclic graphs in every step of timing optimization. Logical effort computation is one of the other method proposed for timing optimization [16]. However, it has two limitations. First, it requires estimation of input capacitance, of which circuits with complex branches or multiple paths have
difficulty in accurate estimation. Second, it optimizes timing at the cost of increased area [17]. Simple exact algorithm [18] is another approach where series and parallel transistors are grouped for iterative parametric sweep analysis to identify optimal transistor sizes. Although this method works for dynamic CMOS logic, it is not a deterministic approach, as it does not guarantee timing convergence. Moreover, its simulation time increases at a quadratic rate with the number of transistors in the design. Most importantly, none of these algorithms account for process variations.

B. Process Variations

Methods to mitigate the effect of process variations in CMOS circuits were proposed in [6], [7], [9], and [19]–[21]. These methods deal with statistical variations and are not optimal for designs with large number of parameter variations [19]. The adaptive body biasing method was presented in [6] and [20] to compensate for variation tolerance, where each die is provided with a unique bias voltage to reduce the variance of frequency variation. However, this method does not minimize intradie variations, as each block requires a unique bias voltage. Another limitation is the increasing leakage power, caused by the reduction of threshold voltage. Programmable keepers were proposed to compensate for process variations in [21]. This method works for designs with large number of parallel stacks (similar to the NO gates). However, it requires additional hardware to program the keeper transistors for other designs. Significant disadvantage of this keeper approach is that during transition from precharge to evaluation, considerable noise can be imposed on the dynamic node through coupling capacitances between the dynamic node and other switching transistors in the circuit, making it extremely vulnerable [9].

C. Noise Tolerance

There has been research performed to improve the noise immunity in dynamic CMOS circuits [13], [22]–[24]. Recently, the twin-transistor technique was proposed to pull up the source voltage in a noise-dependent manner [13]. As illustrated in Fig. 3, the twin-transistor method requires an extra transistor for every pull-down transistor. The drain nodes of the additional nMOS transistors are connected to the inputs. The charge injected by the noise is drained through the additional nMOS transistors, and the source voltage of pull-down network is raised to enhance the noise tolerance. A current mirror-based keeper technique as shown in Fig. 4 was proposed for a better process tracking [22]. It consists of a replica transistor whose width is a safety factor times the total nMOS pull down logic width. The gate of this transistor is connected to the source and the leakage current is mirrored to the dynamic node through the pMOS current mirror transistors. Although this technique provides excellent tracking of delay, the contention is still high because the keeper is strongly ON during the beginning of the evaluation phase.

The multikeeper technique was proposed in [23] to achieve similar process tracking by adaptively switching the number of keeper transistors. However, this method has disadvantages of high routing complexity, increased dynamic node capacitance, and high contention current. An adaptive keeper design based on rate-sensing technique, as shown in Fig. 5, was proposed in [24]. While this method promises promising results, it has difficulty of finding an accurate reference rate to compute the unique biased voltage for each logic gate. While all these existing techniques can enhance the noise tolerance of dynamic CMOS circuits, they all have different tradeoffs in terms of speed, power, and area in order to achieve the noise tolerance goal. Especially, as the requirement of noise tolerance increases along with the advance of process technology, the design overhead will further increase dramatically. In addition, none of these noise-tolerant techniques include timing optimization in its design constraint.

IV. DELAY UNCERTAINTY DUE TO PROCESS VARIATIONS

Process variations are broadly classified into two types: inter-die variations representing variations from chip to chip for the same circuit and intra-die variations representing the process variations at different locations on the same chip.

Research has shown that intra-die variations primarily impact the mean delay and inter-die variations impact the vari-
paths to obtain the performance uncertainty characteristics.

In nanoscale CMOS circuits, where process variations occur in a highly uncorrelated and multidimensional space, the true (physics-based) performance uncertainty measurement and analysis involves tens of thousands, if not hundreds of thousands, of variables [19]. On the other hand, SPICE-level Monte Carlo simulations with 1000 runs are performed on the design netlist with the process and design physical parameters, which are mentioned previously, changed in the range of ±3σ (σ of process parameters provided by the CMOS technology library). Delay profiles obtained from these simulations are used to compute the average (μ), standard deviation (σ), uncertainty (Δ), and sensitivity (δ) due to process variations.

