Optimal Bounded-Skew Steiner Trees to Minimize Maximum k-Active Dynamic Power

Hamed Fatemi[1], Andrew B. Kahng[2,3], Minsoo Kim[3] and Jose Pineda de Gyvez[1]

[1]NXP Semiconductors, San Jose, CA, USA
[2]CSE and [3]ECE Departments, UC San Diego, La Jolla, CA, USA
hamed@fatemi.net, {abk, mik226}@ucsd.edu, jose.pineda.de.gyvez@nxp.com

ABSTRACT

Static Random-Access Memory (SRAM) is a key component of modern systems-on-chip (SOCs), appearing in on-chip cache memories, FIFOs, and register files. Increasingly, modern SOCs embed more memory hierarchies and various modules which require on-chip memory accesses due to the high cost of off-chip memory accesses, and the lower power density of memory fabrics that helps reduce need for "dark silicon". For such memory-dominated chips, the product specification and electronic device designers will focus on the *maximum* power consumption across all power usage scenarios, where a portion of memories are active and others are turned off by clock/power gates. In this work, we introduce and study *k-active dynamic power minimization* in bounded-skew trees, where we seek to minimize the maximum dynamic power consumption when at most k clock sinks are active. The sizes of SRAM blocks and the SOC die, relative to buffer distances in advanced nodes, effectively linearize clock power and wirelength of clock subtrees. We can therefore apply an extension of a flow-based ILP for bounded-skew Steiner tree construction, introduced at SLIP-2018 [1]. We also introduce and study *k-consecutive-active dynamic power minimization* in scenarios where only consecutively-indexed clock sinks can be active simultaneously. Further, we demonstrate how non-uniform underlying grids enable the ILP to more flexibly capture locations of terminals of trees. Finally, we study the potential tree cost reduction benefit of flexible clock source locations rather than fixed source locations. Our experimental results give new insight into the tradeoff of maximum k-(consecutive)-active dynamic power and wirelength, and of skew and wirelength.

ACM Reference Format:
Hamed Fatemi[1], Andrew B. Kahng[2,3], Minsoo Kim[3] and Jose Pineda de Gyvez[1]. 2020. Optimal Bounded-Skew Steiner Trees to Minimize Maximum k-Active Dynamic Power. In *System-Level Interconnect - Problems and Pathfinding Workshop (SLIP '20), November 5, 2020, San Diego, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3414622.3431908

1 INTRODUCTION

Static Random-Access Memory (SRAM) is a key component of modern systems-on-chip (SOCs), as a result of several inexorable trends. Firstly, off-chip memory accesses entail long latencies and large energy consumption. Thus, modern SOCs have larger memory hierarchies and integrate more modules, trading away high-cost off-chip memory accesses for lower-cost on-chip memory accesses. For example, microcontroller units contain multiple "masters", such as central processing units (CPUs), memory protection units (MPUs), floating-point processing units (FPUs), digital signal processors (DSPs), etc. More than one of these masters may access memory components simultaneously in many applications. Secondly, the lower power density of memory fabrics can help reduce a need for "dark silicon". Lastly, even as advanced technology nodes are aggressively scaled down, SRAM scaling has been slower than other (logic) scaling: device scaling (e.g., to single-fin transistors in 6T bitcells) causes susceptibility to variations and read/write instability [26]. Due to these trends, as well as emergence of new chip architectures for AI and machine learning, the area occupied by SRAM blocks now commonly approaches or exceeds 60% of the total die area in modern SOCs [13].

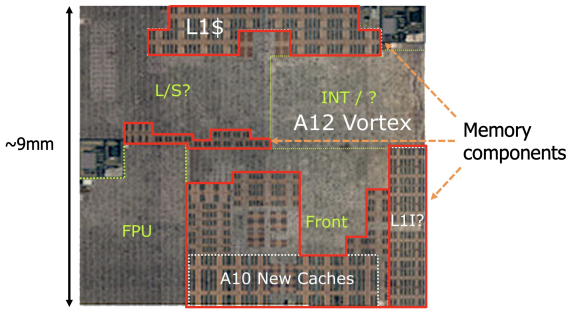


Figure 1: An example floorplan view of a memory-dominant chip [28]. The red boxes indicate memory components in the chip.

Figure 1 shows a die shot of a recent SOC product in the mobile application processor space. The reference [28] notes that the width and height of the SOC is around 9mm, and that there is significantly more memory content relative to the previous-generation product.

k-Active Clock Power Minimization

Low-power design, always a significant challenge, becomes more important in modern SOCs. Clock distribution is an important aspect of any low-power methodology. And, reducing the power consumed by clock distribution to SRAMs is a growing concern in *memory-dominant* designs. Figure 2 shows a "cartoon" view of a *memory-dominant* SOC floorplan and a tree structure for clock distribution; the figure captures the physical design challenges of clock distribution to memory blocks. The figure also shows a motivation for our proposed k-active dynamic power minimization problem formulation.

With memory-dominant designs, memory clusters (sometimes referred to as "banks") are defined by designers; SRAM instances in the same cluster are placed close together at the floorplan stage (four memories in a single cluster are highlighted in Figure 2). Importantly, in many low-power SOC products, the product specification guaranteed to OEM customers and other electronic product designers will focus on the **maximum** power consumption across all power usage scenarios wherein a portion of memories is "active"

SLIP '20, November 5, 2020, San Diego, CA, USA

ACM ISBN 978-1-4503-8106-2/20/11…$15.00
https://doi.org/10.1145/3414622.3431908

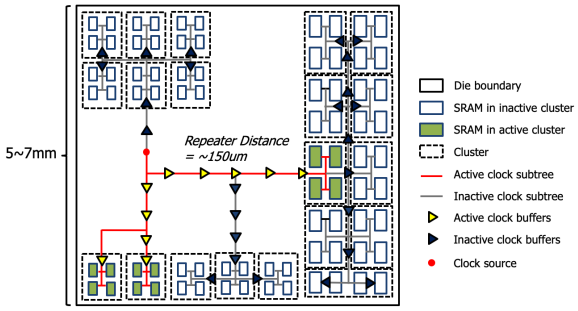


Figure 2: "Cartoon" of floorplan and clock tree structures for a memory-dominant SOC design, which motivates our k-active dynamic power minimization formulation. Active SRAMs and repeater components are marked in green; and active clock subtrees, from the clock source to active clock sinks, are marked in red. Other inactive clock subtrees are gated by power/clock gates.

