Synthesis of Clock Networks with a Mode Reconfigurable Topology and No Short Circuit Current

Necati Uysal, Juan Ariel Cabrera, Rickard Ewetz
University of Central Florida, Orlando, FL, 32816, USA
necati@knights.ucf.edu,rickard.ewetz@ucf.edu

ABSTRACT

Circuits deployed in the Internet of Things operate in low and high performance modes to cater to variable frequency and power requirements. Consequently, the clock networks for such circuits must be synthesized meeting drastically different timing constraints under variations in the different modes. The overall power consumption and robustness to variations of a clock network is determined by the topology. However, state-of-the-art clock networks use the same topology in every mode, despite that the timing constraints in the low and high performance modes are very different. In this paper, we propose a clock network with a mode reconfigurable topology (MRT) for circuits with positive-edge triggered sequential elements. In high performance modes, the required robustness to variations is provided by reconfiguring the MRT structure into a near-tree. In low performance modes, the MRT structure is reconfigured into a tree to save power. Non-tree (or near-tree) structures provide robustness to variations by appropriately constructing multiple alternative paths from the clock source to the clock sinks, which neutralizes the negative impact of variations. In MRT structures, OR-gates are used to join multiple alternative paths into a single path. Consequently, the MRT structures consume no short circuit power because there is only one gate driving each net. Moreover, it is straightforward to reconfigure MRT structures into a tree by gating the clock signal in part of the structure. Compared with state-of-the-art near-tree structures, MRT structures have 8% lower power consumption and similar robustness to variations in high performance modes. In low performance modes, the power consumption is 16% smaller when reconfiguration is used.

ACM Reference Format:
Necati Uysal, Juan Ariel Cabrera, Rickard Ewetz. 2020. Synthesis of Clock Networks with a Mode Reconfigurable Topology and No Short Circuit Current. In *Proceedings of the 2020 International Symposium on Physical Design (ISPD '20), March 29-April 1, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3372780.3375559

1 INTRODUCTION

Modern VLSI circuits are required to operate in low and high performance modes to deliver configurable frequency and power consumption. Every synchronous VLSI circuit is required to be synchronized by a clock network. The clock network delivers a clock signal from a clock source to the sequential elements or clock sinks. Clock skew is the difference in the arrival time between a pair of clock sinks. In every operational mode, there is a timing constraint in the form of a skew constraint between each pair of clock sink that are only separated by combinational logic in the data and control paths. In each mode, the skew constraints must be satisfied under variations to guarantee the functional correctness.

Constructing clock networks with small nominal skews is not too difficult [1, 3, 23, 24]. However, it is very challenging to meet tight skew constraints while the clock network is under the influence of process, voltage and temperature (PVT) variations [2, 21, 22]. In particular, for circuits required to operate in low performance and high performance modes. Many recent studies are focused on constructing clock trees that utilize useful skew to satisfy timing constraints under the influence of variations in multiple modes [9]. However, for circuits with strict requirements on the clock frequency (in the high performance modes), it may be impossible to satisfy the timing constraints using a clock tree.

The overall power consumption and robustness to variations of a clock network is determined by the topology, which can be in the form of a tree, near-tree, or non-tree. Clock trees consume the least power but are vulnerable to variations. Clock networks with a non-tree or near-tree topology have multiple paths from the clock source to the clock sinks. When inserted appropriately, the alternative paths neutralize the negative impact of variations. However, clock networks with a non-tree topology (as clock meshes) may consume 3X-5X more power than a clock tree [20, 26, 28]. In the past decade, clock networks with a near-tree topologies have been investigated, which promise significant improvements in robustness while the power consumption is similar to that of a clock tree [8, 14, 16, 17]. In [14, 16, 25], near-tree structures were constructed by inserting cross-links. In [12], multiple tree structures were fused to create a multilevel fusion tree. In [8], the tree fusion technique was extended to create a locally-merged structure by fusing multiple subtrees at internal nodes of a clock tree. A limitation of near-tree structures is that there are multiple gates driving the same net of wires, which may result in short circuit current. In particular, when voltage and frequency scaling is applied to save power in the low performance modes. Moreover, a drawback of all state-of-the-art clock networks is that the same topology is used in every operational mode, even though the timing constraints are significantly looser in the low performance modes, i.e., a non-tree or near-tree topology is not required.

In this paper, a clock network with mode reconfigurable topology (MRT) is proposed for circuits that operate in a low and high

This research was in part supported by NSF award CCF-1755825.

ISPD '20, March 29-April 1, 2020, Taipei, Taiwan
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7091-2/20/03…$15.00
https://doi.org/10.1145/3372780.3375559

performance mode and only have positive-edge triggered sequential elements. In the high performance mode, the MRT structure is reconfigured into near-tree topology to provide robustness to variations. In the low performance mode, the MRT structure is reconfigured into tree topology to save power. Moreover, OR-gates are used to join the multiple alternative paths into a single path, in order to provide robustness to variations. Consequently, there is no short circuit current because there is only a single gate driving each net of wires. In fact, this is the first near-tree (or non-tree) topology without any short circuit current, even if voltage scaling is applied in the low performance mode. To significantly improve the robustness, the OR-gates are required to be implemented using parallel INV-gates connected in series with an NAND-gate. It is important to observe that the OR-gates mainly improve the robustness of the delivery of the rising-edge of the clock signal (see details in Section 4.1). Nevertheless, the overall robustness is improved because positive-edge triggered sequential elements have extremely loose timing constraints on the falling-edge of the clock signal. Moreover, it is straightforward to reconfigure MRT structures into a tree topology by gating the clock signal in part of the structure. Lastly, MRT structures can be efficiently constructed by leveraging tree construction algorithms and clock tree optimization techniques developed in previous studies. The experimental results show that compared with state-of-the-art near-tree structures, the MRT structures have 8% and 16% smaller power consumption in the high and low performance mode, respectively. The reductions in power consumption are obtained without degrading the robustness to variations.

