# Obstacle-aware Clock-tree Shaping during Placement

Dong-Jin Lee and Igor L. Markov
EECS Department, University of Michigan, Ann Arbor, MI
University of Michigan, 2260 Hayward St., Ann Arbor, MI 48109-2121
ejdjsy@eecs.umich.edu, imarkov@eecs.umich.edu

***Abstract*—Traditional IC design flows optimize clock networks before signal-net routing are limited by the quality of register placement. Existing publications also reflect this bias and focus mostly on clock routing. The few known techniques for register placement exhibit significant limitations and do not account for recent progress in large-scale placement and obstacle-aware clock-network synthesis.**

**In this work, we integrate clock network synthesis within global placement by optimizing register locations. We propose (1) obstacle-aware virtual clock-tree synthesis; (2) arboreal clock-net contraction force with virtual-node insertion, which can handle multiple clock domains and gated clocks; (3) an obstacle-avoidance force. Our work is validated on large benchmarks with numerous macro blocks. Experimental results indicate that our software implementation, called Lopper, prunes clock-tree branches to reduce their length by 30.0%∼36.6% and average total dynamic power consumption by 6.8%∼11.6% versus conventional wirelength-driven approaches. SPICE-driven simulations show that our methods improve robustness of clock trees.**

## I. Introduction

Power consumption is one of the primary optimization objectives for modern IC designs [23]. It includes three basic components: *short-circuit* power, *leakage* power and *net-switching* power [15]. Net-switching power is usually the largest contributor, and clock networks are often responsible for over 30% of total power consumption due to their high capacitance and frequent switching [5], [6], [18], [28]. The quality of clock networks is greatly affected by register placement, but mainstream literature on placement and most commercial EDA tools have largely overlooked this fact by focusing on wirelength of signal nets [11], routability [31] and circuit timing [7]. As far as we know, high-quality register placement cannot be achieved by easy pre- or post-processing of existing techniques. To this end, most appropriate changes to cell locations that reduce the clock network may depend on the current structure of the clock network, which is not accounted for in existing placement tools. However, over-emphasizing the placement of clock sinks may harm the overall design performance by making signal nets longer.

Our analysis of prior work reveals serious limitations in published techniques. Some methods coerce the placer into shortening the clock tree by capturing portions of the clock tree with the half-perimeter wirelength (HPWL) objective, which is usually applied only to signal nets [4], [32]. This idea overlooks the fact that low-skew clock trees exhibit much greater wirelength than signal nets with the same bounding box. To make matters worse, the HPWL estimate does not offer much fidelity for clock-tree lengths, as we show in Figure 2. Furthermore, a handful of existing publications that optimize clock networks during placement (reviewed in Section II) do not reflect recent progress in large-scale placement and clock-network synthesis, and do not compare their results with best-of-breed software. In most cases, they are evaluated on small benchmarks without routing/buffering obstacles rather than on modern ASIC or SoC designs with many macro blocks. *Our research addresses these gaps in the literature by developing a set of new techniques for clock-net optimization during placement and evaluating these techniques against leading academic software*. We extended the ISPD 2005 benchmark suite toward clock-network synthesis, with the largest benchmark including 2.1M standard cells and 327K registers. The benchmarks include numerous macros, which we interpret as routing obstacles.

To optimize the trade-off between clock network minimization and traditional placement objectives, we propose a new placement methodology based on *obstacle-aware virtual clock-tree synthesis* that extends force-directed placement by adding *a arboreal clock-net force* using virtual nodes. *A key challenge addressed in our work is preserving the quality of global placement when adding clock-net optimizations.* We also accommodate multiple clock domains and gated clocks. Our algorithms are integrated into the SimPL placer [10], which currently produces lowest-wirelength placements on the ISPD‘05 benchmarks. The quality of register placement is evaluated by Contango 2.0 [14] – the winner of the ISPD 2010 contest. Experimental results show that our method can reduce clock-network capacitance by 30.0%∼36.6% while reducing the overall dynamic power of the IC by 6.8%∼11.6% compared to conventional approaches.

Modern CPU designs demand low-power clock networks, yet also impose stringent skew limits, especially in the presence of process, voltage and temperature (PVT) variation for sub-45$nm$ CMOS technologies. Clock networks that are robust to PVT variations are usually not power efficient. To this end, our proposed methodology integrates variation-sensitive virtual clock-tree construction into the primary optimization

objective of global placement, and therefore produces more robust clock trees without increasing power. Empirically, we increase clock-tree yield by 24.6% compared to state-of-the-art wirelength-driven optimizations.

The remainder of this paper is organized as follows. Section II covers prior work and limitations of existing techniques. Section III reviews the optimization objective for clock-net optimization in placement subject to dynamic-power reduction. Section IV describes our proposed techniques for high-quality register placement. Section V describes our methodology for integrating proposed techniques into a state-of-the-art placer used in industry and academia. Our empirical results are described in Section VI. Conclusions are given in Section VII.

## II. Prior work

Recent clock-network synthesis tools often construct initial trees with a simple delay model (e.g., Elmore) and then perform SPICE-accurate tuning [12], [14], [16], [24].

**Clock-network optimization after placement** can be performed by clustering nearby flip-flops [3], [22] to share inverters (inside flip-flops) and shorten the clock tree. This clustering does not adversely affect signal nets, but is rather limited by the locations of combinational gates. In high-performance CPUs, flip-flops are often replaced by single latches, which reduces savings from clock-sink clustering.

**Clock-network optimization during placement.** To address the apparent conflict between clock-net optimization and traditional placement objectives, some researchers proposed techniques and algorithms for better register placement without intrusive interference in traditional placement objectives. Lu [17] proposed several techniques including Manhattan ring-based register guidance, center-of-gravity constraints for registers, pseudo-pins and register-cluster contraction. Cheon [4] proposed power-aware placement that performs both activity-based register clustering and activity-based net weighting to simultaneously reduce the clock and signal net-switching power. In order to reduce the clock network size, Wang [32] proposed dynamic clock-tree building (DCTB), multi level bounding box (MLBB) and multi level attractive force (MLAF), and integrated them into a force-directed placement (FDP) framework [30].

**Limitations of existing techniques.** Clock-net optimization during placement seeks better register locations but should not harm total wirelength of signal nets. A naive method is to increase the weight of the clock net and pull all registers together. Unfortunately, this method increases routing congestion and hot spots, and also leads to poor signal-net wirelength when dealing with more than several hundred registers [4], [32]. To definitively resolve the conflict between clock-net minimization and traditional placement objectives, careful problem formulation is essential.

Prior approaches to clock-net minimization in placement form two families. *Manhattan-ring guidance methods* commit registers to certain guidance locations and try to pull the registers close to the nearest such locations during placement [17]. However, such methods do poorly in the presence of numerous obstacles, e.g., macro-blocks, or when register locations found by the global placer are not uniformly distributed. In other words, guidance rings cannot accurately predict ideal locations for register clusters. Figure 1 illustrates how Manhattan-ring methods fail. In Figure 1(b), the sink group A is attracted by the closest Manhattan ring. The sinks in A are erroneously guided toward the obstacle. The sink group B and the related standard cells have heavy connections to the bottom macro block. However, the two bottom Manhattan rings encourage the sinks in B to move away from the center of B, which will likely increase signal-net wirelength significantly.

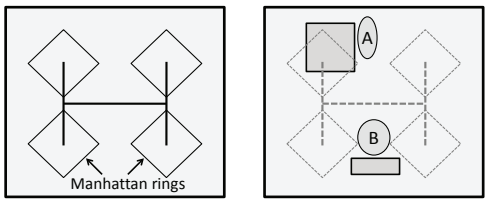



Fig. 1. Two examples of Manhattan rings proposed in [17]. (a) Zero-skew Manhattan rings driven by an H-tree. (b) Manhattan rings on the design with obstacles. Obstacles are indicated by darker boxes, two sink groups (A, B) are represented as ellipses.

The second family of approaches performs clock-network synthesis using register locations from intermediate placement results. Specific techniques [4], [32] often simplify the structure of the clock network and bias the placement process to optimize such simplified networks. However, clock trees generated by those techniques are not realistic and very different from those generated by leading software. In the DCTB algorithm [32], the essential parameters of clock network synthesis, such as sink capacitance and wire capacitance/resistance, are ignored, and the cost function is derived by only considering Manhattan length between sinks or nodes. The quick CTS algorithm in [4] relies on simple heuristic clustering for topology generation and is more simple-minded than standard DME algorithms, which minimize wirelength with zero or bounded skew based on Elmore delay. Furthermore, all previous work ignores the presence of routing obstacles, common in modern IC designs, and this ignorance can undermine end results (Sections IV and VI).