(marked in green in Figure 2) and all other memories are turned off by clock/power gates. Figure 30 of [25] shows current consumption of active memory partitions (clusters) for an MCU product. The figure shows that the maximum difference of current consumption across the partitions is around 30%. Thus, minimization of maximum power consumption for active memory clusters is a real and significant issue at the SOC product level. Figure 2 also illustrates the insertion of repeaters in long global segments of the clock distribution. In recent technology nodes, the repeater distance is typically at most 150 to 200um, while die width and height can be 5 to 8mm or even more. The sizes of SRAM blocks and the SOC die, relative to repeater distances in advanced nodes, effectively "linearize" the calculation of clock power and buffered RC delay for top-level clock distribution.[1]

k-Active Dynamic Power Minimization Problem. In usage scenarios where only k memory clusters are active and the rest are turned off by power/clock gates, the clock power consumption depends on the locations of active clusters and the wirelengths of active clock subtrees. Among all such scenarios, the worst-case power becomes the maximum *k-active dynamic power*. We thus propose and study the *k-active dynamic power minimization problem*, where we seek to minimize the maximum dynamic power consumption when at most k clock sinks are active. Based on the linearity of power consumption and buffered RC delay in top-level clock distribution, we can model k-active dynamic power and skew based on the union of source-sink paths to a given set of k active clock sinks. We formally state this problem in Section 3.2 and show an ILP-based formulation.

k-Consecutive-Active Dynamic Power Minimization Problem. To address potentially more realistic scenarios, we remove unrealistic power usage scenarios which overconstrain the optimization and result in suboptimal clock tree constructions. Figure 2 shows a scenario where there are two neighboring clusters, and a third cluster is placed far away from the others. Depending on the SOC architecture and fine-grain access given to applications, the optimization may not need to worry about scenarios where two simultaneously active clusters or sinks are placed far apart. We thus introduce the *k-consecutive-active dynamic power minimization problem* which simplifies constraints in our ILP formulation, since only sinks that are physically close can be active simultaneously. In our study below, we index clock sinks (SRAM clusters) based on geometric information, and establish k-active power constraints only for groups of k active sinks that have consecutive indices. We formulate this problem in Section 3.3.

Flow-Based Integer Linear Programs (ILP) for Bounded-Skew Tree. In the recent work of [1], the authors propose a flow-based ILP formulation to find optimal (spanning and Steiner) trees, given locations of clock source, sinks and skew constraints. We review the work in detail in Section 3.1. We build on the work to address the k-active and k-consecutive-active dynamic power minimization problems. We also apply non-uniform grids to the previous ILP to obtain better results which more closely reflect routing channels in the SOC floorplan, and locations of memories. Finally, we study a scenario where we seek the optimal location of a *flexible* clock source rather than a fixed clock source; we show how to generally formulate the ILP for optimal tree construction when there are *flexible* and/or *forbidden* locations for the clock source. We discuss non-uniform grids and flexible or forbidden clock source locations in Sections 3.4 and 3.5, respectively.

We make the following contributions.

- Building on the work of [1], we use a flow-based integer linear programming (ILP) approach. We add new constraints to minimize a weighted sum of total clock distribution wirelengths and the maximum wirelengths of subtrees that reach k-active sinks. Our formulation addresses wirelengths of subtrees based on "k-active dynamic power".
- Our ILP formulation also addresses k-consecutive-active dynamic power of clock trees to remove unrealistic power usage scenarios.
- We apply non-uniform grids to the ILP problem to more accurately model terminal locations without incurring additional complexity increase.
- Going beyond pre-fixed clock source locations, we study effects of *flexible* as well as *forbidden* clock source locations. The former helps to reduce overall wirelength and wirelengths of subtrees with k-active sinks. The latter reflects scenarios where clock source locations are unavailable due to SOC floorplan constraints.

The remainder of this paper is organized as follows. Section 2 summarizes related previous works. Section 3 describes our main problem formulation and its several variants. Section 4 describes the setup and results for our experiments. We conclude in Section 5.

2 RELATED WORK

In this section, we review previous related works. Since we address power minimization problems for clock tree constructions, we first review the literature for zero-skew tree (ZST) and bounded-skew tree (BST) construction problems for clock trees. We also note several example approaches for low-power buffered clock tree construction in physical design.

Zero-Skew Tree (ZST) and Bounded-Skew Tree (BST) Construction. [4] [5] [12] [16] propose deferred-merge embedding (DME)-based zero-skew tree (ZST) constructions which subsequently became the foundation of many works for the bounded-skew tree construction problem. The works [9] [15] [17] address the bounded-skew Steiner tree construction problem with heuristic approaches. [7] uses hierarchical clustering for the ZST problem with dynamic programming. [21] proposes approximation algorithms for ZST and BST. Another part of the literature addresses optimal or approximate rectilinear Steiner minimal tree (RSMT) constructions. [2] proposes ILP (set covering) formulations to find a Steiner tree given a set of terminals and designated Steiner points. Other works [18] finds optimal Steiner trees with specified topology satisfying lower and upper pathlength bounds, using linear programming. [20] proposes exact algorithms to find Steiner trees based on mathematical programming and dynamic programming. [8] proposes FLUTE, a fast lookup table-based RSMT algorithm. FLUTE is optimal when the number of terminals is less than ten, and for the

[1]Power and delay are linear in distance, when long global interconnects are buffered with a regular buffer interval [3].

range of practical instance sizes offers an extremely competitive runtime-quality tradeoff point for the RSMT problem. [1] studies the skew-cost tradeoff and suboptimality in previous interconnect tree constructions, and proposes an optimal flow-based ILP approach.

Low-Power Buffered Clock Tree. Many works have been proposed to minimize clock power in physical design. In some situations, considering only wirelengths in clock trees as a proxy for cost (power) is not enough for real-world designs due to buffers on interconnects. [19] addresses buffered clock power minimization by proposing an algorithm to design a tree topology and simultaneously insert buffers. [11] proposes a methodology for power-aware clock tree planning, named *LPClock*, which constructs clock trees heuristically and inserts clock gating cells to minimize clock power. [6] proposes a heuristic approach to construct a low-power zero-skew clock network by buffer and clock gate insertions, given terminal locations and activity information. By contrast, in this work, we focus on clock tree constructions in memory-dominant SOC designs. By considering SRAM blocks, SOC die sizes and buffer distances in advanced nodes, we observe that both the clock power and the signal propagation delay in clock subtrees can be modeled as linear phenomena.

We are not aware of previous works that study minimization of subtrees across subsets of k sinks. Below, we point out that traditional bounded-skew or skew-wirelength tradeoffs do not directly address this new objective. On the other hand, directly minimizing the cost of k-sink clock subtrees in memory-dominated SOC designs dimensions can reduce clock distribution power in actual operation of IC products. Our work can provide guidance for clock distribution in such designs.

3 PROBLEM FORMULATION

In this section, we describe our problem formulation. First, we review the formulation of [1]; we then modify it to address the k-active dynamic power minimization problem. Second, we formulate the k-consecutive-active dynamic power minimization problem. Third, we incorporate non-uniform grids into the ILP formulation to more flexibly reflect clock source and sink locations. Last, we propose both *flexible* and *forbidden* clock source locations to further guide the choice of optimal clock source locations for reduced tree cost and/or subject to layout constraints. Table 1 shows the notations that we use in what follows.