The remainder of the paper is organized, as follows: preliminaries are presented in Section 2. Previous work is reviewed in Section 3. The proposed MRT structure is given in Section 4 and the methodology is given in Section 5. The experimental results are presented in Section 6. The paper is concluded in Section 7.

2 PROBLEM FORMULATION

Timing constraints: Synchronous circuits are required to meet setup and hold time constraints between any pair of clock sinks that are only separated by combinational logic in the data and control paths. Moreover, positive-edge triggered sequential elements only have tight timing constraints on the delivery of the positive edge of the clock signal [27]. For circuits with high frequency requirements, a clock tree would require a too large portion (10%-30%) of the clock period T for guardbands to variations, which dramatically increases the size of the data and control paths. Therefore, a clock network with a near-tree or a non-tree topology is required for such circuits [1]. This paper is targeting circuits with a low and high-performance mode, which aim to have a clock frequency (in the high-performance mode) that requires the use of a clock network with a near-tree or a non-tree topology. For circuits with high frequency requirements, the setup time is the critical constraint, as studies have shown that hold time constraints can be satisfied by optimizing the combinational logic [10]. Moreover, the maximum delay through the combinational logic in the control and data paths are optimized to meet a constraint on the maximum propagation delay. Consequently, the clock network is required to deliver the clock signal to the clock sinks meeting a uniform skew constraint B in the high performance mode under variations [11, 21, 22].

Voltage and frequency scaling: Next, voltage and frequency scaling is applied to minimize the power consumption while meeting the timing constraints in the low performance mode. The dynamic power consumption P is calculated using, $P = C_L \cdot V_{DD}^2 \cdot f$, where C_L is the switching capacitance, V_{DD} is the supply voltage, and f is the clock frequency. When the clock frequency is reduced in the low performance modes, timing margins are introduced in the limiting setup time constraints, which allows the supply voltage to be scaled down until the maximum propagation delay through the combinational logic becomes too long. In particular, the propagation delay through combinational gates is proportional to $V_{DD}/(V_{DD} - V_t)^2$, where V_t is the threshold voltage; Note that supply voltage scaling also impacts the timing of the clock network. Furthermore, several recent studies attempt to squeeze out additional power savings by performing clock tree optimization, where multiple modes are simultaneously optimized to improve timing and power [7, 18].

Problem formulation: This paper considers the problem of constructing a clock network connecting a clock source to a set of clock sinks (with specified xy-locations) using wires, buffers, clock gates, and OR-gates from a given technology library. The clock network is required to operate in a low and a high performance mode and meet skew constraints in both modes. Transition time constraints are also required to be satisfied.

- The primary objective is for the positive-edge of the clock signal to meet the smallest skew bound B under variations in the high performance mode while minimizing the total capacitance (or capacitive cost). This problem is approached by constructing a clock network with a near-tree topology and zero nominal skew.
- The secondary objective is to minimize the power consumption in the low performance mode using voltage scaling and reconfiguration proposed in this paper. Note that it is easy to satisfy timing constraints in the low performance mode because a slower clock frequency is used.

3 PREVIOUS STUDIES

The topology of a clock network can be in the form of a tree [1, 3, 4, 23, 24], a near-tree [8, 14, 16], or a non-tree [20, 26, 28], as illustrated in Figure 1. Near-tree structures can be in the form of clock trees with cross links, clock spines [19], multilevel fusion trees [12], and locally-merged structures [8]. In Table 1, we compare the properties of clock networks with different topologies.

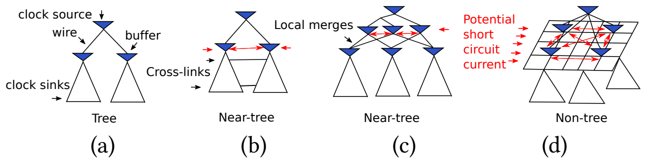


Figure 1: Clock networks with different topologies.

The trade-off between robustness and power consumption for tree and non-tree (or near-tree) topologies is well known [8, 12, 14, 16, 20, 28]. Clearly, near-tree structures seem advantageous to

[1]Note that non-tree and near-tree topologies are only advantageous if tree structures require excessive guardbands.

non-tree structures as clock meshes. Nevertheless, near-tree structures are not widely (or at all) adopted in the industry. This may be due to that i) short circuit current may be introduced when voltage scaling is applied and ii) near-tree structures are not compatible with state-of-the-art EDA tools. These challenges stem from that near-tree structures have multiple gates driving the same net of wires, as shown in Figure 1. One of the reasons why clock meshes have been successfully adopted in the industry is that symmetric clock trees are traditionally used to drive the mesh grid. In addition, symmetric structures are more gracefully degraded by voltage scaling than non-symmetric structures. Moreover, there has already been substantial investments in specialized EDA tools for the synthesis and analysis of such structures. An intuitive idea to reduce the power consumption in low performance modes is to turn-off part of the non-tree (or near-tree) structure using clock gates while still guaranteeing that the clock signal is delivered using at least one path from the clock source to each clock sink. However, this is impossible in state-of-the-art non-tree (or near-tree) structures because it would create static short circuit current.

Table 1: Comparison between clock networks with tree, near-tree, and non-tree topologies. '*' voltage scaling can be applied due to symmetric structure. '' specialized EDA tools have been developed.**

Topology	Work	Robustness to variations	Power	Compatible with		
				Voltage scaling	EDA tools	Reconfig-uration
Tree	[1, 3, 24]	low	**small**	**Yes**	**Yes**	No
Near-tree	[8, 12, 16]	**high**	medium	No	No	No
Non-tree	[20, 26, 28]	**high**	very large	**Yes***	**Yes****	No
MRT (near-tree)	This work	**high**	medium	**Yes**	**Yes**	**Yes**

In this paper, we propose a clock network with a mode reconfigurable topology (MRT) with properties as labeled in Table 1. The MRT structure has similar performance as near-tree structures in earlier studies but multiple paths are joined using OR-gates instead of wires. Consequently, there is only one gate driving each net of wires. Therefore, it is easy to understand that the structure is compatible with voltage scaling, state-of-the-art EDA tools, and topology reconfiguration.