V. MOSFET-BASED KEEPER

Circuits designed using MOSFET devices that exhibit negative differential resistance (NDR) have been studied extensively in the literature [26], [27]. Several systematic methods have been developed to construct NDR circuits using transistors, constituting a pool of MOSFET-based keepers (Mk).

A two-transistor simple NDR circuit illustrated in Fig. 8 was proposed in [26]. It is composed of a cross-coupled depletion-mode nMOS transistor and enhancement-mode pMOS transistor. Since the gate of nMOS transistor is connected to the dynamic node S, the current through the two transistors will be cut off immediately when the voltage at S drops to the turn-off voltage of nMOS transistor. It is noted that in our application the gate of transistor K1 connects to a constant voltage source, the power-supply node.

A typical ac I-V characteristic of the proposed keeper together with its corresponding dc I-V characteristic is shown in Fig. 9 [28]. It can be seen that the two curves are very close to each other and they reach their peaks at approximately the same voltage value across the keeper. This is a distinct contrast to the case of the feedback keepers presented earlier. It is mainly because of the fact that the gate of nMOS transistor is directly connected to the dynamic node S, and therefore is able to cut off the current through the keeper instantaneously when the voltage at S drops. The slight difference between the two I-V curves is caused by the small amount of time required to discharge the parasitic capacitance of the internal node residing between the two transistors [28]. In addition to its relatively similar ac and dc characteristics, this MOSFET-
based keeper offers other advantages such as no requirement to compute unique bias voltages for each design, limited area overhead, and delay contention. Accordingly, this MOSFET-based keeper has been used in our process variation aware LBMP transistor sizing algorithm proposed in the next section.

Previous research in [28] has shown that peak current in the Mk is crucial to ascertain optimal performance in terms of noise tolerance. Careful sizing of transistors (\(K_1\) and \(K_2\)) in the Mk will help to achieve this goal peak current and increase the noise tolerance. The initial step in sizing these transistors is finding all the unique timing paths \(p_i = \{T_i, T_{i+1}, \ldots\\}\) with respect to each output. The timing paths for output \(Y_1\) are \(p_2 = \{T_0, T_3, T_5\}\) and \(p_3 = \{T_4, T_5\}\).

Next, a transistor weight \(w_i\) in the range of 0.05–0.5 is assigned to each transistor relative to its distance from the dynamic output node. For instance, the circuit in Fig. 8 is comprised of four transistor stacks relative to their distance from the output nodes: stack-1, closest to the dynamic output node, includes transistors \(T_2\) and \(T_5\); stack-2 includes transistors \(T_1\) and \(T_4\); stack-3 includes transistor \(T_3\); and stack-4 includes transistor \(T_0\). Accordingly, transistors in stacks 1–4 are assigned weights of 0.05, 0.1, 0.2, and 0.4, respectively. Later, the MOSFET-sizing factor \(M_p\) for each path is found through

\[
M_p = \max(S_i \times w_i, S_{i+1} \times w_{i+1}, \ldots),
\]

where \(S_i\) and \(w_i\) are the size and weight of each transistor in a timing path. For instance, the sizing factors for path-2 and path-3 are

\[
M_2 = \max(S_0 \times w_0, S_3 \times w_3), \quad M_3 = \max(S_4 \times w_4, S_5 \times w_5).
\]

Upon computing these MOSFET factors, widths of keeper transistors \(K_1\) and \(K_2\) are found through

\[
K_1 = \sum_{i=0}^{P} M_i, \quad K_2 = f \times K_1,
\]

where \(f\) is a technology-dependent initial sizing factor between 1.1 and 1.5. The widths of keeper transistors in Fig. 8 are \(K_1 = M_2 + M_3\) and \(K_2 = f \times K_1\).

VI. LBMPs for Timing Optimization and Noise Tolerance

The delay of dynamic circuit is highly dependent on the number and size (width) of transistors in the critical path. Increasing width of transistors in a path will increase the discharging current and reduce the output pull-down path delay. However, increasing width of transistors to reduce one path delay may increase the capacitive load of channel-connected transistors on other paths and, substantially increase their delays. This complexity increases along with the number of paths present in the circuit. A 2-b weighted binary-to-thermometric converter (WBTC) used in high-performance binary adders is shown in Fig. 10 and its 34 timing paths are listed in Table I. This design is used as an example to explain the path delay optimization complexity and the process variation-aware LBMPs transistor sizing algorithm presented in Fig. 11.