Table 1: Notations

Notation	Meaning
p_i	i^{th} terminal ($p_i \in P$, p_1 is a root (source))
v_i	i^{th} vertex ($v_i \in V$, $P \subseteq V$)
e_{uw}	a directed edge from vertex v_u to vertex v_w ($e_{uw} \in E$)
λ_{uw}	0-1 indicator of whether e_{uw} is in a tree T
c_{uw}	cost of edge e_{uw}
f^i_{uw}	0-1 indicator of whether the flow to p_i goes through e_{uw}
d^i_u	pathlength at v_u along unique v_1-p_i path
m_{uw}	Manhattan distance from v_u to v_w
B	skew bound
L	lower bound on pathlength from source v_1
S_i	pathlength of p_1-p_i path
$S_{i,j}$	pathlength of p_1-p_i-p_j tree
$C_{i,j}$	length of common edges of p_1-p_i path and p_1-p_j path
$f^{i,j}_{uw}$	0-1 indicator of whether the flow to p_i and/or p_j goes through e_{uw}
$PMax(1)$	pathlength of the longest source-to-sink (p_i) path (p_1-p_i)
$PMax(2)$	length of the longest source-to-two-sinks (p_i, p_j) subtree (p_1-$p_i \cup p_1$-p_j)
M	a large constant

3.1 Review of the ILP Formulation of [1]

We now review the ILP formulation for optimal bounded-skew spanning and Steiner tree construction in the work [1]. Let x_i and y_i be coordinates of terminal p_i in the Manhattan plane. P denotes a set of terminals p_i over which a tree is to be constructed, where $i = \{1, 2, ..., |P|\}$. By convention, p_1 is a root (source) of the tree and the other terminals are leaves (sinks). A graph $G = (V, E)$ is constructed which includes terminals in P along with the Hanan grid [14] and additional *half-integer* grids (centered lines) between the predefined Hanan grids.[2] Every point at the intersection of two gridlines corresponds to a vertex $v_i \in V$, and directed edges e_{uw} from vertex v_u to vertex v_w are defined where v_u and v_w are neighboring vertices in the grid. Thus, each vertex v_i can have up to four incoming edges and four outgoing edges. The objective is to minimize the total cost of edges (i.e., total wirelength) in the constructed tree T.

Minimize: $\sum_{u,w} \lambda_{uw} \cdot c_{uw}$

Subject to:

$$\lambda_{uw} \geq f^i_{uw},\ \forall p_i \in P,\ e_{uw} \in E \tag{1}$$

$$\sum_u f^i_{uw} - \sum_u f^i_{wu} = \begin{cases} 1 \text{ if } v_w = v_1, \forall v_u \in V,\ p_i \in P,\ i \neq 1 \\ -1 \text{ else if } v_w = v_i \\ 0 \text{ otherwise} \end{cases} \tag{2}$$

$$\sum_{u,w} c_{uw} \cdot f^i_{uw} \geq L,\ \forall p_i \in P,\ i \neq 1 \tag{3}$$

$$\sum_{u,w} c_{uw} \cdot f^i_{uw} \leq L + B,\ \forall p_i \in P,\ i \neq 1 \tag{4}$$

Given a set of terminals P including source p_1, and pathlength lower and upper bounds L and $L + B$, where B is a skew bound, the ILP finds a spanning or Steiner tree over P that satisfies the pathlength bounds while minimizing $\sum_{u,w} \lambda_{uw} \cdot c_{uw}$.

f^i_{uw} denotes a binary indicator of whether the flow to p_i goes through e_{uw}. Constraint (1) enforces $\lambda_{uw} = 1$ when any flow goes through e_{uw}. Constraint (2) enforces flow conservation at all vertices except for the source (where the outgoing flow exceeds the incoming flow by the number of sinks) and any sink (where the sum of incoming flow exceeds the outgoing flow by one).

Skew bounds are enforced by Constraints (3) and (4), which apply the lower bound (L) and the upper bound ($L + B$) to each source-to-sink pathlength.

$$\begin{cases} d^i_u = 0 \text{ if } v_u = v_1, \forall\ p_i \in P,\ i \neq 1 \\ d^i_u \geq L \text{ else if } v_u = v_i, \forall\ p_i \in P,\ i \neq 1 \\ d^i_u \leq L + B \text{ otherwise } \forall\ p_i \in P,\ i \neq 1 \end{cases} \tag{5}$$

$$d^i_u + M \cdot (1 - f^i_{wu}) \geq d^i_w + c_{wu},\ \forall v_u, v_w \in V, u \neq w,\ p_i \in P \tag{6}$$

$$d^i_u - M \cdot (1 - f^i_{wu}) \leq d^i_w + c_{wu},\ \forall v_u, v_w \in V, u \neq w,\ p_i \in P \tag{7}$$

Constraints (5) to (7) prevent cycles in the output, i.e., they enforce a tree solution topology. Without these constraints, cycles can be introduced by detouring to satisfy skew constraints. To avoid cycles, we define a variable d^i_u which denotes pathlength from the source to vertex v_u for a path to sink p_i. For Constraint (5), the pathlength is zero when vertex v_u is the source. The pathlength must be maintained between lower and upper bounds for any vertices. Constraints (6) and (7) are added for pathlength d^i_u. When $f^i_{wu} = 1$, a pathlength increases by the edge cost c_{wu}, which means that d^i_u will be infinite in the presence of a cycle, and will not satisfy Constraint (5).

$$\text{if } m_{1u} + m_{ui} > L + B,\ \forall v_u \in V,\ u \neq 1, ..., |P|,\ p_i \in P$$
$$f^i_{uw} = 0,\ f^i_{wu} = 0,\ \forall e_{uw} \in E,\ e_{wu} \in E \tag{8}$$

$$\text{if } m_{1u} + m_{uu'} + m_{u'i} > L + B \ \&\&\ m_{1u'} + m_{u'u} + m_{ui} > L + B$$
$$f^i_{uw} + f^i_{u'w'} = 1,\ f^i_{wu} + f^i_{u'w'} = 1,\ \forall e_{uw} \in E,\ e_{wu} \in E \tag{9}$$
$$f^i_{uw} + f^i_{w'u'} = 1,\ f^i_{wu} + f^i_{w'u'} = 1,\ \forall e_{u'w'} \in E,\ e_{w'u'} \in E \tag{10}$$

[2] In the work of [1], optimal spanning and Steiner trees are constructed on Hanan grids or half-integer grids.

Constraints (8) to (10) are added to improve ILP runtime by special handling of vertices and edges that are irrelevant to the desired tree solution. Constraint (8) makes flow variables f^i_{uw} and f^i_{wu} zero when the sum of Manhattan distance from source p_1 to vertex v_u and from the vertex to sink p_i larger than upper bound $L + B$. This is because the pathlength of path p_1-p_i going through vertex v_u is always larger than the upper bound. Constraints (9) and (10) address a forbidden scenario involving two vertices. When the sum of the pathlengths from p_1 going through non-terminal vertices v_u and v'_u to sink p_i is larger than the upper bound $L + B$, then any flows for incoming and outgoing edges going through v_u and v'_u cannot exist simultaneously.