4 PROPOSED MRT STRUCTURE

In this section, the proposed MRT structure is introduced using Figure 2. In Section 4.1, it is explained how the MRT structures in the form of a near-tree improves the robustness to variations. In Section 4.2, it is explained how power is saved when the MRT structure is reconfigured to a tree.

The MRT structure in the figure operates in two modes and is constructed using buffers, wires, OR-gates and a clock gate. The input to the MRT structure is a mode control signal and the clock signal from the clock source. The mode control signal specifies the mode that the circuit is operating. In the high performance mode 1, the clock gate is transparent and the entire MRT structure is used to deliver the clock signal to the clock sinks, i.e., the topology is in the form of a near-tree. In the low performance mode 2, the clock signal is gated and the part of the MRT structure that is used to deliver the clock signal is in the form of a tree topology. In Figure 2,

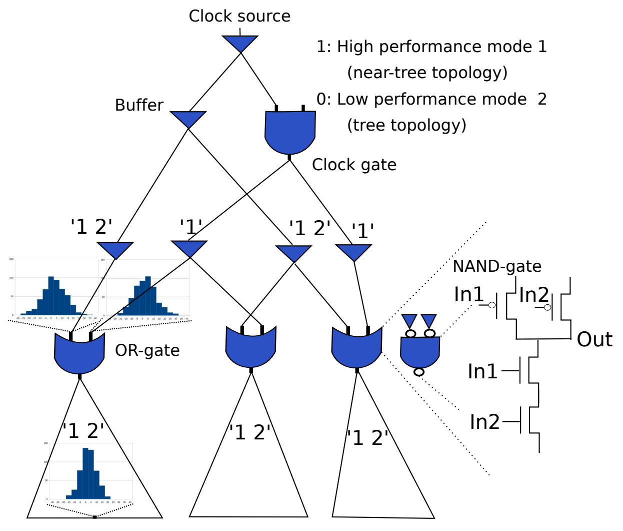


Figure 2: Proposed MRT structure.

a few gates and subtrees are labeled '1' and '1 2' to indicate the modes where the respective components are used to deliver the clock signal.

4.1 Robustness of near-tree to variations

The OR-gates improve the robustness of the rising-edge of the clock signal. Recall that the negative edge of the clock signal is not involved in the timing constraints for positive-edge triggered sequential elements. The robustness of the positive edge is improved because the OR-gates are implemented using two parallel INV-gates connected in series with an NAND-gate, as illustrated in the bottom right part of Figure 2. The two input NAND-gate has two parallel PMOS transistors that may charge the net attached to the output of the NAND-gate. Under nominal conditions, the MRT structure is designed such that the clock signal arrives at the same time to the separate input pins of each OR-gate. Hence, both PMOS transistors in the NAND-gate will equally contribute to charging the downstream net. Under variations, it is expected that the arrival time of the clock signal to the separate input pins will be different. The end-result is that the PMOS transistor that is turned-on first (last) will charge the net more (less) compared within the nominal case. A simple but effective model to estimate the robustness improvement of an OR-gate is to approximate the effective arrival time of the clock signal to the OR-gate with the mean of the two separate arrival times, i.e., an OR-gate will improve the robustness to variations by averaging out the negative effect of the variations. Of course, the model only holds when the last PMOS transistor to be turned-on is activated before the net downstream of the OR-gate is completely charged by the other PMOS transistor. Therefore, the proposed OR-gate is preferred to the traditional OR-gate consisting of an NOR-gate connected in series with an INV-gate, i.e., the capacitance of the internal node in the traditional OR-gate is smaller than the capacitance of the downstream subtree, which means that the internal node will be charged faster and less variations can be neutralized.

To confirm that OR-gates improve the robustness of MRT structures, the structure in the figure is simulated using NGSPICE [15] simulations while applying various forms of correlated variations using the Monte Carlo framework in [8]. The arrival time distribution at three different points in the network are shown in Figure 2. It is concluded that the OR-gates improve the robustness to variations because the variance of the arrival time distribution below the OR-gate is close to half of the variance of the distribution above the OR-gate.

4.2 Reducing power in low performance modes

The MRT structure in the figure can be reconfigured from near-tree to a tree based on the active mode. The reconfiguration is enabled with no short circuit current because each net of wires is connected to only a single driving gate. In fact, this is the first near-tree (or non-tree) structure that can be configured based on the active mode.

Clearly, the reconfiguration introduces a trade-off between robustness and power consumption. Nevertheless, high robustness is not required in the low performance mode because the timing constraints are looser. However, the degraded robustness may reduce the amount of power savings that can be achieved using voltage scaling in the low performance mode. The dynamic power consumption of a VLSI circuit is calculated, as follows:

$$P = C_{comb} \cdot V_{DD}^2 \cdot f \cdot \alpha_{comb} + C_{clk} \cdot V_{DD}^2 \cdot f \cdot \alpha_{clk}, \quad (1)$$

where f is the clock frequency; C_{comb} and C_{clk} are the total capacitance of the combinational logic and the clock network, respectively. α_{comb} and α_{clk} are the activity factors of the combinational logic and clock network, respectively. V_{DD} is the supply voltage.