Previous publications that simplify clock-tree synthesis during placement [4], [32] cluster clock trees and represent these clusters with bounding boxes to model clock network reduction by placement objectives. Typically, registers are clustered at one or multiple levels based on the structure of the reference (simplified) clock tree, and bounding boxes are created for each cluster. The experimental results of [4], [32] show that bounding boxes are helpful for clock-net size reduction. However, we argue below that this method fails to represent clock-net reduction problem in placement.

Bounding boxes are represented by fake nets during placement and are optimized to reduce HPWL [10], [26]. The HPWL objective is relevant to placement because it estimates the lengths of signal routes reasonably well. However, clock routing is very different from signal-net routing and requires longer routes to ensure low skew. Therefore, HPWL does not offer accurate estimates of clock-tree lengths. Figure 2

shows that reducing HPWL of the clock net may increase the total length of the clock tree, demonstrating that the HPWL estimates lack not only accuracy, but also fidelity.

The authors of [32] adapted MLAF to compensate for the drawback of MLBB. However, we show in Section IV-B that MLAF offers only a partial solution to this problem.

## III. Optimization objective

Let $\mathcal{N}$ be the set of signal nets, and let $\mathcal{E}$ be the set of clock-net edges. To optimize clock networks in placement, we minimize the total switching power $P_{sw}$, defined as the sum of $\mathcal{N}$'s switching power $P_{\mathcal{N}}$ and $\mathcal{E}$'s switching power $P_{\mathcal{E}}$

$$P_{sw} = P_{\mathcal{N}} + P_{\mathcal{E}} \tag{1}$$

If activity factors of signal nets and clock-net edges are available, then the total signal-net switching power is

$$P_{\mathcal{N}} = \sum_{n_i \in \mathcal{N}} \alpha_{n_i} HPWL_{n_i} C_n V^2 f \tag{2}$$

and the total clock-net switching power is

$$P_{\mathcal{E}} = \sum_{e_i \in \mathcal{E}} \alpha_{e_i} L_{e_i} C_e V^2 f \tag{3}$$

Here, $\alpha_{n_i}$ and $\alpha_{e_i}$ are the respective signal-net and clock-edge activity factors, $C_n$ and $C_e$ are the respective unit capacitance for signal and clock wires, $V$ is the supply voltage, $f$ is the clock frequency, $HPWL_{n_i}$ is the HPWL of net $n_i$, and $L_{e_i}$ is the Manhattan length of edge $e_i$. Activity factors of clock-net edges are required when multiple clock domains or gated clocks are utilized for given designs, otherwise $\alpha_{e_i} = 1$ as clock edges switch every clock cycle. The handling of gated clocks is discussed in Section V in more detail. If the activity factors of signal nets are not available, the computation of total switching power relies on *clock-power ratio* $\beta$, i.e., clock-net switching power divided by total switching power. In this case, the average activity factor of signal-net $\alpha_{avg}$ can be derived as

$$\alpha_{avg} = \frac{(1-\beta)\sum_{e_i \in \mathcal{E}} L_{e_i} C_e}{\beta \sum_{n_i \in \mathcal{N}} HPWL_{n_i} C_n} \tag{4}$$

$\alpha_{avg}$ is utilized for the activity factors of all the signal nets.

Compared to the work in [32], where the main objective does not capture the power of signal nets, our primary objective function (Formula 1) captures both clock-net and signal-net switching power. The objective function in [4] considers clock-net and signal-net power, but their estimation of clock-net switching power relies on bounding boxes that cannot accurately represent clock-net wirelength. However, our analysis of clock-net power in Section IV-B allows us to explictly represent clock-net power in the primary objective. Thus, the optimization of our primary objective effectively decreases total switching power as decribed in Section VI.

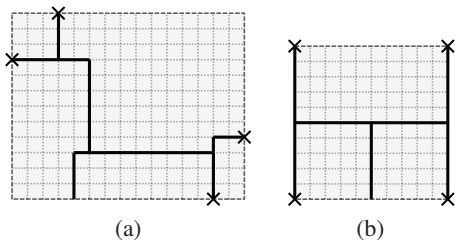



Fig. 2. Bounding boxes of two partial ZST-DME clock trees. (a) HPWL of the bounding box is (15+12)=27. The total wirelength of the inside clock tree is 32. (b) HPWL is (10+10)=20 and the total wirelength of the clock tree is 35. The clock-net wirelength of (b) is greater than (a) although the bounding-box HPWL of (b) is notably smaller than (a) while the source-to-sink wirelength is 15 for all sinks.

## IV. Proposed techniques

We propose a methodology and several new techniques to overcome limitations of prior work and reliably optimize large IC designs with numerous layout obstacles. Our approach consists of two major phases: (*i*) virtual clock-tree synthesis, (*ii*) arboreal clock-net contraction force, which is corrected by an obstacle-avoidance force.

### A. Obstacle-aware virtual clock trees

Our virtual clock-tree synthesis handles macro blocks as wiring obstacles and produces obstacle-avoiding clock trees. The importance of utilizing obstacle-aware clock trees is illustrated in Figure 3 (the contraction forces are described in Section IV-B). Clock-net optimizations without obstacle handling pull clock sinks inside obstacles, which undermines global placement.

Experimental results in [14] show that the difference in total capacitance between initial zero-skew DME trees (based on Elmore delay) and the final SPICE-optimized trees is only 2.2% on average. Hence, initial trees produced by leading clock-network synthesis tools offer reasonably accurate capacitance estimates. To quickly construct a *virtual clock-tree* during placement, our methodology first performs traditional DME-based zero-skew clock-tree synthesis with Elmore delay model, subject to obstacle avoidance. Several techniques are

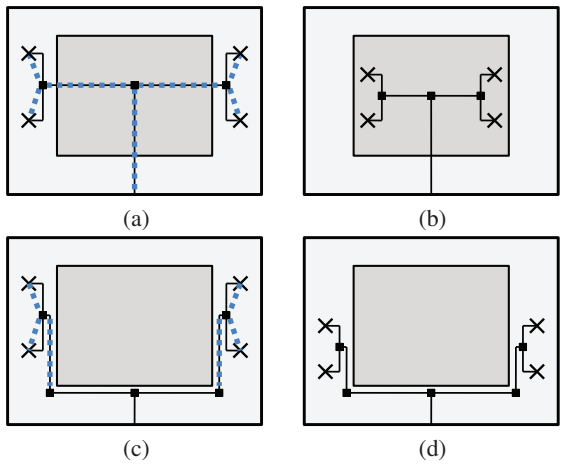
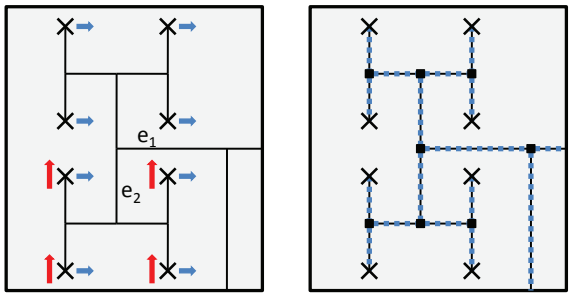



Fig. 3. An example of clock-net optimization with an obstacle. (a) The virtual clock tree and corresponding contraction forces are created without considering the obstacle. (b) The result of a placement iteration with the forces in (a). (c) The obstacle is accounted during virtual clock-tree generation and when establishing additional forces. (d) The result of (c).




(a) (b)

Fig. 4. Two types of forces for clock-net optimization. Registers are indicated by crosses. (a) For each edge, the corresponding downstream registers are given force vectors. Right arrows are the force vectors for reducing $e_1$, and up arrows are the force vectors for reducing $e_2$. (b) Virtual nodes are inserted (squares), and forces are created between each pair of connected nodes (dotted lines).

known for this problem, including direct obstacle-avoiding clock-tree construction [9] and incremental repair of obstacle-unaware trees [12]. Each approach can be used in our methodology, but we found that incremental-repair techniques are simpler and yet produce high-quality trees.[1] Our clock trees target the 45 $nm$ technology used at the ISPD 2010 clock network synthesis contest [27].

### B. Arboreal clock-net contraction force

If the virtual clock network connecting to current register locations faithfully represents a realistic clock network, then optimizing it directly should improve the final clock network produced by a specialized CTS tool after placement is complete. To this end, we extend force-directed placement with new, structurally-defined forces that seek to reduce individual edges of the virtual clock network. This technique communicates current clock-tree structure to the placement algorithm, and also allows the structure to change with placement.

Figure 4(a) illustrates a sample virtual clock tree. To reduce the length of $e_1$ directly, all sinks downstream from $e_1$ can be moved in the direction of reducing the length of $e_1$. For each downstream sink of $e_1$, a force vector needs to be assigned. The force vectors created for $e_1$ should not affect other tree edges.

The sum of magnitudes of force vectors induced by $e_1$ ($F_{e_i}^{sum}$) needs to be carefully controlled to avoid excessive increase in signal-net wirelength. $F_{e_i}^{sum}$ may vary when the activity factors of clock edges differ (e.g., in gated clocks). Figure 4(a) illustrates force vectors. The force from $e_1$ is weaker than the force from $e_2$, $F_{e_1} < F_{e_2}$ since the sum of magnitudes should be same.