3.2 k-Active Dynamic Power Minimization

We extend the approach of [1] to address k-active dynamic power minimization problem, as follows. We first modify the objective function from the ILP formulation in Section 3.1, then we add six constraints to capture k-active dynamic power minimization problem for $k = 1$ and 2. Our objective function is to minimize the weighted sum of total cost of edges in an output tree T and the maximum k-active dynamic power. For k points other than p_1, we say that the k-active dynamic power is the total cost of edges in the union of the corresponding k source-to-sink paths in an output tree T. In addition, for a given output tree T, we define the maximum k-active dynamic power, $PMax(k)$, to be the maximum k-active dynamic power over all k-subsets of $\{p_2, \ldots, p_{|P|}\}$. Thus, our objective function and constraints can be formulated as follows.

$$\textbf{Minimize: } \sum_{u,w} \lambda_{uw} \cdot c_{uw} + \alpha \cdot PMax(1) + \beta \cdot PMax(2)$$

Given a set of terminals P including source p_1, pathlength bounds L and $L+B$, and parameters α and β, we construct a Steiner tree over P that satisfies the pathlength bounds while minimizing $\sum_{u,w} \lambda_{uw} \cdot c_{uw} + \alpha \cdot PMax(1) + \beta \cdot PMax(2)$. The first term of the objective function is the total cost of the edges. $\lambda_{uw} = 1$ when edge $e_{u,w}$ is included in the output tree T. The second and third terms are the weighted k-active dynamic power where $k = 1, 2$ respectively. Based on the principle of inclusion-exclusion, we can extend to any k-active costs from $k = 1$ and $k = 2$, at the cost of additional wirelength summation terms and constraints that govern k-tuples of sinks.

$$S_i \le PMax(1),\ \forall p_i \in P \tag{11}$$
$$S_{i,j} \le PMax(2),\ \forall p_i, p_j \in P \tag{12}$$
$$\sum_{u,w} c_{uw} \cdot f^i_{uw} \le S_i,\ \forall p_i \in P \tag{13}$$
$$S_i + S_j - C_{i,j} \le S_{i,j},\ \forall p_i, p_j \in P \tag{14}$$
$$2 \cdot f^{i,j}_{uw} \le f^i_{uw} + f^j_{uw} \tag{15}$$
$$\sum_{u,w} c_{uw} \cdot f^{i,j}_{uw} = C_{i,j},\ \forall p_i, p_j \in P \tag{16}$$

Constraints (11) to (16) are added for the $PMax(k)$ computation. In Constraint (11), S_i denotes a pathlength from the source p_1 to sink p_i and $PMax(1)$ denotes the maximum pathlength among all S_i. In Constraint (12), $S_{i,j}$ denotes a length of a source-sinks (p_i and p_j) subtree and $PMax(2)$ denotes the longest pathlength for the subtrees. Constraint (13) computes pathlength S_i from p_1 to p_i. In Constraint (14), $C_{i,j}$ is the common pathlength between the source-to-sink paths for p_i and p_j. The constraint is therefore with respect to the total edge length of a source-to-two-sinks (p_i and p_j) subtree. In Constraint (15), a new flow variable $f^{i,j}_{uw}$ is defined, which indicates whether edge e_{uw} is common to both of the source-to-sink paths to p_i and p_j. Constraint (16) computes common pathlength $C_{i,j}$ for sinks p_i and p_j.

3.3 k-Consecutive-Active Dynamic Power Minimization

In this subsection, we describe a k-consecutive-active dynamic power minimization problem. With this variant formulation, we capture locality of simultaneous memory accesses, and can thus relax constraints for unrealistic power usage scenarios which over-constrain the optimization.

In our study, we index clock sinks based on geometric information and consider locality according to adjacent indices. We treat the clock source as the origin, and associate each clock sink with its theta angle with respect to this origin. Sinks are indexed (in counterclockwise order) by theta angle.

We do not add new constraints to the ILP. Instead, we replace four constraints with the following:

$$S_{i,i+1} \le PMax(2),\ \forall p_i, p_{i+1} \in P \tag{17}$$
$$S_i + S_{i+1} - C_{i,i+1} \le S_{i,i+1},\ \forall p_i, p_{i+1} \in P \tag{18}$$
$$2 \cdot f^{i,i+1}_{uw} \le f^i_{uw} + f^{i+1}_{uw} \tag{19}$$
$$\sum_{u,w} c_{uw} \cdot f^{i,i+1}_{uw} = C_{i,i+1},\ \forall p_i, p_{i+1} \in P \tag{20}$$

For the k-consecutive-active dynamic power minimization problem, Constraints (17), (18), (20) and (19) replace Constraints (12), (14), (15) and (16) given in Section 3.2. In this way, constraints from the original k-active dynamic power minimization problem are relaxed for the k-consecutive-active dynamic power minimization problem. Instead of addressing all the possible combinations of k sinks, the relaxed constraints only address local groups of k sinks.

3.4 Non-uniform Grid

In this subsection, we demonstrate how non-uniform underlying grids enable the ILP to more flexibly capture locations of terminals and available routing resources in the Steiner tree construction. A non-uniform grid is created for a given floorplan and macro (SRAM) placement.[3] We consider channels between macros as well as clock pin locations (or midpoints of macros in clusters) of macros in our non-uniform grids; in the limiting case, the grid can be said to approach a *channel intersection graph* [10]. Removal of unusable edges in the grid can make the problem sparser and improve ILP runtimes.

Figure 3 shows uniform and non-uniform grids on an example floorplan from a RISC-V based design.[4] Figure 3(a) shows uniform grids with locations of terminals. Note that we would like to accurately capture the locations of clock pins (terminals) on SRAMs. However, in a uniform grid these locations must be snapped to nearest grid intersections. This causes a discrepancy between desired terminal locations and the terminal locations actually used on the underlying uniform grids, leading to suboptimal tree solutions. On the other hand, Figure 3(b) shows a non-uniform grid based on the macro channels and clock source locations, with removal of unusable grids where terminals are not located. Figure 3(c) shows non-uniform grids after removing unusable grids. In this way, we can use desired terminal locations of clock pins for our tree construction and have more flexibility of terminal locations without adding more complexity to the ILP.

[3] As in [1], a graph is constructed based on the Hanan grid and additional centered lines between the predefined grids.

[4] We synthesize netlists of *SweRV_wrapper* design [27] [30] using [29], and perform place-and-route using [22], in a foundry 14nm FinFET enablement. We perform macro placement by using [22] and manually adjust the macro placement. The design has 28 macros and 74K standard-cell instances.

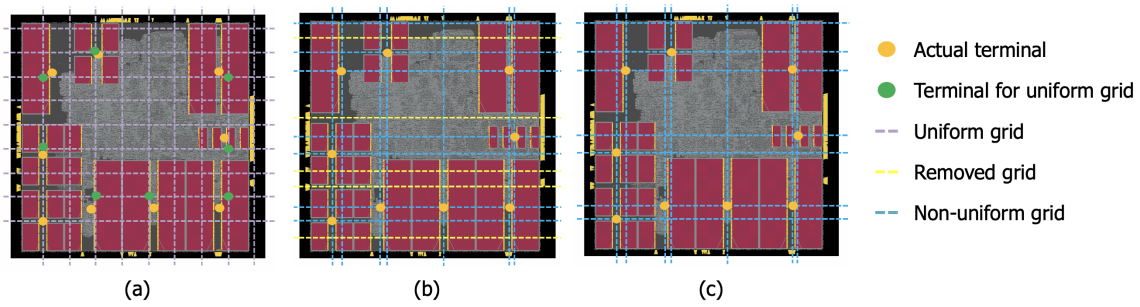


Figure 3: Terminals with uniform and non-uniform grids on an example floorplan. (a) Terminals with uniform grids. Actual terminals are located at yellow dots and the adjusted clock sinks are located at green dots. (b) Terminals with non-uniform grids based on channel intersection graph. The yellow dashed lines indicate unusable grids where terminals are not located. (c) Terminals with non-uniform grids after the unusable grids (the yellow dashed lines in (b)) are removed.