Normally, voltage and frequency scaling is performed by updating the clock frequency and then reducing the supply voltage until the timing constraints are not satisfied. For MRT structures, we propose to combine voltage and frequency scaling with reconfiguration of the topology. First, the MRT structure is reconfigured into a tree topology. Second, traditional voltage and frequency scaling is performed. The reconfiguration of the topology introduces a trade-off between voltage scaling (V_{DD}) and switching capacitance in the clock network (C_{clk}). The explanation is that the reconfiguration introduces a small nominal skew between some pairs of clock sinks. The skew is introduced because MRT structures may use OR-gates with different number of input pins. Consequently, there are smaller timing margins available for voltage scaling, which results in that a slightly higher supply voltage is required to be used. However, it is easy to understand that the supply voltage is only marginally higher because the propagation delay of any gate is proportional to $V_{DD}/(V_{DD} - V_t)^2$. Therefore, when V_{DD} is close to the threshold voltage V_t, large timing margins are required to marginally reduce the supply voltage further. It is expected to be better to use a portion of the timing margins obtained from the frequency scaling to reconfigure the clock network to reduce (C_{clk}), despite that supply voltage scaling reduces the power consumption of both the combinational logic and the clock network in Eq (1). Moreover, the switching activity of the clock network α_{clk} is higher than that of the combinational logic α_{comb}.

5 METHODOLOGY

The flow for constructing the proposed MRT structure is shown in Figure 3. The MRT structures are constructed stage-by stage using a bottom-up process based on the classic zero-skew DME paradigm [2, 4, 5]. A stage consists of a device (a buffer or an OR-gate) driving a subtree of wires connected to the stage-sinks. The stage-sinks of the first stage are the clock sinks and the stage-sinks of any other stage is the input pins of the devices driving the previous stage. The bottom-up construction is followed by a top-down embedding of internal nodes using the DME algorithm, a reconfiguration based clock gating step, and a clock network optimization step.

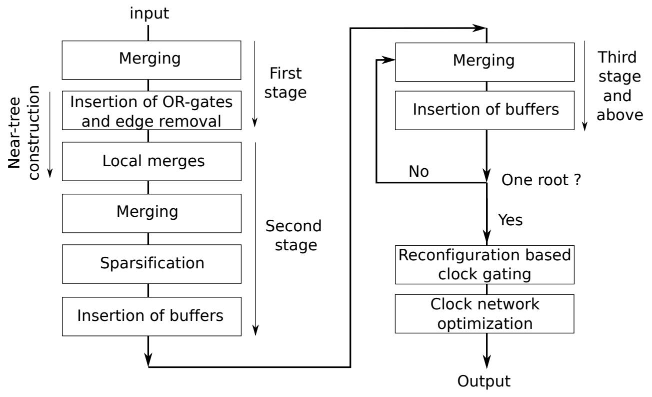


Figure 3: Flow for constructing MRT structures.

In MRT structures, OR-gates are only inserted to drive the subtrees of the first stage, as illustrated in Figure 2 and shown in Figure 3. In [8], it was observed that inserting redundancy (or multiple paths) at the first stage balances robustness to variations and total capacitance. However, we plan to explore inserting OR-gates at other locations in our future work.

The flow shows that the first stage is constructed by merging subtrees (see Zero skew tree construction in Section 5.1) and inserting OR-gates to drive the subtrees (see Section 5.2.1). The second stage is constructed using zero-skew merges or local merges [8] (see Near-tree construction in Section 5.2) followed by merging, sparsification (see Section 5.2.3) and buffer insertion (see Section 5.1). Next, the third and any subsequent stages are constructed by iteratively performing merging and inserting buffers, as illustrated in Figure 3. The process is repeated until there is only a single root remaining and a near-tree MRT is obtained. In the reconfiguration based clock gating step (see Section 5.3), a subset of the buffers are converted into clock gates to enable the MRT structure to be reconfigured from a near-tree into a tree (or close to a tree). In the clock network optimization step (see Section 5.4), the MRT structure is tuned to i) minimize the nominal skew close to zero and to ii) minimize the skew between input pins of the same OR-gate, which improves robustness.

5.1 Zero skew tree construction [1, 3, 23, 24]

In this section, it is explained how an ZST with buffers can be constructed based on iteratively performing a *merging* phase and an *insertion of buffers* phase, which is shown in Figure 3. The method is based on techniques proposed in [1, 4, 23, 24].

Merging: The merging step is used to construct a set of zero skew subtrees (or ZSTs) from a set of stage-sinks using the DME algorithm. The construction is based on iteratively merging the pair of subtrees that require the least amount of wire length to be joined into a larger subtree using a *zero-skew merge*. A zero-skew merge joins a pair of subtrees with the smallest possible wirelength while ensuring that the delay to all the sinks in the formed subtree are equal under the Elmore delay model. After a pair of subtrees have been joined, the transition time is evaluated. If the transition time constraint is violated, the subtree pair is unmerged and both subtrees are locked from further merging. Otherwise, the newly formed subtree is allowed to be joined with additional subtrees. The process is repeated until all subtrees are locked from further merging [1, 3, 4, 23, 24].

Insertion of buffers: The input to the buffer insertion is a set of locked subtrees from the merging step. For each subtree, the minimum sized buffer that can drive the locked subtree is selected to be inserted at the root. Next, a piece of stem wire is inserted between the subtree and the buffer [4].

5.2 Near-tree construction

The near-tree construction step consists of an insertion of OR-gates, a local merges phase, an edge removal phase and a sparsification phase.

5.2.1 Insertion of OR-gates and local merges. In this section, we explain the insertion of the OR-gates to drive the subtrees of the first stage, the selection of the number of input pins for the OR-gates, and how the input pins of the OR-gates are joined using zero-skew merges (or local merges [8]), which is illustrated in Figure 4. The objective is to join subtrees that contain clock sinks only separated by combinational logic in the data and control paths early in the topology to intuitively improve the robustness to variations.