The main problem with this method is that the relative locations of branching nodes from sinks are assumed to be same when the force vectors are created. However, optimal relative locations of the branching nodes change during the optimization. Therefore, placement iterations with fixed force vectors for sinks do not produce optimal locations.

To shorten clock wires, we propose *a arboreal clock-net contraction force with virtual-node insertion*. Our approach creates forces between clock-tree nodes and structurally transfer the forces down to registers. Virtual nodes represent branching nodes in the clock tree and split the clock tree into individual edges, seen as different nets by the placement algorithm. The virtual nodes have zero area and do not create overlap with real cells, so they do not affect the spreading process in force-directed placers. Zero-area nodes may or may not be allowed to overlap with obstacles (if such a node is placed over an obstacle, its overlap has zero area). In our case, virtual nodes should not be placed over obstacles to avoid routing over obstacles.

Compared to the fixed force vectors applied exclusively to sinks, our technique creates forces between flexible nodes and each force seeks to reduce the length of the corresponding clock edge. Unlike in the bounding-box based method, each force is integrated into the placement instance as a two-pin pseudo net, as shown in Figure 4(b).

To reduce dynamic power consumption of the IC, contraction forces are calculated based on the activity factors of the signal nets. When activity factors of signal nets are available, the average activity factor $\alpha_{avg}$ over all nets is

$$\alpha_{avg} = \frac{\sum_{n_i \in \mathcal{N}} \alpha_{n_i} HPWL_{n_i}}{\sum_{n_i \in \mathcal{N}} HPWL_{n_i}} \tag{5}$$

and the weight of signal net $n_i$ is defined as

$$w_{n_i} = \frac{\alpha_{n_i}}{\alpha_{avg}} \tag{6}$$

When activity factors of signal nets are not available, Equation 4 is utilized to compute $\alpha_{avg}$ and $w_{n_i} = 1$ for all signal nets. A two-pin net representing clock-net contraction forces for clock edge $e_i$ is given a weight

$$w_{e_i} = \frac{C_e \alpha_{e_i}}{C_n \alpha_{avg}} \tag{7}$$

and the HPWL of a two-pin net from $e_i$ is equal to the Manhattan length of $e_i$,

$$L_{e_i} = HPWL_{e_i} \tag{8}$$

Note that our primary objective function is a sum of signal-net and clock-net switching power (Formula 1). By combining Formulas 2, 3 and 8, the total switching power is expressed as

$$\Big( \sum_{n_i \in \mathcal{N}} \alpha_{n_i} HPWL_{n_i} C_n + \sum_{e_i \in \mathcal{E}} \alpha_{e_i} HPWL_{e_i} C_e \Big) V^2 f \tag{9}$$

By substituting $\alpha_{n_i}$ and $\alpha_{e_i}$ in terms of $w_{n_i}$ and $w_{e_i}$ (Equations 5, 7), Equation 9 can be rewritten as

$$\alpha_{avg} \Big( \sum_{n_i \in \mathcal{N}} w_{n_i} HPWL_{n_i} C_n + \sum_{e_i \in \mathcal{E}} w_{e_i} HPWL_{e_i} C_n \Big) V^2 f \tag{10}$$

Let $K$ be $\alpha_{avg} C_n V^2 f$, $\mathcal{M} = \mathcal{N} \cup \mathcal{E}$. Then the total switching power of signal nets and current clock nets is,

$$P_{sw} = P_{\mathcal{N}} + P_{\mathcal{E}} = K \sum_{m_i \in \mathcal{M}} w_{m_i} HPWL_{m_i} \tag{11}$$

[1] Extensive empirical studies and the experience of ISPD clock-network synthesis contests suggest that when clock sinks are placed outside the obstacles, the overlaps caused by obstacle-unaware trees can often be fixed with minimal impact on skew and total capacitance, compared to obstacle-aware trees.

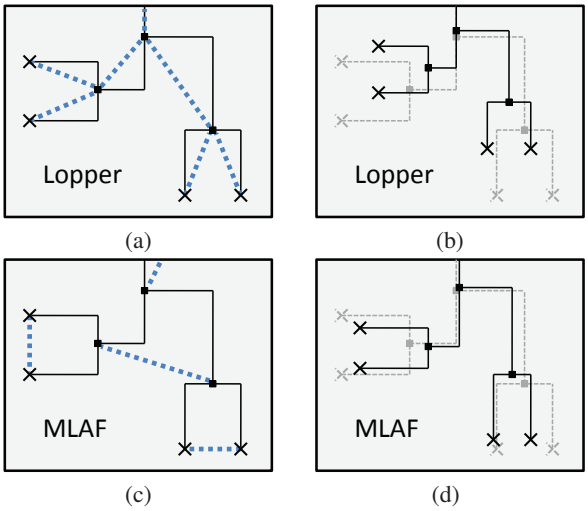



Fig. 5. Comparison between our arboreal clock-net contraction force and MLAF of [32]. (a) Arboreal clock-net contraction forces are generated. (b) The modified register and virtual clock-node locations when forces in (a) are utilized. (c) The forces created by the MLAF algorithm. (d) The modified register and virtual clock-node locations when forces in (c) are utilized. We can observe that the edges between parents and children nodes are poorly handled for the force creation in (c), and our method is more efficient on non H-tree structures (which is common in modern designs).

In other words, our techniques capture the switching-power minimization problem, which can be solved by any high-quality wirelength-driven placer capable of net weighting. Figure 5 compares our technique and MLAF from [32]. MLAF is ineffective in shortening clock nets that significantly differ from H-trees. Additionally, MLAF does not establish exclusive forces that represent the edges between parents and children nodes. Instead, bounding boxes (MLBB) are used with MLAF in [32]. We show in Section II that bounding boxes cannot offer accurate estimates of clock-tree lengths. Also, the authors of [32] did not explain how they assigned weights to MLAF but only hinted that the magnitude of MLAF was similar to the magnitude of forces for signal nets. Detailed comparison between our technique and MLAF is discussed in Section VI-B (Table V).

### C. Obstacle-avoidance force

Given an obstacle-avoiding tree, we modify arboreal clock-net contraction forces to promote obstacle avoidance. Contraction forces based on an obstacle-avoiding clock tree do not necessarily improve every tree edge, as shown in Figure 6. In Figure 6(a), five edges are derived from a virtual obstacle-aware tree built as in Section IV-A. If we create forces for all the edges, subsequent optimization will produce the tree in Figure 6(b). The force $f_4$ associated with edge $e_4$ is rendered ineffective by the obstacle. Our force-modification algorithm for obstacle avoidance detects these obstacle-detouring edges and eliminates the contraction forces for them.[2] In this example, $e_4$ and $e_5$ are excluded from force construction, and the result is illustrated in Figure 6(c).

[2]Consider a clock-tree edge that does not cross a given obstacle. The edge *detours* the obstacle if the straight line connecting the ends of the edge crosses the obstacle.

## V. Proposed Methodology

We integrate our techniques into SimPL, a flat, force-directed quadratic placer [10]. Recall that analytic placers first minimize a function of interconnect length, neglecting overlaps between standard cells and macros. This initial step places many cells in densely populated regions. Clock-net contraction forces are ineffective at this step for two reasons: (*i*) the current virtual clock network may differ greatly from the final clock network. (*ii*) the contraction forces may restrict the spreading of the registers at the center of the design due to their high net weight. Therefore, our techniques are invoked between signal-net wirelength-driven global placement and detailed placement (including legalization).

Our clock-net optimization during placement is referred to as *Lopper*, and described in Figure 7.

### A. The Lopper flow

**At each iteration of Lopper**, a new virtual clock tree is generated based on current register locations. We discard the previous virtual clock tree based on the following observation. The topology of a clock tree and the embedding of its wires minimize (*i*) skew as the primary objective, (*ii*) total wirelength as the secondary objective. When an iteration of Lopper is performed, the locations of the registers are modified in order to reduce the total wirelength of the given virtual clock tree. Since registers are displaced by different amounts (due to different connectivities), keeping the previous clock-tree structure would risk a large increase in skew. Therefore we regenerate the virtual clock tree for each iteration to obtain an optimal virtual clock tree with the current register locations. The tree topology typically undergoes only moderate changes, while branching nodes relocate to reduce skew.

Early placement iterations may greatly displace the registers, suggesting that effective clock-net wirelength reduction requires moving registers over the obstacles. In this case, obstacle-aware virtual CTS and obstacle-avoidance force may undermine the potential improvement. Therefore, Lopper ignores obstacles until average displacement of registers becomes small.