3.5 Flexible or Forbidden Clock Source Location

Given fixed locations of a clock source and sinks, we find an optimal solution to minimize costs while skew constraints are satisfied. However, in real designs, we might be able to move a clock source location unless fixed locations are hard constraints. For example, if a clock generator is placed and fixed at a specific area or pin assignments for top-level implementations are given, we cannot move the clock source location. Otherwise, we can achieve better tree solutions by moving clock source locations. To this end, we describe enablement of flexible clock source locations.

To allow all vertices (except sinks) to be sources, we define a clock source at a new virtual vertex v_0. In the graph $G = (V, E)$ constructed according to Section 3.2, we add virtual edges from the virtual vertex v_0 to every vertex v_i. Thus, each vertex v_i has up to five incoming edges and four outgoing edges. We then set the cost of each virtual edge to zero. The vertex that ends up connected to the virtual vertex through a virtual edge becomes the actual clock source. We define two constraints for flexible clock source locations, as follows.

$$\sum_{0,u} \lambda_{0u} = 1, \ \forall e_{0u} \in E \tag{21}$$

$$\lambda_{0u} = 0, \ \forall v_u = v_i, p_i \in P, i \neq 1 \tag{22}$$

Constraint (21) ensures that only one virtual edge is used in tree solution T. Constraint (22) is to avoid flows from a virtual vertex (clock source) to any sinks with zero cost.

Figure 4 shows example trees ($|P| = 12$) with a fixed clock source and a movable clock source. Figure 4(a) shows an optimal tree with the fixed clock source at (3, 5.6). The total wirelength of the tree solution is 25.6 when lower bound $L = 3.44$ and upper bound $L + B = 8.6$. In this example, we set $\alpha = 1, \beta = 0.5$ in the weighted sum of total wirelength and k-active dynamic power. Given a virtual clock source, the tree solution in (b) makes a connection from the virtual source to the vertex at (2.4, 3), which becomes the actual clock source. Figure 4(c) shows a combined tree with the tree from Figure 4(b) and the original clock source location from Figure 4(a). As seen in the data table of Figure 4, the tree of Figure 4(c) has larger cost than the tree of Figure 4(a), as we expect (tree (a) is optimal for this particular source location).

On the other hand, even if clock sources can be flexibly moved, they might not be placeable at certain locations due to geometric constraints. To address this, the ILP can be modified to reflect a "forbidden area" wherein clock sources cannot be placed. The following constraint is added to define the forbidden area for clock sources.

$$\lambda_{0u} = 0, \ \forall v_u \in V_r, \tag{23}$$

Constraint (23) prevents connections from the virtual edge to any vertices within a given forbidden area, where V_r denotes the set of vertices in the forbidden area. Figure 4(d) shows an example tree with flexible clock source locations and a forbidden area for the source. In this example, we set the forbidden area to be $2 \leq x \leq 4, 2 \leq y \leq 4$ – e.g., reflecting a situation where clock pins or clock generator cannot be placed in the middle of the die area.

4 EXPERIMENTAL SETUP AND RESULTS

4.1 Experimental Setup

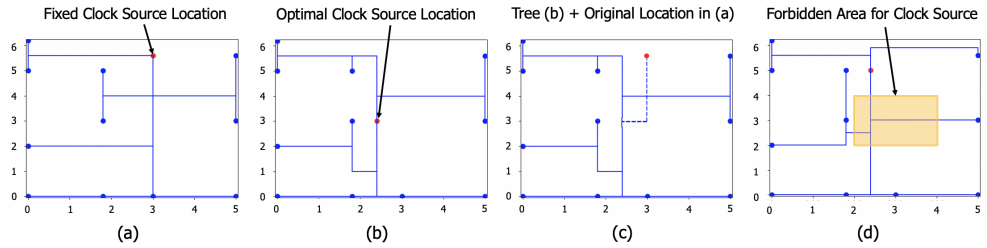
We implement our work in C++ with CPLEX 12.8.0 [24] and Gurobi 9.0.1 [23] as our ILP solvers. Our experiments are performed with four threads on a 2.6GHz Intel Xeon server. In our objective function, we set the weighting factors, $\alpha = 0, 1, 100$ and $\beta = 0, 0.5, 100$. For lower bounds, we set lower bound $L = \{4, 5, 6, 7, 8, 9\}$. For our runtime study, we set lower bound $L = M - B$ to avoid an infeasiblility, where M denotes the maximum Manhattan distance between source and sink. Also, we define skew bound B with M multiplied by 0.2, 0.4, 0.6, 0.8 (i.e., $B = 0.2 \cdot M, 0.4 \cdot M, 0.6 \cdot M, 0.8 \cdot M$). We also set unbounded cases with zero lower bound and infinite upper bound. In our experiments, we set two hours as a runtime limit of our ILP.[5]

4.2 Experimental Results

Study of weighting factors (α and β) in an objective function. We show example trees and scatter plots for skew-wirelength, $PMax(1)$-wirelength and $PMax(2)$-wirelength tradeoffs with various weighting factors α, β in our objective function. When $\alpha = \beta = 0$, the results are the same as those of the previous work [1]. In this way, we can compare our results to the work [1]. We observe that in practice, there are not many *non-dominated* trees for a given instance, across all values of α, β, lower and upper bounds. A tree T_1 is said to be *dominated* by tree T_2 if the wirelength, skew, $PMax(1)$ and $PMax(2)$ metrics of T_2 are all equal to or better than those of T_1. Figure 5 shows an example set of non-dominated solution trees for one instance in our experiment.

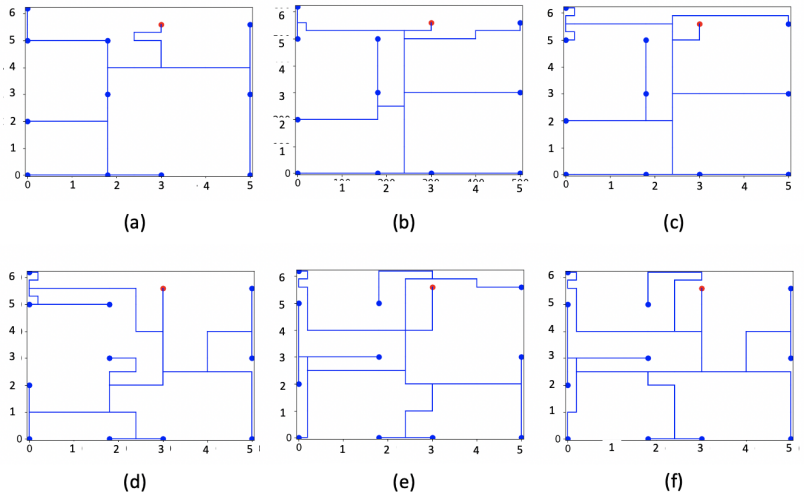
Figure 6 shows example trees with various weighting factors in our objective function. Given a 12-terminal pointset, lower bound $L = 6$ and skew bound $B = 0.4 \cdot M$, we find tree solutions where $\alpha = \beta = 0$, $\alpha = 100, \beta = 0$ and $\alpha = 0, \beta = 100$, respectively. We set lower bound $L = 4, 5, 6, 7, 8, 9$ in this experiment. Figure 6(a) shows the tree solution which the total wirelength of the tree is minimized. Figure 6(b) shows the tree solution which $PMax(1)$ is minimized. The solution has the minimum total wirelength among all solutions with minimum $PMax(1)$. Compared to Figure 6(a), $PMax(1)$

[5]The runtime limit is not applied to our runtime study in Table 2.