Consider the two timing neighboring paths: path-18 (\(T_{28} \rightarrow T_7 \rightarrow T_{12} \rightarrow T_{18}\)) and path-29 (\(T_{28} \rightarrow T_0 \rightarrow T_4 \rightarrow T_{11} \rightarrow T_{15} \rightarrow T_{16}\)) in Fig. 10. A test was performed to optimize path-18 by gradually increasing widths of \(T_7, T_8, T_{12}, \) and \(T_{18}\). It was observed that the delay of path-18 reduced by 4%, but delay of path-29 increased by 9.3%. This is a result of transistors on path-29 being channel-connected to transistors on path-18. For instance, \(T_4\) and \(T_{11}\) are channel-connected to \(T_7\) and \(T_8\), and \(T_{15}\) and \(T_{16}\) are channel-connected to \(T_{12}\) and \(T_{18}\). Increasing widths of \(T_7, T_8, T_{12},\) and \(T_{18}\) in path-18 increased the capacitive load of \(T_4, T_{11}, T_{15},\) and \(T_{16}\) and further increased delay of path-29. This illustrates that increasing widths of
transistors on one path increases capacitive load and delay of other path. Our transistor sizing algorithm computes the number of paths a transistor is present in and increases the sizes of transistors that appear in the most number of paths to reduce delays of most paths.

In a stack of transistors connected in series, the discharge time of a transistor increases with its relative distance from the output. Accordingly, the transistor sizes are made progressively larger, starting from a minimum-size transistor, which is closest to the output. The width of the next-to-last transistor is scaled up by a factor. In the proposed LBMP sizing algorithm, a transistor weight ($w$) in the range of 0.05-0.5 is assigned to each transistor relative to its distance from the dynamic output node. For instance, the 2-b WBTC in Fig. 10 is comprised of seven transistor stacks relative to their distance from the dynamic output node. Stack-1, closest to the dynamic output node includes transistors $T_0$, $T_4$, $T_{13}$, $T_{21}$, $T_{25}$, and $T_{27}$. Stack-2 includes transistors $T_0$, $T_{11}$, $T_{14}$, and $T_{23}$. Stack-3 includes transistors $T_2$, $T_8$, $T_{13}$, $T_{22}$, and $T_{24}$. Stack-4 includes transistors $T_3$, $T_{12}$, $T_{15}$, and $T_{28}$. Stack-5 includes transistors $T_4$, $T_{14}$, and $T_{15}$. Stack-6 includes transistors $T_7$ and $T_{10}$. Accordingly, transistors in stacks 1–7 are assigned weights of 0.05, 0.1, 0.15, 0.2, 0.3, 0.4, and 0.5, respectively. For designs with different number of stacks, weights are evenly distributed in the range 0.05–0.5 relative to its distance from the dynamic output node with a weight of 0.05 assigned to stack closest to the dynamic output node, and a weight of 0.5 assigned to stack farthest from the dynamic output node.