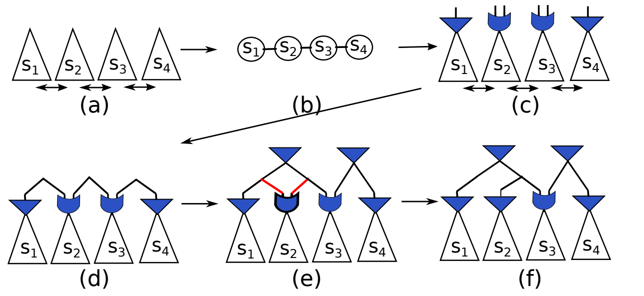


Figure 4: (a) Subtrees. (b) SRG. (c) Driver devices are inserted. (d) Multiple subtrees are merged. (e) Second stage subtrees are constructed. (f) Subtrees after sparsification

The subtrees constructed in the first stage are shown in Figure 4(a). The arrows in the figure illustrate that the corresponding subtree pair contains a pair of clock sinks only separated by combinational logic in the data and control paths. Next, a Sequential Relation Graph (SRG) [8] is formed to capture the subtrees and the arrows in Figure 4(a). The SRG of the subtrees in Figure 4(a) is shown in Figure 4(b). The number of input pins of OR-gates are limited up to four in order to save the cost. Therefore, if any vertex in the SRG has more than four edges, edge removal is performed (as explained in the Section 5.2.2 section) until no vertex has more than four edges. Next, an OR-gate is inserted to drive each subtree and the number of inputs pins of the OR-gate is selected to be equal to the number of edges connected to the corresponding vertex in the SRG. Note that an OR-gate with a single input pin is equivalent to a buffer. Figure 4(c) shows selected OR-gates and buffers for the subtrees in Figure 4(a) and with the SRG in Figure 4(b). Clearly, the motivation for the edge removal is to limit the size of the NMOS transistors within the NAND-gate (inside the OR-gates). Lastly, local merges (zero-skew merges) are performed to join the input pins of OR-gates that are correspondingly connected with edges in the SRG. Figure 4(d) shows the subtrees with OR-gates (or buffers) inserted in Figure 4(c) after local merges have been performed. After the insertion of the OR-gates and the local merges, normal merging is performed (as described in Section 5.1) to construct the remainder of the subtrees. Next, a sparsification is applied to remove excessive redundant path in the second stage of the subtrees (see Section 5.2.3).

5.2.2 Edge removal. In this section, it is explained how edges are removed from an SRG until no vertex is connected to more than four edges and all edges can be realized using a local merge, which is illustrated in Figure 5.

The first step is to remove all edges that cannot be realized using a local merge. This is performed by temporarily inserting a buffer to drive each of the subtrees captured in the SRG. Next, for every edge in the SRG, the corresponding pair of subtrees (with buffers attached at the root) are attempted to be merged using a zero-skew merge. If a pair cannot be merged due to the transition time constraint, the corresponding edge in the SRG is removed. Next, the temporarily inserted buffers are removed.

The second step is to iteratively remove edges from the SRG until no vertex is connected to more than four other vertices. The method of removing edges described below aims to minimize the number of edges that are removed in order to maximize the robustness. Let the weight of a vertex be equal to the number of edges connected to the vertex. Moreover, let the weight of an edge be equal to the sum of the two vertex weights it is connected to. In our implementation, we first find the vertex with the maximum weight, which is marked with a thick black circle in Figure 5(a). Next, the edge with the largest weight that is connected to the selected vertex is chosen to be removed. In Figure 5(a), the first edge to be removed is illustrated with a thick black edge. Next, the process is repeated until the graph in Figure 5(b) is obtained, i.e., no vertex has more than four edges.

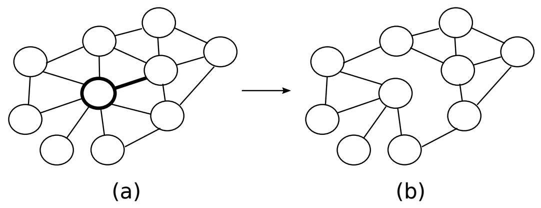


Figure 5: (a) Initial SRG. (b) After edge removal.

5.2.3 Sparsification. The sparsification step is performed to ensure that each subtree of the second stage only contains one pin from any OR-gates, i.e., there is a unique path from the driver of each subtree to each OR-gate, which is shown in (e) and (f) of the Figure 4. The sparsification process is applied to each locked subtree after the construction of the second stage of the network. For each subtree, sinks (or input pins) of the subtree are marked. If two or more pins of the same OR-gate are marked, the input pins are merged into a single input pin. Next, the subtree is removed and reconstructed from the new (smaller) set of pins. If the newly constructed subtree satisfies the transition time constraint, the subtree is kept. Otherwise, the subtree is reverted back into the non-sparsified form. This process is iteratively applied to each second stage subtree. The remainder of the network is constructed by following the zero skew tree construction methodology as explained in Section 5.1.

5.3 Reconfiguration based insertion of clock gates

In this section, it is described how buffers in a constructed MRT structure are selected to be converted into clock gates, to enable the MRT structure to be reconfigured from a near-tree into a tree topology (or close to a tree topology), i.e., it is not guaranteed that there only exists a single path from the clock source to every clock sink after the clock gating.

In contrast with traditional clock gating that aims to completely turn-off a portion of the clock network, the reconfiguration based clock gating must ensure that there still exists at least one path from the clock source to each clock sink, which is called a *connectivity constraint.* The conversion of buffers into clock gates is a trade-off between the amount of switching capacitance in the high performance mode, the amount of switching capacitance in the low performance mode, and the amount of overhead circuitry required to provide the mode control signal to the clock gates. The switching capacitance in the high performance mode is slightly increased when clock gates are inserted because clock gates have larger internal capacitance than buffers. Naturally, clock gates reduce the capacitance in the low performance mode, as part of MRT structure is turned-off. In this paper, the buffer to clock gate conversion is formulated as an optimization problem, called an MRT gating (MRTG) problem, as follows:

$$\min \alpha C_H + \beta C_L + \gamma N_g \tag{2}$$

where C_L and C_H are the switching capacitance in the low and high performance mode, respectively. N_g is the number of inserted clock gates. α, β, γ are parameters used to regulate the trade-off between the different terms in the objective. The ratio between α and β is specified based on the portion of the time the circuit operates in the high and low performance mode. γ is used to avoid inserting clock gates that only gate a small part of the MRT structure.