Metric	(a)	(b)	(c)	(d)
WL	25.6	24.4	27.6	25.4
Skew	5	2	2	4.1
PMax(1)	8.6	5.6	8.8	7.6
PMax(2)	12.2	11.2	14.4	11.4
Cost	40.3	35.6	43.6	38.7

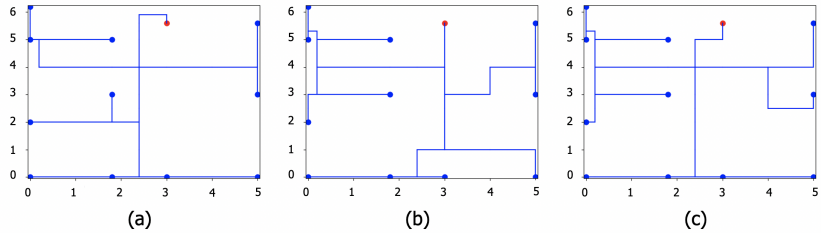
Figure 4: Example trees for flexible clock source locations when $\alpha = 1, \beta = 0.5, L = 3.44, B = 0.6 \cdot M$, and $L + B = 8.6$. (a) A tree with a fixed clock source at (3, 5.6). (b) A tree with a movable clock source, with optimal source location at (2.4, 3). (c) A tree with the tree from (b) and the original source location from (a). (d) A tree with a movable clock source that has a forbidden area ($2 \leq x \leq 4, 2 \leq y \leq 4$, yellow box).



Metric	(a)	(b)	(c)
WL	24.4	26.3	27.3
Skew	4.8	4.7	3.8
PMax(1)	9.8	8.8	8.8
PMax(2)	15.8	12.1	12.8

Metric	(e)	(f)	(g)
WL	32.4	36.3	38.6
Skew	1.6	1.8	1.5
PMax(1)	8.6	8.8	8.6
PMax(2)	15.6	14.0	14.4

Figure 5: An example set of non-dominated trees for a 12-terminal pointset. The non-dominated tree solutions have the weighting factors and bounds as follows. (a) $\alpha = \beta = 0, L = 5$ and $B = 0.6 \cdot M$. (b) $\alpha = 0, \beta = 100, L = 4$ and $B = 0.6 \cdot M$. (c) $\alpha = 0, \beta = 100, L = 5$ and $B = 0.6 \cdot M$. (d) $\alpha = 100, \beta = 0, L = 7$ and $B = 0.4 \cdot M$. (e) $\alpha = 0, \beta = 100, L = 7$ and $B = 0.4 \cdot M$. (f) $\alpha = 0, \beta = 100, L = 7$ and $B = 0.2 \cdot M$.



Metric	(a)	(b)	(c)
WL	**26.6**	28.7	28.7
Skew	3.2	2.4	2.6
PMax(1)	9.4	**8.6**	8.8
PMax(2)	14.6	14	**13.6**

Figure 6: Example trees with various weighting factors in our objective function where $L = 6, B = 0.4 \cdot M$ and $L + B = 9.44$ for a given 12-terminal pointset (a) A tree solution with $\alpha = \beta = 0$ to minimize the total wirelength only. (b) A tree solution with $\alpha = 100, \beta = 0$ to minimize $PMax(1)$. The solution has the minimum total wirelength among the solutions with the minimum $PMax(1)$. (c) A tree solution with $\alpha = 0, \beta = 100$ to minimize $PMax(2)$. The solution has the minimum total wirelength among the solutions with the minimum $PMax(2)$.

decreases from 9.4 to 8.6 while the total wirelength increases. Figure 6(c) shows the tree solution which $PMax(2)$ is minimized. The solution has the minimum total wirelength among all solutions with minimum $PMax(2)$. Compared to Figure 6(a), $PMax(2)$ decreases from 14.6 to 13.6 while the total wirelength increases. From this example, we can see that solution trees can be optimized for $PMax(1)$ and $PMax(2)$ by varying the weighting factors in the objective function.

Figure 7 shows scatter plots for skew-wirelength, $PMax(1)$-wirelength and $PMax(2)$-wirelength. Tradeoff curves with various weighting factors are shown in Figure 7(a). Compared to the cases that minimize only total wirelength ($\alpha = 0$ and $\beta = 0$, i.e., as in the work of [1]), the tradeoff curves for nonzero weighting factors form a "bundle" of tradeoffs curves. In terms of the skew-wirelength tradeoff, we see that considering k-active sinks does not affect skew much. Figure 7(b) gives a scatter plot for $PMax(1)$-wirelength, showing that $PMax(1)$ decreases when *alpha* is nonzero. Similarly, Figure 7(c) gives a scatter plot for $PMax(2)$-wirelength, showing that $PMax(2)$ decreases when *beta* is nonzero. Figure 7(b) includes

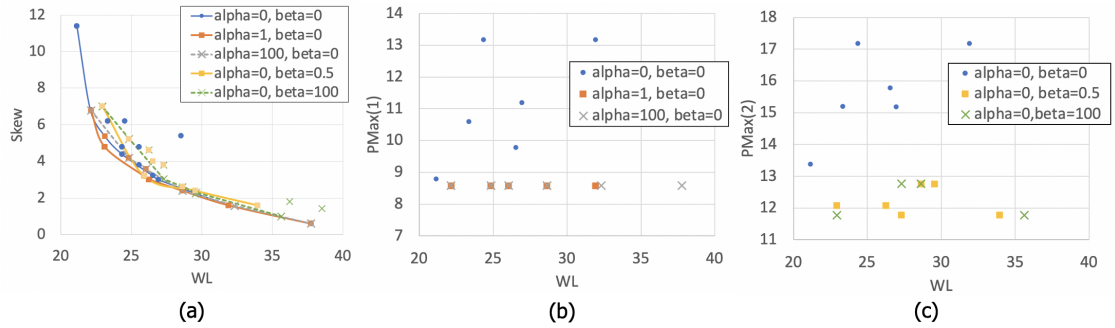


Figure 7: Scatter plots for (a) skew-wirelength, (b) $PMax(1)$-wirelength and (c) $PMax(2)$-wirelength for the 12-terminal pointset with various weighting factors α and β. Additional points that do not correspond to the figure legend are obtainable with other values of α, β, L and B.

the solutions with nonzero α and Figure 7(c) includes the solutions with nonzero β to show improved $PMax(1)$ and $PMax(2)$, respectively.

Study of k-consecutive-active sinks. As described above, we index sinks according to theta angle about the clock source, in counterclockwise order. Figure 8 shows scatter plots for skew-wirelength, $PMax(1)$-wirelength and $PMax(2)$-wirelength tradeoffs with k-active sinks and k-consecutive-active sinks. We set weighting factors $\alpha = 1, \beta = 0.5$ and lower bound $L = 4, 5, 6, 7, 8, 9$. Figures 8(a) and (b) respectively show skew-wirelength and $PMax(1)$-wirelength for k-consecutive-active sink solutions, similar to what was presented for k-active sink solutions. Figure 8(c) shows that $PMax(2)$ values of k-consecutive-active sinks are less than those of k-active sinks. This is because $PMax(2)$ of k-consecutive-active sinks only considers local subtrees with consecutive indices of clock sinks. (Note that the k-consecutive-active sinks optimization will lead to different solutions than the k-active sinks optimization only when the β weighting term for $PMax(2)$ is nonzero.)