**Global placement** typically continues while HPWL continues improving, but clock-tree reduction in Lopper requires

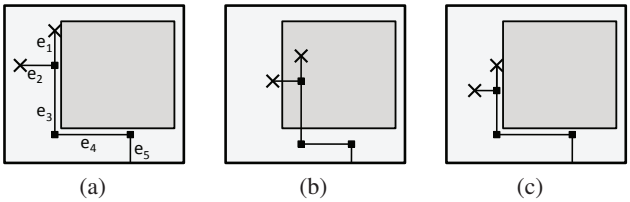



Fig. 6. Obstacle-avoidance force. (a) Five edges of an obstacle-aware virtual clock tree. (b) The result when all the edges are utilized for contraction forces. (c) The result when $e_4$ and $e_5$ are excluded from force construction.

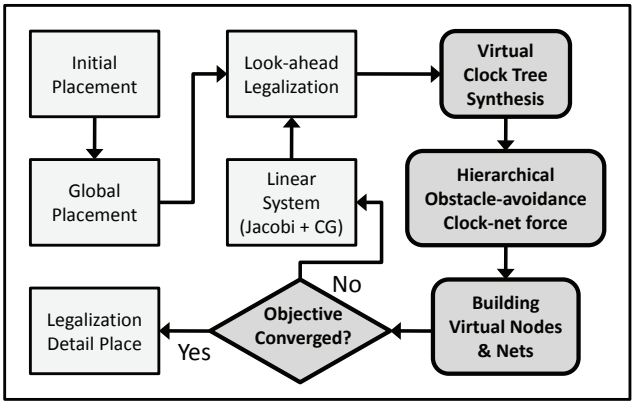



Fig. 7. Key steps of Lopper integrated into the SimPL placer, as indicated with darker rounded boxes and a lozenge. Plain boxes represent the SimPL steps.

a different convergence criterion. After each iteration, total switching power is calculated and compared to previous values. Lopper is invoked repeatedly until total switching power (Equation 1) stops reducing.

**Legalization and detailed placement** are applied after Lopper is complete. It is important to preserve the virtual nodes and two-pin nets that represent the clock-net contraction forces during detailed placement because detailed placement algorithms usually optimize wirelength and would not have preserved clock-optimized register locations if guided only by signal nets.

### *B. Trade-offs and additional features*

**Quality control.** Our techniques reduce the size of clock networks, but are likely to increase signal-net wirelength. The activity factor of each signal-net $\alpha_{n_i}$ or clock-power ratio $\beta$ are required for Lopper to reduce total switching power. However, even clock-power ratio $\beta$ is hard to estimate before the design is completed and can vary with various applications running on a CPU. Therefore, in our implementation the trade-off between clock-net and signal-net switching power can be easily controlled with a single parameter $\beta$. This simple quality control allows an IC designer to achieve intended total switching power of a chip without changing the algorithm or its internal parameters. Relevant trade-offs are illustrated in Table III.

**Gated clocks and multiple clock domains.** Clock gating is a well-known and often the most effective approach to reduce clock network power dissipation [21]. To extend our techniques to gated clocks and multiple clock domains, each register $s_i$ is given an activity factor $\alpha_{s_i}$ and the activity factors are propagated through the tree. The activity factor of an edge is the highest activity factor of its child edge or register (see Figure 8). Without clock gating, all registers are given activity factors 1.0, which are propagated to all tree edges.

Once activity factors are propagated to tree edges in each clock tree, they are used to calculate net weights that represent clock-net contraction forces in Equation 7. Registers that switch less frequently due to clock gating will be more affected

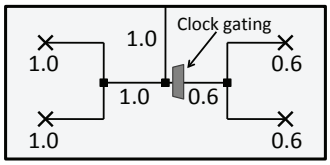



Fig. 8. Activity-factor propagation for gated clocks. Registers are indicated with crosses. Tree edges and registers are labeled with activity factors.

by signal nets than normal registers without clock gating. Our technique does not track the locations of gaters assuming that the final clock tree and the gaters are constructed after register placement. While we have not experimented with gater placement, we do not believe that it will affect results reported in our work.

**Flexible integration.** Through the Lopper flow, forces for clock-net optimization are represented in placement instances by virtual nodes and nets. No support for clock-net optimization is required in the placement algorithm. Therefore, Lopper can integrate any fast obstacle-aware clock-tree synthesis technique into any iterative high-performance wirelength-driven placer capable of net weighting.

**Integration into timing-driven placement.** Timing-driven placement optimizes cell locations to satisfy timing constraints, while minimizing interconnect [8, Chapter 8]. During timing-driven placement, delays of timing-critical nets are carefully controlled to prevent timing violations. Lopper can be integrated into timing-driven placers thanks to the flexibility of proposed integration. For example, a common approach to timing-driven placement increases weights of critical nets.[3] To this end, Formula 6 can be extended to include the *criticality* $c_{n_i}$ of net $n_i$ which represents how important this net is to satisfying timing constraints.

$$w_{n_i} = \frac{\alpha_{n_i} c_{n_i}}{\alpha_{avg}} \tag{12}$$

If weights of critical nets exceed clock-net weights (i.e., $w_{n_i} > w_{e_i}$), the wirelength of critical nets is affected less when reducing clock-net wirelength. During the Lopper flow, only the wirelengths of less critical nets increase to reduce clock-net switching power. Therefore, optimization for total switching-power reduction can be performed without violating timing constraints.

**Congestion-aware clock-net optimization.** Our methods may increase signal-net length to reduce clock-net length subject to total switching-power reduction. However, longer signal nets may aggravate routing congestion. To prevent wirelength increase in highly congested areas, we extend Formula 12 to include the *congestion factor* $g_{n_i}$ of net $n_i$ which represents how much the area around $n_i$ is congested.

$$w_{n_i} = \frac{\alpha_{n_i} c_{n_i} g_{n_i}}{\alpha_{avg}} \tag{13}$$

$g_{n_i}$ is 1 when there is no possible congestion around $n_i$. $g_{n_i}$ increases as the congestion around $n_i$ becomes worse.

[3]This approach is used in our work to illustrate the compatibility of Lopper with timing-driven placement. Many other approaches exist [8, Chapter 8].

By including congestion factors in our primary optimization objective, we can avoid signal-net length increase in highly congested regions.

**High-quality register placement for robustness to variations.** In practice, not only low nominal skew but also robustness to PVT variations are essential for building high-quality clock networks. When making a clock network more robust, one uses large buffers and/or redundant wiring, which increases total capacitance and dynamic power. Because clock networks consume a large portion of total power, it is important to limit the maximum clock-network power. *When clock-network power is limited by design, one of the most effective techniques to improve robustness to PVT variations is to optimize register placement.* Since our virtual clock-tree construction algorithm is variation sensitive[4] and integrated into our primary optimization objective during placement, our register placement is sensitive to PVT variations. Extensive SPICE simulations confirm that the Lopper flow significantly improves robustness of clock trees when clock-network power is limited (see Section VI-C).

## VI. Empirical validation

The benchmarks used in prior publications on clock-tree optimization during placement exhibit the following problems: (1) Empirical validation of each existing publication relies on one benchmark suite which is not utilized by any other work. Most of the benchmarks are inaccessible to public, therefore comparisons to new techniques are impossible. (2) The benchmark designs are based on unrealistically small placement instances. None of the prior publications provide results on a design with more than 1M standard cells, which is common in modern modern ASIC designs. (3) Macro blocks became essential components, and many IC designs include more than hundreds of macros with fixed locations after floorplanning [1]. However, prior publications used the benchmarks without macro blocks or ignored macro blocks present in the benchmarks [32]. (4) Reference placement tools used for comparison are often outdated [17] or self-implemented [32]. Such comparisons risk not being representative of state-of-the-art EDA tools.

In this section, we propose a new benchmark set that addresses the above pitfalls. Our experimental results offer full comparisons with leading academic wirelength-driven placers and a known technique for register placement (MLAF). The quality of register locations is validated by a leading academic clock-network synthesis tool.