In the next section, we explain how the MRTG gating problem is solved while the connectivity constraint is satisfied.

5.3.1 Solving the MRTG problem. In this section, we provide a scalable and practical solution to solving the MRTG problem. However, we note that the MRTG problem can be solved optimally using an mixed integer programming (MIP) formulation. In this paper, the MRTG problem is solved by considering all buffers as candidates for conversion into clock gates. First, the algorithm removes all candidates that violate the connectivity constraint, which can be checked using a breadth first search (BFS) per candidate.

Let $\triangle cost(b)$ be the change in the objective function in Eq (2) of converting a candidate buffer b into a clock gate. Next, the candidate buffer with the smallest $\triangle cost(b)$ is converted into a clock gate. This process is iteratively repeated until no candidates with a negative $\triangle cost(b)$ remain. Note that the connectivity constraint must be reevaluated (for each candidate) after every insertion of a clock gate. Next, a clock network optimization step in Section 5.4 is performed to remove the timing violations.

5.4 Clock network optimization

State-of-the-art clock tree optimization techniques are based on specifying and realizing delay adjustments remove the timing violations [13]. The specified delay adjustments are realized by inserting delay buffers and detour wires. Clock network optimization for MRT structures is performed using a two-phase approach. In the first phase, the input pins of the OR-gates are viewed as the clock sinks and the skew between pins of the same OR-gate is minimized by specifying and realizing delay adjustments, which improves the robustness of MRT structures to variations. In the second phase, the nominal skew is minimized by specifying and realizing delay adjustment below the OR-gates.Note that it is easy to adapt clock tree optimization algorithms to the MRT structure because there is only a single gate driving each net.

6 EXPERIMENTAL RESULTS

The proposed methodology is implemented in C++ and experimental evaluation is performed on a quad core 3.4 GHz Linux machine with 32GB of memory. The properties of the buffers, the OR-gates and the wires are obtained from a 45 nm technology library [21]. The benchmarks in [6] are used to synthesize our clock networks. The details about benchmark circuits (number of clock sinks, number of skew constraints, and clock period) are shown in Table 2. The transition time constraint for all of the structures is set to 100 ps. The clock periods correspond to a clock frequency of $5GHz$ and $1GHz$ in the high and low performance mode, respectively.

To demonstrate the effectiveness of the proposed framework, clock tree structures (labeled 'Tree'), near-tree structures (labeled 'Near-tree') and the MRT structures (labeled 'MRT') are constructed and evaluated. The structures are constructed using different modifications of the flow in Figure 3. The Tree structures are constructed by disabling the near-tree construction step and the reconfiguration step. These Tree structures are similar to the clock trees in [24]. The Near-tree structures are constructed to mimic the locally-merged structures in [8], which are illustrated in Figure 1(c). The locally-merged structures were selected because they were reported to outperform clock trees with cross-links and clock meshes in [8]. The Near-tree structures are constructed by replacing the OR-gate insertion with a sink-splitting step [12] and disabling the reconfiguration step. The MRT structures are constructed using the full flow in Figure 3.

Table 2: Benchmarks in [6].

Circuit	Sinks	Skew constraints	Clock period (ps)	
(name)	(num)	(num)	T_H	T_L
s1423	74	78	200	1000
s5378	179	175	200	1000
s15850	597	318	200	1000
msp	683	44990	200	1000
fpu	715	16263	200	1000
usbf	1765	33438	200	1000
pci bridge32	3582	141074	200	1000
des peft	8808	17152	200	1000

Table 3: Evaluation of the structures in the high performance mode.

Benchmark	Structure	Power	Timing yield		Run-time
			B_{100}	B_{95}	
		(mW)	(ps)	(ps)	(min)
msp	Tree [24]	14.91	16.93	12.06	8
	Near-tree [8]	27.28	5.76	4.81	9
	MRT	21.20	7.33	5.90	9
fpu	Tree [24]	17.07	15.72	10.46	4
	Near-tree [8]	33.06	10.86	7.43	4
	MRT	25.29	10.11	6.90	4
s1423	Tree [24]	37.10	30.19	22.43	3
	Near-tree [8]	42.56	16.24	12.42	3
	MRT	44.56	13.86	9.92	4
s5378	Tree [24]	51.85	24.40	16.37	2
	Near-tree [8]	62.30	20.83	12.83	4
	MRT	71.70	19.71	12.29	5
s15850	Tree [24]	120.96	35.25	22.54	7
	Near-tree [8]	199.63	29.74	20.66	21
	MRT	220.89	26.30	19.08	23
usbf	Tree [24]	37.88	22.63	17.10	6
	Near-tree [8]	57.55	8.66	7.31	10
	MRT	50.15	10.52	8.09	9
pci bridge32	Tree [24]	74.87	26.10	18.80	49
	Near-tree [8]	104.10	9.98	8.15	50
	MRT	107.22	19.62	14.15	80
des	Tree [24]	153.96	34.03	19.94	79
	Near-tree [8]	254.28	22.59	15.96	216
	MRT	193.12	14.40	11.26	105
Norm.	Tree [24]	1.00	1.00	1.00	1.00
	Near-tree [8]	1.54	**0.58**	**0.63**	1.61
	MRT	**1.42**	**0.59**	**0.62**	1.68