Study of flexible or forbidden clock source locations. Figure 9 shows skew-wirelength, $PMax(1)$-wirelength and $PMax(2)$-wirelength scatter plots for flexible and fixed clock source locations for the 12-terminal pointset in Figure 4. In this experiment, we set weighting factors $\alpha = 1, \beta = 0.5$ and lower bound $L = 4, 5, 6, 7, 8, 9$. Figures 9(a) to (c) shows that tree solutions with flexible clock source locations have better metrics (total wirelength, $PMax(1)$ and $PMax(2)$) than those with fixed clock source locations. Adding a forbidden area (reducing flexibility) worsens results in (c), but skew-wirelength, $PMax(1)$-wirelength and $PMax(2)$-wirelength remain superior to when the clock source location is fixed.

Study of runtime. In order to compare ILP runtime, we define our pointsets with $P = \{8, 10, 12, 14, 16\}$, where $|P|$ is the number of terminals of trees. For this experiment, for each $|P|$ we randomly generate 10 sets of terminals within a seven by seven grid. We also generate random grid edge costs in horizontal and vertical directions (i.e., between consecutive x- or y-coordinate values), from a uniform distribution over the interval [0.5, 2.0]. We use two different ILP solvers [23] [24] in this experiment.[6]

Table 2 shows the average ILP runtime. As shown in Table 2, ILP runtime increases when the number of terminals increases in the most of cases. Even though there are outliers due to infeasiblity of some of the problems, runtime increases when skew constraints get tighter (i.e., smaller skew bound).[7] The runtime for Solver A is smaller than for Solver B in most of the cases.

[6]In order to avoid benchmarking the tools, below we report tool names only as Solver A and Solver B.

[7]We cannot explain why the runtime for unbounded cases is smaller than for any bounded cases in [1]. In our studies, if we remove constraints for lower and upper bounds, the runtime slightly increases compared to the cases of $B = 0.8 \cdot M$ in Table 2.

Table 2: Average ILP runtime when $\alpha = 1, \beta = 0.5, L = M - B$. M denotes the maximum distance between source and sink.

ILP Solver	Skew bound	$\|P\| = 8$	$\|P\| = 10$	$\|P\| = 12$	$\|P\| = 14$	$\|P\| = 16$
Solver A	Unbounded	9.30	37.21	151.23	288.77	488.97
	$0.8 \cdot M$	10.07	12.17	32.69	56.57	100.31
	$0.6 \cdot M$	22.63	31.34	66.38	110.35	175.87
	$0.4 \cdot M$	71.29	96.41	153.19	1082.44	2454.79
	$0.2 \cdot M$	20.19	638.81	2912.25	14979.97	35566.68
Solver B	Unbounded	5.9	90.57	329.92	642.07	1454.97
	$0.8 \cdot M$	2.43	10.78	68.09	156.59	567.08
	$0.6 \cdot M$	7.87	91.53	619.80	757.56	1583.59
	$0.4 \cdot M$	96.93	1606.85	2091.18	11528.67	27122.51
	$0.2 \cdot M$	37.02	17800.59	24265.66	74113.28	107977.12

We also compare runtime for fixed and flexible clock source locations with non-uniform grids. Table 3 shows ILP runtime for fixed and flexible clock source locations with two different non-uniform grids, defined as follows. *Sparse* denotes non-uniform grids after removal of unusable edges as in Figure 3(c). *Dense* denotes non-uniform grids where unusable edges are retained, as in Figure 3(b). Due to the inclusion of a virtual source and edges, runs with flexible clock source location have longer runtimes than corresponding runs with fixed locations. We see that ILP solution runtime is reduced by removing unusable edges.

Table 3: ILP runtime for fixed and flexible clock source locations in Figure 4 and non-uniform grids. We set variables $\alpha = 1, \beta = 0.5, L = M - B$. M denotes the maximum distance between source and sink. *Sparse* denotes non-uniform grids after removing unusable edges. *Dense* denotes non-uniform grids with unusable edges retained.

Skew bound	Sparse		Dense	
	Fixed	Flexible	Fixed	Flexible
Unbounded	95.82	278.87	1964.88	2073.29
$0.8 \cdot M$	7.13	132.41	117.19	1581.06
$0.6 \cdot M$	34.89	432.06	500.11	3280.04
$0.4 \cdot M$	143.31	36781.35	2607.33	163050.29

5 CONCLUSION

In this work, we have proposed a new k-active dynamic power minimization problem that arises in clock distribution for memory-dominant SOC designs. We observe that in modern technologies the scale of SRAM blocks and SOC die, relative to global repeater distances, effectively "linearizes" the problem and makes it amenable to combinatorial optimization.

By extending the recent work of [1], we formulate an ILP for the k-active dynamic power minimization problem as well as a k-consecutive-active dynamic power minimization variant, The ILP minimizes the weighted sum of the total costs (wirelengths) and the maximum power across all sets of k active sinks. We also propose non-uniform grids to capture flexible locations of terminals in an output tree, and the handling of both flexible and forbidden clock

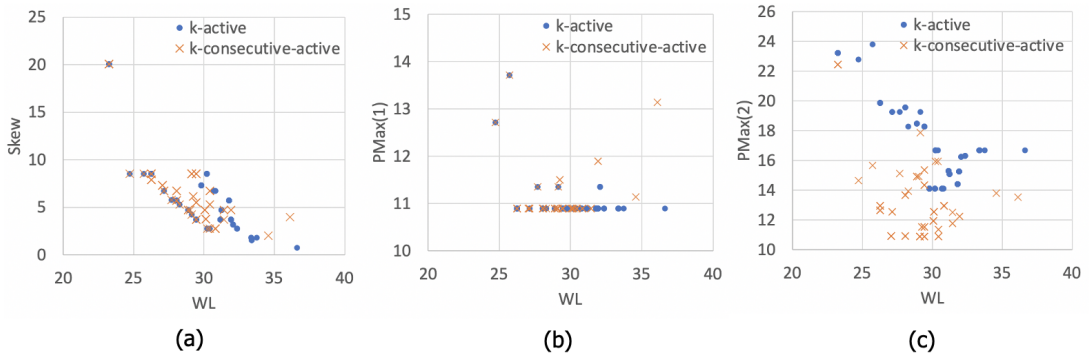


Figure 8: Scatter plots for (a) skew-wirelength, (b) $PMax(1)$-wirelength and (c) $PMax(2)$-wirelength for k-active sinks and k-consecutive-active sinks.