### A. Experimental setup

The ISPD 2005 placement contest benchmark suite is being used extensively in placement research, and the academic community consistently advanced physical design techniques using the ISPD'05 benchmarks. These benchmarks are directly derived from industrial ASIC designs, with circuit sizes ranging from 210K to 2.1M placeable objects. We adapted eight designs from the ISPD'05 benchmarks and created register lists in which 15% of standard cells are selected to be registers. We selected the number 15% based on the industrial designs introduced in [4], where the average 14.65% of cells are registers. The largest benchmark has 327K registers. Fixed macro blocks are viewed as routing and placement blockages during clock-network synthesis.[5] Some macro blocks that create routing dead space are slightly resized and/or repositioned to eliminate dead space (see Figure 9). This modification is so small that the impact on density and timing is negligible. The benchmarks are mapped to the Nangate 45 $nm$ open cell library [20] to facilitate clock-network synthesis with parameters from ISPD 2010 CNS contest. The standard-cell height (or row height) is set to 1.4 $\mu m$ according to the 45 $nm$ library.[6] Clock-power ratio $\beta$ is set to 0.3 for clock network optimization during placement based on the industrial circuits from [4], where clock power is responsible for 31.9% of total power on average. For each circuit, the average activity factor of signal nets is calculated based on the signal-net and clock-net wirelength of the placement produced by SimPL [10] using Formula 4. The unit-wire capacitances for signal-net and clock-net ($C_n$, $C_e$) are set to $0.1fF/\mu m$, $0.2fF/\mu m$ respectively based on the 45 $nm$ technology model from the ISPD'10 contest [27] and the Nangate open-cell library [20]. Supply voltage and clock frequency are set to $1.0V$ and $2GHz$. The coordinate of clock source is set to the bottom left corner

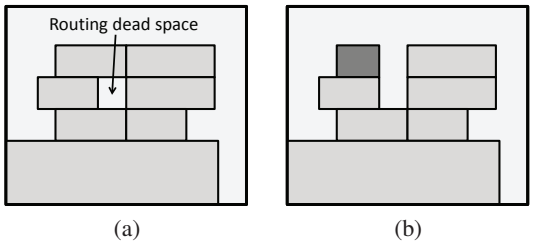



Fig. 9. An example of routing dead space that can be found in the ISPD'05 benchmarks. (a) Routing dead space is created by enclosing macro blocks. (b) One macro block is modified to open the space.

[4]When clock-network power is limited, the DME-based initial-tree construction algorithm in [14] is variation sensitive in the sense that it generate low-skew trees with optimizing wirelength. When initial wirelength of a clock tree is small, more optimizations for enhancing robustness are possible within given power limit. Our virtual clock-tree construction algorithm is adapted from this variation-sensitive initial-tree construction algorithm.

[5]When macro blocks act as placement blockages but routing is allowed above them, the load-capacitance-aware obstacle-avoidance algorithm in [12] can be utilized to detour the clock-tree wires that $(i)$ cross macro blocks, $(ii)$ but cannot be driven by the buffers outside macro blocks.

[6]Unit length in the ISPD'05 benchmark corresponds to approximately $117nm$ in the new benchmark set.

| Name | Cells | Regs | Macros | CoreX ($mm$) | CoreY ($mm$) | Area ($mm^2$) |
|---|---|---|---|---|---|---|
| clkad1 | 210K | 32K | 56 | 1.247 | 1.246 | 1.554 |
| clkad2 | 255K | 38K | 177 | 1.640 | 1.638 | 2.686 |
| clkad3 | 451K | 68K | 721 | 2.706 | 2.722 | 7.363 |
| clkad4 | 494K | 74K | 1329 | 2.706 | 2.722 | 7.363 |
| clkbb1 | 278K | 42K | 30 | 1.247 | 1.246 | 1.554 |
| clkbb2 | 535K | 84K | 923 | 2.181 | 2.192 | 4.781 |
| clkbb3 | 1095K | 165K | 666 | 3.231 | 3.242 | 10.47 |
| clkbb4 | 2169K | 327K | 639 | 3.756 | 3.772 | 14.16 |

TABLE I
THE NEW CLKISPD'05 BENCHMARKS.

| | | FastPlace3 | | | mPL6 | | | SimPL 101 | | | SimPL+Lopper | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bench | $\alpha_{avg}$ | ClkWL ($mm$) | HPWL ($m$) | Pwr ($mW$) | ClkWL ($mm$) | HPWL ($m$) | Pwr ($mW$) | ClkWL ($mm$) | HPWL ($m$) | Pwr ($mW$) | ClkWL ($mm$) | HPWL ($m$) | Pwr ($mW$) | ⏲ (min) |
| clkad1 | 0.109 | 214.7 | 9.119 | 285.5 | 248.2 | 9.092 | 298.3 | 209.1 | 8.968 | 279.9 | **152.3** | 9.233 | **263.0** | 3.58 |
| clkad2 | 0.099 | 236.2 | 10.92 | 310.1 | 267.0 | 10.74 | 318.9 | 223.1 | 10.54 | 297.6 | **161.0** | 10.83 | **278.4** | 5.51 |
| clkad3 | 0.091 | 469.3 | 24.95 | 640.8 | 467.6 | 24.99 | 640.8 | 468.5 | 24.08 | 624.7 | **326.9** | 24.90 | **583.0** | 10.7 |
| clkad4 | 0.112 | 540.9 | 23.12 | 732.9 | 615.6 | 22.62 | 751.6 | 519.4 | 21.70 | 692.6 | **354.4** | 22.32 | **640.4** | 11.5 |
| clkbb1 | 0.099 | 250.5 | 11.24 | 323.6 | 245.1 | 11.29 | 322.5 | 238.2 | 11.18 | 317.6 | **166.3** | 11.53 | **295.7** | 5.20 |
| clkbb2 | 0.149 | 539.2 | 18.07 | 752.6 | 514.1 | 17.77 | 733.6 | 533.2 | 16.75 | 710.9 | **371.2** | 17.26 | **661.4** | 13.1 |
| clkbb3 | 0.103 | 892.6 | 42.65 | 1236 | 1032 | 40.15 | 1240 | 866.3 | 39.22 | 1155 | **602.2** | 40.97 | **1085** | 29.8 |
| clkbb4 | 0.093 | 1907 | 97.32 | 2575 | 2119 | 96.77 | 2650 | 1855 | 92.96 | 2473 | **1266** | 95.21 | **2279** | 86.9 |
| Avg | | 1.03× | 1.05× | 1.04× | 1.11× | 1.03× | 1.06× | 1.00× | 1.00× | 1.00× | **0.70**× | 1.03× | **0.93**× | 1.81× |

TABLE II
RESULTS ON THE CLKISPD'05 BENCHMARK SUITE. CLKWL REPRESENTS TOTAL WIRELENGTH OF A CLOCK NETWORK SYNTHESIZED BY THE INITIAL PHASE OF CONTANGO 2.0 [14]. HPWL IS TOTAL HPWL OF SIGNAL NETS. PWR IS TOTAL NET-SWITCHING POWER. SIMPL+LOPPER IS 4.16× FASTER THAN MPL6 AND 1.51×, 1.81× SLOWER THAN FASTPLACE3, SIMPL RESPECTIVELY.

of core area except when it is blocked by macros. When the desired location is blocked, we move the clock source to the closest unblocked coordinate. Since many academic placers handle the ISPD'05 benchmarks, a direct comparison of clock-network quality and signal-net wirelength is possible. The new benchmarks (referred to as *CLKISPD'05*, downloadable from http://vlsicad.eecs.umich.edu/BK/CLKISPD05bench [13]) are described in Table I.

The quality of clock networks based on the final register locations of each placer is evaluated by Contango 2.0 [14]. Contango 2.0 is the winner of the ISPD 2009 and 2010 Clock Network Synthesis (CNS) contests and produces clock trees with less than 7.5 $ps$ skew in the presence of variation on the ISPD'10 CNS benchmarks. During our experiments in Section VI-B , we exclude SPICE-accurate tuning in Contango 2.0 for two reasons: (1) the designs from the ISPD'05 benchmarks are too large to run SPICE simulations, (2) the average added capacitance during the SPICE-driven optimization on the ISPD'10 CNS benchmarks is 2.2% of total clock-net capacitance (including sink, wire and buffer capacitance), suggesting that the initial trees optimized for Elmore delay provide good estimates of power consumption. In Section VI-C, we present experimental results for clock trees with SPICE-driven optimizations on the modified benchmark set. Insertion delay and skew from SPICE simulations are reported as well as total capacitance of optimized clock trees with driving buffers. We also present robustness analysis of different register placements when total capacitance is limited for clock networks in the presence of variations.

### B. Empirical results

Table II compares results of our methodology to the leading academic placers on the CLKISPD'05 benchmarks. The results of SimPL [10] are used as reference for comparison. $\alpha_{avg}$ is computed for each benchmark based on the given $\beta = 0.3$ as described in Section VI-A, and total wire-switching power is calculated based on $\alpha_{avg}$. Power consumed inside macro blocks is ignored since it cannot be optimized during placement and is not available in original ISPD benchmark data. On average, the combination of SimPL and Lopper reduces total clock-tree length by 30.0%, total wire-switching power by 6.8% while the total HPWL of the signal nets only increases by 3.1% compared to SimPL. Compared to FastPlace3 [29] and mPL6 [2], our methodology reduces the total clock-net wirelength by 32.1%, 36.6%, total wire-switching power by 10.5%, 11.6% respectively, while the total signal-net HPWL is smaller than that produced by FastPlace3 by 1.4% and very similar to that produced by mPL6. Our methodology shows consistent improvement for the benchmarks considered, with various configurations of macro blocks. Figure 10 compares two clock trees based on different register placements from SimPL and our method. Registers are locally clustered by our method to reduce clock-net wirelength. The size of a cluster is automatically determined based on the clock-tree topology and connectivity between registers and combinational logic.