Since the magnitude of leakage charge and input noise result in significant performance degradation of high-performance dynamic CMOS circuits, answering this challenge is the initial step in LBMP algorithm. A MOSFET-based keeper presented in the previous section is designed and inserted at each of the dynamic CMOS output node. As increasing width of transistor that appears in the most number of paths reduces overall delay, the number of paths a transistor is present in is computed and denoted as repeats. The next step in LBMP sizing algorithm is to compute the repeat and weight profiles of all transistors $\{T_0, \ldots, T_n\}$. Upon computing these profiles, all transistor sizes are initialized with $S = S_{\text{init}} \times f^*$ for faster optimization convergence, where $f$ is a technology-dependent initial sizing factor between 1.1 and 1.5, and $u$ is the relative position of transistor from dynamic output node. With this initial transistor sizing, process variations are induced to obtain delay distribution of all paths. The transistors on the top 20% critical paths based on delay ($\mu + \alpha \sigma$) are grouped to set-$x$, and their widths are increased by an optimization formula, $S_{\text{new}} = S_{\text{old}}(1 + r_j W_j / (1 + r_j))$, where $r_j$ and $w_j$ are the repeat and weight profiles of transistor $j$. As the delay of critical path is dependent on the capacitive load of channel-connected transistors, reducing this capacitive load from neighboring paths reduces the overall delay. The first-order connection transistors in set-$x$ are identified and grouped to set-$y$. Then transistors in set-$y$ are excluded from set-$y$, if any, and sizes of set-$y$ transistors are decreased by an optimization formula, $S_{\text{new}} = S_{\text{old}}(1 - (r_j W_j / (1 + r_j)))$. Next, the circuit is updated with these new sizes in the pull-down network, and sizes of transistors in the MOSFET-based keeper are updated with $K_t = \sum M_j$ and $K_b = f \times K_t$. While computing transistor sizes in every iteration, the LBMP transistor sizing algorithm considers delay distribution and transistor sizes from the previous iterations to identify paths that: 1) act as a load through transistor channel connection to the critical path, and 2) can sacrifice its path-delay without increasing the worst-case delay of the circuit in the new iteration. Once these paths are identified, sizes of transistors in these paths are decreased to reduce the channel capacitive load on the critical path. This will reinforce that the worst-case delay in the new iteration is lower or at least the same compared to the previous iteration in the LBMP algorithm.

Fig. 10. 2-b WBTC.

**TABLE I**

<table>
<thead>
<tr>
<th>Path No.</th>
<th>Transistors</th>
<th>Transistors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>2</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>3</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>4</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>5</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>6</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>7</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>8</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>9</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>10</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>11</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>12</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>13</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>14</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>15</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>16</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
<tr>
<td>17</td>
<td>$T_0$, $T_4$, $T_{13}$, $T_{21}$</td>
<td>$T_3$, $T_8$, $T_{12}$, $T_{28}$</td>
</tr>
</tbody>
</table>
Next, in order to find the footer transistor size, the first step is to identify all the first-order connection transistors \(T_1, T_2, \ldots, T_n\) to the footer transistor \(T_f\). Later, the footer transistor size \(S_f\) is found through \[ S_f = \sum_{l=1}^{n} S_l \times W_l, \] where \(S_l\) is the width of transistor \(T_l\). Once sizes of all the transistors are updated, process variations are induced to obtain new delay profiles for further timing optimization. This algorithm is repeated until the delay converges to an acceptable value.

VII. PERFORMANCE MEASUREMENT OF TIMING, NOISE TOLERANCE, DELAY UNCERTAINTY AND SENSITIVITY

A 2-b WBTC used in high-performance binary adders is shown in Fig. 10. This circuit is used as an example to illustrate the complexity of transistor-sizing optimization with multiple timing paths. With less than 50 transistors, the 2-b WBTC has 34 timing paths and of which path delays change significantly with different transistor sizes. Prior to optimization, the worst-case delay of 2-b WBTC was 328 ps from path-1 \(T_{28}, T_0, T_4, T_{11}, T_{22},\) and \(T_{26}\). The top 20% critical paths are path-1, 2, 5, 8, 26, and 29. Widths of all transistors in these critical paths are initially increased by a ratio \(S_i\) to their initial values. For example, the sizes of transistors \(T_{22}, T_11, T_4,\) and \(T_0\) in path-1 are increased to \(120 \times 1.1 = 132\) nm, \(120 \times 1.2 = 145\) nm, \(120 \times 1.3 = 160\) nm, and \(120 \times 1.4 = 176\) nm, respectively. After initial transistor sizing, process variations are induced and delay distribution of each path are obtained. Accordingly, transistor sizes are updated using the respective formulas and simulations are performed to obtain a new critical path order. After a few iterations of the LBMP sizing algorithm, the worst-case delay of 2-b WBTC converged to an optimal delay of 194 ps, which accounts for a 40.8% improvement.