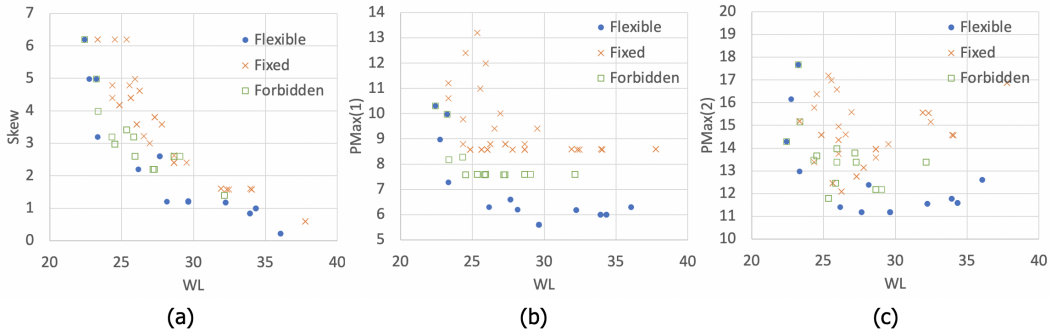


Figure 9: Scatter plots for (a) skew-wirelength, (b) $PMax(1)$-wirelength and (c) $PMax(2)$-wirelength for *fixed* and *flexible* clock source locations, as well as flexible clock source locations with a *forbidden* $2 \leq x \leq 4, 2 \leq y \leq 4$ area.

source locations to improve solution quality and solver runtime. Our experimental results give new insights into the tradeoff for maximum k-(consecutive)-active dynamic power and wirelengths of subtrees as well as skew and total wirelengths.

Our future directions include (1) implementations of our tree solutions as a guide for clock tree synthesis in commercial place-and-route methodologies; (2) heuristics to combine ILP and, e.g., problem decomposition strategies to address larger-scale instances of the k-(consecutive)-active dynamic power minimization problem; and (3) runtime improvements of ILP solution, particularly with flexible clock tree locations.

ACKNOWLEDGMENTS

Research at UCSD is supported by the U.S. National Science Foundation, U.S. DARPA, Samsung, Qualcomm, NXP Semiconductors, Mentor Graphics, and the C-DEN Center. We thank Kwangsoo Han, Christopher Moyes and Alex Zelikovsky for helpful discussions and for providing source code from their work [1].

REFERENCES

[1] K. Han, A. B. Kahng, C. Moyes and A. Zelikovsky, "A Study of Optimal Cost-Skew Tradeoff and Remaining Suboptimality in Interconnect Tree Constructions", *Proc. ACM/IEEE International Workshop on System-Level Interconnect Prediction*, 2018, pp. 2:1-2:8.
[2] Y. P. Aneja, "An Integer Linear Programming Approach to the Steiner Problem in Graphs", *Networks* 10(2) (1980), pp. 167-178.
[3] H. B. Bakoglu and J. D. Meindl, "Optimal Interconnect Circuits for VLSI", *IEEE Trans. Electron Devices* ED-32(5) (1985), pp. 903–909.
[4] K. Boese and A. B. Kahng, "Zero-Skew Clock Routing Trees With Minimum Wirelength", *Proc. IEEE ASIC Conf.*, 1992, pp. 1.1.1 - 1.1.5.
[5] T. H. Chao, Y. C. Hsu, J. M. Ho, K. D. Boese and A. B. Kahng, "Zero Skew Clock Routing With Minimum Wirelength", *IEEE Trans. on Circuits and Systems* 39(11) (1992), pp. 799-814.
[6] W.-C. Chao and W.-K. Mak, "Low-Power Gated and Buffered Clock Network Construction", *ACM TODAES* 13(1) (2008), pp. 1-20.
[7] G. Chen and E. F. Y. Young, "Dim Sum: Light Clock Tree by Small Diameter Sum", *Proc. DATE*, 2019, pp. 174-179.
[8] C. Chu and Y.-C. Wong, "FLUTE: Fast Lookup Table Based Rectilinear Steiner Minimal Tree Algorithm for VLSI Design", *IEEE Trans. on CAD* 27(1) (2008), pp. 70-83.
[9] J. Cong, A. B. Kahng, C. K. Koh and C.-W. A. Tsao, "Bounded-Skew Clock and Steiner Routing", *ACM TODAES* 3(3) (1998), pp. 341-388.
[10] W. M. Dai, T. Asano and E. S. Kuh, "Routing Region Definition and Ordering Scheme for Building-Block Layout", *IEEE Trans. on CAD* 4(3) (1985), pp. 189-197.
[11] M. Donno, E. Macii and L. Mazzoni, "Power-Aware Clock Tree Planning", *Proc. ISPD*, 2004, pp. 138-147.
[12] M. Edahiro, "A Clustering-Based Optimization Algorithm in Zero-Skew Routings", *Proc. DAC*, 1993, pp. 612-616.
[13] B. Giraud, O. Thomas, A. Amara, A. Vladimirescu and M. Belleville, *SRAM Circuit Design*, Dordrecht, Springer, 2009.
[14] M. Hanan, "On Steiner's Problem with Rectilinear Distance", *SIAM J. Appl. Math.* 14(2), pp. 255–265.
[15] J.-H. Huang, A. B. Kahng and C.-W. A. Tsao, "On the Bounded-Skew Clock and Steiner Routing Problems", *Proc. DAC*, 1995, pp. 508-513.
[16] A. B. Kahng and C.-W. A. Tsao, "Planar-DME: A Single-Layer Zero-Skew Clock Tree Router", *IEEE Trans. on CAD* 15(1) (1996), pp. 8-19.
[17] A. B. Kahng and C.-W. A. Tsao, "Practical Bounded-Skew Clock Routing", *J. VLSI Signal Processing* 16 (1997), pp. 199-215.
[18] J. Oh, I. Pyo and M. Pedram, "Constructing Lower and Upper Bounded Delay Routing Trees Using Linear Programming", *Proc. DAC*, 1996, pp. 401-404.
[19] A. Vittal and M. Marek-Sadowska, "Low-Power Buffered Clock Tree Design", *IEEE Trans. on CAD* 16(9) (1997), pp. 965-975.
[20] D. M. Warme, P. Winter and M. Zachariasen, "Exact Algorithms for Plane Steiner Tree Problems: A Computational Study", in *Advances in Steiner Trees* (D. Z. Du, J. M. Smith and J. H. Rubinstein, eds.), Kluwer Academic Publishers, 2000, pp. 81-116.
[21] A. Z. Zelikovsky and I. I. Mandoiu, "Practical Approximation Algorithms for Zero- and Bounded-skew Trees", *SIAM J. Disc. Math.* 15(1) (2002), pp. 97-111.
[22] Cadence Innovus User Guide. http://www.cadence.com
[23] Gurobi Optimizer Reference Manual. http://www.gurobi.com
[24] IBM ILOG CPLEX. http://www.ilog.com/products/cplex
[25] i.MX RT600 Crossover MCU with Arm Cortex-M33 and DSP Cores Data Sheet. https://www.nxp.com/docs/en/data-sheet/DS-RT600.pdf
[26] Scaling The Lowly SRAM. https://semiengineering.com/scaling-the-lowly-sram
[27] SweRV RISC-V Core 1.1 from Western Digital. https://github.com/westerndigitalcorporation/swerv_eh1
[28] The iPhone XS & XS Max Review: Unveiling the Silicon Secrets. https://www.anandtech.com/show/13392/the-iphone-xs-xs-max-review-unveiling-the-silicon-secrets/2
[29] Synopsys Design Compiler User Guide. http://www.synopsys.com
[30] The OpenROAD Project GitHub Repository. https://github.com/The-OpenROAD-Project