To further study the relative significance of clock-power ratio $\beta$, we show in Table III the impact of varying $\beta$ on the benchmark *clkad1*. The average activity factor of signal nets $\alpha_{avg}$ is computed based on the reference layout and utilized for computing the total wire-switching power. The performance of Lopper is improved when clock networks consume a greater portion of total power. Table III also shows that reducing clock networks does not necessarily reduce the total switching power. For example, the result for $\beta = 0.6$ consumes 109.6 $mW$ for total wire-switching power, but if the same circuit is used for the applications with $\beta = 0.1$, the total wire-switching power computed by Equations 1 - 3

| $\beta$ | $\alpha_{avg}$ | Orig. P ($mW$) | ClkWL ($mm$) | HPWL ($m$) | Pwr ($mW$) | Pwr (Rel) |
|---|---|---|---|---|---|---|
| Orig | - | - | 209.1 | 8.968 | - | - |
| 0.1 | 0.420 | 837.0 | 184.2 | 9.073 | **835.8** | **0.999** |
| 0.15 | 0.264 | 557.2 | 173.5 | 9.128 | **551.3** | **0.990** |
| 0.2 | 0.187 | 419.1 | 165.7 | 9.188 | **409.9** | **0.978** |
| 0.25 | 0.140 | 334.8 | 158.0 | 9.225 | **321.5** | **0.960** |
| 0.3 | 0.109 | 279.9 | 152.3 | 9.233 | **262.2** | **0.939** |
| 0.35 | 0.087 | 239.7 | 151.0 | 9.280 | **221.9** | **0.925** |
| 0.4 | 0.070 | 209.2 | 144.8 | 9.305 | **188.2** | **0.900** |
| 0.45 | 0.057 | 185.9 | 144.5 | 9.316 | **164.0** | **0.882** |
| 0.5 | 0.047 | 168.0 | 139.5 | 9.342 | **143.6** | **0.854** |
| 0.55 | 0.038 | 151.8 | 135.7 | 9.343 | **125.3** | **0.826** |
| 0.6 | 0.031 | 139.3 | 128.0 | 9.425 | **109.6** | **0.787** |

TABLE III
THE RESULTS ON *clkad1* WITH VARIOUS CLOCK POWER RATIOS $\beta$. THE SPECIFICATIONS OF THE REFERENCE PLACEMENT PRODUCED BY SIMPL ARE IN THE ROW *Orig*. $\alpha_{avg}$ IS CALCULATED BASED ON $\beta$ AND REFERENCE PLACEMENT PRODUCED BY SIMPL. TOTAL WIRE-SWITCHING POWER VALUES OF THE REFERENCE PLACEMENT WITH THE CORRESPONDING $\beta$ ARE REPRESENTED IN THE COLUMN *Orig. P*. THE RELATIVE POWER RATIOS ARE INDICATED WITH *Rel*.

| | Orig. Flow | | w/o OAVCT | | w/o OAF | |
|---|---|---|---|---|---|---|
| Bench | ClkWL ($mm$) | Pwr ($mW$) | ClkWL ($mm$) | Pwr ($mW$) | ClkWL ($mm$) | Pwr ($mW$) |
| clkad1 | 152.3 | 263.0 | 165.7 | 267.8 | 158.5 | 265.3 |
| clkad2 | 161.0 | 278.4 | 170.9 | 285.5 | 163.7 | 278.7 |
| clkad3 | 326.9 | 583.0 | 362.1 | 595.1 | 340.8 | 587.4 |
| clkad4 | 354.4 | 640.4 | 403.1 | 657.2 | 379.8 | 649.4 |
| clkbb1 | 166.3 | 295.7 | 172.6 | 297.4 | 169.1 | 296.4 |
| clkbb2 | 371.2 | 661.4 | 411.2 | 673.8 | 389.9 | 666.7 |
| clkbb3 | 602.2 | 1085 | 663.1 | 1104 | 627.2 | 1093 |
| clkbb4 | 1266 | 2279 | 1412 | 2331 | 1328 | 2102 |
| Avg | **1.0**× | **1.0**× | +**9.5**% | +**1.8**% | +**4.1**% | +**0.7**% |

TABLE IV
IMPACT OF EXCLUDING OBSTACLE-AWARE VIRTUAL CLOCK TREES (OAVCT), OBSTACLE AVOIDANCE FORCES (OAF). OAVCT AND OAF ARE EXCLUDED IN THE COLUMNS UNDER "W/O OAVCT". ONLY OAF IS REMOVED IN "W/O OAF"

is 842.9 $mW$, which is greater than the switching power of the reference placement 837.0 $mW$. This implies that clock-net optimization must utilize activity factors of signal nets or clock-power ratios to reduce total switching power.

Table IV shows the impact of obstacle-aware virtual clock trees (OAVCT) and obstacle avoidance forces (OAF). When OAVCT is excluded, DME trees without obstacle handling are utilized for the remaining flow. The results indicate that 9.5% of clock-net wirelength can be reduced on average by utilizing obstacle-aware trees. The advantage of OAVCT is reduced on benchmarks with a few obstacles such as *clkbb1* where a few obstacles exist at the top left corner of the chip. Obstacle-avoidance forces reduce clock-net length by 4.1% and total switching power by 0.7%.

Table V compares results of our technique to the technique called MLAF on MLBB [32]. We re-implemented their MLAF algorithm and integrated it into the SimPL placer [10] instead of the FDP framework [30] they utilized. Since their DCTB algorithm cannot process obstacles, our obstacle-aware virtual clock-tree generation algorithm in Section IV-A is utilized for the MLAF algorithm. In terms of clock-net wirelength and net-switching power, the average gain from the MLAF technique is limited by 43.5%, 30.6% of the improvement of our technique respectively, which means that our arboreal clock-net contraction force is 3.3× more effective for switching-power reduction than MLAF. Our comparison to MLAF concludes that explicit and structural representation of clock-net force and an accurate weighting function are important to achieve competitive register placement.

| | SIMPL+MLAF | | |
|---|---|---|---|
| Bench | ClkWL ($mm$) | HPWL ($m$) | Pwr ($mW$) |
| clkad1 | 182.4 (46.9%) | 9.194 (85.3%) | 274.2 (33.7%) |
| clkad2 | 200.9 (35.8%) | 10.76 (76.2%) | 293.0 (24.0%) |
| clkad3 | 402.5 (46.6%) | 24.71 (76.9%) | 609.8 (35.7%) |
| clkad4 | 449.5 (42.4%) | 22.24 (86.9%) | 676.6 (30.7%) |
| clkbb1 | 203.8 (47.9%) | 11.48 (84.9%) | 309.7 (36.1%) |
| clkbb2 | 473.8 (36.7%) | 17.16 (80.0%) | 699.3 (23.4%) |
| clkbb3 | 743.5 (46.5%) | 40.81 (91.0%) | 1139 (22.9%) |
| clkbb4 | 1587 (45.5%) | 94.77 (80.2%) | 2399 (38.1%) |
| Avg | **0.87**× **(43.5**%**)** | **1.03**× **(82.7**%**)** | **0.98**× **(30.6**%**)** |

TABLE V
RESULTS OF THE MLAF TECHNIQUE INTEGRATED INTO SIMPL WITH COMPARISON TO OUR TECHNIQUE. AVERAGE RESULTS ARE COMPARED TO THE RESULTS FOR SIMPL IN TABLE II. THE NUMBERS IN PARENTHESES REPRESENT THE AMOUNT OF REDUCTION(CLKWL, PWR) [INCREASE(HPWL)] ASSUMING 100% REDUCTION [INCREASE] FOR OUR TECHNIQUE. FOR EXAMPLE, [209.1(SIMPL) - 182.4(MLAF)] / [209.1(SIMPL) - 152.3(LOPPER)] = 46.9%.