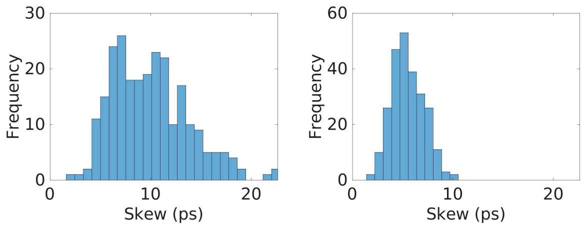
The structures are evaluated in terms of power consumption and timing yield. The power consumption is evaluated using circuit simulations. The timing yield is evaluated by performing Monte Carlo simulations using NGSPICE. Let B_{100} and B_{95} be the skew bound that results in 95% and 100% yield, respectively. The timing yield is computed by simulating each structure with 250 Monte Carlo simulations while applying wire width variations (±5%), voltage variations (±7.5%), channel length variations (±5%) and temperature variations (±15%) around the nominal values. The variations are generated using a five-level quad-tree to introduce spatial correlations. In each simulation, skew and transient time constraints are evaluated. If the chip meets timing constraints, the chip is classified as good. Otherwise, the chip is classified as defective. The yield is calculated by the number of good chips divided by the number of tested chips. B_{100} and B_{95} are obtained from the timing in the Monte Carlo simulations. For all structures, no yield loss is obtained from the transition time constraints.

6.1 Evaluation of high performance mode

In Table 3, we evaluate the performance of the three constructed structures in the high performance mode. The power consumption and the run-time of the synthesis process are reported in the columns labeled as 'Power' and 'Run-time', respectively. The robustness to variations is measured using B_{95} and B_{100}.

Compared with the Tree structures, the Near-tree structures have 42% and 37% smaller B_{100} and B_{95}, respectively. The improvements in robustness to variations stems from that the Near-tree structures have alternative paths from clock source to the clock sinks, which reduces the impact of variations. The improvements in robustness come at the expense of 54% overhead in power consumption, which is expected because there are redundant paths in the topology. The run-time of the Near-tree structure is 61% longer than the Tree structures because the Near-tree structures are larger.

Compared with the Tree structures, the MRT structures have 41% and 38% smaller B_{100} and B_{95}, respectively. The improved robustness stems from that the MRT structures have a near-tree topology and that the OR-gates are assembled using INV-gates and NAND-gates, which average out the negative impact of variations. To illustrate the robustness improvements, we show a histogram of the skew introduced by variations for both the Tree structures and the MRT structures on *usbf* in Figure 6. It can be observed that the MRT structure has a tighter skew distribution than the Tree structure.



(a) Tree (b) MRT

Figure 6: Histogram of skews from Monte Carlo simulations of (a) Tree and (b) MRT structures on *usbf*.

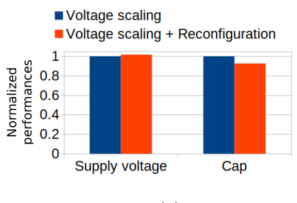
Compared with the Near-tree structures, the MRT structures have similar robustness to variations. The B_{100} and B_{95} of the MRT structures are 1% larger and 1% smaller, respectively. The robustness of the two structures is similar due to that they have a similar amount of redundancy. However, the MRT structures have 12% smaller power consumption than the Near-tree structures. The lower power consumption stems from: (i) The switching capacitance is 8% smaller. The main explanation for the lower switching capacitance is that the locally-merged structures require a slightly stricter transition time constraint to be used when constructing the second stage; (ii) The MRT structures have no short-circuit current because each net only has a single driver. In the Near-tree structures, there are multiple devices that drive the same net. The run-time of the two structures are similar. In addition, the MRT structures are compatible with standard EDA tools and topology reconfiguration, which is evaluated in Section 6.2. The only drawback of the MRT structures is that a mode control signal is required to be delivered to the clock gates. However, the wire length used to deliver the mode control signal is only 4% of the total wirelength of the MRT structures. Moreover, this wire length does not contribute to the dynamic power consumption.

Based on the discussion above, a clock tree provides the best synchronization solution if it can satisfy the timing constraints without excessive guardbands. In this paper, we focus on circuits where a higher robustness to variations is required in the high performance mode. For such circuits, we conclude that the proposed MRT structures are advantageous to the Near-tree structures in [8].

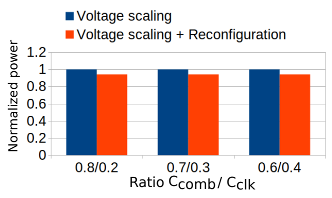
6.2 Evaluation of low performance mode

In this section, we evaluated the performance of the structures in the low performance mode. First, we compare voltage scaling with applying voltage scaling combined with reconfiguration. Next, we compare the performance of the MRT structures with the Near-tree structures. Recall, the clock frequency is scaled down in the low performance mode and it is easy to meet timing constraints. Instead, the main objective is to minimize the power consumption.

In Figure 7, we evaluated the performance of the MRT structures when voltage scaling and voltage scaling combined with reconfiguration is applied. The two techniques are compared in terms of the normalized supply voltage and the normalized switching capacitance in Figure 7(a). It can be observed that the reconfiguration reduces the switching capacitance by 8%, which is expected because parts of the topology are turned-off. However, the reconfiguration reduces the available timing margins with 4%, which results in that a 2% higher supply voltage is required to be used. As mentioned earlier, only a minor increase in the supply voltage is expected because the propagation delays through the combinational logic increases rapidly when V_{DD} is approaching V_t.



(a)



(b)

Figure 7: Evaluation of voltage scaling vs reconfiguration combined with voltage scaling in terms of (a) supply voltage and switching capacitance (b) sensitivity to the C_{comb}/C_{clk} ratio.

In Figure 7(b), the normalized total circuit power is computed using Eq (1) in Section 4.2. α_{comb} and α_{clk} are set to 0.1 and 1, respectively. V_{DD} in the high performance mode is equal to $1V$ and V_t for the technology is $0.4V$ [21]. The total circuit power savings for different ratios between C_{comb} and C_{clk} are shown in Figure 7(b). Compared with only applying voltage scaling, the figure shows that the normalized power is reduced by between 5% and 6% using reconfiguration combined with voltage scaling. Naturally, larger power savings are obtained when the C_{comb}/C_{clk} ratio is smaller.