The LBMP transistor sizing algorithm has been implemented by considering both mean delay as well as a linear combination of mean and the standard deviation \((\mu + \sigma)\) of the delay distribution for the path-ranking criteria. Results obtained in Fig. 12 shows that the linear function of \((\mu + \sigma)\) results in a better performance with lower mean delay and standard deviation from process variations. Fig. 13 shows the significance of LBMP sizing algorithm through comparison of delay distribution in preoptimized and postoptimized 2-b WBTC design, where the delay reduced from 328 ps to 194 ps and delay uncertainty reduced from 133 ps to 36 ps, an improvement of 34.9% and 73%, respectively. Efficiency of the LBMP sizing algorithm is further illustrated in Fig. 14 through a deterministic improvement of worst-case delay \((\mu + \sigma)\), delay uncertainty \((\lambda)\), and delay sensitivity \((\delta)\) in the 2-b WBTC. In order to demonstrate the effectiveness of MOSFET-based keeper and updating footer transistor sizes, analysis was performed in two cases. In case-1, LBMP optimization was performed without updating the footer transistor size and without using the MOSFET-based keeper. In case-2, the MOSFET-based keeper was inserted at each dynamic output node, and the sizes of the keeper and footer transistors were updated as necessary.

In addition to optimizing delay, the LBMP transistor sizing algorithm also optimizes power consumption as shown in Fig. 15. The distribution of average power consumption due to process variations in the preoptimized 2-b WBTC is shown in Fig. 15 (red color), with an average of 0.13 \(\mu W\). When the 2-b WBTC was optimized as in case-1, delay reduced from 328 ps to 194 ps, and the average power increased by 553% to 0.85 \(\mu W\) in Fig. 15 (blue color) due to charge leakage and contention. Performing optimization as in case-2 reduced the average power to 0.22 \(\mu W\) in Fig. 15 (brown color) with almost the same delay, 196 ps, further highlighting the significance of LBMP algorithm.

Fig. 16 shows the robustness of the LBMP algorithm to gate input noise level. The 2-b WBTC is used as the test case in this paper. The load capacitance of each output in this design is 25fF and the clock frequency used in simulation is 500 MHz. The normalized gate delay versus the maximum input noise voltage through SPICE simulations is shown in Fig. 16. It is clear that designs optimized with LBMP
TABLE II

<table>
<thead>
<tr>
<th>Design</th>
<th>Pre-Optimized Delay (ps)</th>
<th>Post-Optimized Delay (ps)</th>
<th>Delay Improvement (%)</th>
<th>Pre-Optimized Power (μW)</th>
<th>Post-Optimized Power (μW)</th>
<th>Power Increase (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Case-1</td>
<td>Case-2</td>
<td>Case-1</td>
<td>Case-2</td>
<td>Case-1</td>
<td>Case-2</td>
</tr>
<tr>
<td>2-b WBTC</td>
<td>328.3</td>
<td>194.0</td>
<td>96.2</td>
<td>82.0</td>
<td>40.2</td>
<td>0.13</td>
</tr>
<tr>
<td>74181-ALU</td>
<td>211.5</td>
<td>106.9</td>
<td>99.9</td>
<td>49.4</td>
<td>48.5</td>
<td>0.07</td>
</tr>
<tr>
<td>C03WLA</td>
<td>193.5</td>
<td>97.5</td>
<td>97.5</td>
<td>52.7</td>
<td>48.9</td>
<td>0.09</td>
</tr>
<tr>
<td>C540-CC1</td>
<td>60.19</td>
<td>37.5</td>
<td>31.5</td>
<td>57.6</td>
<td>47.7</td>
<td>0.06</td>
</tr>
<tr>
<td>C540-CC2</td>
<td>62.9</td>
<td>42.4</td>
<td>118.1</td>
<td>23.5</td>
<td>27.1</td>
<td>0.18</td>
</tr>
<tr>
<td>C515-CP2</td>
<td>105.1</td>
<td>91.1</td>
<td>81.6</td>
<td>23.8</td>
<td>22.7</td>
<td>0.08</td>
</tr>
<tr>
<td>C515-CP4</td>
<td>111.7</td>
<td>101.2</td>
<td>89.2</td>
<td>27.3</td>
<td>20.1</td>
<td>0.15</td>
</tr>
<tr>
<td>C515-CP2</td>
<td>60.19</td>
<td>37.5</td>
<td>31.5</td>
<td>57.6</td>
<td>47.7</td>
<td>0.06</td>
</tr>
<tr>
<td>C5512-CC17</td>
<td>58.03</td>
<td>37.4</td>
<td>42.0</td>
<td>35.4</td>
<td>27.5</td>
<td>0.13</td>
</tr>
<tr>
<td>C5512-CC24</td>
<td>218.4</td>
<td>105.5</td>
<td>115.3</td>
<td>57.7</td>
<td>47.2</td>
<td>0.05</td>
</tr>
<tr>
<td>C5512-CC17</td>
<td>60.19</td>
<td>37.5</td>
<td>31.5</td>
<td>57.6</td>
<td>47.7</td>
<td>0.06</td>
</tr>
</tbody>
</table>