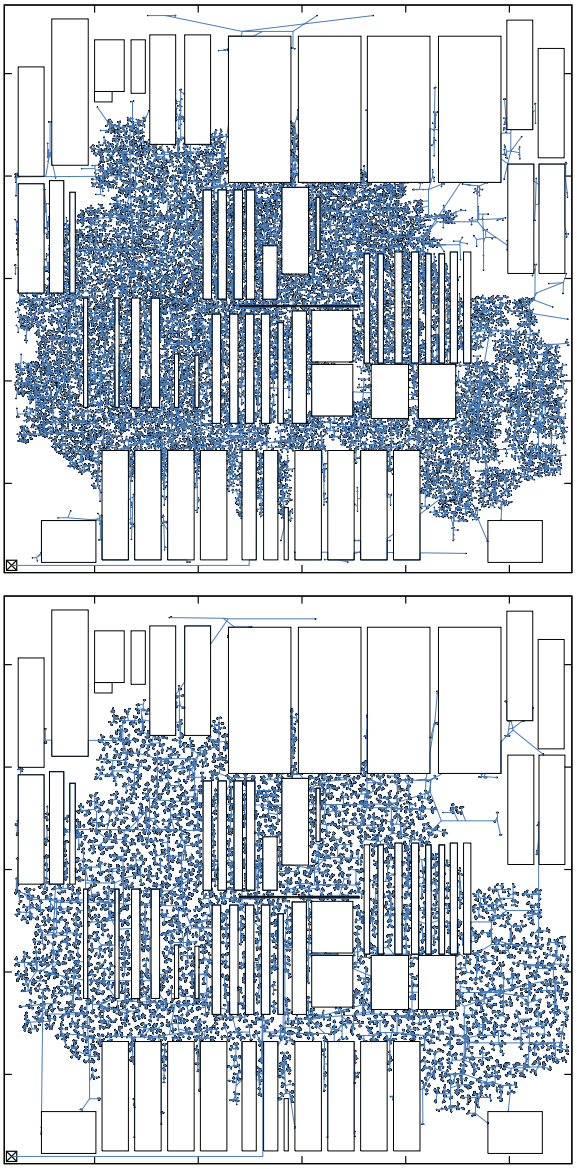


Fig. 10. Clock trees for clkad1, based on a SimPL register placement (top) and produced by proposed techniques (bottom). The respective clock-tree wirelengths based on SimPL and our method are 209.13 $mm$ and 152.27 $mm$. The total switching power of SimPL and our method are 279.9 $mW$ and 263.0 $mW$ respectively.

### C. SPICE-driven validation

Insertion delay and skew are important metrics when evaluating the quality of clock trees. Accurate analysis of these metrics requires SPICE simulations. The CLKISPD'05 benchmark set has up to 327K registers in one benchmark

| | | FastPlace3 | | | mPL6 | | | SimPL 101 | | | SimPL+Lopper | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bench | Regs | Ins. D. ($ps$) | Skew ($ps$) | Cap. ($pF$) | Ins. D. ($ps$) | Skew ($ps$) | Cap ($pF$) | Ins. D. ($ps$) | Skew ($ps$) | Cap. ($pF$) | Ins. D. ($ps$) | Skew ($ps$) | Cap. ($pF$) |
| clkad1_s | 2114 | 386.3 | 2.702 | 29.90 | 388.2 | 2.629 | 34.09 | 381.7 | 3.730 | 26.61 | 369.0 | 0.962 | **16.70** |
| clkad2_s | 2550 | 406.8 | 3.329 | 35.95 | 414.2 | 2.471 | 36.24 | 405.8 | 3.152 | 31.52 | 402.9 | 2.163 | **20.53** |
| clkad3_s | 4516 | 453.7 | 3.921 | 68.08 | 468.0 | 3.642 | 77.36 | 453.4 | 3.770 | 65.95 | 452.3 | 2.208 | **45.00** |
| clkad4_s | 4960 | 455.4 | 3.502 | 77.84 | 470.0 | 2.065 | 86.28 | 454.8 | 3.568 | 71.66 | 446.6 | 2.216 | **44.44** |
| clkbb1_s | 2781 | 385.0 | 2.711 | 29.86 | 387.0 | 2.139 | 35.61 | 386.1 | 5.166 | 29.10 | 385.9 | 1.754 | **18.35** |
| clkbb2_s | 5578 | 445.1 | 8.153 | 78.34 | 444.4 | 5.984 | 83.77 | 444.4 | 1.876 | 75.18 | 431.4 | 2.537 | **47.18** |
| clkbb3_s | 10968 | 489.8 | 3.118 | 125.7 | 513.0 | 7.140 | 150.2 | 497.6 | 2.796 | 120.7 | 494.2 | 2.286 | **77.75** |
| clkbb4_s | 21773 | 523.9 | 2.318 | 268.6 | 522.7 | 2.956 | 284.0 | 523.5 | 3.511 | 250.8 | 510.9 | 2.852 | **156.0** |
| Avg | | 1.016× | 1.83× | 1.68× | 1.033× | 1.77× | 1.87× | 1.016× | 1.85× | 1.57× | 1.000× | 1.00× | **1.00**× |

TABLE VI
RESULTS OF SPICE-DRIVEN OPTIMIZATIONS ON THE MODIFIED CLKISPD'05 BENCHMARK SUITE. REGS REPRESENTS THE NUMBER OF REGISTERS IN EACH BENCHMARK. INS. D. IS INSERTION DELAY AND SKEW IS NOMINAL LOCAL SKEW DEFINED IN [14] WITH LOCAL SKEW DISTANCE LIMIT $600\mu m$. CAP. REPRESENTS TOTAL CAPACITANCE OF THE CLOCK TREE INCLUDING DRIVING BUFFERS.

and it is impractical to perform SPICE-driven optimizations introduced in [14]. To construct high-quality SPICE-accurate clock trees, we decrease the number of registers in the CLKISPD'05 benchmarks and invoke Contango2.0 on various register placements. The experimental results with SPICE-driven optimizations are described in Table VI.

We adapted wire and buffer libraries from the ISPD 2010 CNS benchmarks. The concept of local skew defined in [14] is utilized to calculate exact skew with the local skew distance limit $600\mu m$. Unlike Table II that reports dynamic power based on only wire capacitance, the total capacitance including driving buffers is reported in Table VI. Same buffering scheme is utilized for all clock trees in this table. The Lopper flow produces high-quality register placement and it is already shown in Table II that the clock-network wirelength is significantly smaller than other methods. Due to compact clock-net wirelength, fewer driving buffers are required for clock-tree synthesis on our register placement, hence the total clock-network capacitance including buffers is 57%∼87% less than other register placements.

Table VI also shows that insertion delay based on our method is 1.6%∼3.3% smaller than other methods. Since insertion delay depends on the path length from the clock source to sinks, the size of layout area is closely related to insertion delay. As shown in Figure 10, the wirelength of clock trees is reduced mainly near leaves, hence the improvement of insertion delay is not proportional to power improvement.[7] In practice, clock trees spend a lot of power near the leaves. The results show that Contango2.0 can reduce nominal skew down to $10ps$ for any register placement. The quality of nominal skew is not highly related to register placement but it depends on optimization performed during CNS. However, compact clock trees are easier to tune therefore average nominal skew reported in Table VI is 77%∼87% smaller on our design.

Table VII shows how the quality of register placement affects robustness of clock networks in the presence of variations. Since robustness analysis requires extensive SPICE simulations, we rebuilt the CLKISPD'05 benchmarks with fewer registers, so that we can run hundreds of Monte-Carlo SPICE simulations. We imposed a capacitance limit when running Contango 2.0, and clock trees are optimized to be as robust as possible within this limit. After building clock trees, we first measure the nominal skew of clock trees without variations. Then we run extensive Monte-Carlo simulations with variations to estimate the impact of variations on clock-tree circuits. The variation model from the ISPD 2010 CNS benchmarks is utilized in the experiments.

[7]When the clock source is at a corner of chip area, often 30%∼50% of insertion delay is due to the tree trunk (the wire that connects the clock source and the root node of the clock tree [12]) and the length of the trunk is largely unaffected by register placement *in practice*.

| | | | SimPL 101 | | | SimPL+Lopper | | |
|---|---|---|---|---|---|---|---|---|
| Bench | Regs | Cap. ($pF$) | Nom. ($ps$) | Mean ($ps$) | Yield (%) | Nom. ($ps$) | Mean ($ps$) | Yield (%) |
| clkad1_v | 1057 | 23 | 1.855 | 6.634 | 72.4 | 2.177 | 4.298 | **99.2** |
| clkad2_v | 1275 | 28 | 1.113 | 5.771 | **90.4** | 3.265 | 5.291 | 89.8 |
| clkad3_v | 1354 | 46 | 1.132 | 7.281 | 60.4 | 2.673 | 6.615 | **72.4** |
| clkad4_v | 1488 | 50 | 2.010 | 8.007 | 43.2 | 1.610 | 6.094 | **82.0** |
| clkbb1_v | 1390 | 26 | 1.014 | 6.342 | 78.8 | 0.994 | 5.104 | **94.6** |
| clkbb2_v | 1115 | 39 | 2.208 | 7.674 | 49.0 | 1.338 | 5.912 | **86.6** |
| clkbb3_v | 1096 | 49 | 1.532 | 7.246 | 58.4 | 1.379 | 6.071 | **82.4** |
| clkbb4_v | 1088 | 67 | 5.226 | 9.607 | 15.4 | 4.207 | 7.342 | **57.8** |
| Avg | | | 2.011 | 7.320 | 58.5 | 2.205 | 5.841 | **83.1** |

TABLE VII
RESULTS OF SPICE SIMULATIONS IN THE PRESENCE OF VARIATIONS. REGS REPRESENTS THE NUMBER OF REGISTERS IN EACH BENCHMARK. CAP. REPRESENTS THE CAPACITANCE LIMIT FOR CLOCK NETWORKS. NOM. REPRESENTS NOMINAL SKEW WITHOUT VARIATION AND MEAN IS AVERAGE SKEW WITH VARIATION. YIELD REPRESENTS THE PERCENTAGE OF ACCEPTABLE RESULTS WITH GIVEN SKEW LIMIT $7.5ps$.