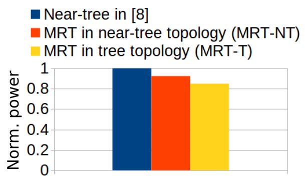


Figure 8: Evaluation of the Near-tree and the MRT structures in the low performance mode.

Next, we focus on comparing the performance of the MRT structures and the Near-tree structures in the low performance mode. The power consumption of the Near-tree structures and the MRT structures in the form of a near-tree (MRT-NT) and a tree (MRT-T) topology is shown in Figure 8. Compared with the Near-tree structure in [8], the MRT-NT structure has 8% lower power consumption. The MRT-T structure has 16% and 8% lower power consumption compared with the Near-tree structure and the MRT-NT structure, respectively. The reconfiguration of the topology based on the active mode is a unique new feature to trade-off between robustness and power consumption.

7 SUMMARY AND FUTURE WORK

In this paper, we proposed the construction of a clock network with a mode reconfigurable topology for positive-edge triggered sequential elements. The proposed MRT structure demonstrates similar robustness with lower costs when compared with state-of-the-art near-tree structures. Moreover, the MRT structure can operate in multiple modes without introducing any short circuit current. In the future, we will explore inserting OR-gates in multiple stages of the topology, integrating the use of useful skew, and handling negative-edge and dual-edge triggered sequential elements.

REFERENCES

[1] Kenneth Boese and Andrew B. Kahng. 1992. Zero-Skew Clock Routing Trees With Minimum Wirelength. In *Proc. of International ASIC Conference and Exhibit*. 17–21.
[2] Shashank Bujimalla and Cheng-Kok Koh. 2011. Synthesis of low power clock trees for handling power-supply variations *(ISPD)*. 37–44.
[3] T.-H. Chao et al. 1992. Zero skew clock net routing *(DAC)*. 518–523.
[4] Yong. P. Chen and D. F. Wong. 1996. An Algorithm for Zero-Skew Clock Tree Routing with Buffer Insertion *(EDTC)*. 230–237.
[5] Masato Edahiro. 1993. A clustering-based optimization algorithm in zero-skew routings *(DAC)*. 612 – 616.
[6] Rickard Ewetz et al. 2015. Benchmark circuits for clock scheduling and synthesis *([Available Online] https://purr.purdue.edu/publications/1759)*.
[7] Rickard Ewetz, Shankarshana Janarthanan, and Cheng-Kok Koh. 2015. Construction of Reconfigurable Clock Trees for MCMM Designs *(DAC)*. 1–6.
[8] Rickard Ewetz and Cheng-Kok Koh. 2015. Cost-Effective Robustness in Clock Networks Using Near-Tree Structures. *TCAD* 34, 4 (2015), 515–528.
[9] S. Held et al. 2003. Clock scheduling and clocktree construction for high performance ASICs *(ICCAD)*. 232–239.
[10] Chau-Chin Huang et al. 2016. Timing-driven Cell Placement Optimization for Early Slack Histogram Compression *(DAC)*. 81:1–81:6.
[11] Myung-Chul Kim et al. 2015. ICCAD-2015 CAD Contest in Incremental Timing-driven Placement and Benchmark Suite *(ICCAD)*. 921–926.
[12] Dong-Jin Lee and Igor L. Markov. 2011. Multilevel tree fusion for robust clock networks *(ICCAD)*. 632–639.
[13] Jianchao Lu and Baris. Taskin. 2009. Post-CTS clock skew scheduling with limited delay buffering *(MWSCAS)*. 224–227.
[14] Tarun Mittal and Cheng-Kok Koh. 2011. Cross link insertion for improving tolerance to variations in clock network synthesis *(ISPD)*. 29–36.
[15] NGSPICE. 2012. [Available Online] http://ngspice.sourceforge.net/. (2012).
[16] Anand Rajaram et al. 2004. Reducing clock skew variability via cross links *(DAC)*. 18–23.
[17] Anand Rajaram et al. 2005. Improved algorithms for link-based non-tree clock networks for skew variability reduction *(ISPD)*. 55–62.
[18] Subhendu Roy et al. 2014. Clock Tree Resynthesis for Multi-corner Multi-mode Timing Closure *(ISPD)*. 69–76.
[19] Hyungjung Seo et al. 2015. Synthesis for Power-Aware Clock Spines *(ICCAD)*. 126–131.
[20] Xin-Wei Shih et al. 2010. High variation-tolerant obstacle-avoiding clock mesh synthesis with symmetrical driving trees *(ICCAD)*. 452–457.
[21] Cliff N. Sze. 2010. ISPD 2010 High Performance Clock Network Synthesis Contest: Benchmark Suite and Results *(ISPD)*. 143–143.
[22] Cliff N. Sze et al. 2009. ISPD 2009 Clock network synthesis contest *(ISPD)*. 149–150.
[23] Chung-Wen Albert Tsao and Cheng-Kok Koh. 2002. UST/DME: a clock tree router for general skew constraints. *TODAES* 7, 3 (2002), 359–379.
[24] R.-S. Tsay. 1991. Exact zero skew *(ICCAD)*. 336–339.
[25] G. Venkataraman et al. 2005. Practical techniques to reduce skew and its variations in buffered clock networks *(ICCAD)*.
[26] Ganesh Venkataraman et al. 2006. Combinatorial algorithms for fast clock mesh optimization *(ICCAD)*. 563–567.
[27] Laung-Terng Wang et al. 2009. *Electronic design automation: synthesis, verification, and test.* Morgan Kaufmann.
[28] Linfu Xiao et al. 2010. Local clock skew minimization using blockage-aware mixed tree-mesh clock network *(ICCAD)*. 458 – 462.