Average (%) | 54.8 | 34.0 | 58.8 | 34.0 | 38.4 | 11.2 |

Fig. 12. Postoptimized delay distribution comparison of 2-b WBTC with different path ranking methods.

Fig. 13. Preoptimized and postoptimized delay distribution comparison of 2-b WBTC.

Fig. 14. Optimization convergence of 2-b WBTC using LBMP sizing algorithm.

The algorithm have significantly reduced performance overhead in comparison with the initial design. The performance benefit of using the MOSFET-based keepers as in case-2 is significantly increased when the noise-tolerance requirement is increased. Comparisons of power consumption for 2-b WBTC at different noise robustness levels and delay are shown in Figs. 17 and 18, respectively. Our simulations have shown that LBMP-optimized design with MOSFET-based smart keeper as in case-2 consumes less power than their counterparts in case-1 due to its faster switching of the internal dynamic node that results in a shorter period of contention. Previous research presented in [5] shows that a decrease in supply voltage degrades cell timing at a quadratic rate; a 5% drop in total rail-to-rail voltage may result in a 15% timing degradation. After the implementation of LBMP transistor sizing algorithm on this design, it was observed that a 20% decrease in supply voltage results in only 4% variation in timing, further demonstrating the algorithm’s robustness to variation in supply voltage. Furthermore, the LBMP sizing algorithm was also implemented on several ISCAS benchmark circuits listed in Table II. When the optimization was performed per case-1, the
LBMP algorithm has shown the average delay improvement by 34.8%, while the power has increased significantly by 169.7% due to leakage and contention. Incorporating the MOSFET-based keepers and updating transistor sizes per case-2 into the LBMP optimization allowed for a faster switching of dynamic nodes, and resolved the contention to reduce the average power increase to an acceptable value of 69.5%.

In addition to optimizing worst-case delay as in Fig. 19, accounting for process variations, the LBMP algorithm also optimizes delay uncertainty ($\Delta = T_{\text{max}} - T_{\text{min}}$) and delay sensitivity ($\delta = \sigma/\mu$), where $T_{\text{max}}$ and $T_{\text{min}}$ are the maximum and minimum delays, and $\sigma$ and $\mu$ are the standard deviation and average in the path-delay distributions due to process variations. Results presented in Figs. 20 and 21 show an average improvement in delay uncertainty by 40.3% and delay sensitivity by 25.1%. NMs determine the allowable maximum noise voltage on the input of a gate so that the output does not corrupt [17]. Comparison of both the low and high NMs (NM, ...
TABLE III