Our register placement leads to more compact clock trees than other methods, and robustness is further enhanced by Contango2.0. The results show that the clock trees based on our techniques offer 24.6% greater yield than the clock trees based on simPL alone when the skew limit is set to $7.5ps$.

## VII. CONCLUSIONS

Despite the increasing significance of power optimization in VLSI, state-of-the-art placement algorithms only optimize signal-net switching power and ignore clock-network switching responsible for over 30% of total power. We propose new techniques and a methodology to optimize total dynamic power during placement for large IC designs with macro blocks. To this end, we advocate obstacle-aware virtual clock-tree synthesis, a arboreal clock-net contraction force with virtual nodes that can handle gated clocks, and an obstacle-avoidance force for clock edges. Our methodology is integrated into the SimPL placer [10], and the total switching power is measured by utilizing Contango 2.0 [14] — both programs are leading academic software. A new set of 45 $nm$

benchmarks is proposed to better represent modern IC designs. Experimental results show that our method lowers the overall dynamic power by significantly reducing clock-net switching power. Other benefits of our optimizations include smaller insertion delay in clock trees, diminished sensitivity to process variations, and reduced supply voltage noise.

## REFERENCES


[1] C. J. Alpert, D. P. Mehta, S. S. Sapatnekar, "Handbook of Algorithms for Physical Design Automation," *CRC Press*, 2009.

[2] T. F. Chan et al, "mPL6: Enhanced Multilevel Mixed- Size Placement," *ISPD*'06, pp. 212-214.

[3] Y.-T. Chang et al, "Post-Placement Power Optimization with Multi-Bit Flip-Flops," *ICCAD*'10, pp. 218-223.

[4] Y. Cheon, P.-H. Ho, A. B. Kahng, S. Reda and Q. Wang, "Power-Aware Placement," *DAC*'05, pp. 795-800.

[5] M. Donno, E. Macci, and L. Mazzoni, "Power-Aware Clock Tree Planning," *ISPD*'04, pp. 138-147.

[6] P. E. Gronowski, W. J. Bowhill, R. P. Preston, M. K. Gowan, and R. L. Allmon, "High-Performance Microproccesor Design," *IEEE JSSC*, 33(5) (1998), pp. 676-686.

[7] W. Hou, X. Hong, W. Wu and Y. Cai, "A path-based timing-driven quadratic placement algorithm," *ASPDAC*'03, pp. 745-748.

[8] A. B. Kahng, J. Lienig, I. L. Markov and J. Hu, "VLSI Physical Design: From Graph Partitioning to Timing Closure," Springer 2011.

[9] A. B. Kahng,C.-W. Tsao, "Practical Bounded-Skew Clock Routing," *J. VLSI Signal Proc.* 16(1997), pp.199-215.

[10] M.-C. Kim, D.-J. Lee and I. L. Markov, "SimPL: An Effective Placement Algorithm," *ICCAD*'10, pp. 649-656.

[11] J. M. Kleinhans, G. Sigl, F. M. Johannes and K. J. Antreich, "GORDIAN: VLSI Placement by Quadratic Programming and Slicing Optimization," *IEEE Trans. on CAD*, 10(3): 356-365, 1991.

[12] D.-J. Lee, I. L. Markov, "Contango: Integrated Optimization of SoC Clock Networks," *DATE*'10, pp. 1468-1473.

[13] D.-J. Lee, I. L. Markov, The CLKISPD'05 Benchmark Suite, 2011. Downloadable from http://vlsicad.eecs.umich.edu/BK/CLKISPD05bench

[14] D.-J. Lee, M.-C. Kim and I. L. Markov, "Low-Power Clock Trees for CPUs," *ICCAD*'10, pp. 444-451.

[15] Y. Liu, X. Hong, Y. Cai, W. Wu, "CEP: A Clock-Driven ECO Placement Algorithm for Standard-Cell Layout," *ASIC*'01, pp. 118-121.

[16] J. Lu et al, "A Dual-MST Approach for Clock Network Synthesis," *ASPDAC*'10, pp. 467-473.

[17] Y. Lu et al, "Navigating Registers in Placement for Clock Network Minimization," *DAC*'05, pp. 176-181.

[18] N. Magen, A. Kolodny, U. Weiser, and N. Shamir, "Interconnect-power Dissipation in a Microprocessor," *SLIP*'04, pp. 7-13.

[19] G. J. Nam, C. J. Alpert, P. Villarrubia, B. Winter and M. Yildiz, "The ISPD2005 Placement Contest and Benchmark Suite," *ISPD*'05, pp. 216-220.

[20] Nangate Inc. Open Cell Library v2009 07, 2009. Downloadable from http://www.nangate.com/openlibrary

[21] J. Oh and M. Pedram, "Gated Clock Routing for Low-Power Microprocessor Design", *IEEE Trans. on CAD*, Vol. 20, No. 6, pp. 715-722, 2001.

[22] R.P. Pokala, R.A. Feretich and R.W. McGuffin, "Physical Synthesis for Performance Optimization", *ASIC*'92, pp. 34-37.

[23] J. M. Rabaey, A. Chandrakasan, B. Nikolic, "Digital Integrated Circuits: A Design Perspective," *Prentice Hall*, Second Edition, 2003.

[24] X.-W. Shih et al, "Blockage-Avoiding Buffered Clock-Tree Synthesis for Clock Latency-Range and Skew Minimization," *ASPDAC*'10, pp. 395-400.

[25] G. Sigl, K. Doll and F. M. Johannes, "Analytical Placement: A Linear or a Quadratic Objective Function?" *DAC*'91, pp. 427-431.

[26] P. Spindler, U. Schlichtmann, F. M. Johannes, "Kraftwerk2 - A Fast Force-Directed Quadratic Placement Approach Using an Accurate Net Model," *IEEE Trans. on CAD*, 27(8) 2008, pp. 1398-1411.

[27] C. N. Sze, "ISPD 2010 High-Performance Clock Network Synthesis Contest: Benchmark Suite and Results," *ISPD*'10, pp. 143-143.

[28] V. Tiwari, D. Singh, S. Rajgopal, G. Mehta, R. Patel, and F. Baez, "Reducing Power in High-Performance Microprocessors," *DAC*'98, pp. 732-737.

[29] N. Viswanathan, M.Pan, C.Chu, "FastPlace 3.0: A Fast Multilevel Quadratic Placement Algorithm with Placement Congestion Control," *ASPDAC*'07, pp. 135-140.

[30] K. P. Vorwerk, A. Kennings and A. Vannelli, "Engineering details of a stable force-directed placer," *ICCAD*'04, pp. 795-800.

[31] M. Wang and M. Sarrafzadeh, "Congestion minimization during placement," *ISPD*'99, pp. 145-150.

[32] Y. Wang, Q. Zhou, X. Hong and Y. Cai, "Clock-Tree Aware Placement Based on Dynamic Clock-Tree Building," *ISCAS*'07, pp. 2040-2043.


**Dong-Jin Lee** was born in Gwangju, South Korea. He received the B.S. degree in electrical engineering from Seoul National University, South Korea, in 2006 and the M.S. and Ph.D. degrees in Electrical Engineering from University of Michigan, Ann Arbor, in 2008 and 2011 respectively. His research interests include EDA clock network synthesis, placement and routing algorithms. He is the winner of the ISPD 2009 and 2010 clock-newtwork synthesis contests and the recipient of the IEEE/ACM William J. McCalla Best Paper Award at ICCAD 2010. In 2009 and 2010, he worked as a research intern at Texas Instruments.

**Igor L. Markov** is an associate professor of Electrical Engineering and Computer Science at the University of Michigan. He received his Ph.D. in Computer Science from UCLA. He is a member of the Executive Board of ACM SIGDA, Editorial Board member of the Communications of ACM and IEEE Design & Test, as well as several ACM and IEEE Transactions. He chaired tracks at DAC, ICCAD, ICCD, DATE and GLSVLSI. Prof. Markov researches computers that make computers. He has co-authored three books and more than 180 refereed publications, some of which were honored by the best-paper awards at the Design Automation and Test in Europe Conference (DATE), the Int'l Symposium on Physical Design (ISPD) and IEEE Trans. on Computer-Aided Design. During the 2011 redesign of the ACM Computing Classification System, Prof. Markov lead the effort on the Hardware tree. Prof. Markov is the recipient of a DAC Fellowship, an ACM SIGDA Outstanding New Faculty award, an NSF CAREER award, an IBM Partnership Award, a Microsoft A. Richard Newton Breakthrough Research Award, and the inaugural IEEE CEDA Early Career Award.