NM Comparison in Preoptimized and Postoptimized Designs

<table>
<thead>
<tr>
<th>Design</th>
<th>Pre-Optimized</th>
<th>Post-Optimized</th>
<th>NM</th>
<th>Improvement (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NM_L/NM_H (V)</td>
<td>NM_L/NM_H (V)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-b WBTC</td>
<td>0.45/0.45</td>
<td>0.50/0.50</td>
<td>11.1</td>
<td></td>
</tr>
<tr>
<td>74181-ALU</td>
<td>0.45/0.45</td>
<td>0.50/0.50</td>
<td>11.1</td>
<td></td>
</tr>
<tr>
<td>c2670-CLA</td>
<td>0.35/0.35</td>
<td>0.40/0.35</td>
<td>7.1</td>
<td></td>
</tr>
<tr>
<td>c3540-CC5</td>
<td>0.40/0.45</td>
<td>0.55/0.55</td>
<td>29.8</td>
<td></td>
</tr>
<tr>
<td>c3540-CC9</td>
<td>0.30/0.30</td>
<td>0.35/0.50</td>
<td>41.6</td>
<td></td>
</tr>
<tr>
<td>c5315-CalP2</td>
<td>0.35/0.35</td>
<td>0.50/0.40</td>
<td>28.5</td>
<td></td>
</tr>
<tr>
<td>c5315-CB4</td>
<td>0.35/0.40</td>
<td>0.40/0.50</td>
<td>19.6</td>
<td></td>
</tr>
<tr>
<td>c5315-GLC4</td>
<td>0.40/0.40</td>
<td>0.50/0.50</td>
<td>25.0</td>
<td></td>
</tr>
<tr>
<td>c5315-CCG20</td>
<td>0.40/0.45</td>
<td>0.50/0.55</td>
<td>16.6</td>
<td></td>
</tr>
<tr>
<td>c7552-GLC34</td>
<td>0.35/0.35</td>
<td>0.25/0.45</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>c7552-GLC5</td>
<td>0.40/0.40</td>
<td>0.40/0.50</td>
<td>12.5</td>
<td></td>
</tr>
<tr>
<td>c7552-CCG20</td>
<td>0.40/0.45</td>
<td>0.50/0.55</td>
<td>20.8</td>
<td>Average 19.4</td>
</tr>
</tbody>
</table>

Fig. 20. Delay uncertainty improvement from LBMP transistor sizing algorithm.

Fig. 21. Delay sensitivity improvement from LBMP transistor sizing algorithm.

Effective performance optimization techniques are vital to the success of very large-scale integrated circuits as timing optimization and noise tolerance become ever-increasing problems with the relentless scaling of CMOS process technology. A desirable performance optimization technique should be able to improve the circuit robustness against all these challenges, be suitable for all logic functions, and have very low overhead in silicon area, delay, and power consumption. In this paper, such a performance optimization technique was proposed. First, we identified the complexity in timing optimization of dynamic CMOS logic circuits with high number of timing paths. Second, we proposed a robust to process variation-aware load balance of multiple paths algorithm for timing optimization of dynamic CMOS logic. And last, we demonstrated the robustness of algorithm in optimization of delay, delay uncertainty, sensitivity, noise tolerance, and power consumption.

It has been shown that the proposed technique improved performance of dynamic CMOS logic with little overhead in area and power. Simulation results on several ISCAS and other benchmark circuits have shown an average improvement in worst-case delay by 34%, delay uncertainty by 40.3%, delay sensitivity by 25.1%, and NMs by 19.4%. Furthermore, in contrast to existing techniques, the optimization method does not modify or change the pull-down network in dynamic CMOS circuits. It is fairly easy to be adopted in circuit design practice.

REFERENCES

YELAMARTHI AND CHEN: TIMING OPTIMIZATION AND NOISE TOLERANCE

11


Kumar Yelamarthi (S’02–M’04) received the Ph.D. degree in electrical engineering from Wright State University, Dayton, OH, in 2004.

He is currently an Assistant Professor of electrical engineering with Central Michigan University, Mount Pleasant, MI. He has written over 50 publications in both technical and educational fields. His current research interests include timing optimization, computer-aided design, semiconductor process variations, multidisciplinary very-large-scale integrated design, and engineering education.

Dr. Yelamarthi was a Technical Reviewer of several IEEE/ASME/ASEE International Conferences and Journals. He is a member of Tau Beta Pi, Engineering Honor Society, and Omicron Delta Kappa National Leadership Honor Society.

Chien-In Henry Chen (S’89–M’04) received the Ph.D. degree in electrical engineering from the University of Minnesota, Minneapolis, in 1989.

Since 1989, he has been with Wright State University, Dayton, OH, involved in computer-aided design and testing of digital, mixed-signal integrated circuit and system-on-a-chip, very large-scale integration (VLSI) and field-programmable gate array (FPGA) designs and architecture. His research was supported by DoD, Federal agencies, industrial companies, and the State of Ohio. He has written over 110 publications in IEEE Journals, International Journals, and IEEE Conference Proceedings.

Dr. Chen was a Technical Committee Member of the IEEE International AS/ACM Conference, the IEEE International Instrumentation and Measurement Technology Conference, and the IEEE International Symposium on Circuits and Systems. He is an Editor of VLSI Design and the Journal of Computers. He is an Associate Editor of Integration: The VLSI Journal and the International Journal of Advances in Computing Technology.